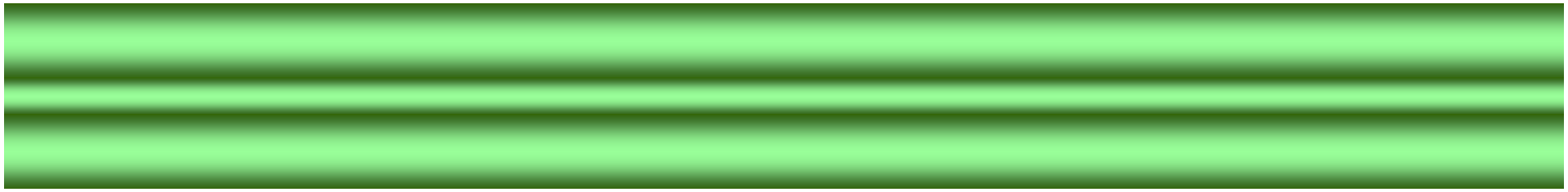
EE414 Embedded Systems

Ch 4. Standard Single Purpose Processors: Peripherals

Part 5/5: Display & Keyboard Interface



Byung Kook Kim
School of Electrical Engineering
Korea Advanced Institute of Science and Technology

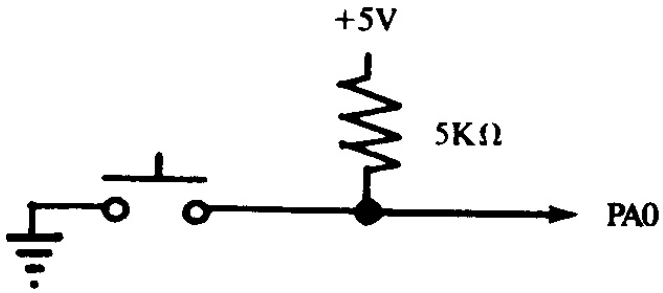
Overview

Part 5/5: Display & Keyboard Interface

- 4.31 Keyboard Interface
- 4.32 Touchscreen
- 4.33 Display Interface
- 4.34 LCD Display
- 4.35 Bit Manipulation

4.31 Keyboard Interface

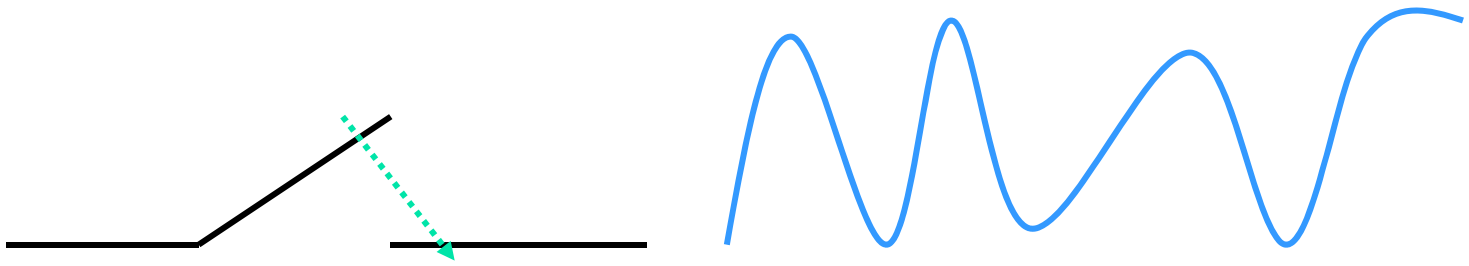
- **A. Switch**
  - Two-position switch
    - Common device: Alone, in groups, and keyboards
    - Interfacing with pull-up resistor
      - Switch open: Out = Vcc
      - Switch closed: Out = GND



  - Three-position switch
    - Common, Normally open, Normally closed.

Switch Debouncing

- Switch bounce
  - Just after the contact position is changed, there is a brief period during which the position of the switch is indefinite.
  - If the contacts are coming together, they may contact and then separate several times.
  - A switch must be debounced to multiple contacts caused by mechanical bouncing:



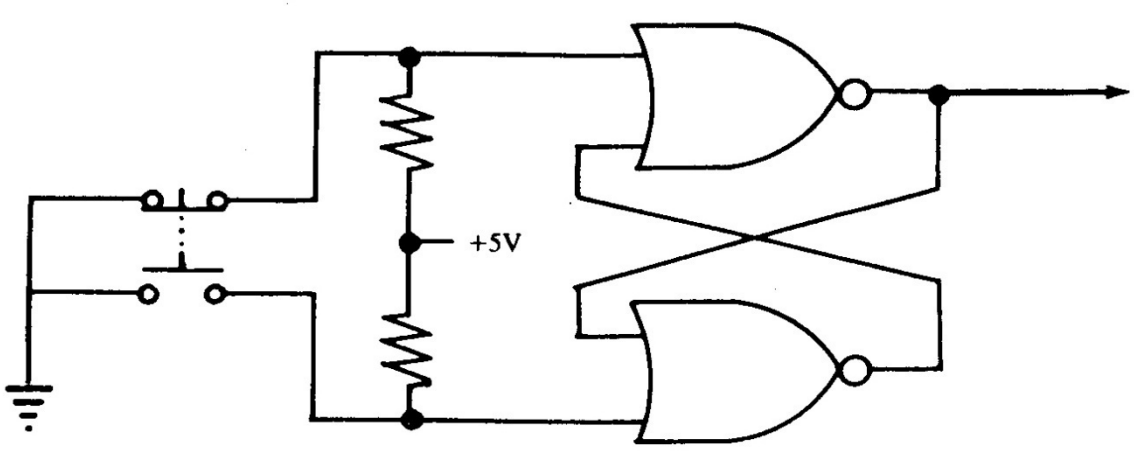
Switch Debouncing (II)

- **Debouncing solution (I)**
  - Delay
    - Test the switch both before and after the delay.
      - If they disagree, the switch is moving.
      - If they agree, the switch is probably in a stable position.
    - This delay is generally from 1 to about 20 ms.
      - Depending on the switch structure.
    - Delay routine:

```
delay:    sub #1, R7        ; Enter with delay count in R7
          bne delay         ; Keep counting until 0
          rts               ; Return from subroutine
```

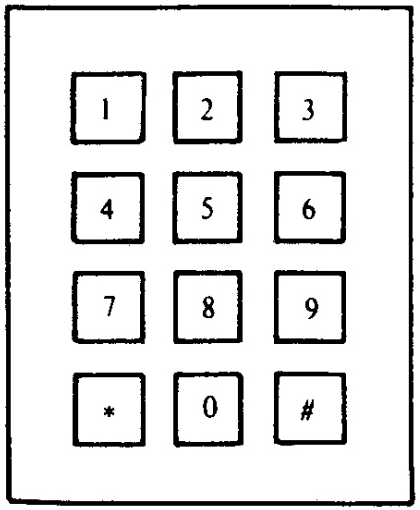
Switch Debouncing (III)

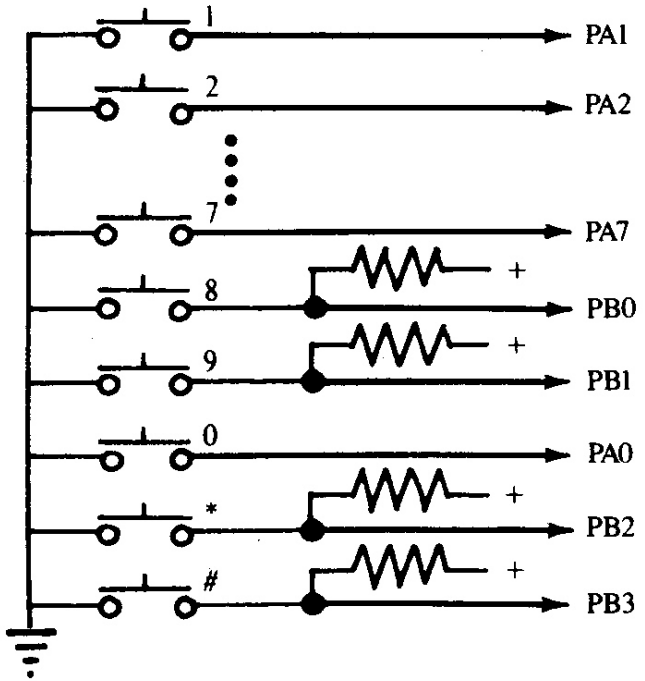
- **Debouncing solution (II)**
  - Hardware debouncing ->
    - Set-reset flip-flop with two NOR gates
    - Three-position switch
    - Assume that the bounces are not so terrible that the switch recontacts the side it already has left.



B. Telephone Keyboard

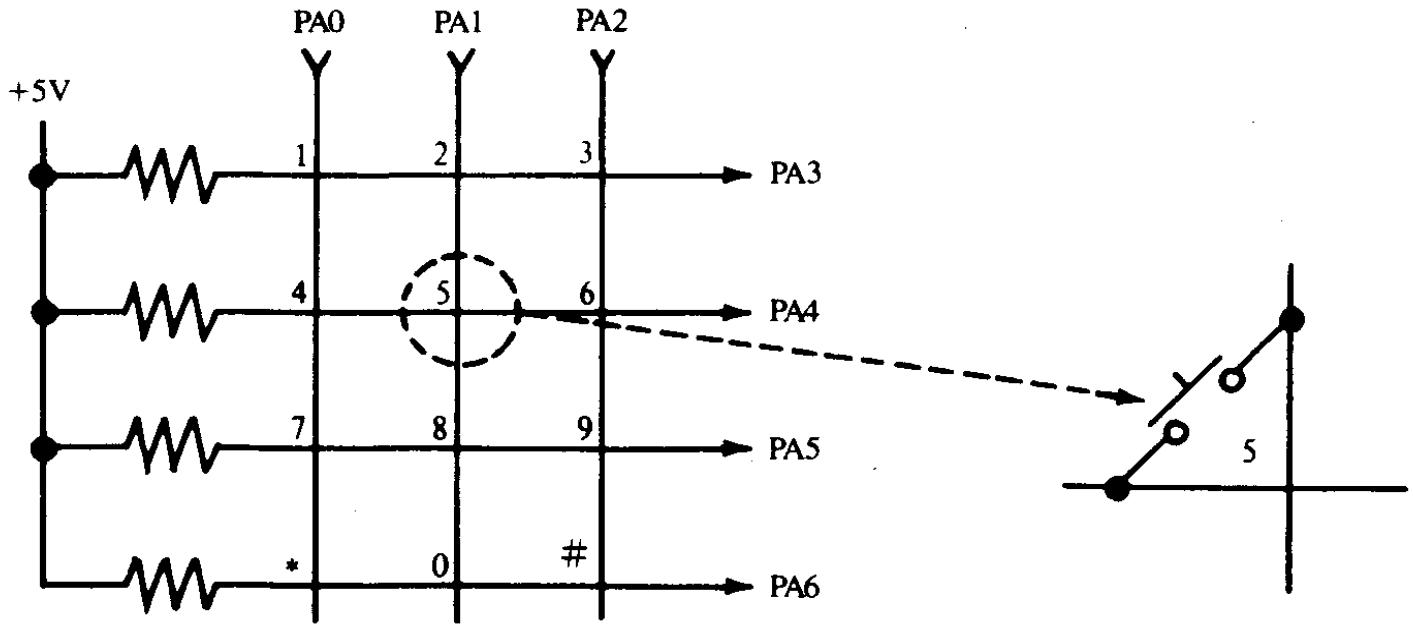
- Standard telephone pushbutton keyboard →
  - 12 buttons: 0 – 9, *, #
  - 12 parallel input bits are required.





C. Telephone Keyboard Grid

- **Reduce wires**: m*n switches using only m+n wires
  - 12 switches with 4+3=7 wires
  - 80 switches with 10+8=18 or 9+9=18 wires!



Program for Keyboard Grid

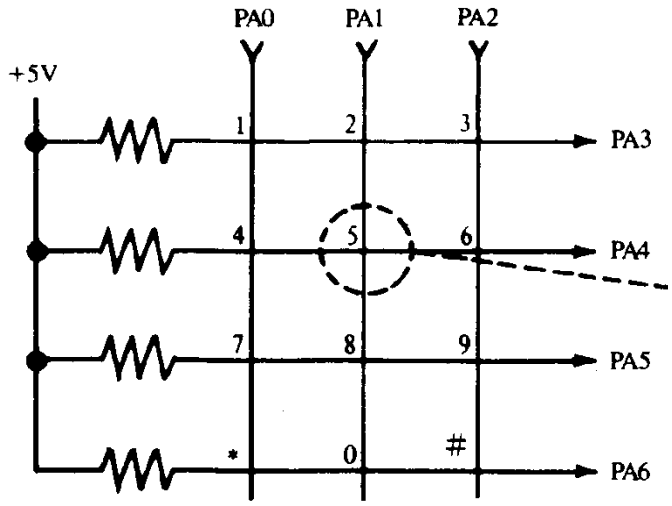
- **Notes**
  - Read one column at a time
    - Force vertical line to be low & read horizontal lines.
- **Algorithm**
  - Make PA2-0 = 011
  - Read horizontal lines. If there is a zero, a key is down in this column.
  - If not, PA2-0 = 101. If there is a zero, a key is down in this column.
  - If not, PA2-0 = 110. If there is a zero, a key is down in this column.
  - *Debouncing*: If a key is found down, save the code, delay, and repeat the reading process.
  - *False Key*: If the newly-read code does not match the old code, save the new code as the old code and go back to Step 1 for another try.
  - If the newly-read code does match the old code, go on to search table for the value of the key.

Program for Keyboard Grid (II)

- Codes for the keyboard grid (Search table)

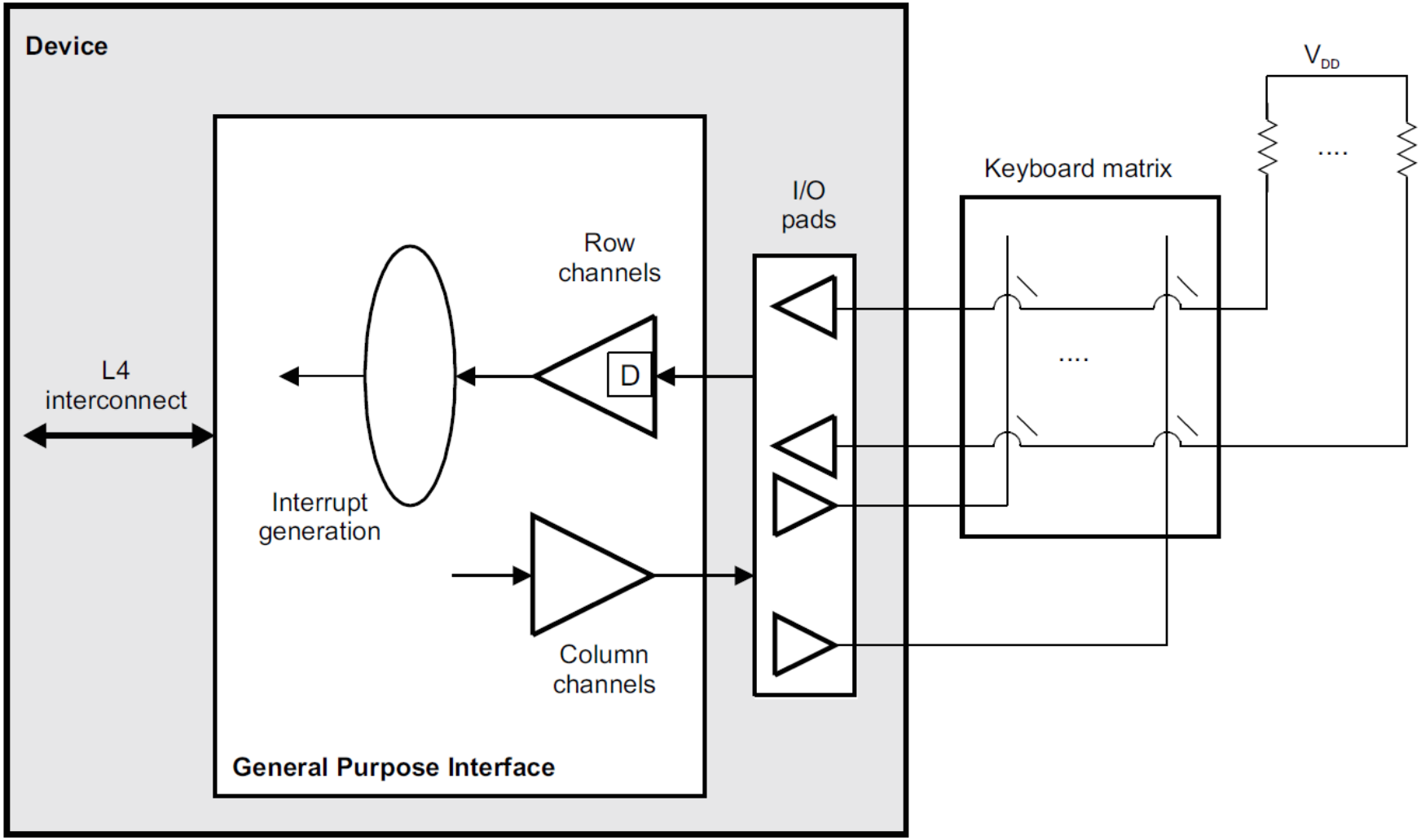
Key	PA6 - 0	Hex
0	0 1 1 1 1 0 1	3D
1	1 1 1 0 1 1 0	76
2	1 1 1 0 1 0 1	75
3	1 1 1 0 0 1 1	73
4	1 1 0 1 1 1 0	6E
5	1 1 0 1 1 0 1	6D
6	1 1 0 1 0 1 1	6B
7	1 0 1 1 1 1 0	5E
8	1 0 1 1 1 0 1	5D
9	1 0 1 1 0 1 1	5B
*	0 1 1 1 1 1 0	3E
#	0 1 1 1 0 1 1	3B

Bidirectional Keyboard Grid Algorithm



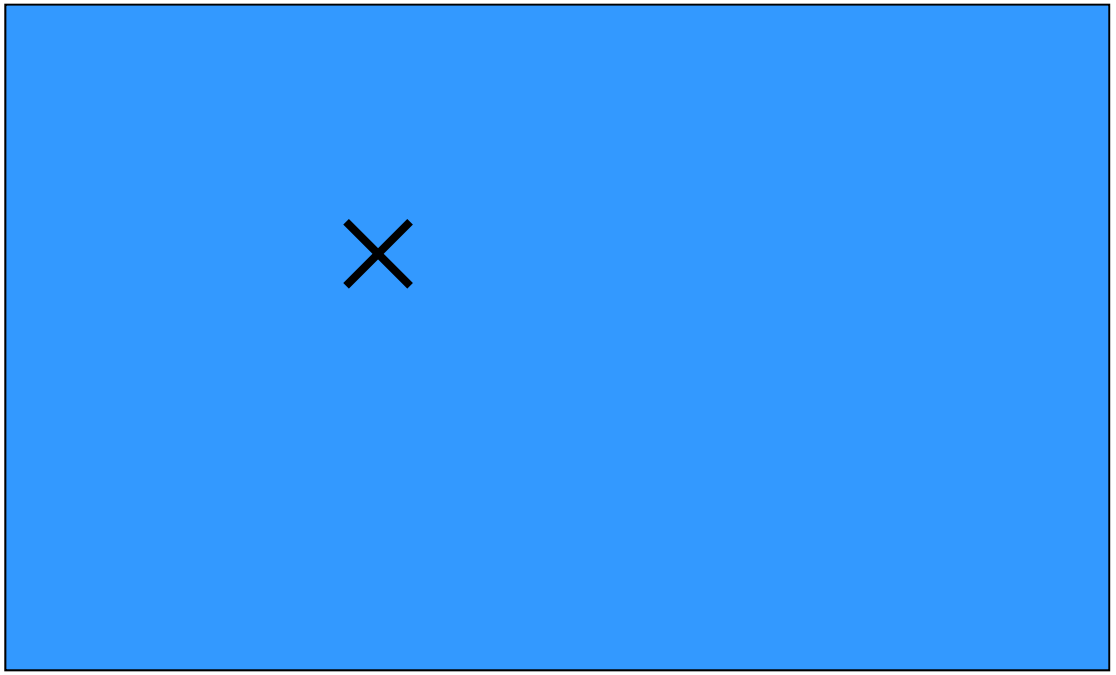
- **Advantages**
  - Only two read operations without column scanning
- **Algorithm**
  - Make PA2-0 output and PA6-3 as input
  - Force PA2-0 low and read to x.
  - Make PA6-3 output and PA2-0 input
  - Force PA6-3 low and read to y.
  - Combine x and y into one byte. The result is the same code as the table in slide 9.

General-Purpose Interface Used as a Keyboard Interface

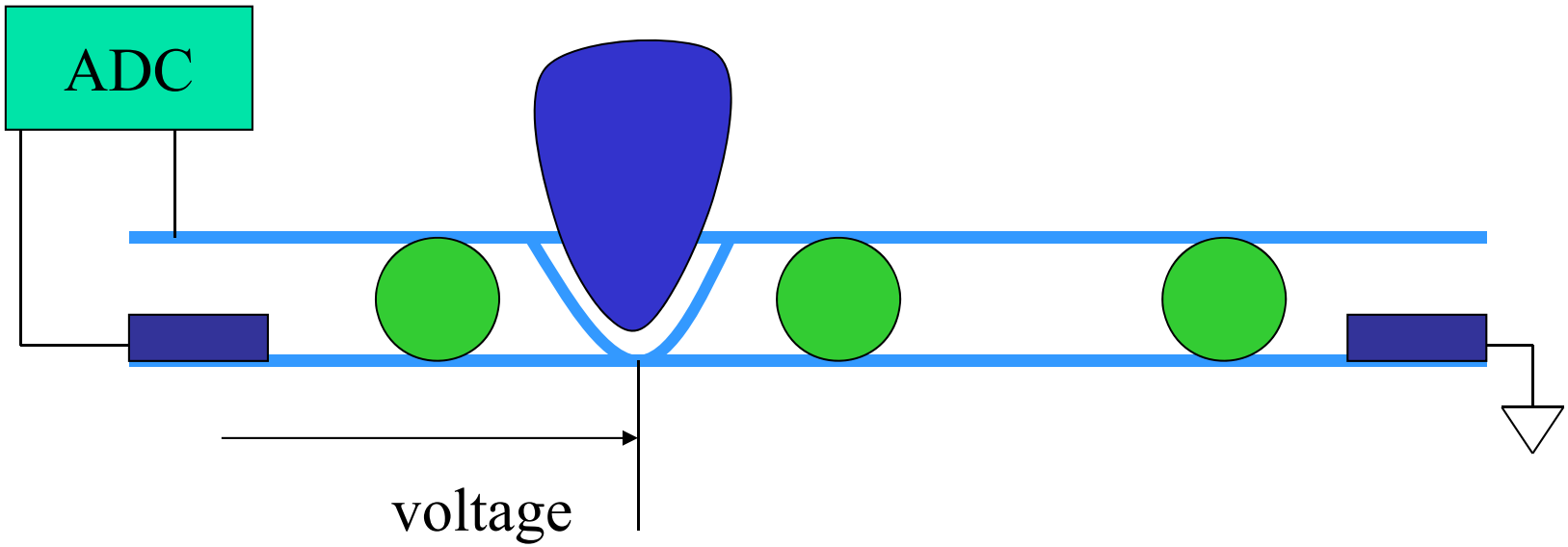


4.32 Touchscreen

- Includes input and output device.
- Input device is a two-dimensional voltmeter:

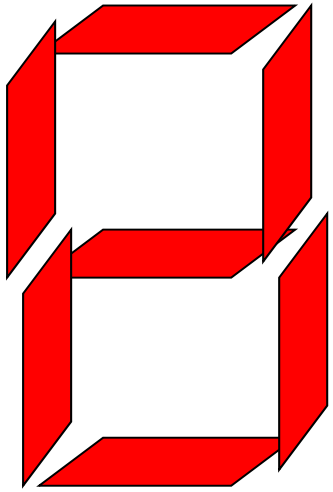


Touchscreen Position Sensing

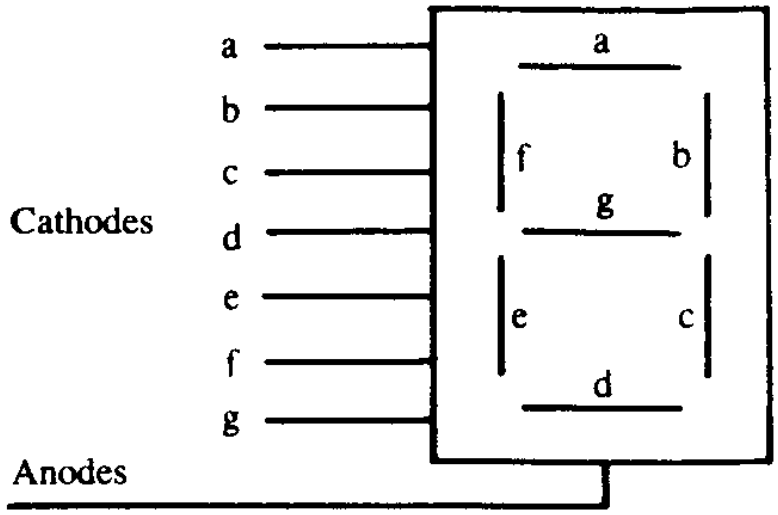


4.33 Display Interface

- **A. Display of digits**
  - Seven-segment display →
  - 16-segment display
  - Array of dots →

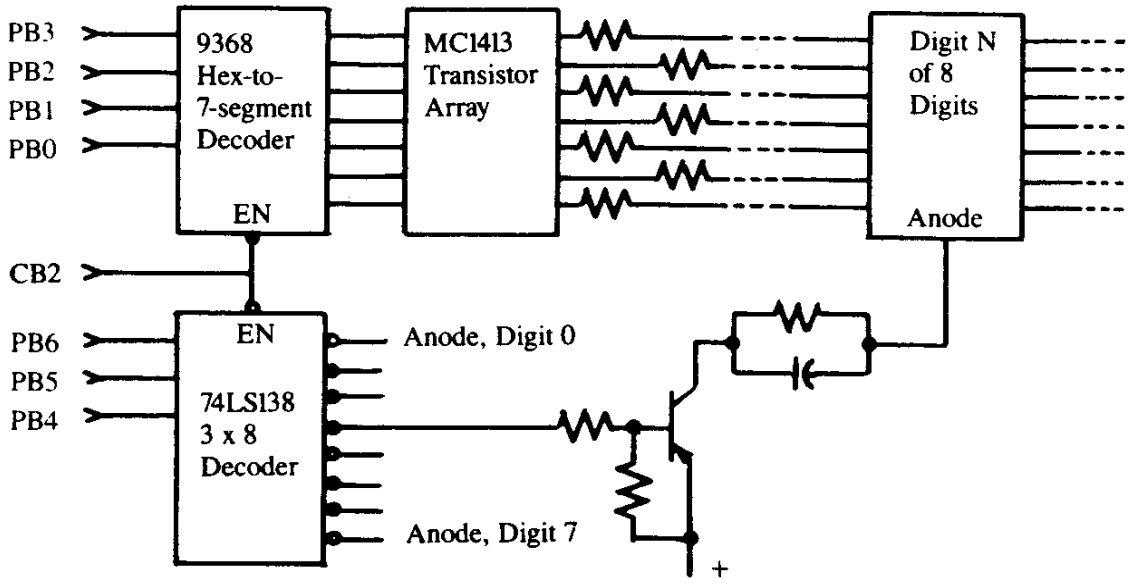


- Types
  - Common anode →
  - Common cathode



Display Interface (II)

- Multiple-digit seven-segment display
  - Ex: 8 digits →
    - 7 x 8 = 56 wires
    - 7 + 8 = 15 wires
    - 4 + 3 = 7 wires



Display Interface (III)

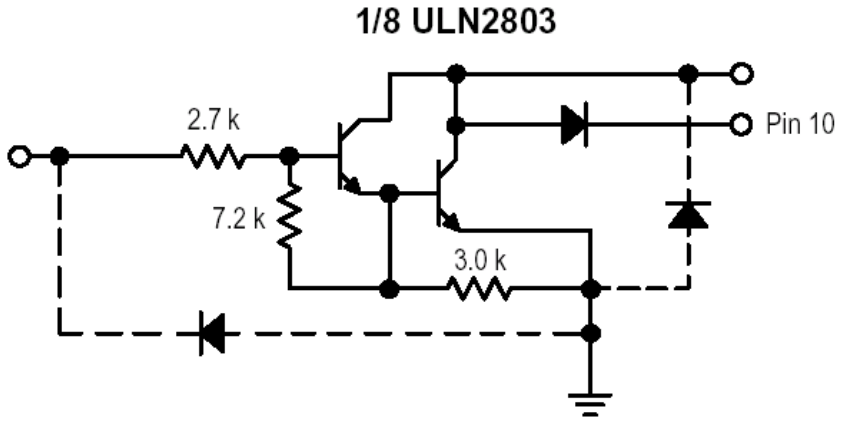
- **Algorithm**

```
Given c[i], i=0, 1, …, 7
For digit d=0, 1, …, 7
    Output digit selection d
    Turn on: Output 7-segment display c[d]
    Delay
    Turn off 7-segment display
End of d
```

- **Ghost elimination**
  - Place turn-on and turn-off side by side after the move
    - (Include red lines as above).

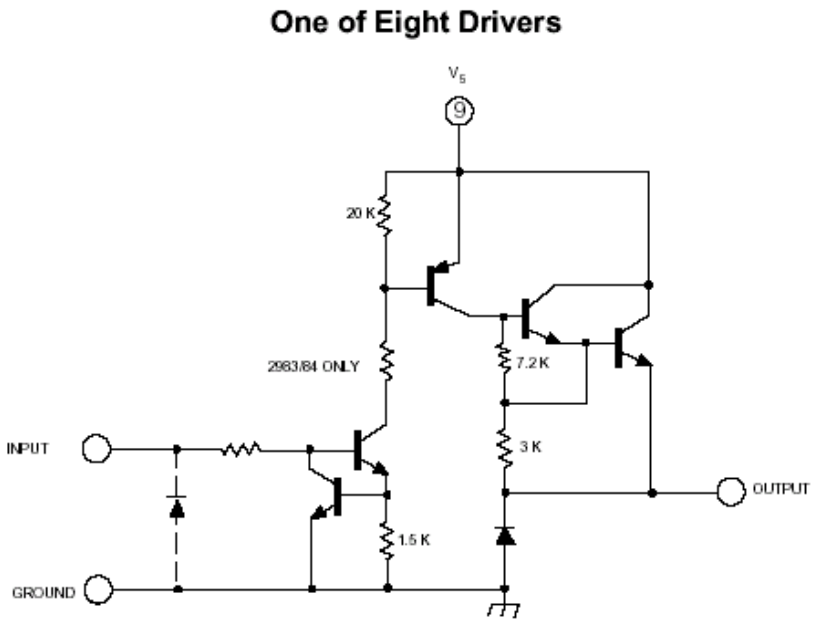
Display Interface (IV)

- **Driver ICs - ULN2803**
  - **8-bit 50V 500mA TTL-input NPN darlington drivers.**
  - The drivers need no power supply; the VDD pin is the common cathode of the eight integrated protection diodes.
  - The eight NPN Darlington connected transistors in this family of arrays are ideally suited for interfacing between low logic level digital circuitry (such as TTL, CMOS or PMOS/NMOS) and the higher current/voltage requirements of lamps, relays, printer hammers or other similar loads for a broad range of computer, industrial, and consumer applications.
  - All devices feature open–collector outputs and free wheeling clamp diodes for transient suppression.



Display Interface (V)

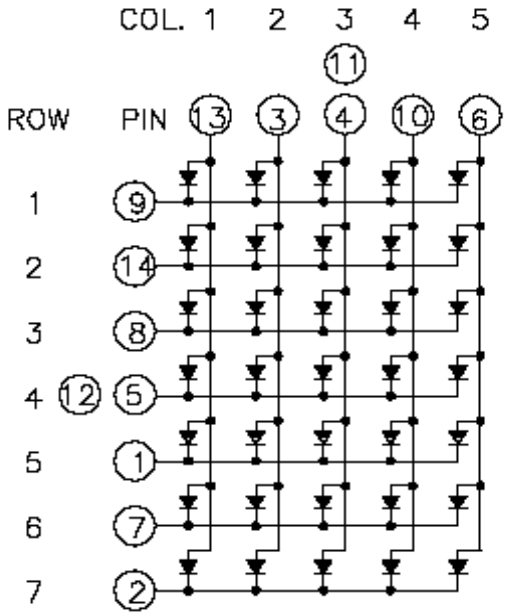
- **Driver ICs – uDN2981**
  - ***8-CHANNEL SOURCE DRIVERS***
  - Recommended for high-side switching applications that benefit from separate logic and load grounds, these devices encompass load supply voltages to 80 V and output currents to -500 mA.
  - These 8-channel source drivers are useful for interfacing between low-level logic and high-current loads.
  - Typical loads include relays, solenoids, lamps, stepper and/or servo motors, print hammers, and LEDs.



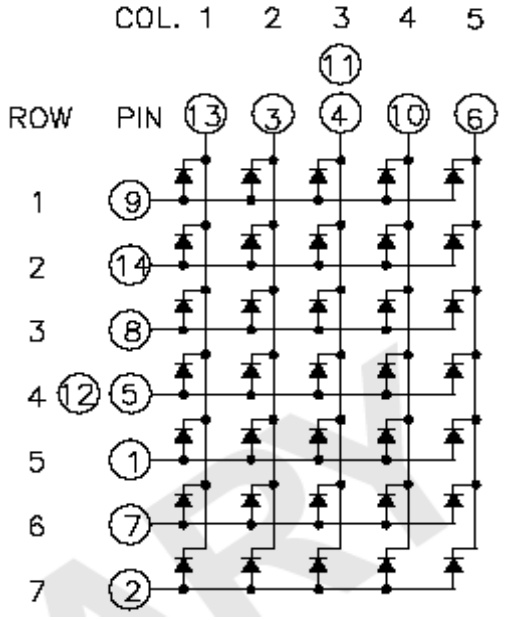
Display Interface (VI)

- **B. Dot matrix display**
  - English 1 x 20 characters
    - 7 x 100 dot matrix display
  - Korean 1 x 20 characters
    - 16 x 320 dot matrix display

A.LTP-2057A



B.LTP-2157A



Display Interface (VII)

- **C. Color display**
  - Red & Green LED: red, green, yellow
  - Color LED: red, green, blue LEDs
  - 1024 x 768 True color
    - 1024 x 768 x 3 bytes for RGB

4.34 LCD Display

- **LCD (liquid crystal display)**
  - Advantages
    - Thinner and lighter
    - Draw much less power than cathode ray tubes (CRTs)
  - LCD History
    - Liquid crystals were first discovered in 1888, by Austrian botanist **Friedrich Reinitzer**.
    - Reinitzer observed that when he melted a curious cholesterol-like substance (**cholesteryl benzoate**), it first became a cloudy liquid and then cleared up as its temperature rose.
    - Upon cooling, the liquid turned blue before finally crystallizing.
    - Eighty years passed before **RCA** made the first experimental LCD in 1968.

LCD Display (II)

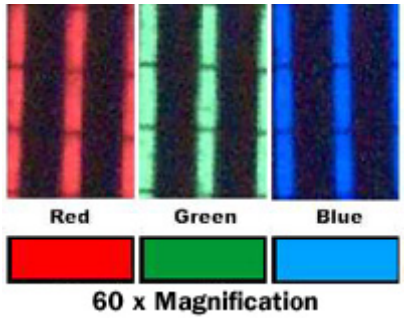
- **Principles on LCD**
  - Liquid crystals are closer to a liquid state than a solid.
    - It takes a fair amount of heat to change a suitable substance from a solid into a liquid crystal, and it only takes a little more heat to turn that same liquid crystal into a real liquid.
    - Very sensitive to **temperature:** used to make thermometers and mood rings.
  - Depending on the temperature and particular nature of a substance, liquid crystals can be in one of several distinct phases.
    - **Nematic phase:** Affected by **electric current**.
    - A particular sort of nematic liquid crystal, called **twisted nematics** (TN), is naturally twisted.
    - Applying an electric current to these liquid crystals will untwist them to varying degrees, depending on the current's voltage.

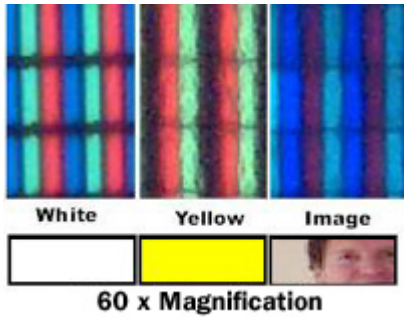
LCD Display (III)

- **LCD Systems**
  - **Common-plane-based** LCDs
    - Good for simple displays that need to show the same information over and over again.
    - Watches and microwave timers
  - **Passive-matrix** LCDs (TN, STN LCDs)
    - Use a simple grid to supply the charge to a particular pixel on the display.
    - **Slow response time** and **imprecise voltage control**.
  - **Active-matrix** LCDs
    - Depend on matrix of **thin film transistors** (TFT): tiny switching transistors and capacitors.
    - To address a particular pixel, the proper row is switched on, and then a charge is sent down the correct column: **gray scale**.
    - Most displays today offer 256 levels of brightness per pixel.

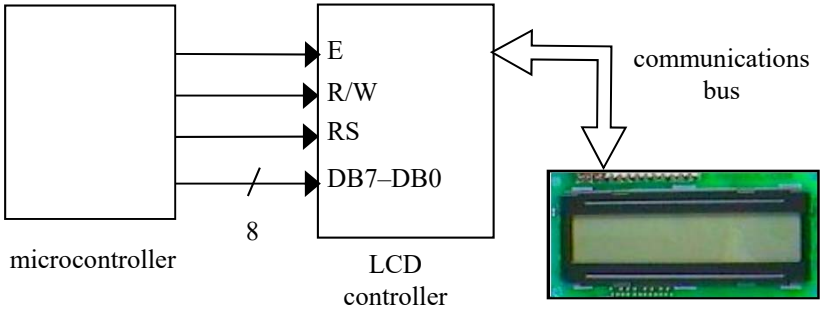
LCD Display (IV)

- **LCD Color**
  - Must have **three subpixels** with red, green and blue color filters to create each color pixel.
  - Through the careful control and variation of the voltage applied, the intensity of each subpixel can range over **256 shades**.
  - Combining the subpixels produces a possible palette of **16.8 million colors** (256 shades of red x 256 shades of green x 256 shades of blue).
  - Enormous number of transistors.
    - Ex: A typical laptop computer supports resolutions up to 1,024x768.
    - 1,024 columns by 768 rows by 3 subpixels, we get 2,359,296 transistors etched onto the glass!
    - If there is a problem with any of these transistors, it creates a "bad pixel" on the display.





LCD Controller




```
void WriteChar(char c){

  RS = 1;                  /* indicate data being sent */
  DATA_BUS = c;            /* send data to LCD */
  EnableLCD(45);           /* toggle the LCD with appropriate delay */
}
```

CODES	
I/D = 1 cursor moves left	DL = 1 8-bit
I/D = 0 cursor moves right	DL = 0 4-bit
S = 1 with display shift	N = 1 2 rows
S/C =1 display shift	N = 0 1 row
S/C = 0 cursor movement	F = 1 5x10 dots
R/L = 1 shift to right	F = 0 5x7 dots
R/L = 0 shift to left	

RS	R/W	DB_7	DB_6	DB_5	DB_4	DB_3	DB_2	DB_1	DB_0	Description
0	0	0	0	0	0	0	0	0	1	Clears all display, return cursor home
0	0	0	0	0	0	0	0	1	*	Returns cursor home
0	0	0	0	0	0	0	1	I/D	S	Sets cursor move direction and/or specifies not to shift display
0	0	0	0	0	0	1	D	C	B	ON/OFF of all display(D), cursor ON/OFF (C), and blink position (B)
0	0	0	0	0	1	S/C	R/L	*	*	Move cursor and shifts display
0	0	0	0	1	DL	N	F	*	*	Sets interface data length, number of display lines, and character font
1	0	WRITE DATA								Writes Data

4.35 Bit Manipulation

- Changing i-th bit
  - var = var EOR (1 << i);
- Clear lower half-byte
  - var = var & 0xf0;
- Combine bit groups
  - var = (x & 0xff00) | (y & 0xff);
- Swap half-bytes
  - ROL.B #4, R0
- Binary to BCD (2 digits)
  - Bcd = ((bin/10)<<4) | (bin%10);
- BCD to binary (2 digits)
  - Bin = (bcd>>4)*10 + (bcd&0x0f);

References

- Keyboard interface, display interface, and bit manipulation
  - William Eccles, “Microprocessor systems – A 16-bit approach”, Addison Wesley, 1985.
- Display interface & LCD display
  - Search Internet