Input/Output Ports and Interfacing

ELEC 330

Digital Systems Engineering

Dr. Ron Hayne

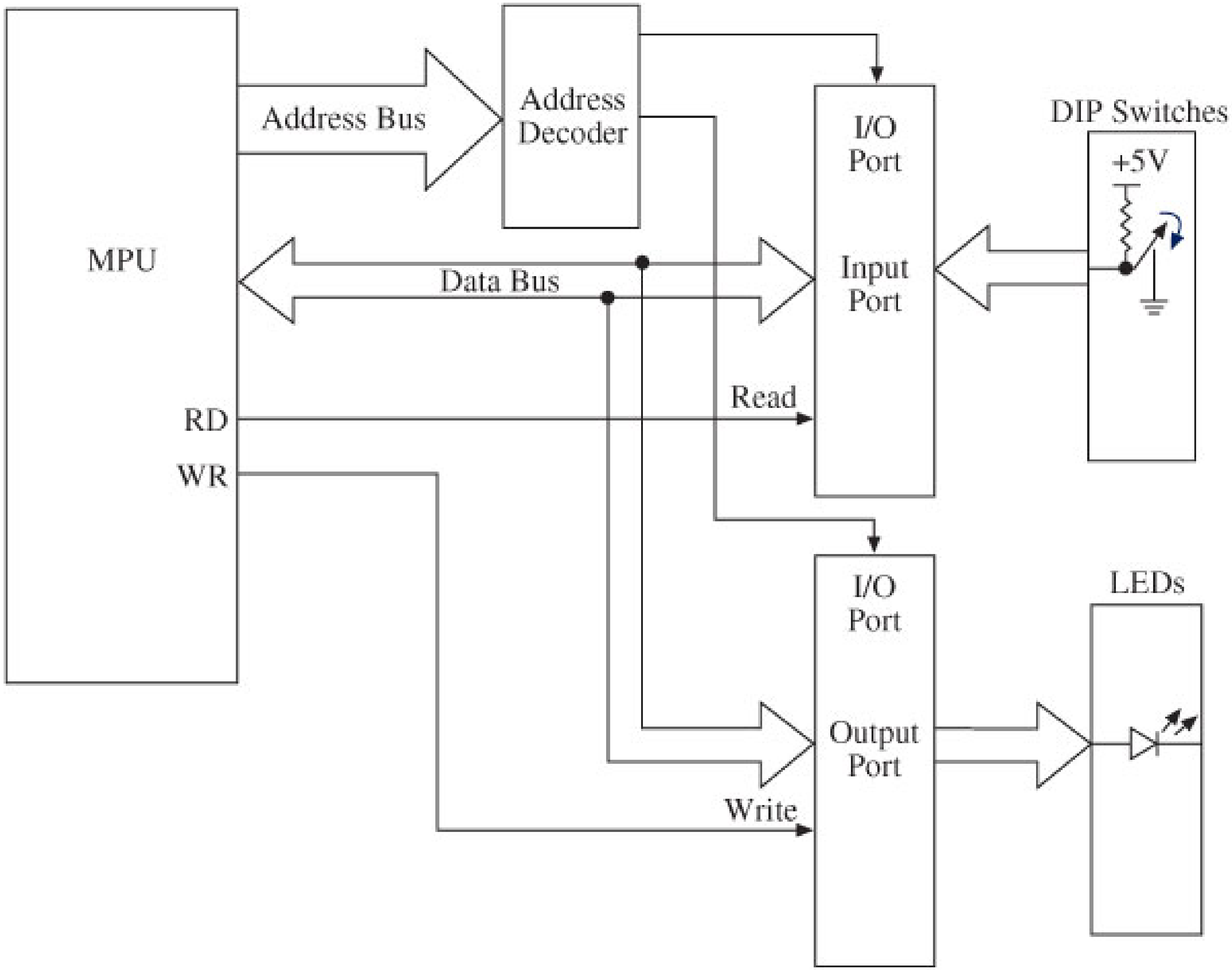
Images Courtesy of Ramesh Gaonkar and Delmar Learning

THE CITADEL
DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

Basic I/O Concepts

- Peripherals such as LEDs and keypads are essential components of microcontroller-based systems
- Input devices
  - Provide digital information to an MPU
  - Examples: switch, keyboard, scanner, and digital camera
- Output devices
  - Receive digital information from an MPU
  - Examples: LED, seven-segment display, LCD, and printer
- Devices are interfaced to an MPU using I/O ports

I/O Interfacing



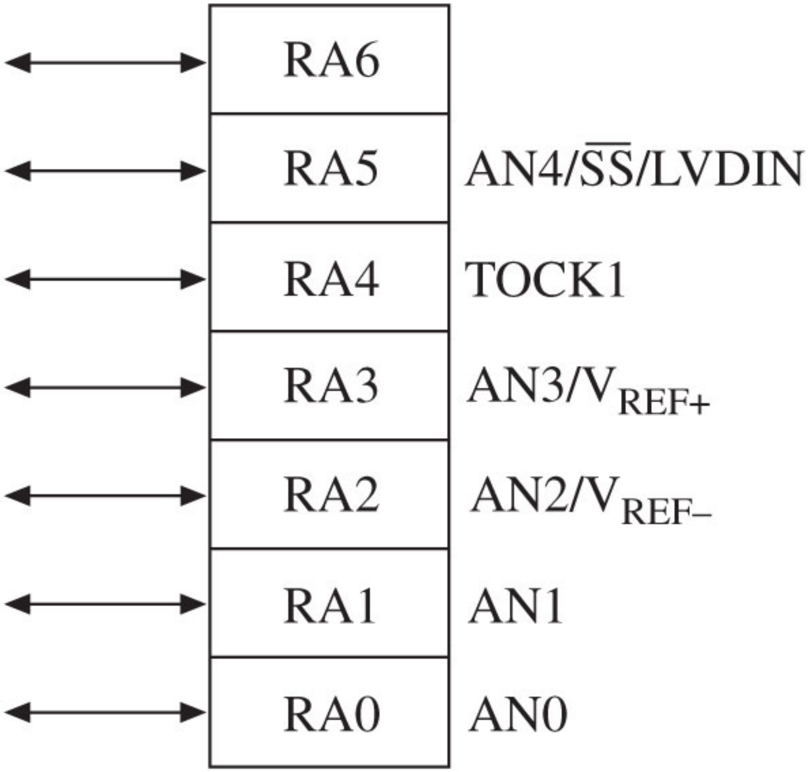
Interfacing and Addressing

- I/O ports
  - Buffers and latches on the MCU chip
    - Assigned binary addresses by decoding the address bus
  - Generally bidirectional
    - Internal data direction registers
  - To read binary data from an input peripheral
    - MPU places the address of an input port on the address bus
    - Enables the input port by asserting the RD signal
    - Reads data using the data bus
  - To write binary data to an output peripheral
    - MPU places the address of an output port on the address bus
    - Places data on data bus
    - Asserts the WR signal to enable the output port

PIC18F452/4520 I/O Ports

- MCU includes five I/O ports
  - PORTA, PORTB, PORTC, PORTD, PORTE
- Ports are multiplexed
  - Can be set up to perform various functions
- Each I/O port is associated with several SFRs
  - PORT
    - Functions as a latch or a buffer
  - TRIS
    - Data direction register
    - Logic 0 sets up the pin as an output
    - Logic 1 sets up the pin as an input
  - LAT
    - Output latch similar to PORT

PIC18F452/4520 I/O Ports



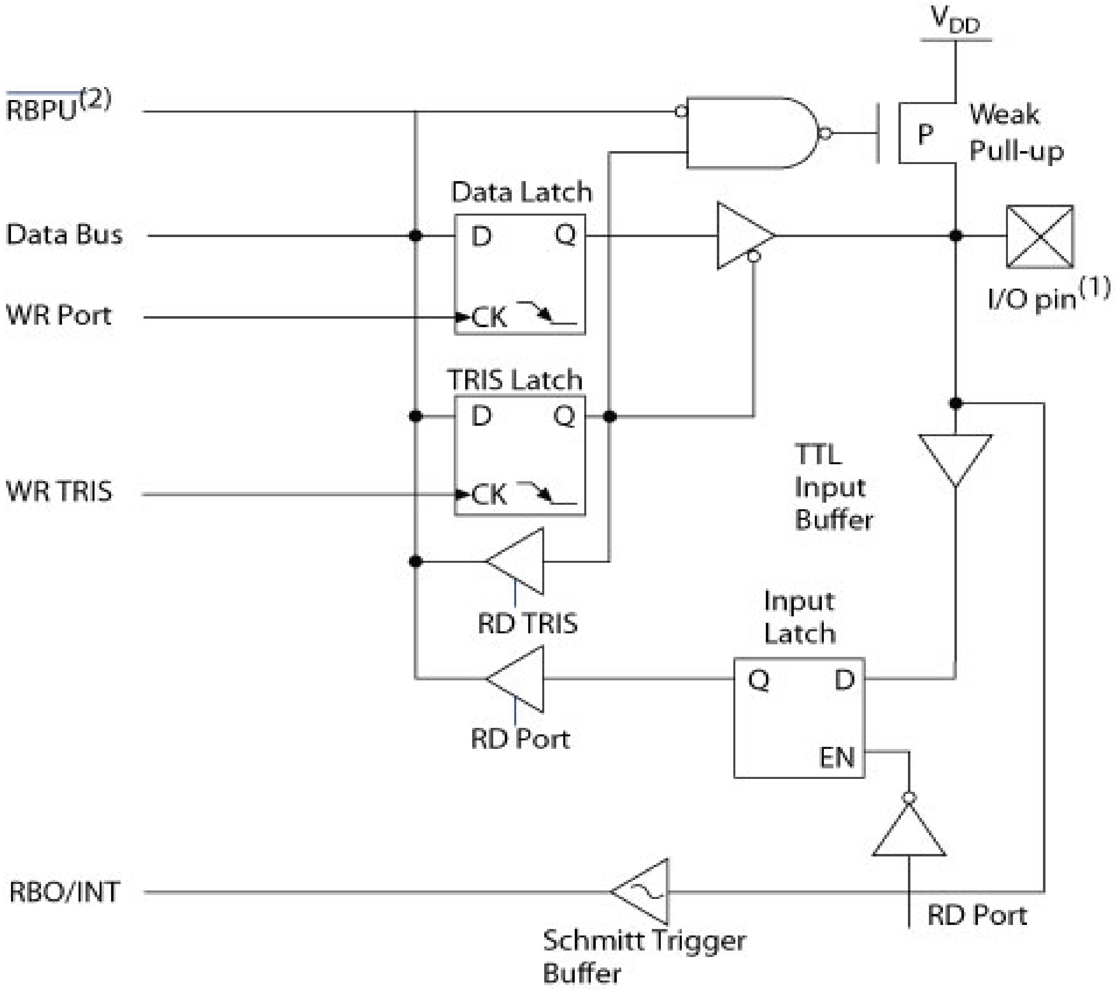
RA6	
RA5	AN4/$\overline{SS}$/LVDIN
RA4	TOCK1
RA3	AN3/V_{REF+}
RA2	AN2/V_{REF-}
RA1	AN1
RA0	AN0

PORTA: Example of Multiple Fns

- Digital I/O: RA6-RA0
- Analog Input: AN0-AN4
- V $_{REF}$+ : A/D Reference Plus V
- V $_{REF}$- : A/D Reference Minus V
- TOCK1: Timer0 Ext. Clock
- SS: SPI Slave Select Input
- LVDIN: Low V Detect Input

PIC18F452/4520 I/O Ports

- PORTB



Note 1: I/O pins have diode protection to VDD and VSS.
2: To enable weak pull-ups, set the appropriate TRIS bit(s) and clear the RBPU bit (Option_REG<7>).

I/O Example

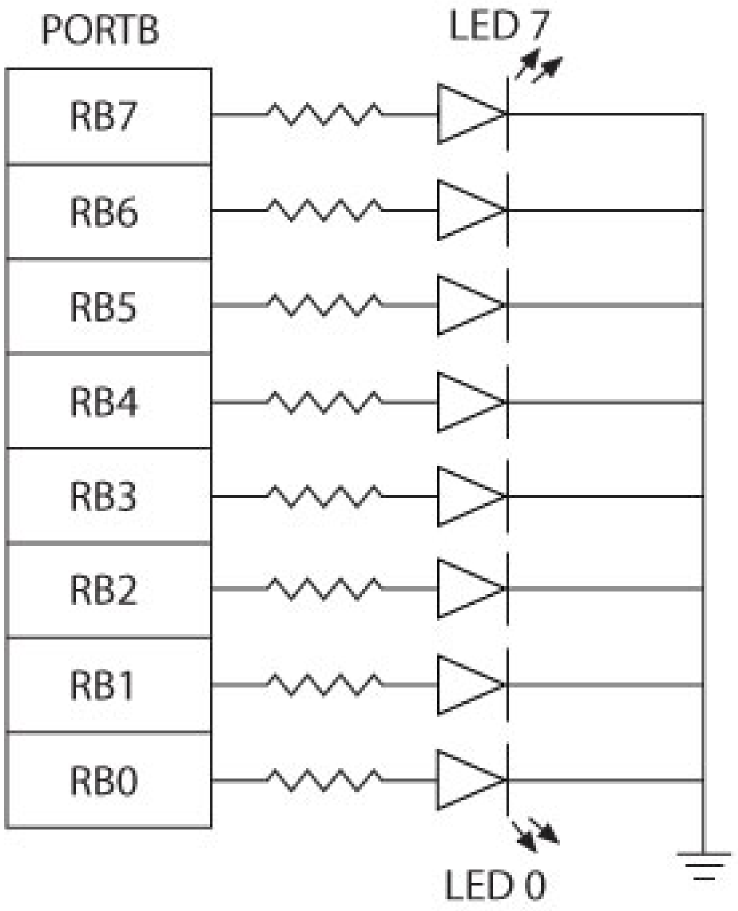
- Write instructions to set up pins RB7-RB4 of PORTB as inputs and pins RB3-RB0 as outputs

Opcode	Operands	Comments
MOVLW	0xF0	;Load B’11110000’ into WREG
MOVWF	TRISB	;Set PORTB TRIS Reg

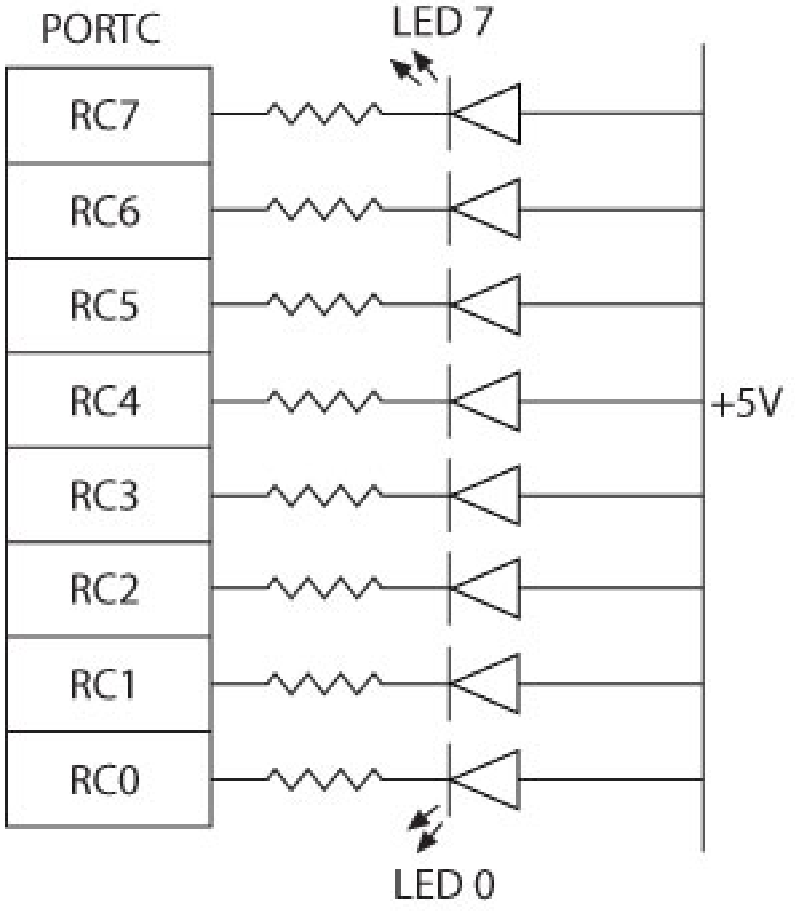
Interfacing Output Peripherals

- Commonly used output peripherals in embedded systems
  - LEDs
  - Seven-Segment Displays
  - LCDs
- Two ways of connecting LEDs to I/O ports
  - Common Cathode
    - LED cathodes are grounded
    - Logic 1 from the I/O port turns on the LEDs
    - Current is supplied by the I/O port called current sourcing
  - Common Anode
    - LED anodes are connected to the power supply
    - Logic 0 from the I/O port turns on the LEDs
    - Current is received by the chip called current sinking

Interfacing Output Peripherals



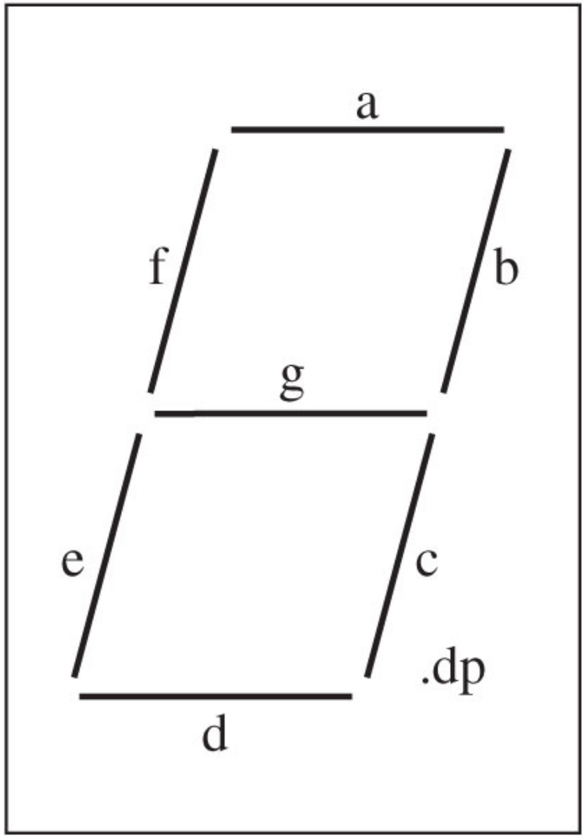
Common Cathode



Common Anode

Seven-Segment Display

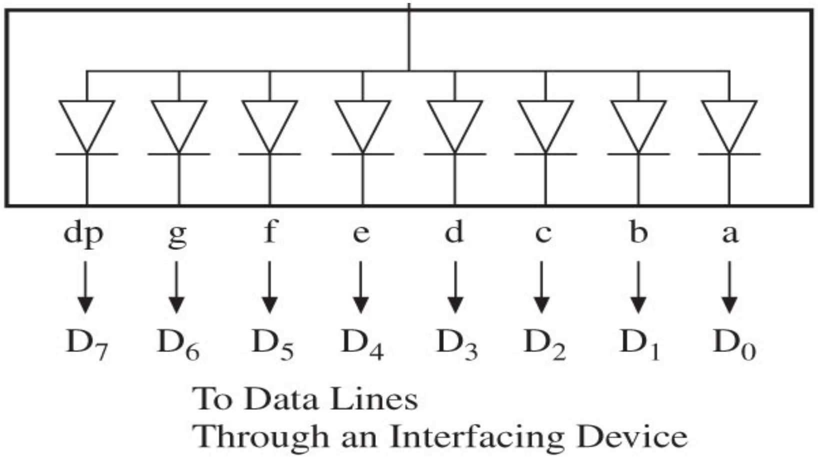
- Seven-segment Displays
  - Used to display BCD digits
    - 0 thru 9
  - A group of 7 LEDs physically mounted in the shape of the number eight
    - Plus a decimal point
  - Each LED is called a segment
    - ‘a’ through ‘g’
  - Two types
    - Common anode
    - Common cathode



Seven-Segment Display

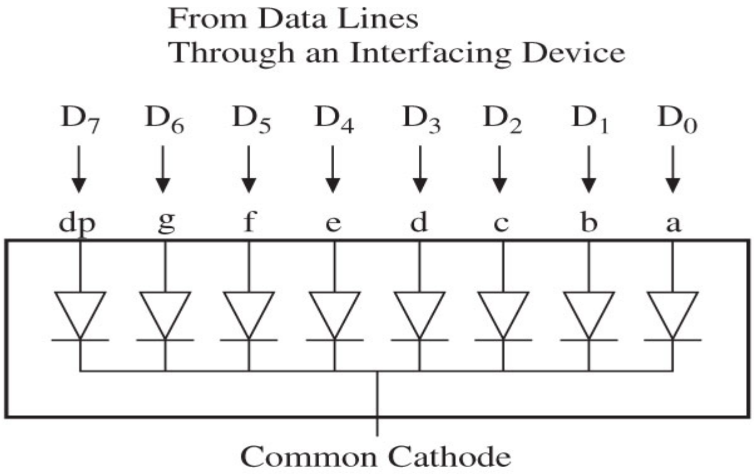
- Common Anode
  - All anodes are connected together to a power supply
  - Cathodes are connected to data lines
- Logic 0 turns on a segment
- Example: To display the digit 1
  - All segments except b and c should be off
  - 11111001 = $F9_H$

Common Anode



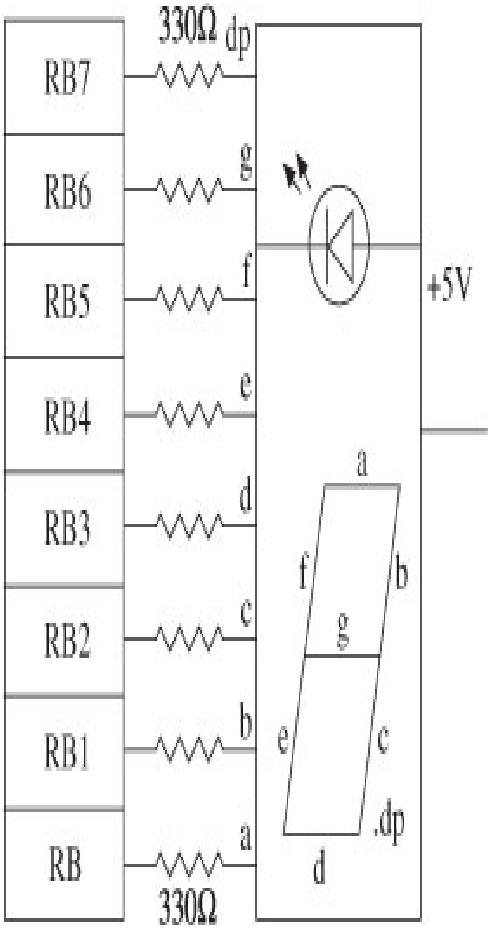
Seven-Segment Display

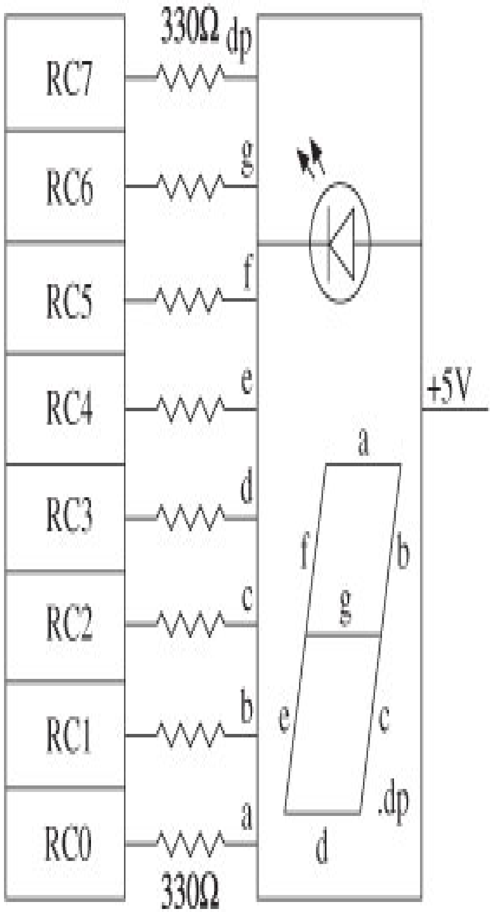
- Common Cathode
  - All cathodes are connected together to ground
  - Anodes are connected to data lines
- Logic 1 turns on a segment
- Example: To display digit 1
  - All segments except b and c should be off
  - $00000110 = 06_H$



Example 9.4

- Interfacing Seven-Segment Display to PORTB
  - Common Anode
  - Table Look-Up





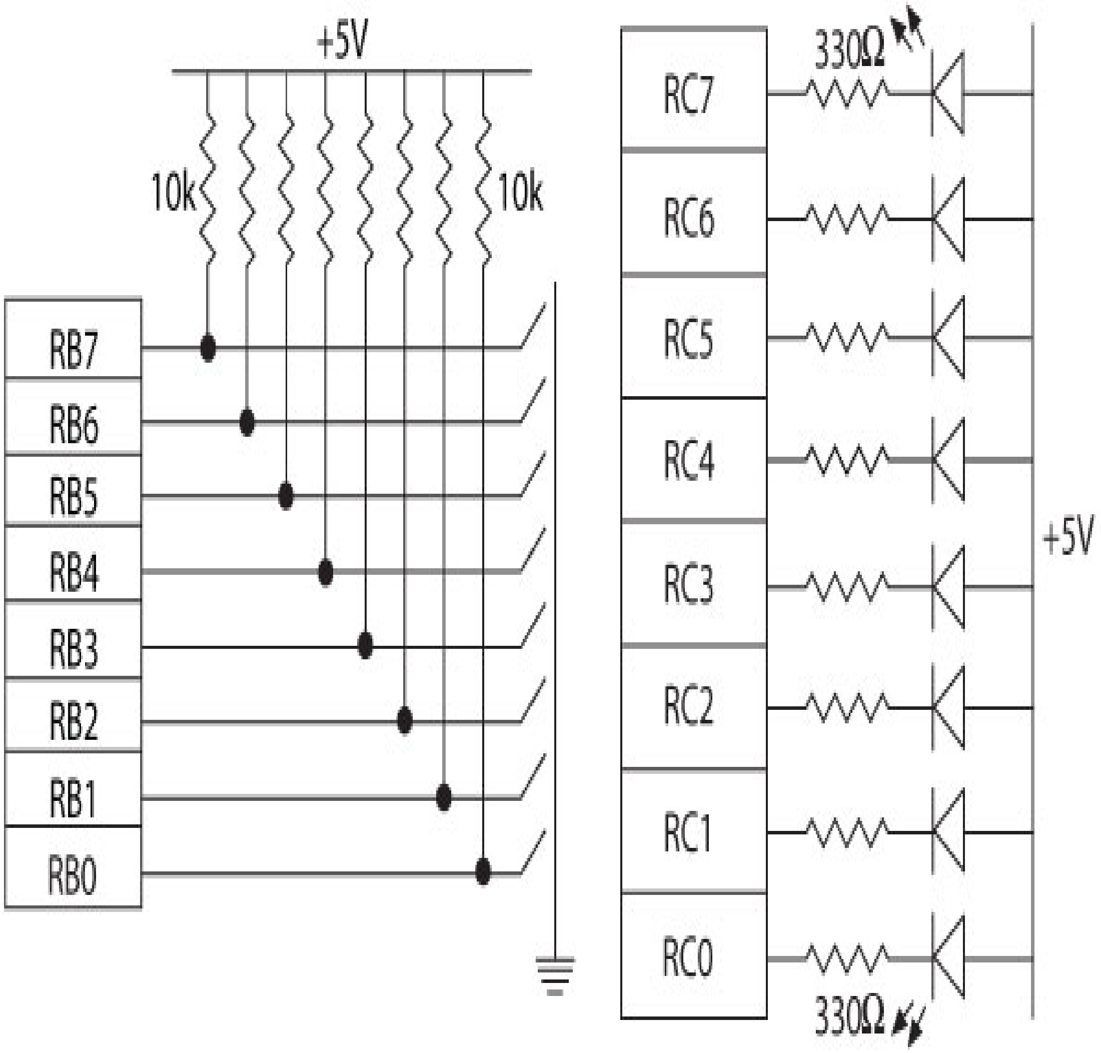
Illustrative Program

- Problem Statement
  - Interface two common-anode seven-segment displays to PORTD and PORTC of the PIC18F
  - Write instructions to implement an up-counter, counting from 00 to 59
  - Display the count on the two seven-segment displays

Interfacing Input Peripherals

- Commonly used input peripherals
  - DIP switches, push-button keys, keyboards, and A/D converters
- DIP switch
  - One side of the switch is tied high
    - To a power supply through a resistor called a pull-up resistor
  - The other side is grounded
  - The logic level changes when the position is switched
- Push-button key
  - Same as the DIP switch except that contact is momentary

Interfacing Dip Switches



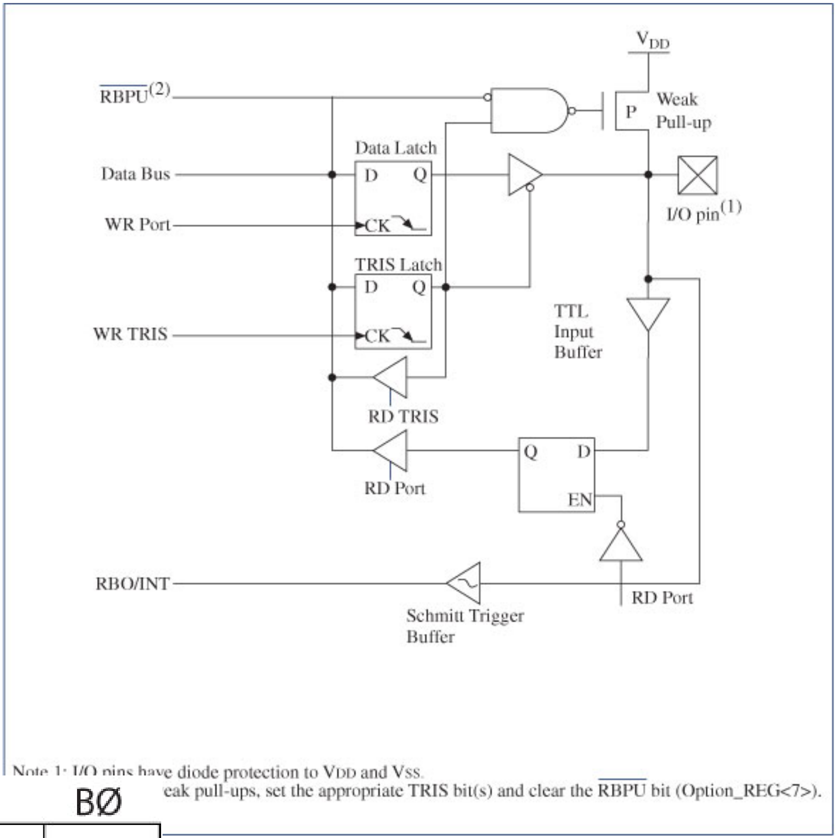
Reading from an I/O Port

- Read input switches on PORTB (RB7-RB4)
  - RB0 set HI (1)
  - Switches Open = LOW (0)
  - Switches Closed = HIGH (1)
- Display on PORTC

Opcode	Operands	Comments
MOVLW	0xF0	;Load B’11110000’ into WREG
MOVWF	TRISB	;Set PORTB TRIS Reg
CLRF	TRISC	;Set PORTC as Output
BSF	PORTB,0	;Set RB0 High
MOVF	PORTB,W	;Read PORTB
MOVWF	PORTC	;Display on PORTC

Internal Pull-Up Resistor

- Turning off the internal FET provides a pull-up resistor
- Bit7 (RBPU) in the INTCON2 register enables or disables the pull-up resistor
    - Instruction to Enable Pull Up Resistors:
      BCF INTCON2,7



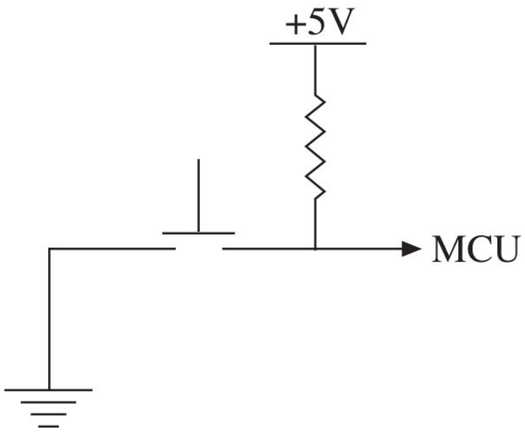
Note 1: I/O pins have diode protection to VDD and VSS.
eak pull-ups, set the appropriate TRIS bit(s) and clear the $\overline{RBPU}$ bit (Option_REG<7>).

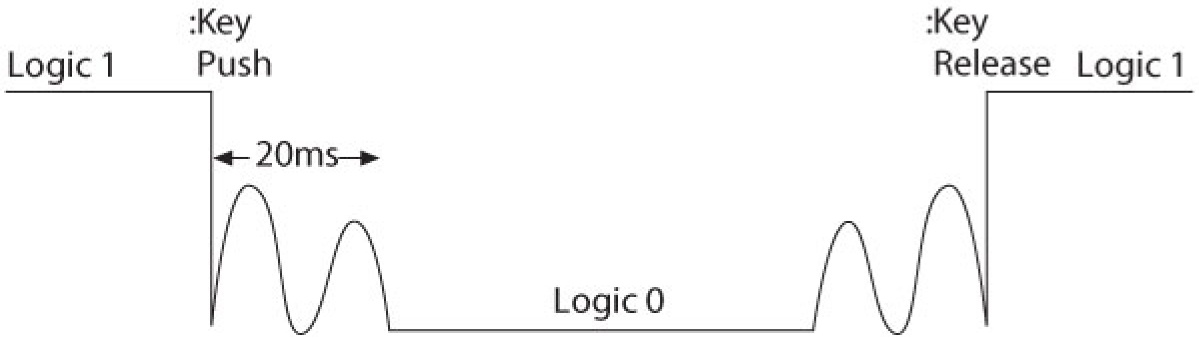
B7	B6	B5	B4	B4	B3	B2	B1	BØ
$\overline{RBPU}$								

$\overline{RBPU}$ = PORTB pull-up resistor enable bit
0 = Pull-up resistors are enabled
1 = Pull-up resistors are disabled

Interfacing Push-Button Keys

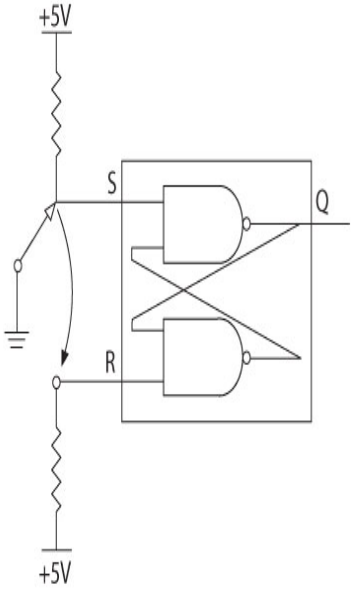
- When a key is pressed (or released), mechanical metal contact bounces momentarily and can be read as multiple inputs
- Key debounce
  - Eliminating reading of one contact as multiple inputs
  - Hardware or Software

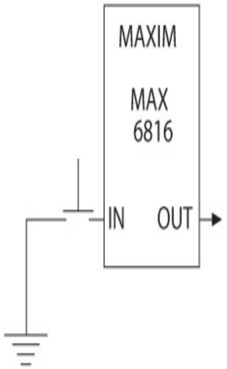




Key Debounce Techniques

- Hardware technique
  - Two NAND gates
    - S-R latch
  - The output of the latch is a pulse without a bounce





- Software technique
  - Wait for 10 to 20 ms after detection of a switch closure
  - If the reading is still the same it is accepted

Interfacing a Matrix Keypad

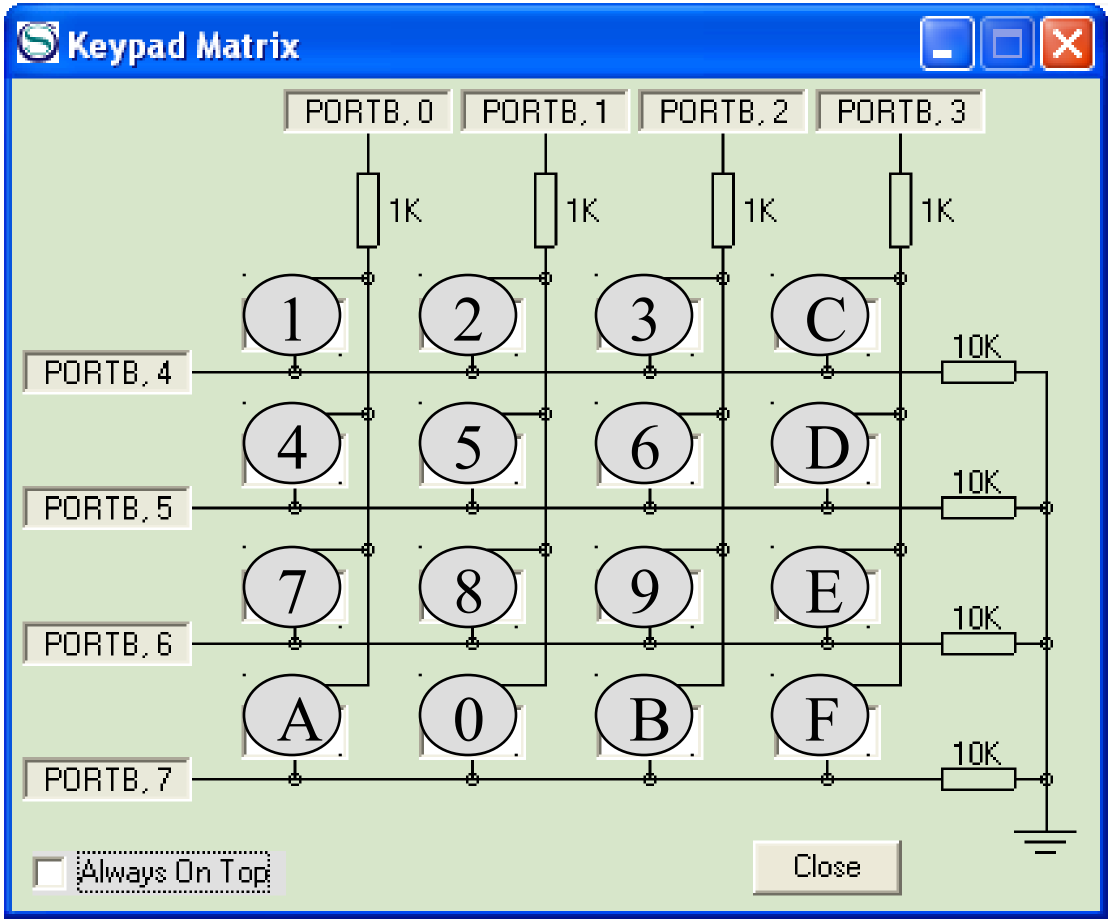
- Problem statement
  - Interface a 4 x 4 Hex keypad to PORTB
  - Write a program to recognize a key pressed and encode the key in its binary value
  - Display binary code on PORTC

Interfacing a Matrix Keypad

- Hardware (PIC18 Simulator)
  - 4 x 4 matrix keypad organized in the row and column format
  - Four columns are connected to the lower half of PORTB (RB0-RB3)
  - Four rows are connected to upper half of PORTB (RB4-RB7)
  - When a key is pressed, it makes a contact with the corresponding row and column

Interfacing a Matrix Keypad

- PIC18 Simulator Keypad Matrix



Interfacing a Matrix Keypad

- Software
  - To recognize and encode the key pressed
    - Set all the columns High by sending ones
    - Check for any key pressed (non-zero)
    - Set one column High at a time
      - Check all the rows in that column
    - Once a key is identified
      - Encode based on its position in the column

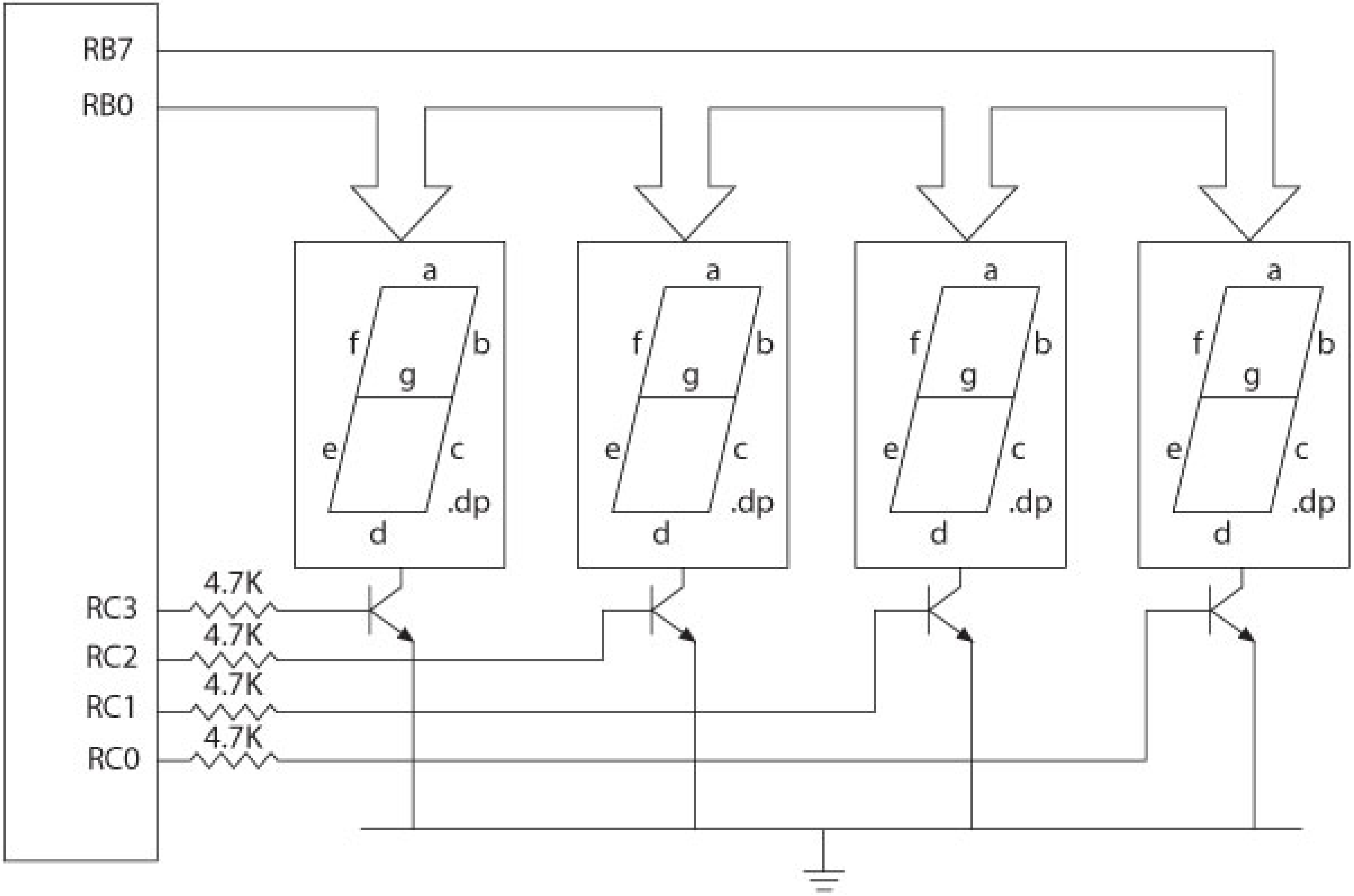
Time Multiplex Scanning Technique

- Problem statement
  - Interface four common cathode seven-segment displays to PORTB and PORTC using the time multiplex scanning technique.
  - Write instructions to display a four-digit number stored in data registers.

Time Multiplex Scanning

- Hardware
  - Eight data lines of PORTB are connected to the anodes of each display
  - Each cathode is connected to PORTC (RC3-RC0) through a transistor
  - Transistors (and LEDs) can be turned on by sending logic 1
  - Each display is turned on and off in a sequence to display a digit

Time Multiplex Scanning



Time Multiplex Scanning

- Software
  - Codes of the numbers to be displayed are stored in data registers in sequence
  - The program gets the codes from the data registers by using the pointer (FSR0) and sends them out to the LED segments through PORTB
  - One display at a time is turned on by sending logic 1 to the corresponding transistor connected to PORTC
  - After an appropriate delay, the first display is turned off and the next display is turned on
  - Turning displays on/off is repeated in sequence

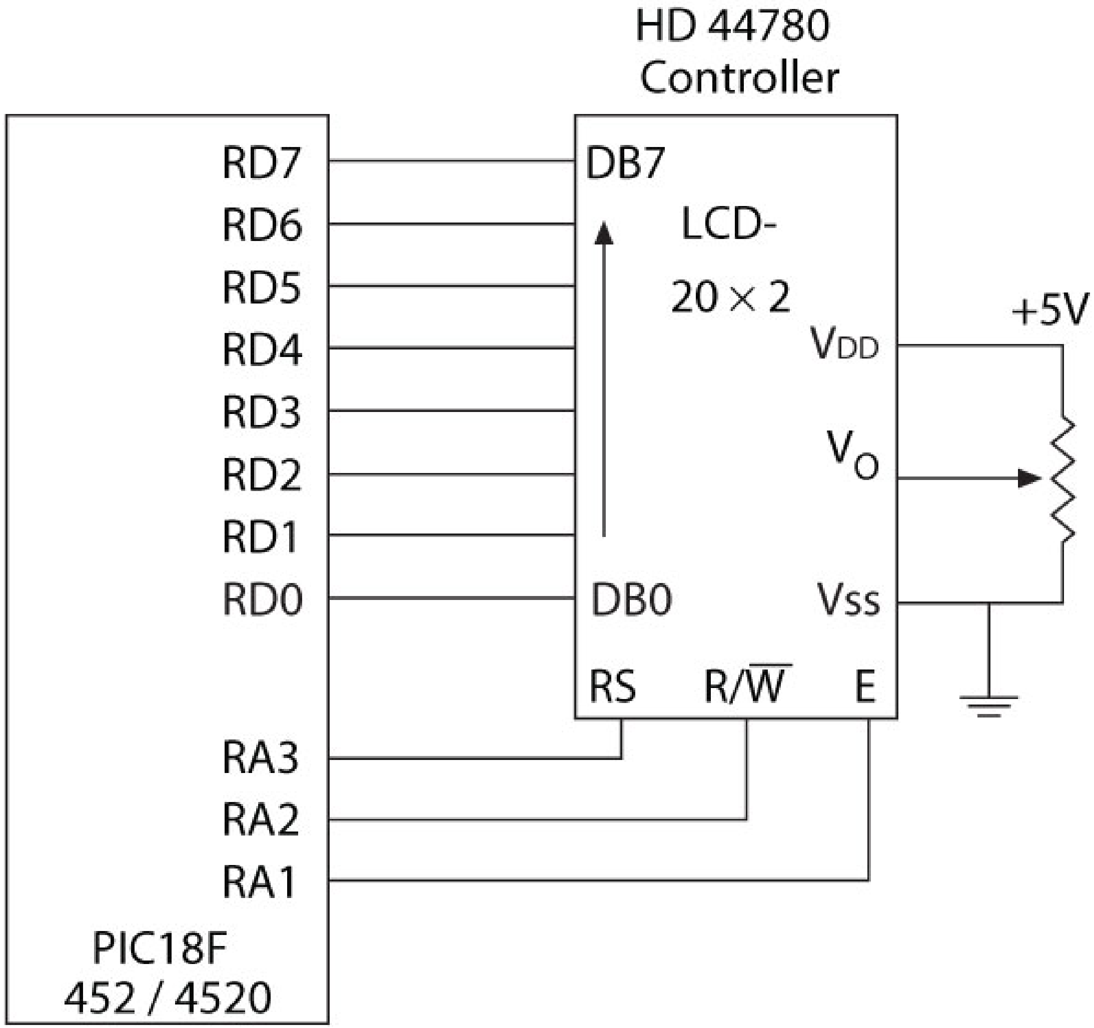
Interfacing LCD

- Problem statement
  - Interface a 2-line x 20 character LCD module with the built-in HD44780 controller to I/O ports of the PIC18 microcontroller.
  - Explain the control signals necessary to read from and write to the LCD.
  - Write a program to display ASCII characters.

Interfacing LCD

- Hardware
  - 20 x 2-line LCD display
    - Two lines with 20 characters per line
  - LCD has a display Data RAM
    - Stores data in 8-bit character code
  - Each register in Data RAM has its own address
    - Corresponds to its position on the line
      - Line 1 is 00_H to 13_H
      - Line 2 is 40_H to 53_H

Interfacing LCD



Interfacing LCD

- Driver HD44780
  - 8-bit data bus (RD7-RD0)
  - Three control signals
    - RS – Register Select (RA3)
    - R/W – Read/Write (RA2)
    - E – Enable (RA1)
  - Three power connections
    - Power, ground, and variable resistor to control brightness

Interfacing LCD

- Can be interfaced either in 8-bit mode or 4-bit mode
  - In 8-bit mode, all eight data lines are connected
  - In 4-bit mode, only four data lines are connected
    - Two transfers per character (or instruction) are needed
- Driver has two 8-bit internal registers
  - Instruction Register (IR) to write instructions to set up LCD
    - Table 9-3
  - Data Register (DR) to write data (ASCII characters)

Interfacing LCD

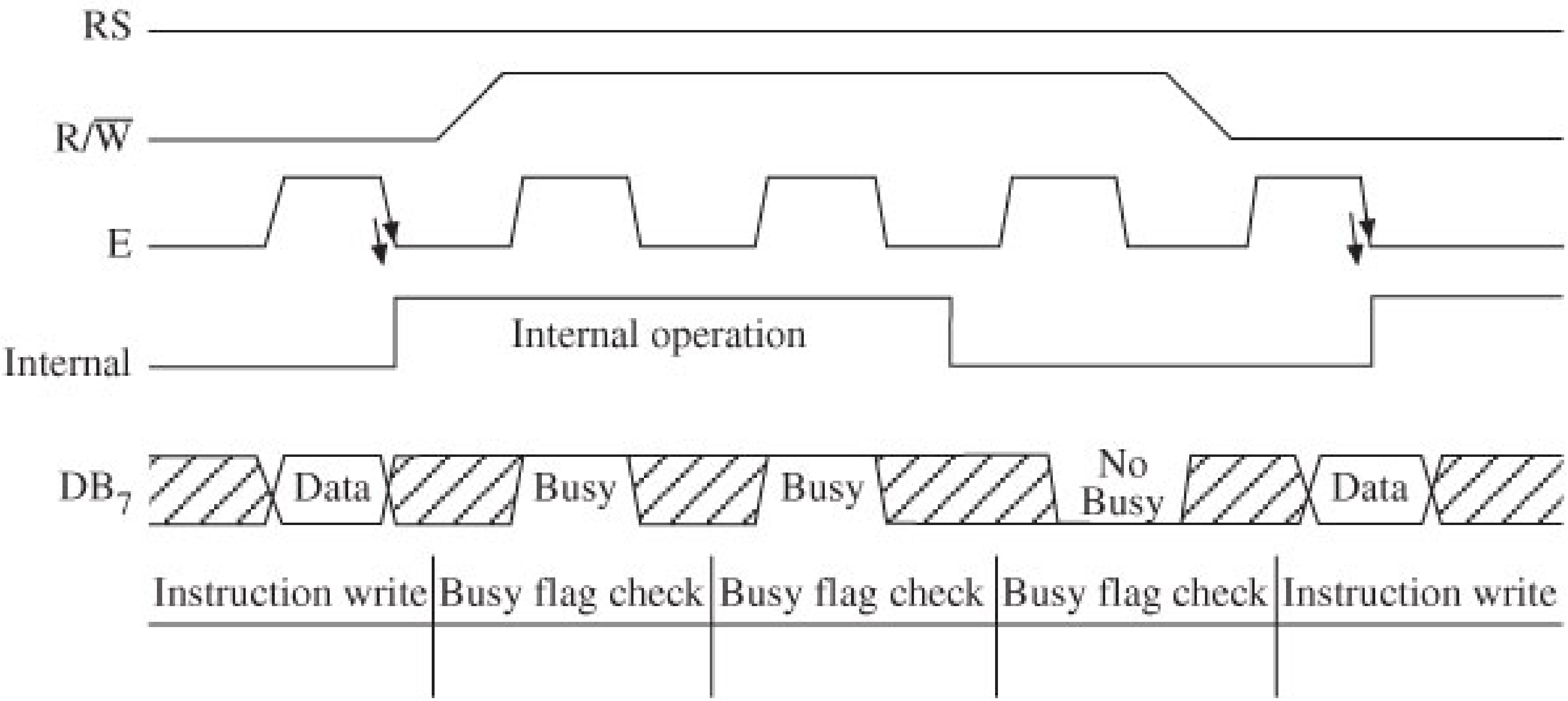
- LCD Operation
  - When the MPU writes an instruction to IR or data to DR, the controller:
    - Sets DB7 high indicating that the controller is busy
    - Sets DB7 low after the completion of the operation
  - The MPU should always check whether DB7 is low before sending an instruction or a data byte

Interfacing LCD

- Writing to or Reading from LCD (Table 9-4)
  - The MPU:
    - Asserts RS low to select IR
    - Asserts RS high to select DR
    - Reads from LCD by asserting the R/W signal high
    - Writes into LCD by asserting the R/W signal low
    - Asserts the E signal high and then low (toggles) to latch a data byte or an instruction

Interfacing LCD

- Timing diagram: writing to LCD



Interfacing LCD

- Software
  - To write into the LCD
    - Send the initial instructions to set up the LCD
      - 4-bit or 8-bit mode
    - Continue to check DB7 until it goes low
    - Write instructions to IR to set up LCD parameters
      - Number of display lines and cursor status
    - Write data to display a message