EE354L

Nexys-4 board

Introduction

Lab Introduction to the Nexys-4 board

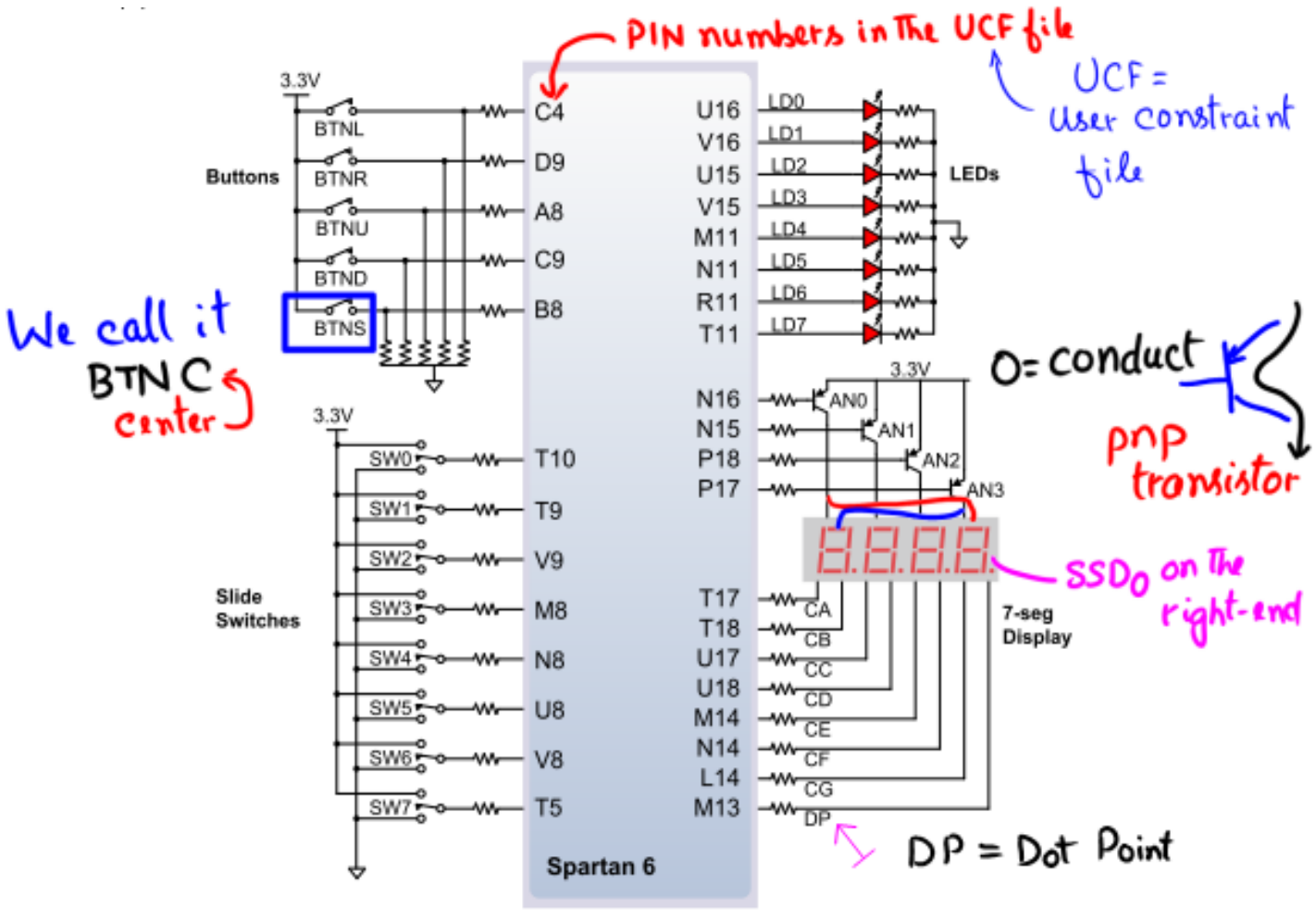
Nexys-4 FPGA board is introduced using the following marked up copy of the Nexys4_rm.pdf:

http://www-classes.usc.edu/engr/ee-s/254/ee254l_lab_manual/Nexys_4_documentation/nexys4_rm.pdf

Basic I/O

Very basic and important for the first few labs

Nexys 3



Nexys 4

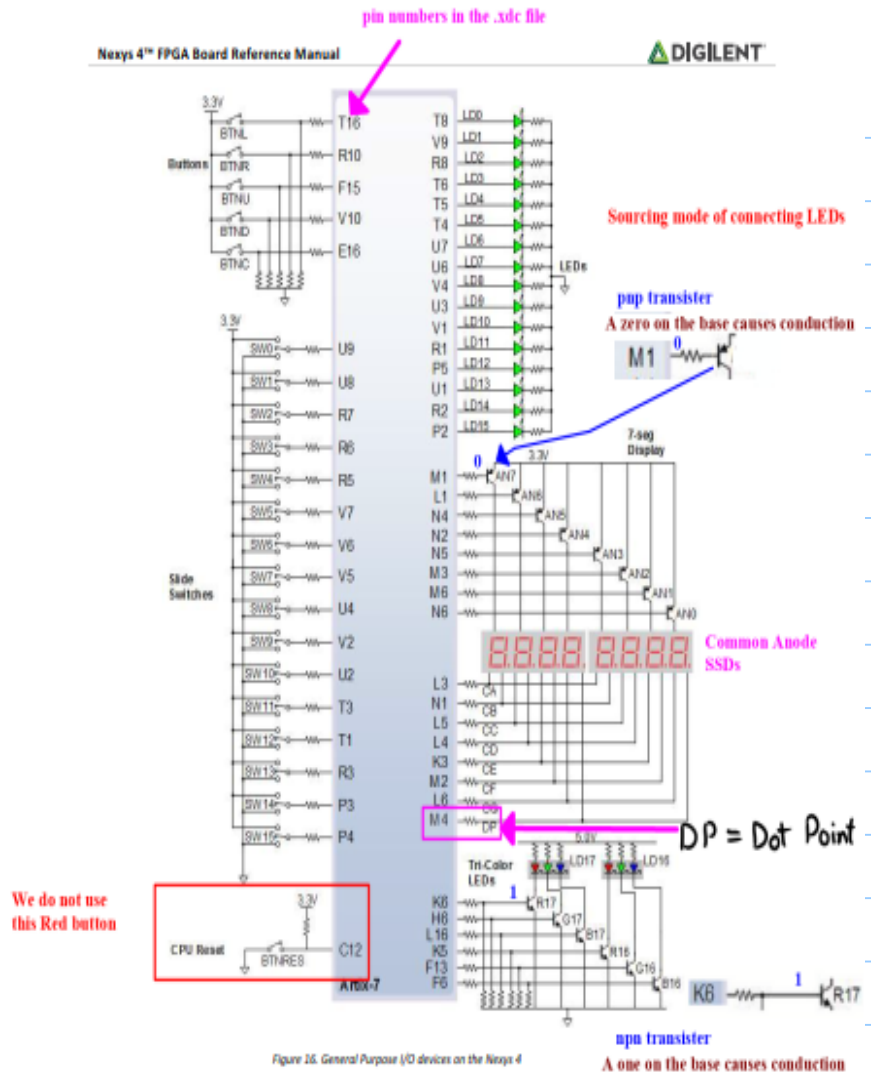
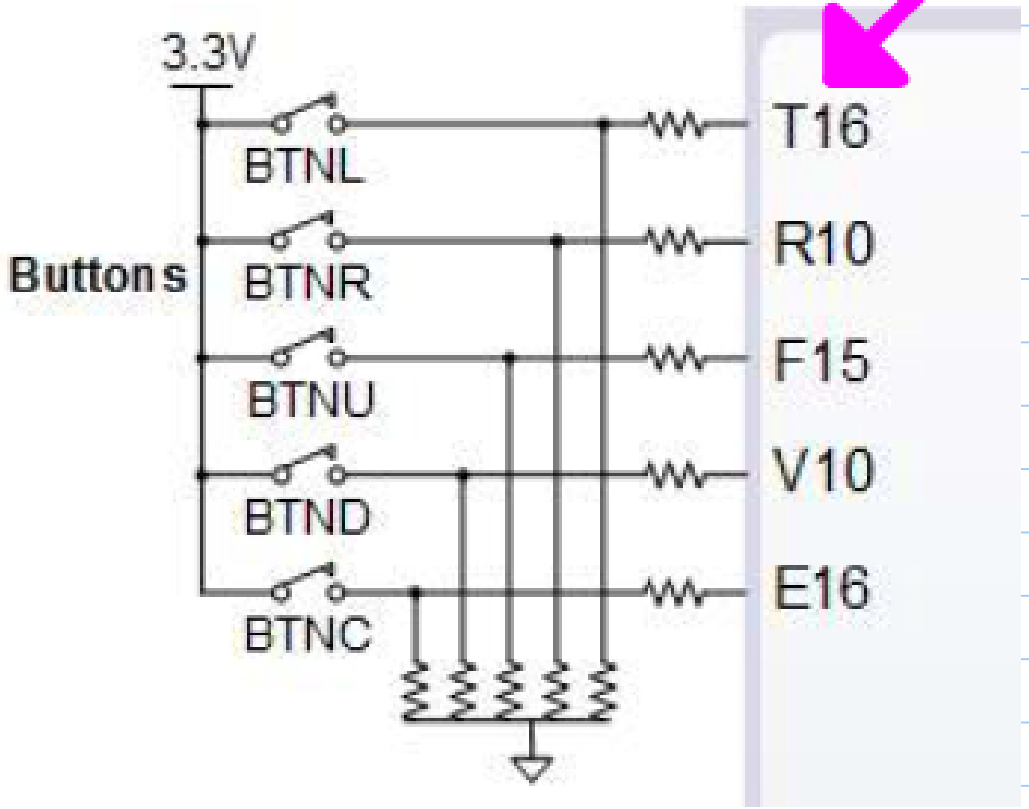
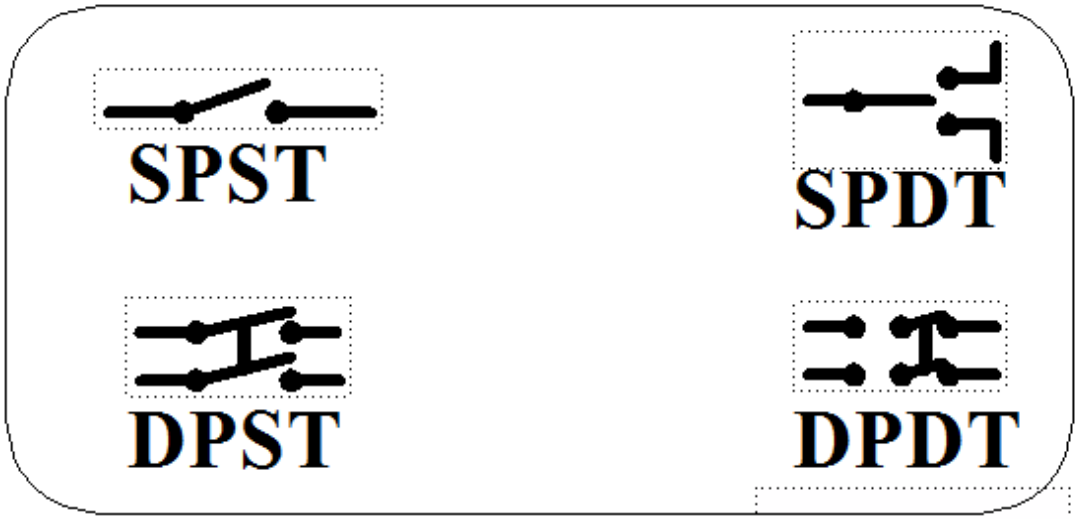


Figure 16. General Purpose I/O devices on the Nexys 4

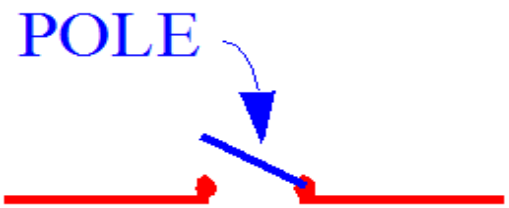
pin numbers in the .xdc file



PB

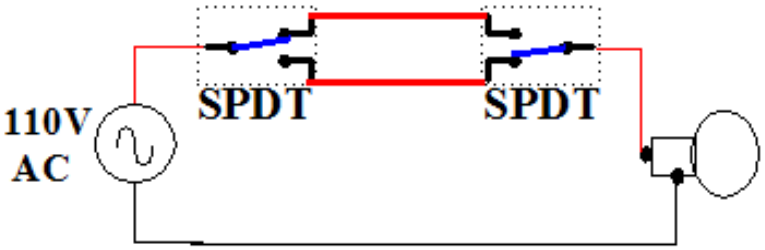


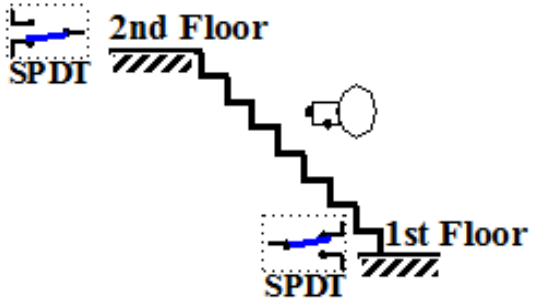
SPST Switch
Single-Pole-Single-Throw Switch



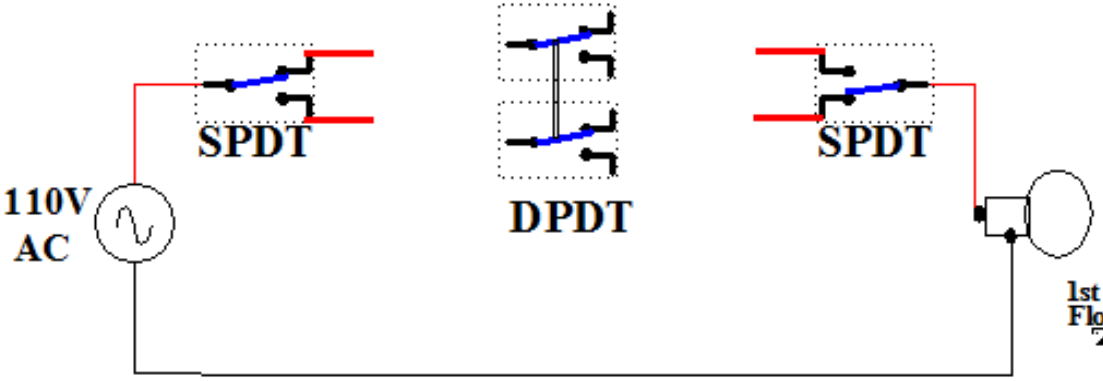
It is called single-throw because it makes contact only on one-side throw.

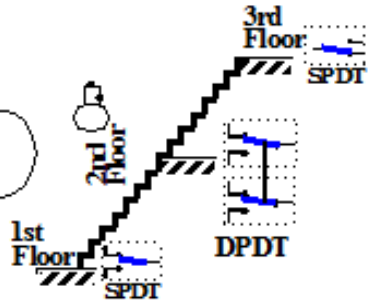
Exercise for fun: Explain how the two-way stair-case light control circuit works.





Now complete the three-way stair-case light control circuit below.



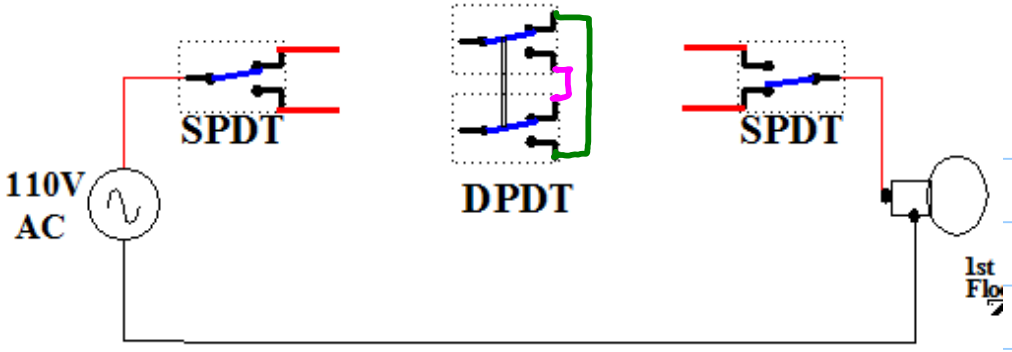


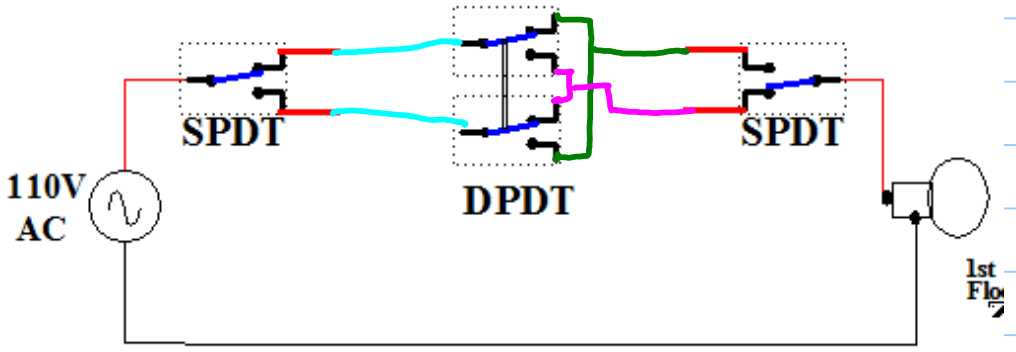
Hint: Wire-up the DPDT so as to make a Straight/Cross connection:


Straight

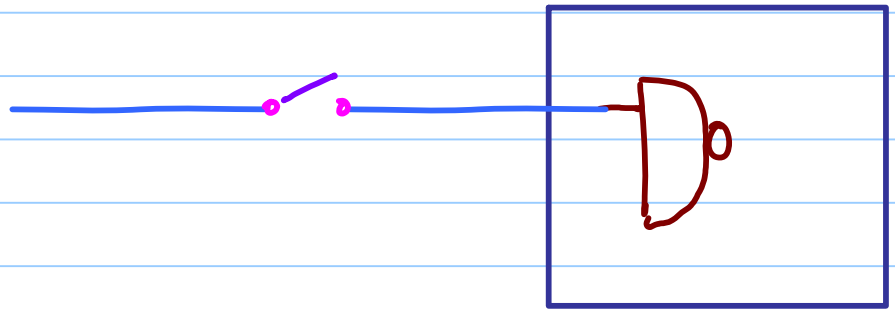

Cross

How many **SPDTs** and how many **DPDTs** you think you need to control one light-bulb common to a **10-floor** staircase. **Total 11 switches.**

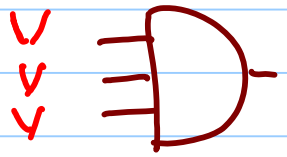

SPDT
DPDT
SPDT
110V
AC
1st


SPDT
SPDT
DPDT
110V
AC
1st

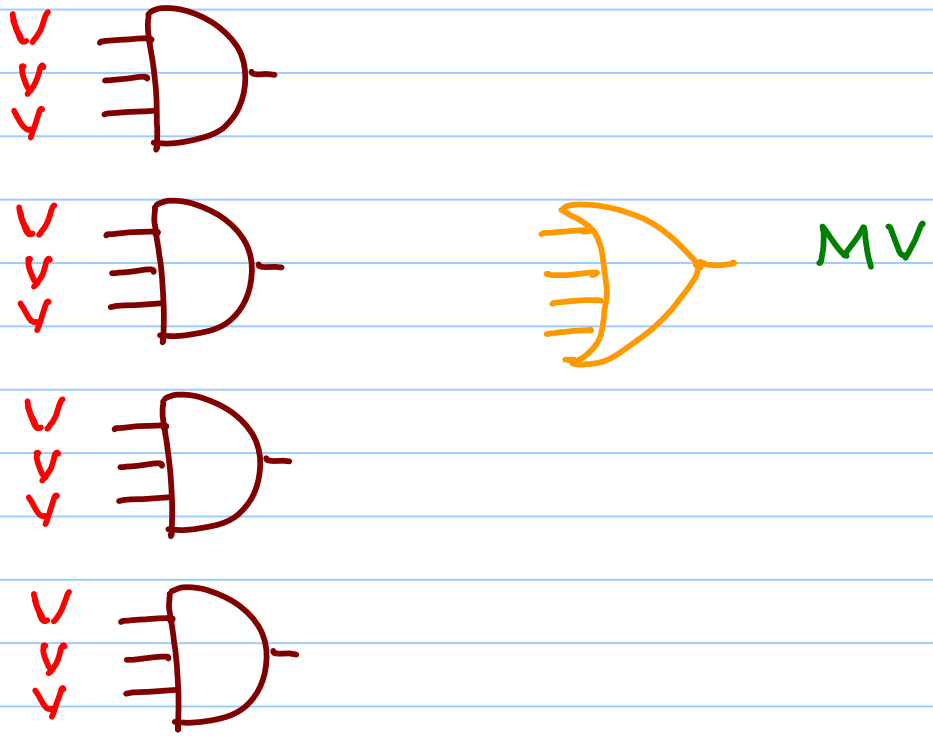
How to connect Push Buttons or SPST switches to a digital circuit ?



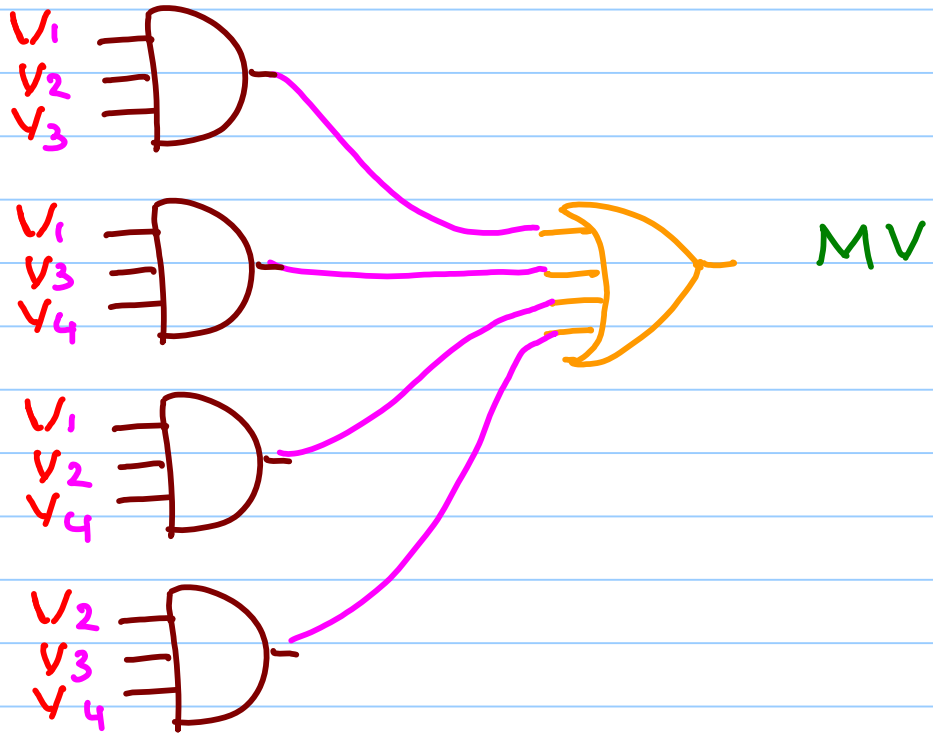
Majority vote V1, V2, V3, V4



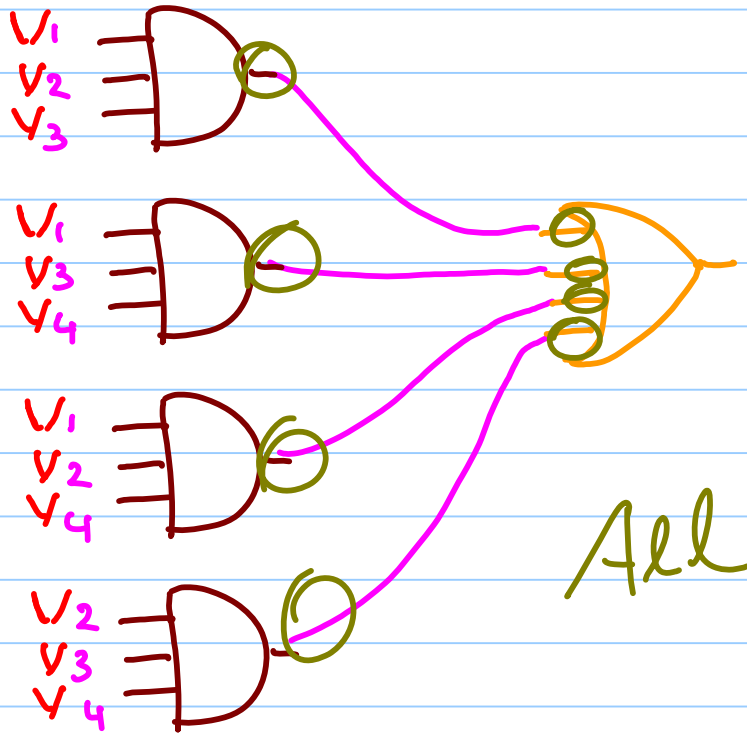
Majority vote V1, V2, V3, V4



Majority vote V1, V2, V3, V4

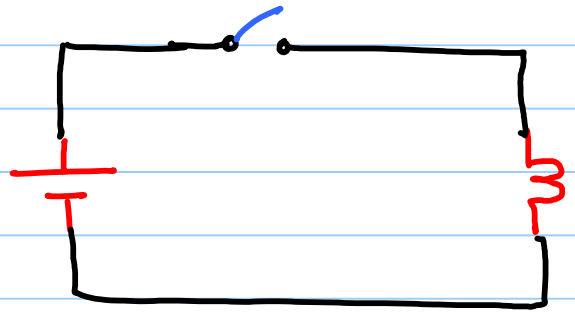


Majority vote V1, V2, V3, V4

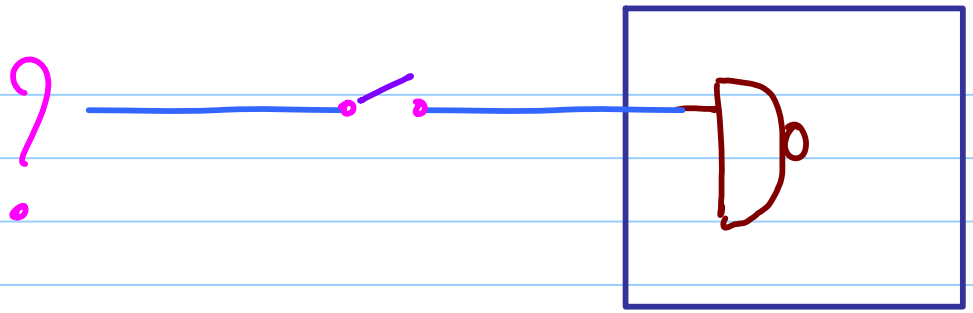


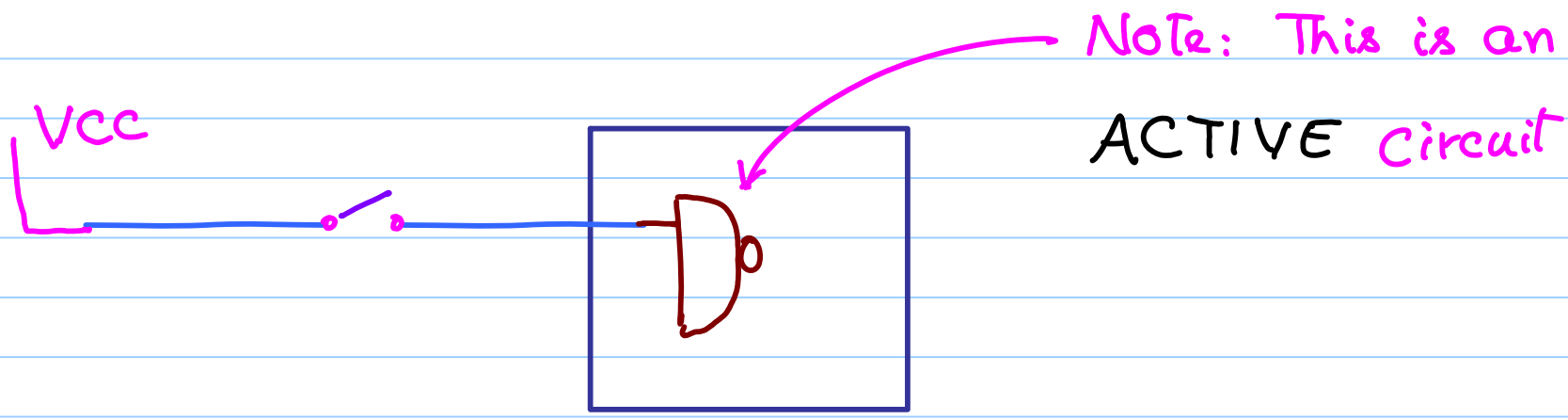
All Nand Gates!

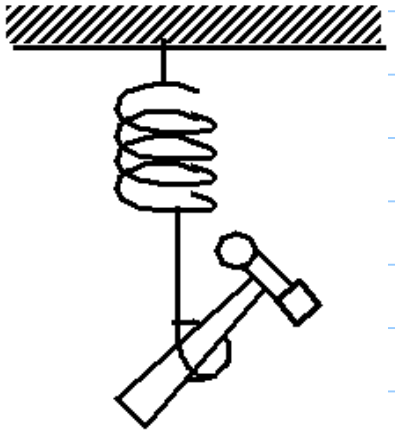
Flash light control switch

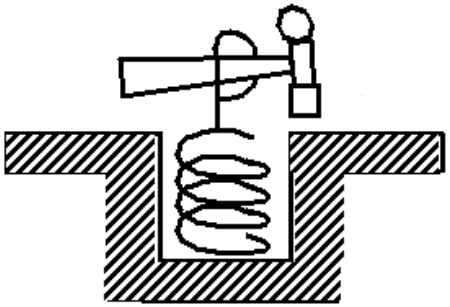


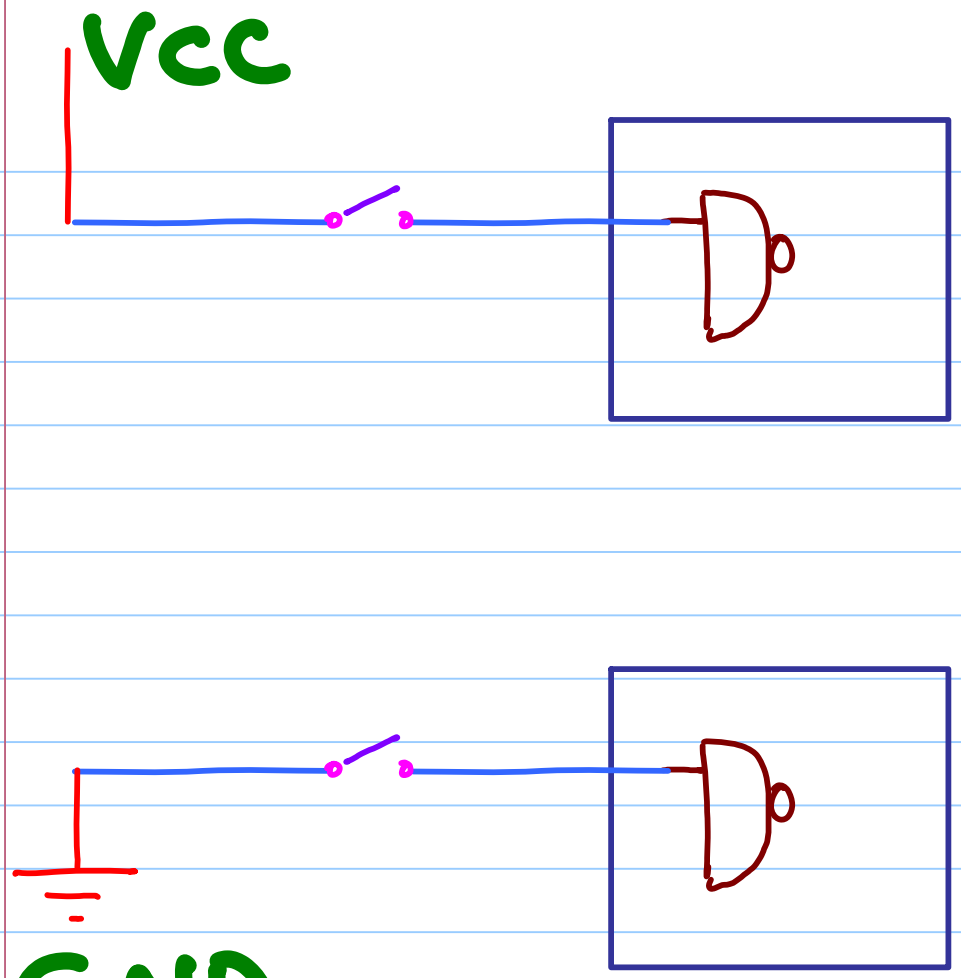
<a href="http://www.learnersdictionary.com/search/flashlight">

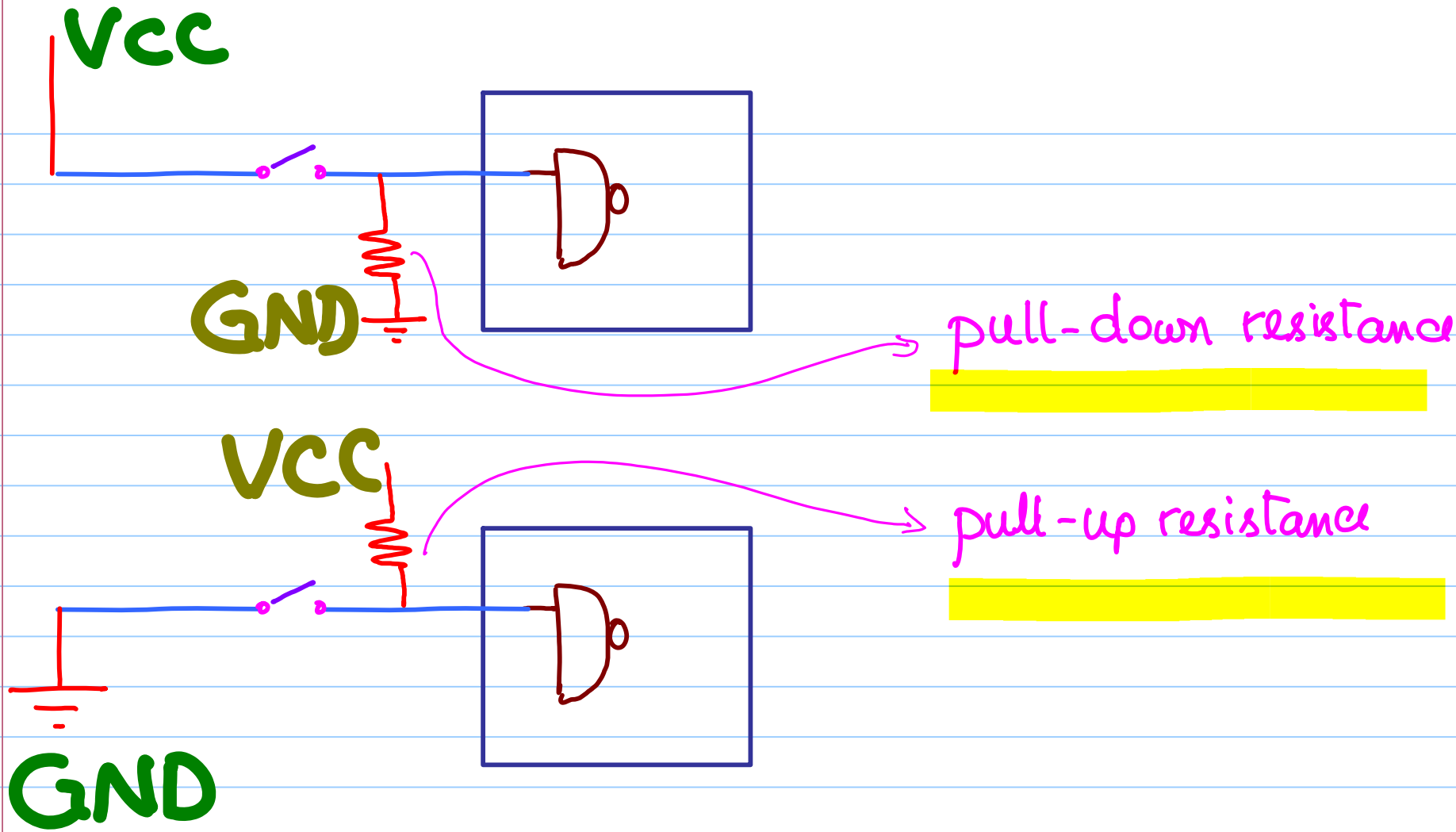

?


Vcc
Note: This is an
ACTIVE circuit



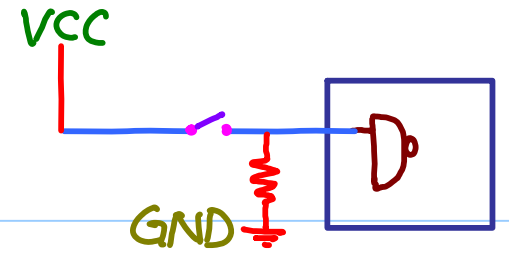



Vcc
GND

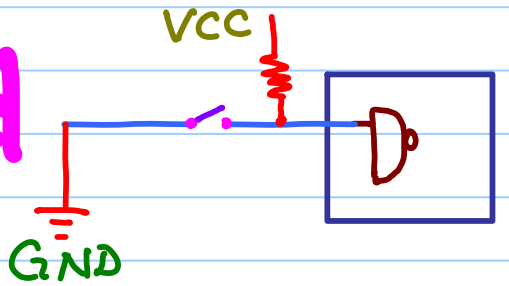

Vcc
GND
pull-down resistance
VCC
pull-up resistance
GND

CMOS

Complementary Metal Oxide Semiconductor

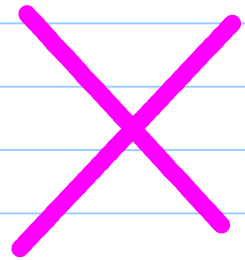


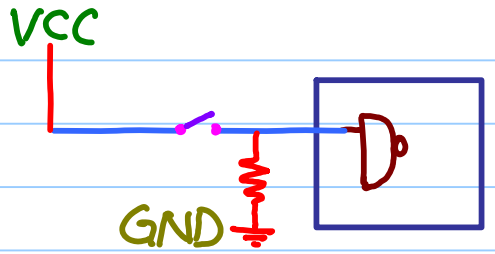
Both are good

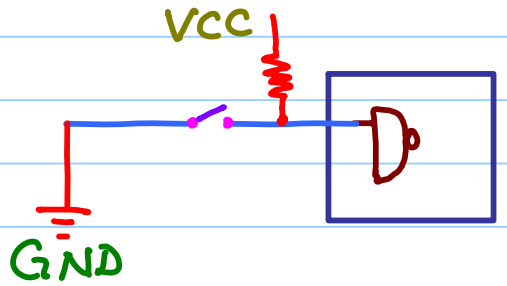


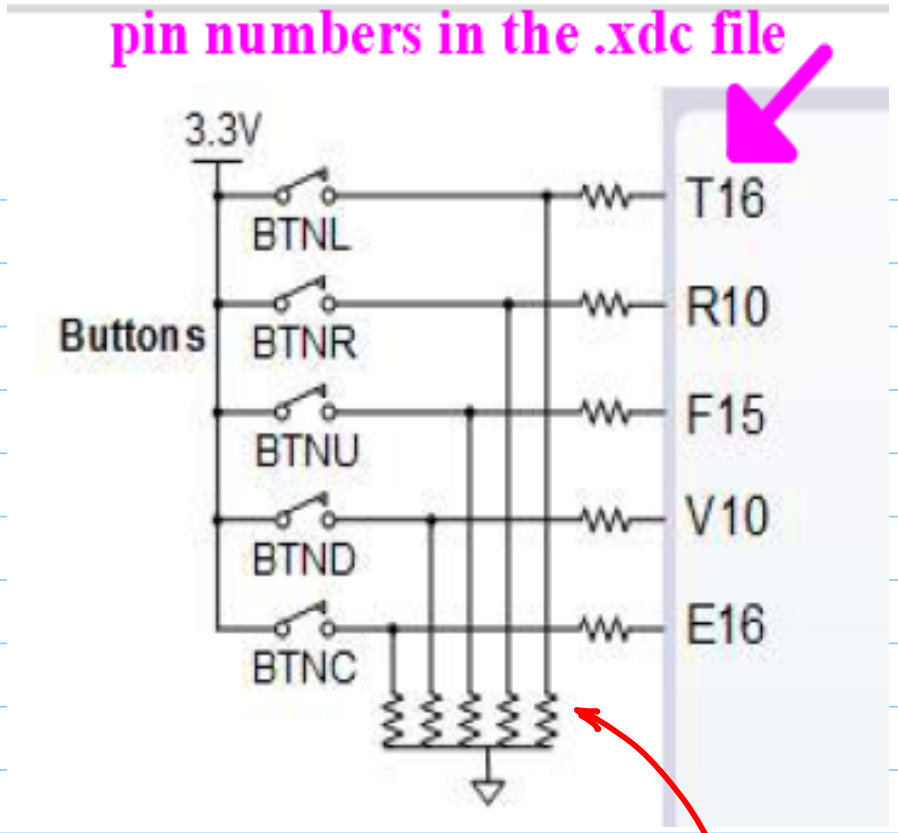
TTL

Transistor Transistor Logic



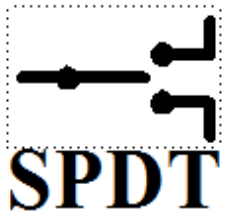




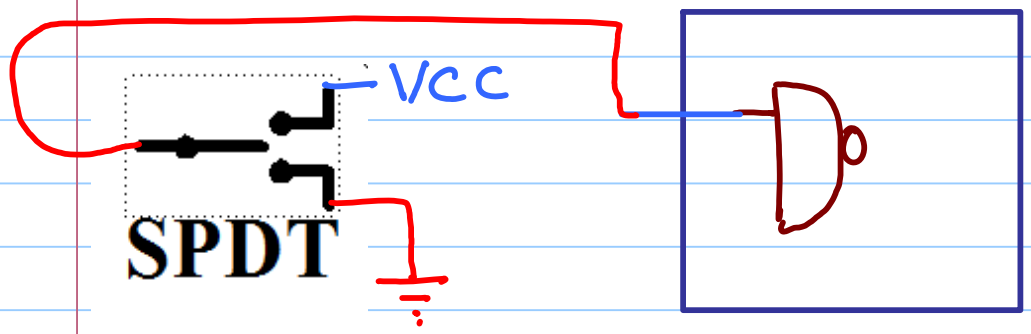

pin numbers in the .xdc file
3.3V
Buttons
BTNL
BTNR
BTNU
BTND
BTNC
T16
R10
F15
V10
E16
pull-down resistances

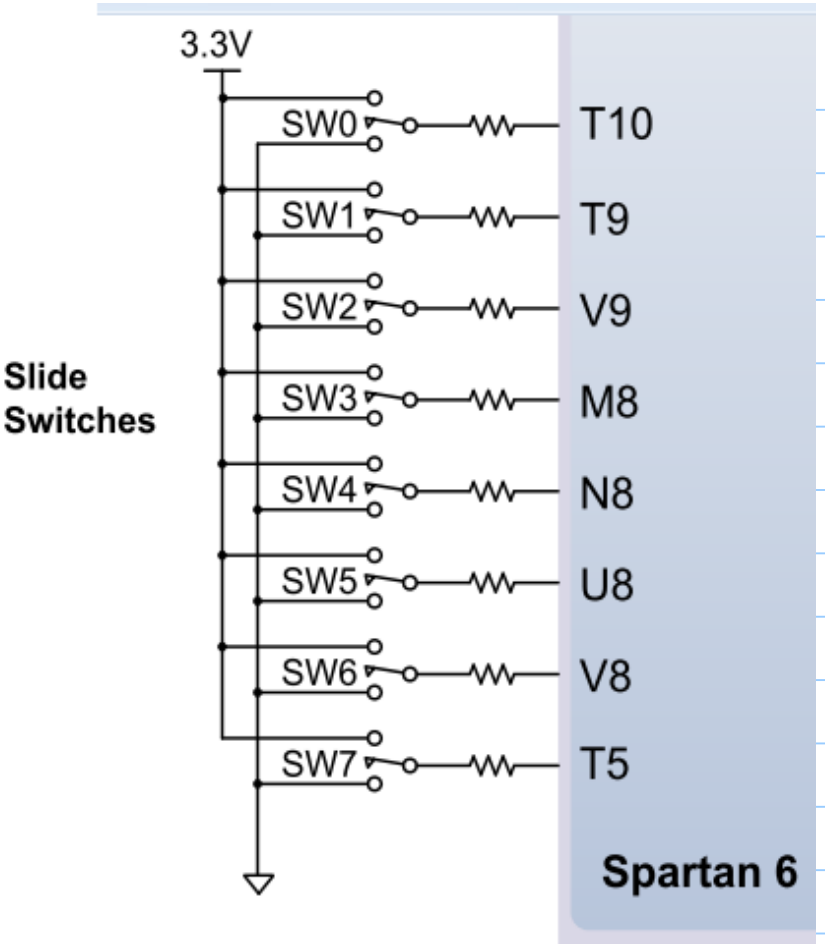
CMOS

Unlike PBs / SPST



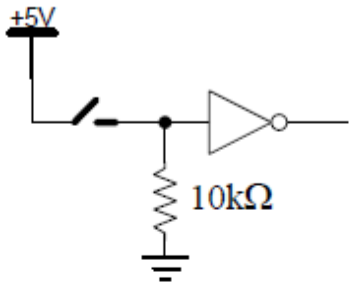
allows an easy connection





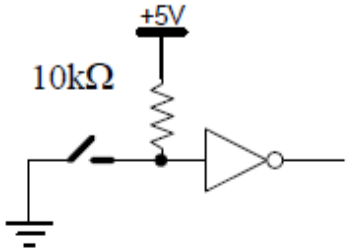
VOTING MACHINE LAB

Method 1



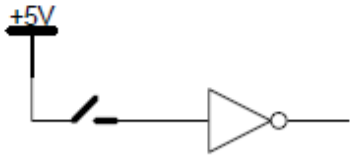
Is this configuration workable?
Is this configuration preferred?

Method 2



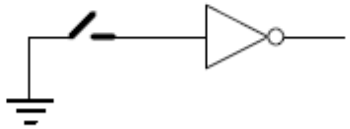
Is this configuration workable?
Is this configuration preferred?

Method 3



Is this configuration workable?
Is this configuration preferred?

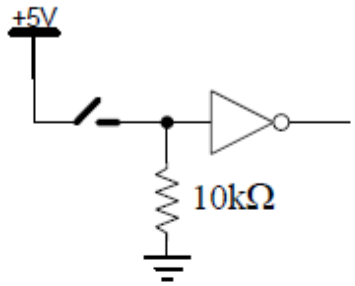
Method 4



Is this configuration workable?
Is this configuration preferred?

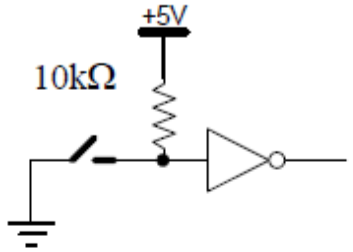
CMOS

Method 1



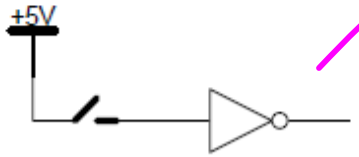
Is this configuration workable?
Is this configuration preferred?

Method 2



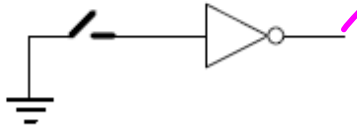
Is this configuration workable?
Is this configuration preferred?

Method 3



Is this configuration workable?
Is this configuration preferred?

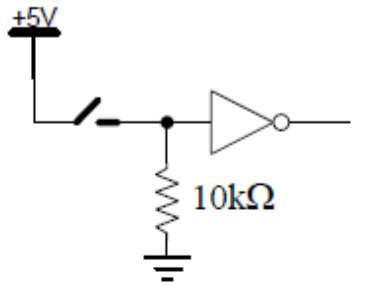
Method 4



Is this configuration workable?
Is this configuration preferred?

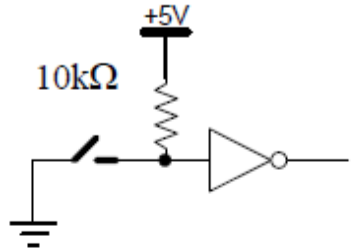
TTL

Method 1



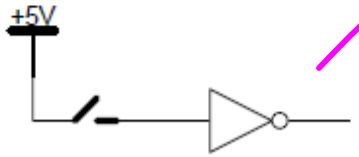
Is this configuration workable?
Is this configuration preferred?

Method 2



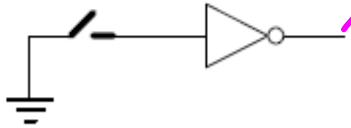
Is this configuration workable?
Is this configuration preferred?

Method 3



Is this configuration workable?
Is this configuration preferred?

Method 4

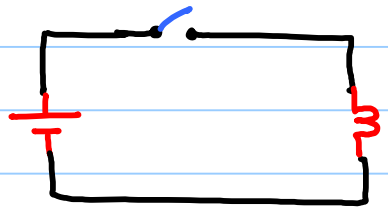


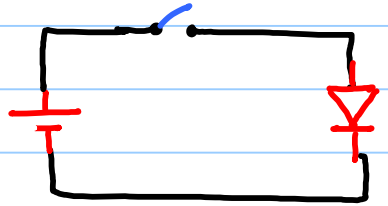
Is this configuration workable?
Is this configuration preferred?

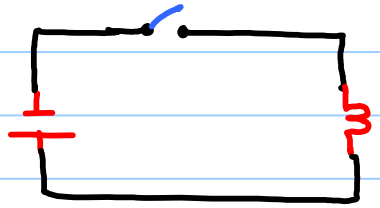
Now let us shift our attention to the

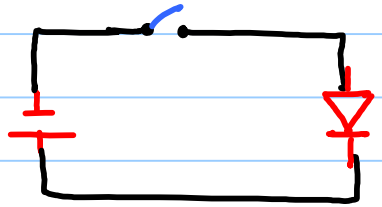
OUTPUTs.

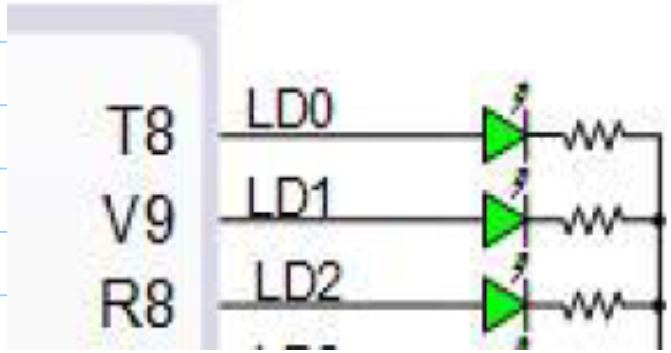
incandescent vs. LED flashlight

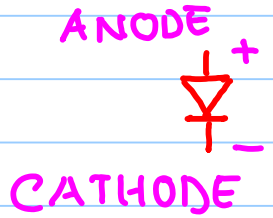


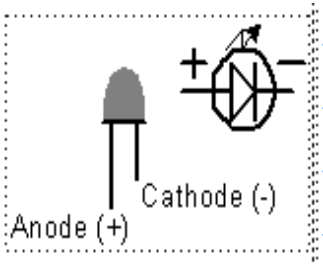








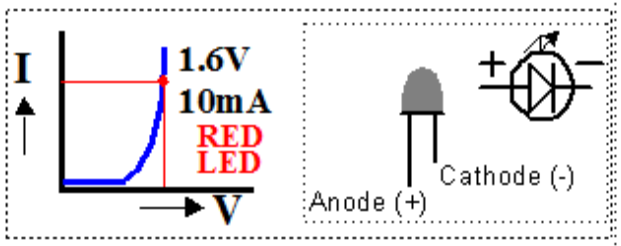




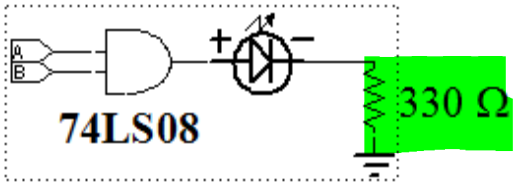
Series current Limiting resistance

18) LEDs (Light-Emitting Diodes)

LED is a non-linear device (see the V-I characteristic). Since current increases steeply once you cross the threshold voltage, we need a current limiting resistance (example: 330 ohms resistance) wired in series with the LED.



Sourcing method to drive an LED is not desirable.

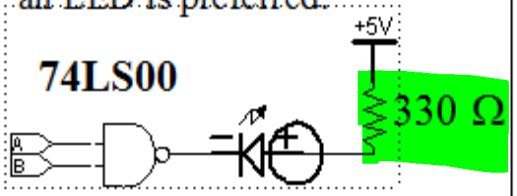


§ Voh (typ) = 3.4V

$$Current = \frac{3.4V - 1.9V}{330\Omega} = 4.5mA$$

4.5mA exceeds $|I_{OHmax}|$ of 0.4mA.

Sinking method to drive an LED is preferred.



§ Typical V_{OL} = 0.25V

$$Current = \frac{5.0V - 1.9V - 0.25V}{330\Omega} = 8.6mA$$

8.6mA is not too far from $|I_{OLmax}|$ of 8.0mA.

Whistle

You can make sound

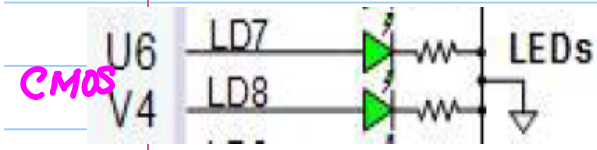
ⓐ by blowing air out

ⓑ by sucking air through it

drawing

CMOS

Complementary Metal Oxide Semiconductor

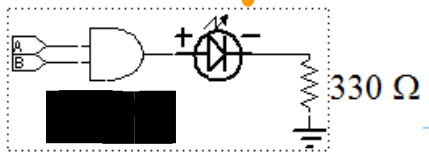


CMOS

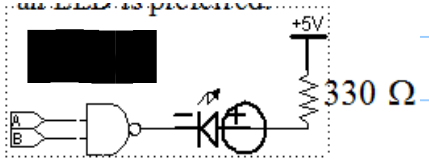
Both are good

sourcing mode

Sourcing Mode



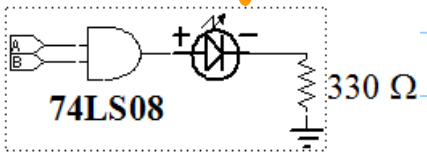
Sinking Mode



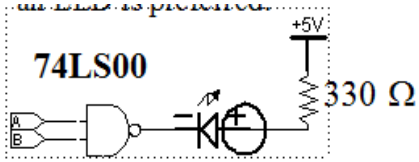
TTL

Transistor Transistor Logic

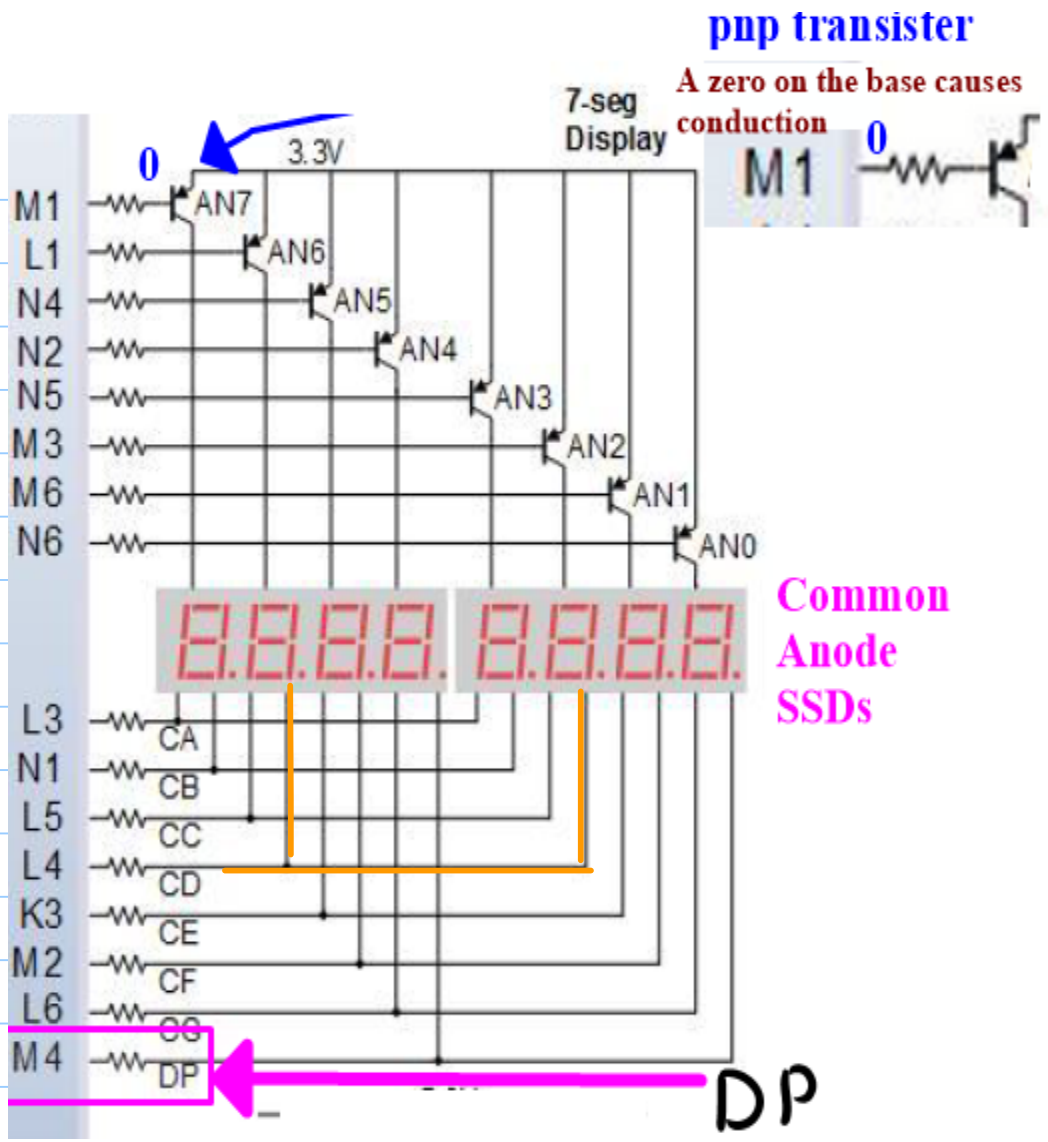
Sourcing Mode

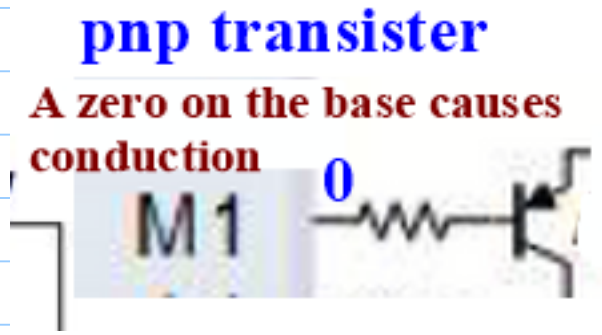


Sinking Mode



Nexys 4

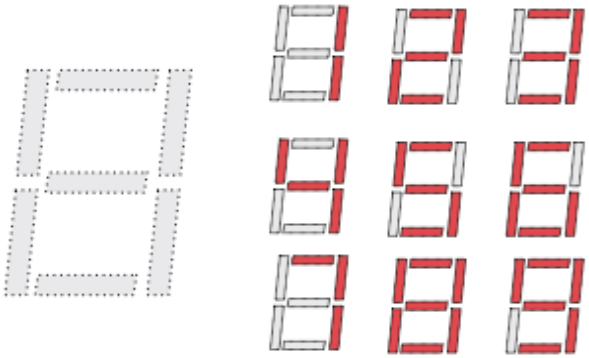




Look at the Cathode CD

Before the 7-Segment display,

Nixie tubes were used.

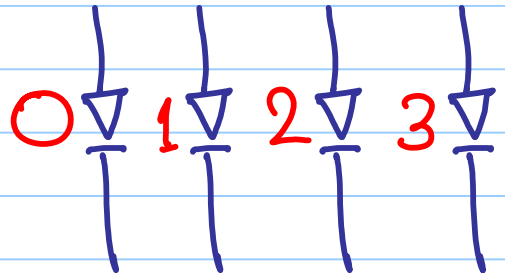


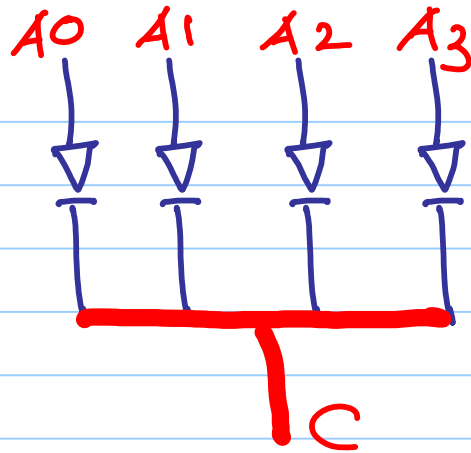
An un-illuminated seven-segment display, and nine illumination patterns corresponding to decimal digits

10 filaments ⇒ 10 cathodes + 1 anode

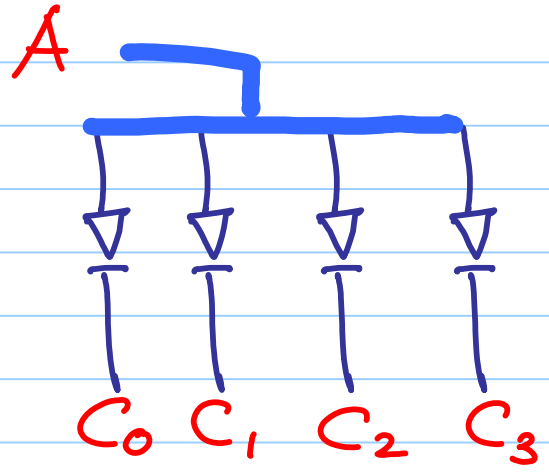
Consider a Nixie with 4 digits

8 terminals ?



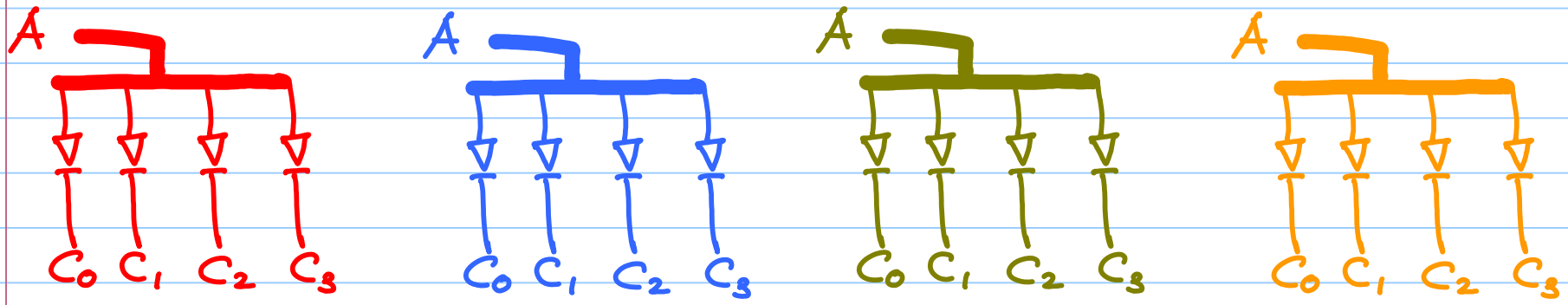


5 terminals

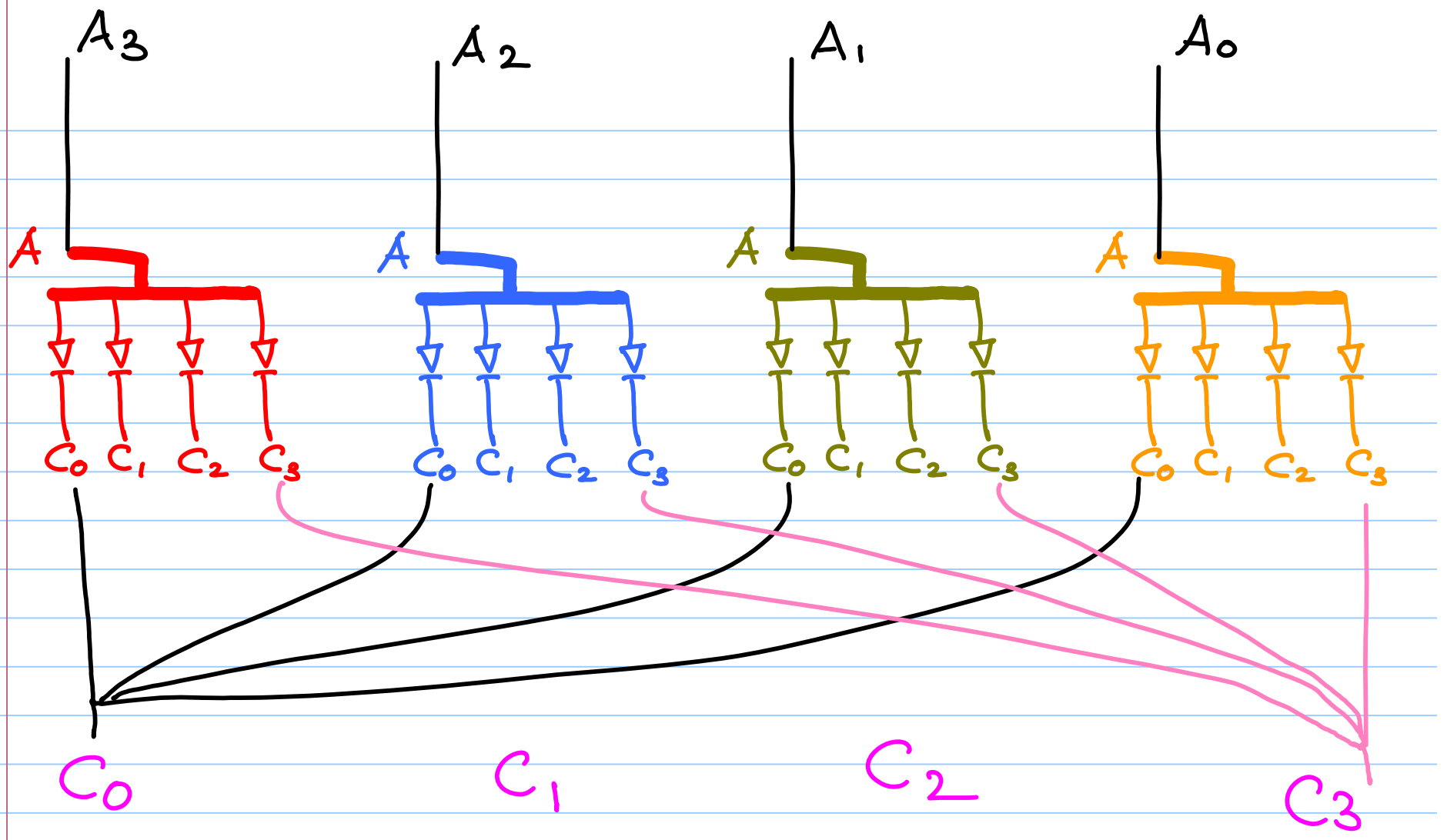


5 terminals

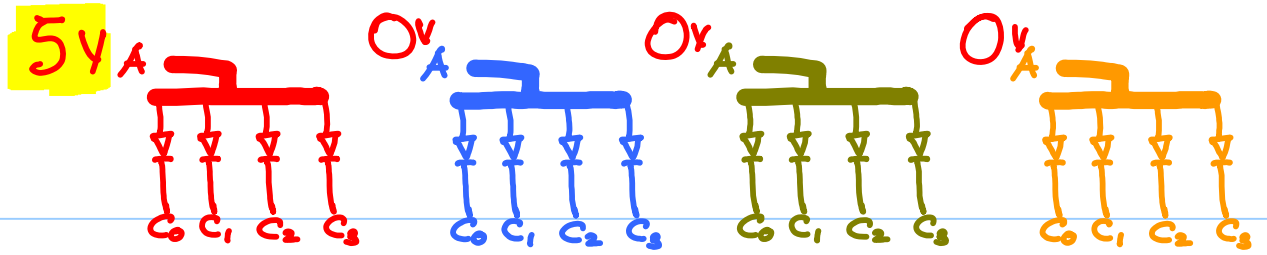
To control 4 such Nixies → 20 Terminals ?



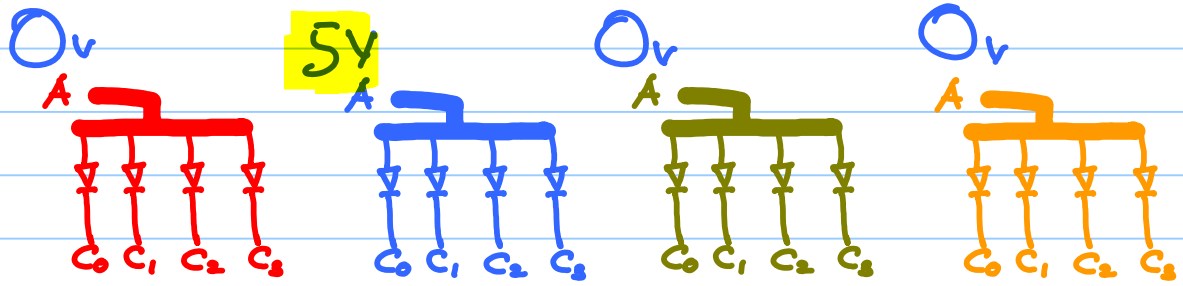
Can you reduce to 8 ?


A3
A2
A1
A0
A
A
A
A
C0 C1 C2 C3
C0 C1 C2 C3
C0 C1 C2 C3
C0 C1 C2 C3
C0
C1
C2
C3

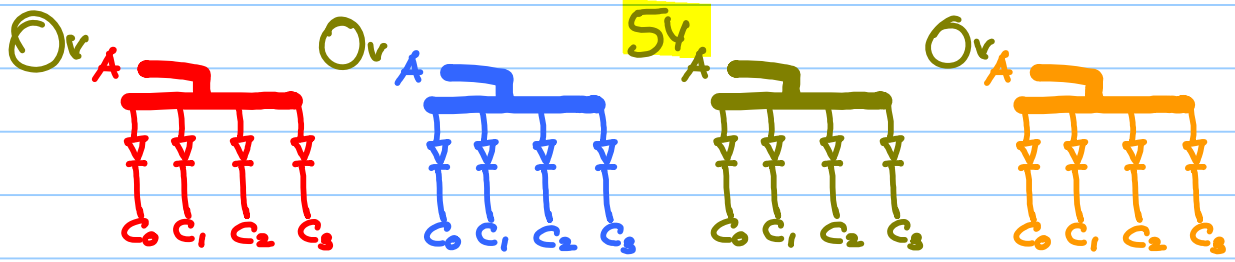
Red



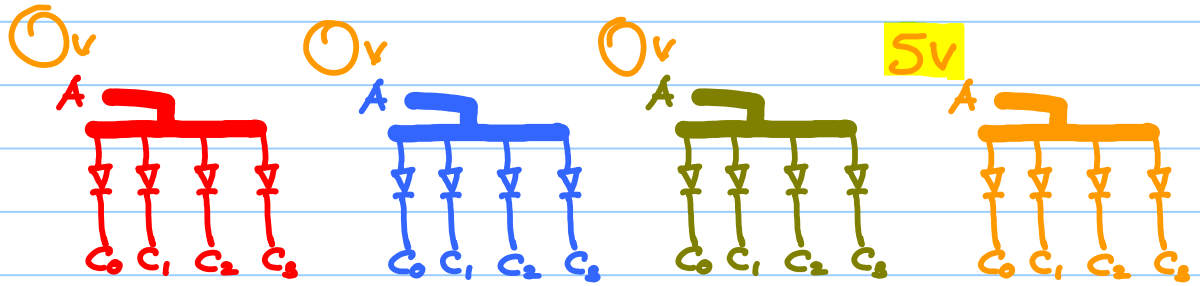
Blue



Green



Orange



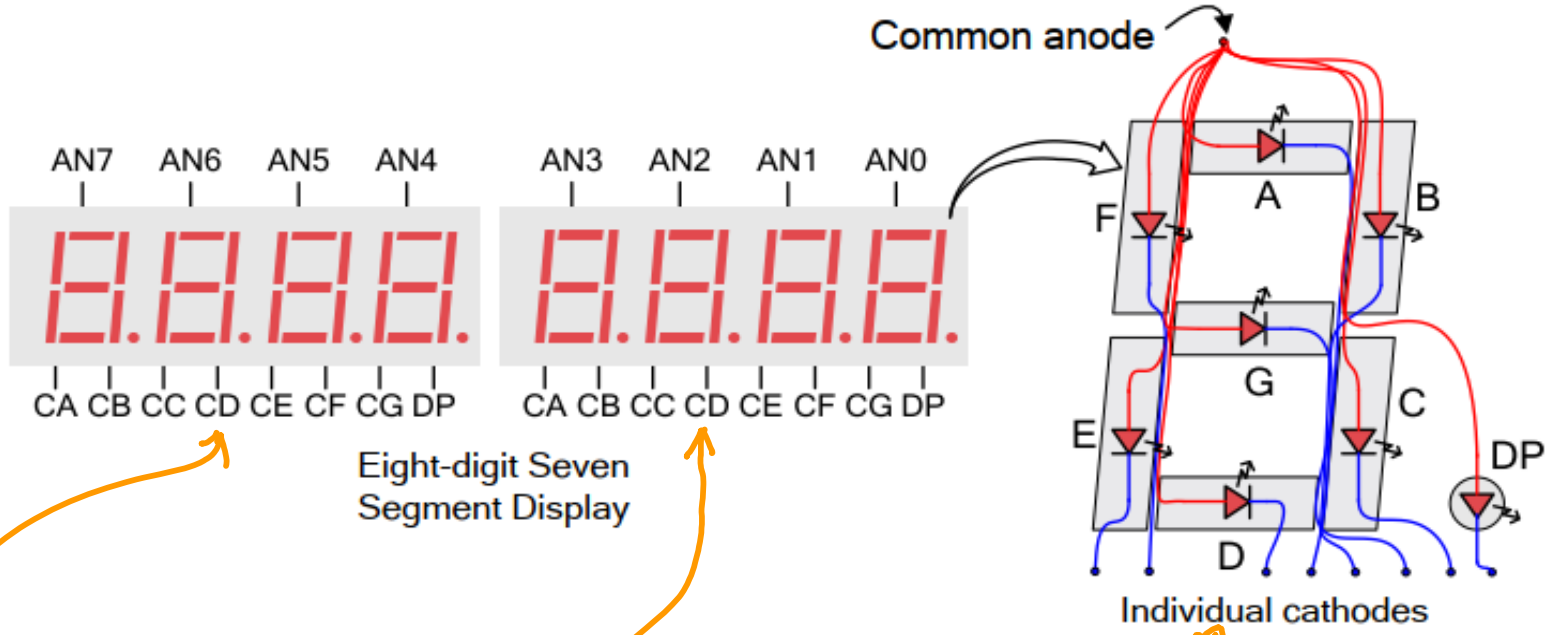
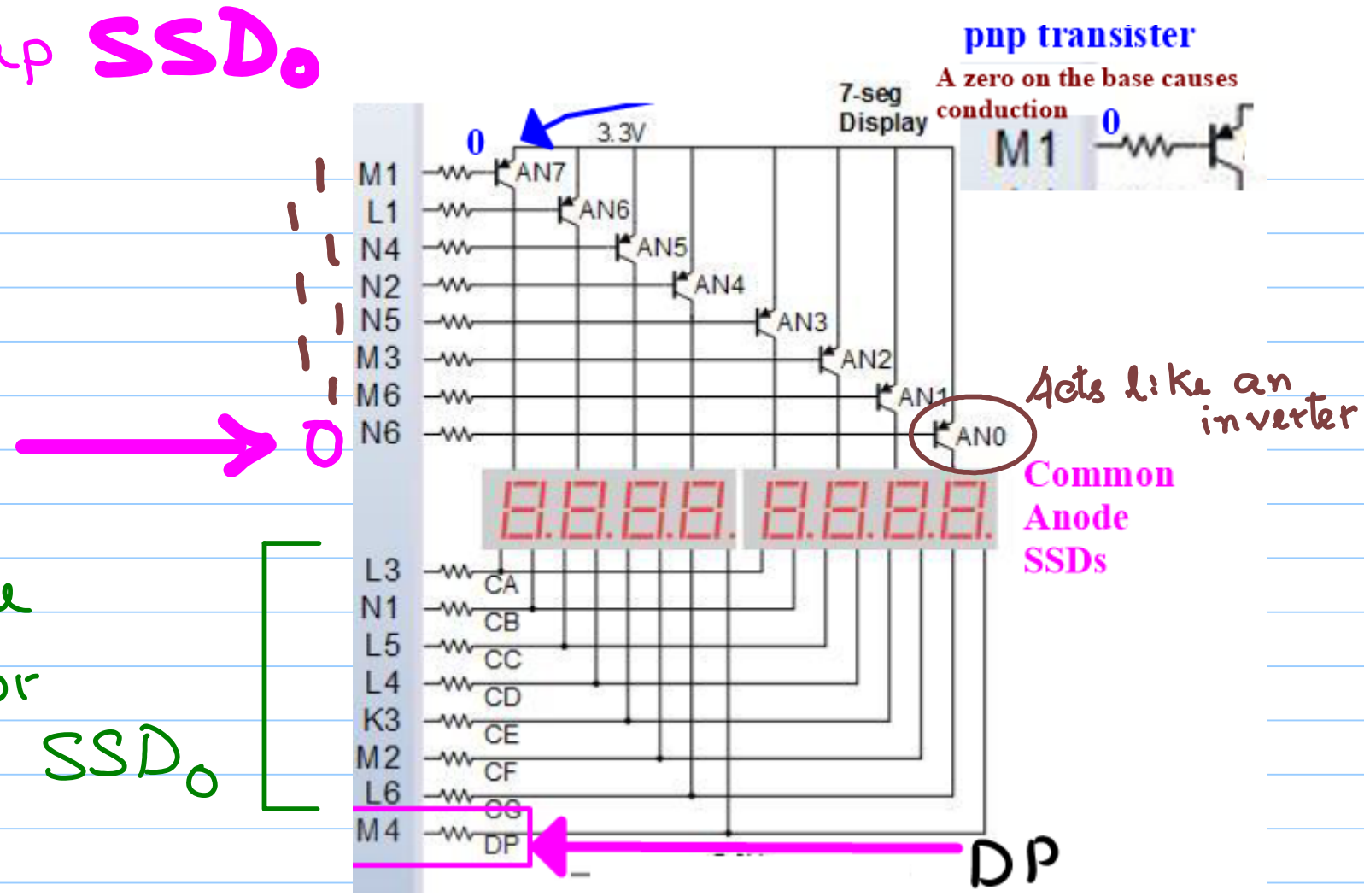


Figure 18. Common anode circuit node

CD is all 4 individual D's joined together

To light up **SSD0**



Code for SSD_0

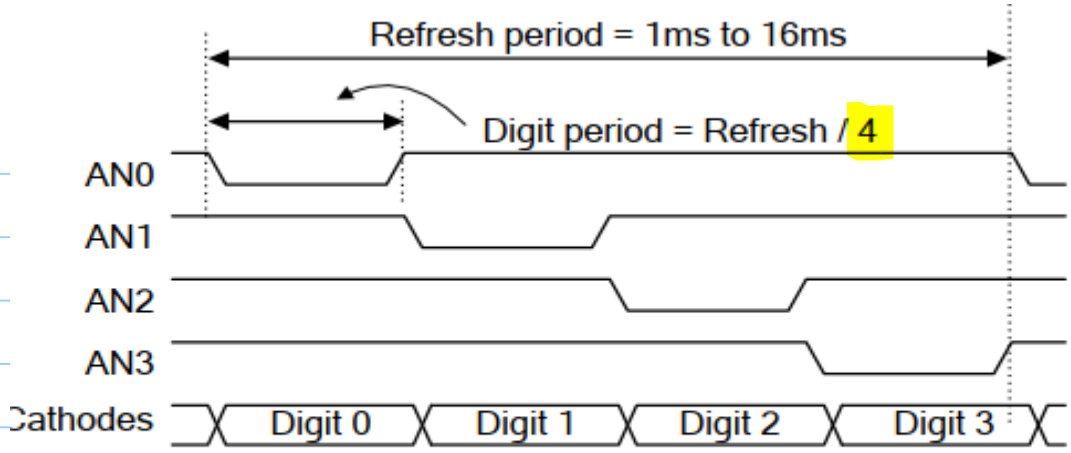


Figure 19. Four digit scanning display controller timing diagram

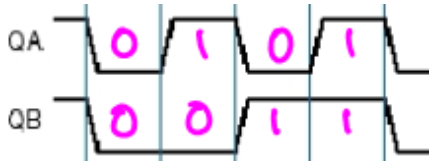
The diagram is drawn for four SSDs (like in the Nexys-3).

But we have 8 SSDs in the Nexys-4.

Nexys 3

```
File      : test_nexys3_verilog.v
Author    : Gandhi Puvvada
```

```
assign An0  = ~(~(sev_seg_clk[1]) && ~(sev_seg_clk[0]));  // when sev_seg_clk = 00
assign An1  = ~(~(sev_seg_clk[1]) &&  (sev_seg_clk[0]));  // when sev_seg_clk = 01
assign An2  = ~( (sev_seg_clk[1]) && ~(sev_seg_clk[0]));  // when sev_seg_clk = 10
assign An3  = ~( (sev_seg_clk[1]) &&  (sev_seg_clk[0]));  // when sev_seg_clk = 11

assign SSD0 = {  Sw3,   Sw2,   Sw1,   Sw0};
assign SSD1 = {  Sw7,   Sw6,   Sw5,   Sw4};
assign SSD2 = { ~Sw3,  ~Sw2,  ~Sw1,  ~Sw0};
assign SSD3 = { ~Sw7,  ~Sw6,  ~Sw5,  ~Sw4};

always @ (sev_seg_clk, SSD0, SSD1, SSD2, SSD3)
begin
   case (sev_seg_clk)
            2'b00: SSD = SSD0;
            2'b01: SSD = SSD1;
            2'b10: SSD = SSD2;
            2'b11: SSD = SSD3;
   endcase
end
```

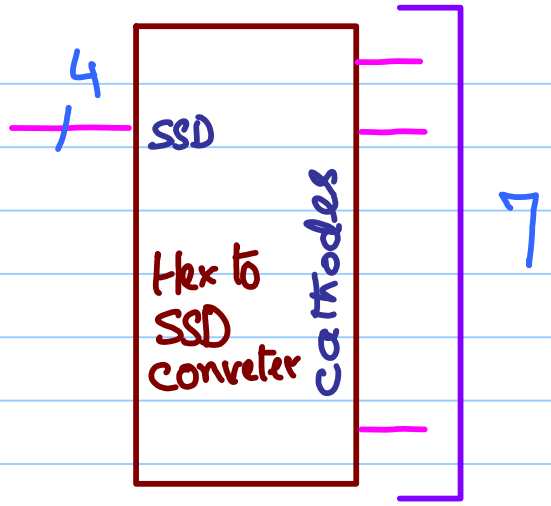


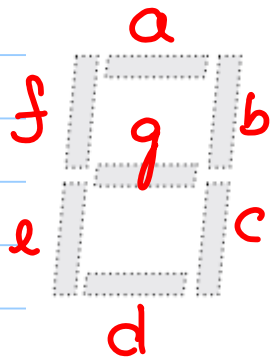
test_nexys4_verilog.v

Nexys 4

```
196  assign AN0  = ~(~(sev_seg_clk[2]) && ~(sev_seg_clk[1]) && ~(sev_seg_clk[0]));
197  assign AN1  = ~(~(sev_seg_clk[2]) && ~(sev_seg_clk[1]) &&  (sev_seg_clk[0]));
198  assign AN2  = ~(~(sev_seg_clk[2]) &&  (sev_seg_clk[1]) && ~(sev_seg_clk[0]));
199  assign AN3  = ~(~(sev_seg_clk[2]) &&  (sev_seg_clk[1]) &&  (sev_seg_clk[0]));
200  assign AN4  = ~( (sev_seg_clk[2]) && ~(sev_seg_clk[1]) && ~(sev_seg_clk[0]));
201  assign AN5  = ~( (sev_seg_clk[2]) && ~(sev_seg_clk[1]) &&  (sev_seg_clk[0]));
202  assign AN6  = ~( (sev_seg_clk[2]) &&  (sev_seg_clk[1]) && ~(sev_seg_clk[0]));
203  assign AN7  = ~( (sev_seg_clk[2]) &&  (sev_seg_clk[1]) &&  (sev_seg_clk[0]));
204
205  assign SSD0 = {   SW3,   SW2,   SW1,   SW0};
206  assign SSD1 = {   SW7,   SW6,   SW5,   SW4};
207  assign SSD2 = {  SW11,  SW10,   SW9,   SW8};
208  assign SSD3 = {  SW15,  SW14,  SW13,  SW12};
209  assign SSD4 = {  ~SW3,  ~SW2,  ~SW1,  ~SW0};
210  assign SSD5 = {  ~SW7,  ~SW6,  ~SW5,  ~SW4};
211  assign SSD6 = { ~SW11, ~SW10,  ~SW9,  ~SW8};
212  assign SSD7 = { ~SW15, ~SW14,  ~SW13, ~SW12};
```

```
214  always @ (sev_seg_clk, SSD0, SSD1, SSD2, SSD3, SSD4, SSD5, SSD6, SSD7)
215  begin
216      case (sev_seg_clk)
217              3'b000: SSD = SSD0;
218              3'b001: SSD = SSD1;
219              3'b010: SSD = SSD2;
220              3'b011: SSD = SSD3;
221              3'b100: SSD = SSD4;
222              3'b101: SSD = SSD5;
223              3'b110: SSD = SSD6;
224              3'b111: SSD = SSD7;
225      endcase
226  end
```

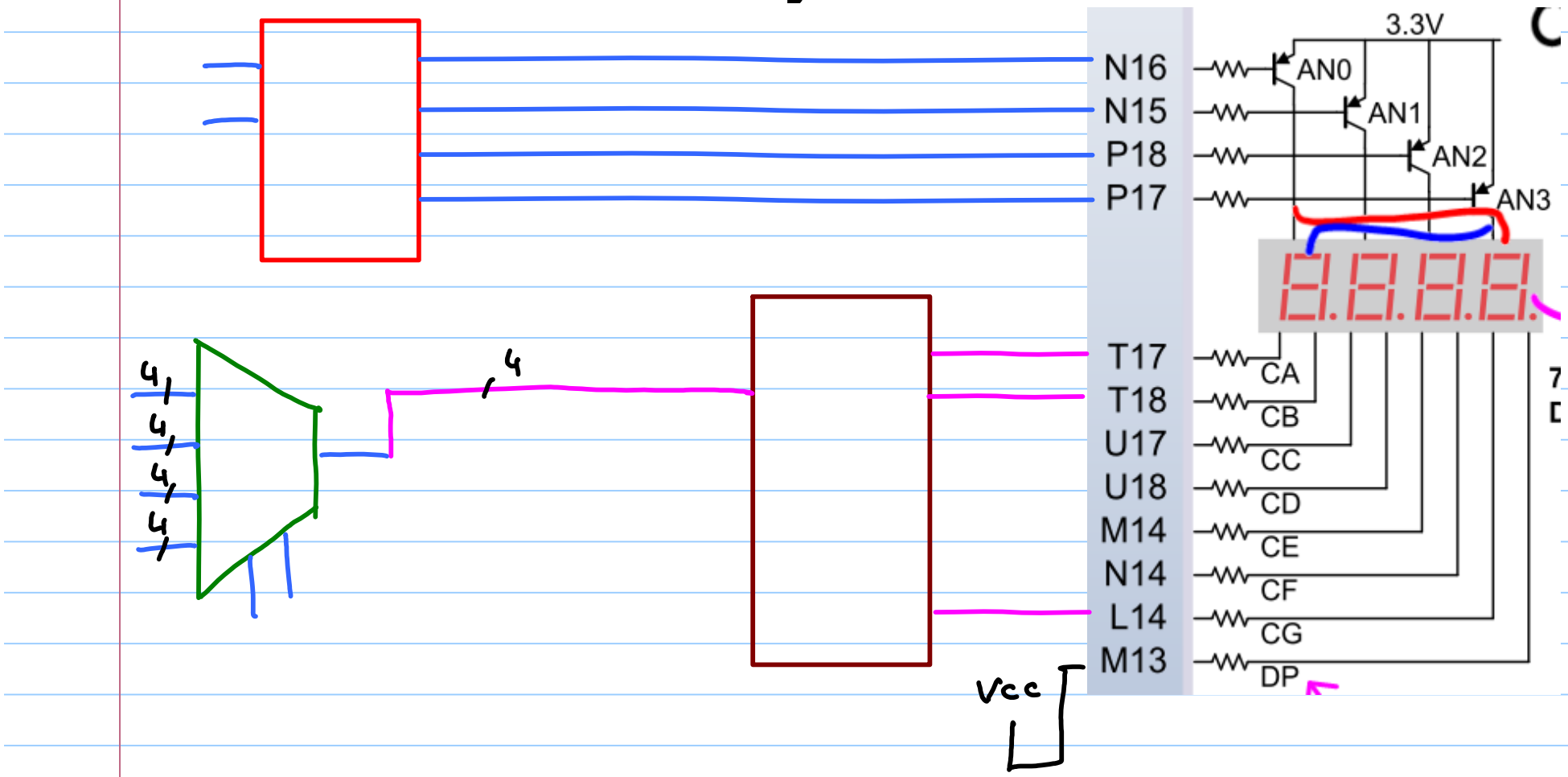


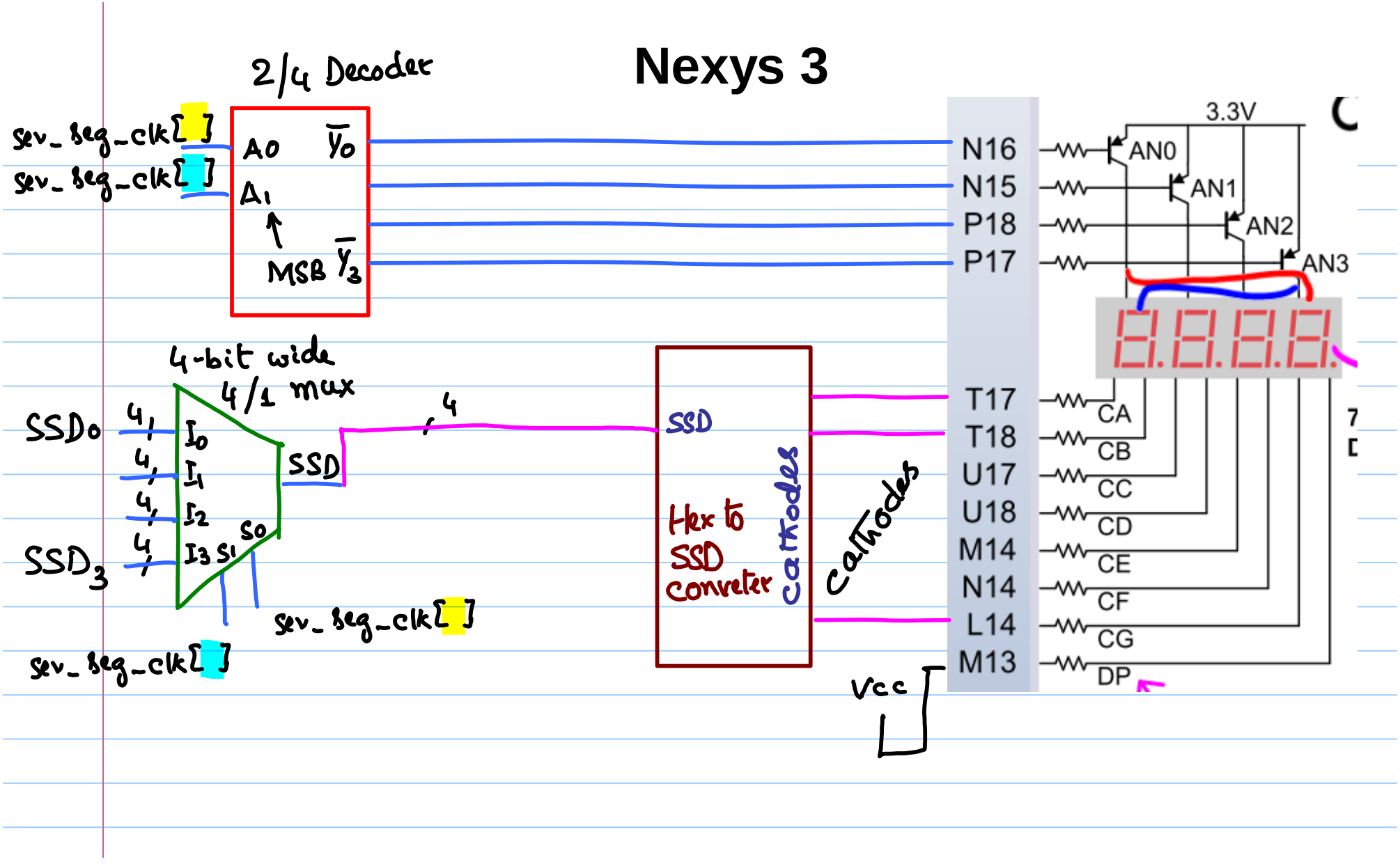


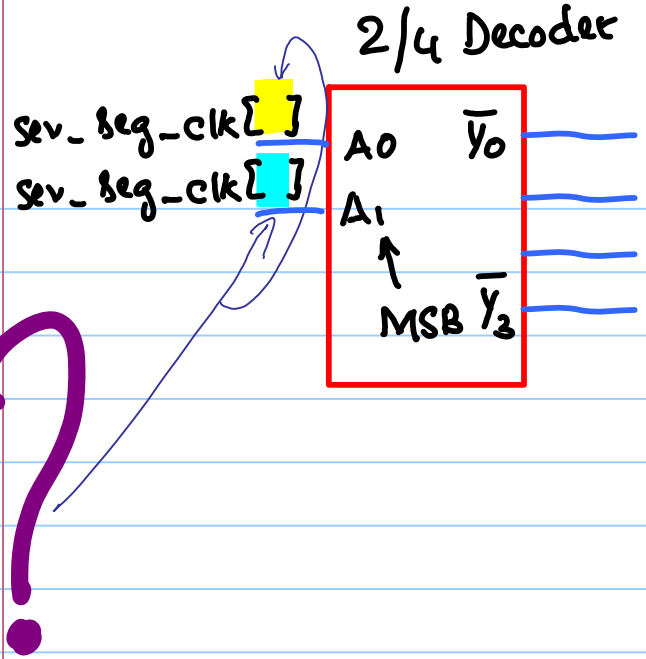

```
232 // Following is Hex-to-SSD conversion.
233 always @ (SSD)
234     begin
235       case (SSD)
236         4'b0000: cathodes = 7'b0000001 ; // 0
237         4'b0001: cathodes = 7'b1001111 ; // 1
238         4'b0010: cathodes = 7'b0010010 ; // 2
239         4'b0011: cathodes = 7'b0000110 ; // 3
240         4'b0100: cathodes = 7'b1001100 ; // 4
241         4'b0101: cathodes = 7'b0100100 ; // 5
242         4'b0110: cathodes = 7'b0100000 ; // 6
243         4'b0111: cathodes = 7'b0001111 ; // 7
244         4'b1000: cathodes = 7'b0000000 ; // 8
245         4'b1001: cathodes = 7'b0000100 ; // 9
246         4'b1010: cathodes = 7'b0001000 ; // A
247         4'b1011: cathodes = 7'b1100000 ; // b
248         4'b1100: cathodes = 7'b0110001 ; // C
249         4'b1101: cathodes = 7'b1000010 ; // d
250         4'b1110: cathodes = 7'b0110000 ; // E
251         4'b1111: cathodes = 7'b0111000 ; // F
252         default: cathodes = 7'bXXXXXXX ; // default
253       endcase
```

// default is actually not needed as we covered all cases

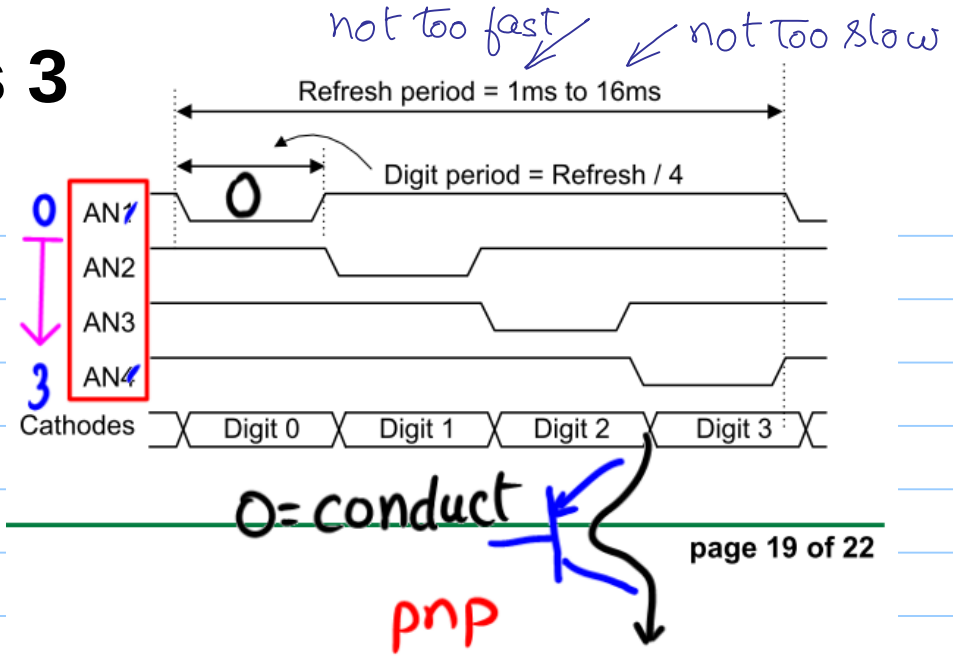
Say a 16-bit counter output needs to be displayed on the 4-digit 7-seg display.




Nexys 3
2/4 Decoder
seg_clk[ ]
seg_clk[ ]
A0
A1
MSB
$\overline{Y}_0$
$\overline{Y}_3$
4-bit wide
4/1 mux
SSD0
SSD3
4
I0
I1
I2
I3
S1
S0
SSD
seg_clk[ ]
seg_clk[ ]
4
SSD
Hex to
SSD
converter
cathodes
cathodes
Vcc
N16
N15
P18
P17
T17
T18
U17
U18
M14
N14
L14
M13
3.3V
AN0
AN1
AN2
AN3
CA
CB
CC
CD
CE
CF
CG
DP



Nexys 3



100MHz

Oscillators/Clocks

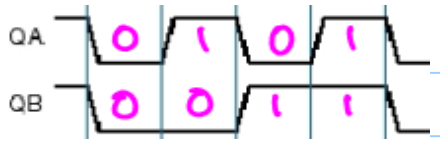
The Nexys3 board includes a single 100MHz CMOS oscillator connected to pin V10 (V10 is the GCLK0 input in bank 2). The input clock can drive any or all of the four clock management tiles in the

Nexys 3

```
File     : test_nexys3_verilog.v

wire          reset;
wire          board_clk;
//wire          sys_clk;
wire [2:0]    slow_bits;
wire [1:0]    sev_seg_clk;
reg  [27:0]   divclk;

    assign reset = BtnC;

//------------
    // Our clock is too fast (100MHz) for SSD scanning
    // create a series of slower "divided" clocks
    // each successive bit is 1/2 frequency
  always @(posedge board_clk, posedge reset)
    begin
        if (reset)
        divclk <= 0;
        else
        divclk <= divclk + 1'b1;
    end
//------------
```

Nexys 3

```
File     : test_nexys3_verilog.v
```



763 Hz

1.3 mS

```
146     assign sev_seg_clk  = divclk[16:15]; // 7 segment display scanning is completed
147                           // every divclk[16] (~763Hz) = (100MHz / 2 **17)
```

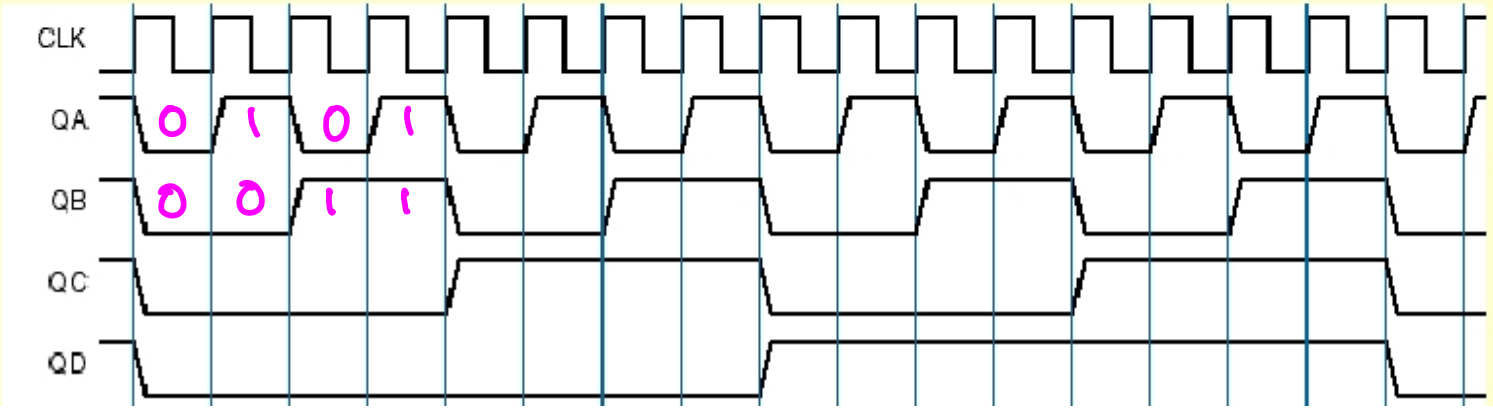
```
assign sev_seg_clk  = divclk[16:15];
```

Nexys 3

Free-running divide-by-16 counter waveforms

board_clk

divclk[0]
divclk[1]
divclk[2]
divclk[3]



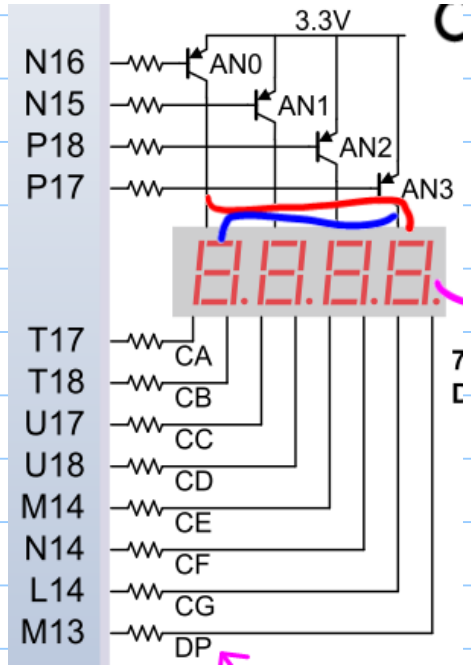
$\frac{1}{2}$ $\frac{1}{4}$ $\frac{1}{8}$ $\frac{1}{16}$

$\frac{1}{2^1}$ $\frac{1}{2^2}$ $\frac{1}{2^3}$ $\frac{1}{2^4}$

$$f_{divclk[n]} = f_{board\_clk} / 2^{n+1}$$

Nexys 3

```
File     : test_nexys3_verilog.v

//assign sys_clk = divclk[18];  // a slow clock for use by students' core design
                                // divclk[18] (~191Hz) = (1000MHz / 2 **19)
```



divclk[25] ——— Dp

```
//------------
  assign Dp = divclk[25]; // The dot point on each SSD flashes
                          // divclk[25] (~1.5Hz) = (100MHz / 2**26)
                         // count the number of flashes for a minute, you should get about 90
```

Walking LEDs

Nexys 3

```
File     : test_nexys3_verilog.v
```

```
assign slow_bits   =  divclk[26:24];

assign {Ld7, Ld6, Ld5, Ld4, Ld3, Ld2, Ld1, Ld0} = leds;
assign leds = button_pressed ? {BtnL, BtnL, BtnU, BtnU, BtnD, BtnD, BtnR, BtnR} :
                    ( divclk[27] ? 8'b11111111 : walking_leds );
// Notice that when divclk[27] is zero, the slow_bits (i.e. divclk[26:24])
// go through a complete sequence of 000-111.

always @ (slow_bits)
  begin
    case (slow_bits)
      3'b000: walking_leds = 8'b00000001 ;
      3'b001: walking_leds = 8'b00000010 ;
      3'b010: walking_leds = 8'b00000100 ;
      3'b011: walking_leds = 8'b00001000 ;
      3'b100: walking_leds = 8'b00010000 ;
      3'b101: walking_leds = 8'b00100000 ;
      3'b110: walking_leds = 8'b01000000 ;
      3'b111: walking_leds = 8'b10000000 ;
      default:walking_leds = 8'bXXXXXXXX ;
    endcase
  end
```