Microcomputers I – CE 320

Mohammad Ghamari, Ph.D.
Electrical and Computer Engineering
Kettering University

Announcement

Lecture 18: Input and Output

Today’s Topics

By the end of this class you should be able to:

- Distinguish between port-mapped I/O and memory-mapped I/O
- Use the digital input/output ports of the Star12

Basic Terminology

- Input/Output (or simply I/O) features allow microprocessors to directly communicate with other devices, such as switches, LCD screens, sensors, keypads, etc.

- Key parameters for I/O:

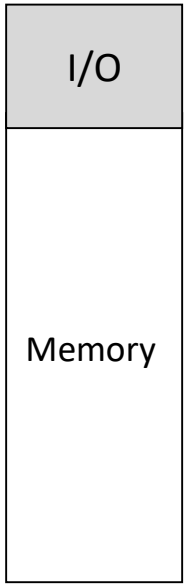
  - **Pin vs. Port**
    - Pin usually refers to a single wire
    - Port usually refers to multiple wires

  - **Direction**
    - Pins/ports need to be configured to define the direction of signal flow

Accessing I/O

- I/O is often accessed by a program just much like accessing memory addresses.

- Processors differ in where the memory addresses are located, and there are two main approaches:

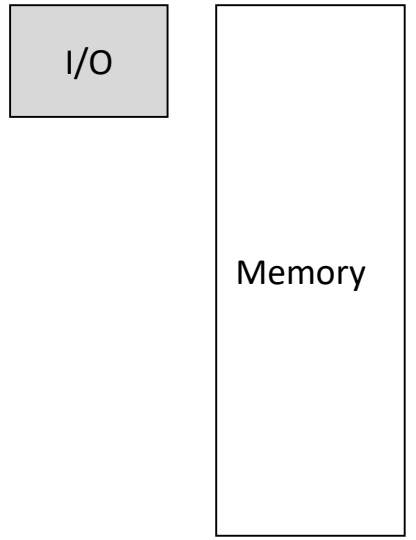


1. **Memory-Mapped I/O**
   - Part of regular memory addresses are used for I/O ports
   - Standard Load/Stores instructions are used
   - Advantage – Easy to use, just need address of port
   - Disadvantage – Must reserve block of addresses in main memory area

Accessing I/O

2. Port-Mapped I/O

- Second memory space is used just for I/O
- Advantage – Special I/O instructions are used, easy to see when I/O is occurring
- Disadvantage – Same addresses used twice, can lead to confusion



I/O in S12 Family

- The ports in the S12 family are named using a somewhat random selection of letters – A, B, E, H, J, K, L, M, P, S, T, U, V, and W.

- Often, these ports can be used for a **specific function**, such as the RS-232 port that communicates with the PCs in lab, SPI, I2C, etc.

- When used as simple I/O that is manually controlled, they are referred to as **general-purpose I/O**.

- The process of controlling a general-purpose I/O port to create a complex pattern (sometimes to simulate a communications protocol that the chip does not have dedicated hardware for) is often called **bit-banging**.

- In this lesson we will limit the discussion to the use of the ports as general purpose I/O on the Dragon12+ board.

Ports B, H, and P

General purpose (used either input or output)

- There are 8 pins in each of these ports (B, H, and P).
- Each port has a corresponding *memory address* that shows the values of the 8 pins.

  PORTB = $0001

  PTH = $0260

  PTP = $0258

- Like many microcontrollers, each pin can be individually programmed to act as an input pin or an output pin.
  - When used as **input ports**, the value is 1 if the voltage at the pin is high, and 0 if the voltage at the pin is low.
  - Performing a **load** from these addresses gives the input values at that moment.
  - When used as **outputs**, store operations send signal to drive the pin to high (when storing 1) or low (when storing 0).

Data Direction Registers (DDR)

- **How do we determine if a general purpose I/O port is being used for input or output?**

- A port that is ***bidirectional*** must have a ***configuration*** method to select either *input* or *output*.

- For the S12, this is done using ***data direction registers (DDR)***. Each port has its own register (with memory address given below), and the pins of each port are configured separately.

  DDRB = $0003

  DDRH = $0262

  DDRP = $025A

Data Direction Registers …

- To configure the ports, store a value into the corresponding DDR based on the settings below:

  1: use the pin as an output

  0: use the pin as an input

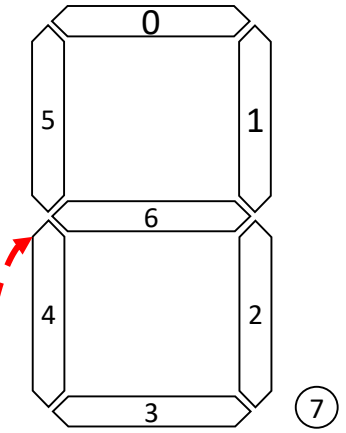
NOTE:

- When a pin is configured for an input, storing a value to its data bit is ignored.
- When configured as an output, the voltage at the physical pin is ignored (as far as we are concerned in Micros I).

Ports B, H, and P in the Dragon12+ Board

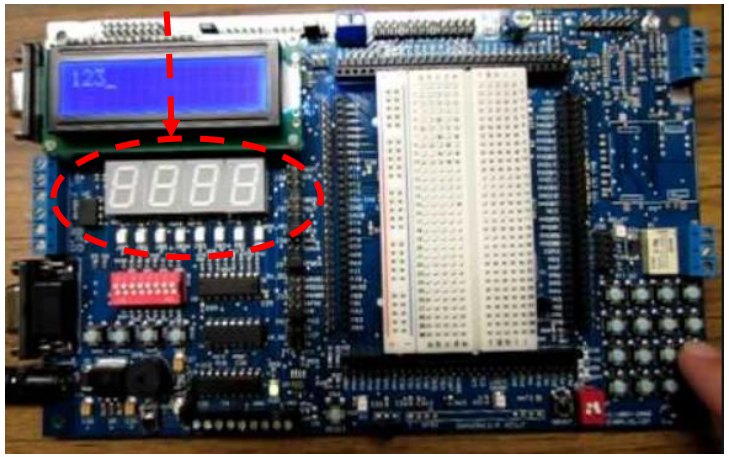
Port B – 7 segment digits

- The S12 processor in the Dragon12+ boards have already been connected to hardware. We will briefly discuss each.



Port B

- This port supplies the values to the 7-segment digits.
- Each digit actually has 8 LEDs including the decimal point.
- The diagram shows which bit controls each LED.
- The pins of Port B are connected to *all four digits* in the Dragon12+ board.
  - This means it cannot output two different patterns on two 7-segment displays at the same time.



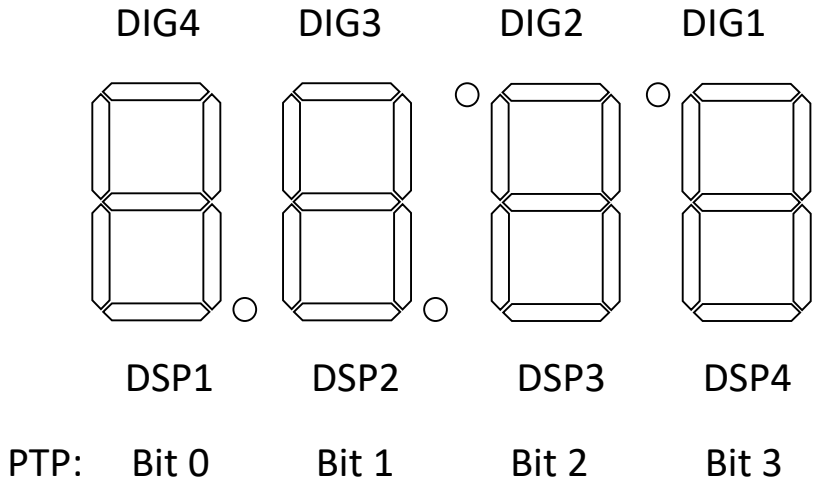
Dragon12 board

Ports B, H, and P in the Dragon12+ Board …

Port P – Selecting a 7-segment digit

Port P

- Port P is used to *select* which of the four 7-segment LED digits are *enabled*.
- Remember that the display pattern is determined by Port B.
- Digits that aren't enabled will have all LEDs off.
- This shows which bit in PTP controls each 7-segment display. Note that only 4 bits of Port P are used.



Enable/Disable

- To *enable* a digit with the Dragon12+ board's hardware, PTP *outputs a 0*.
- To *disable* a digit, PTP must *output a 1*. Note that this is ***active-low logic***.

Ports B, H, and P in the Dragon12+ Board …

Port H – switch input

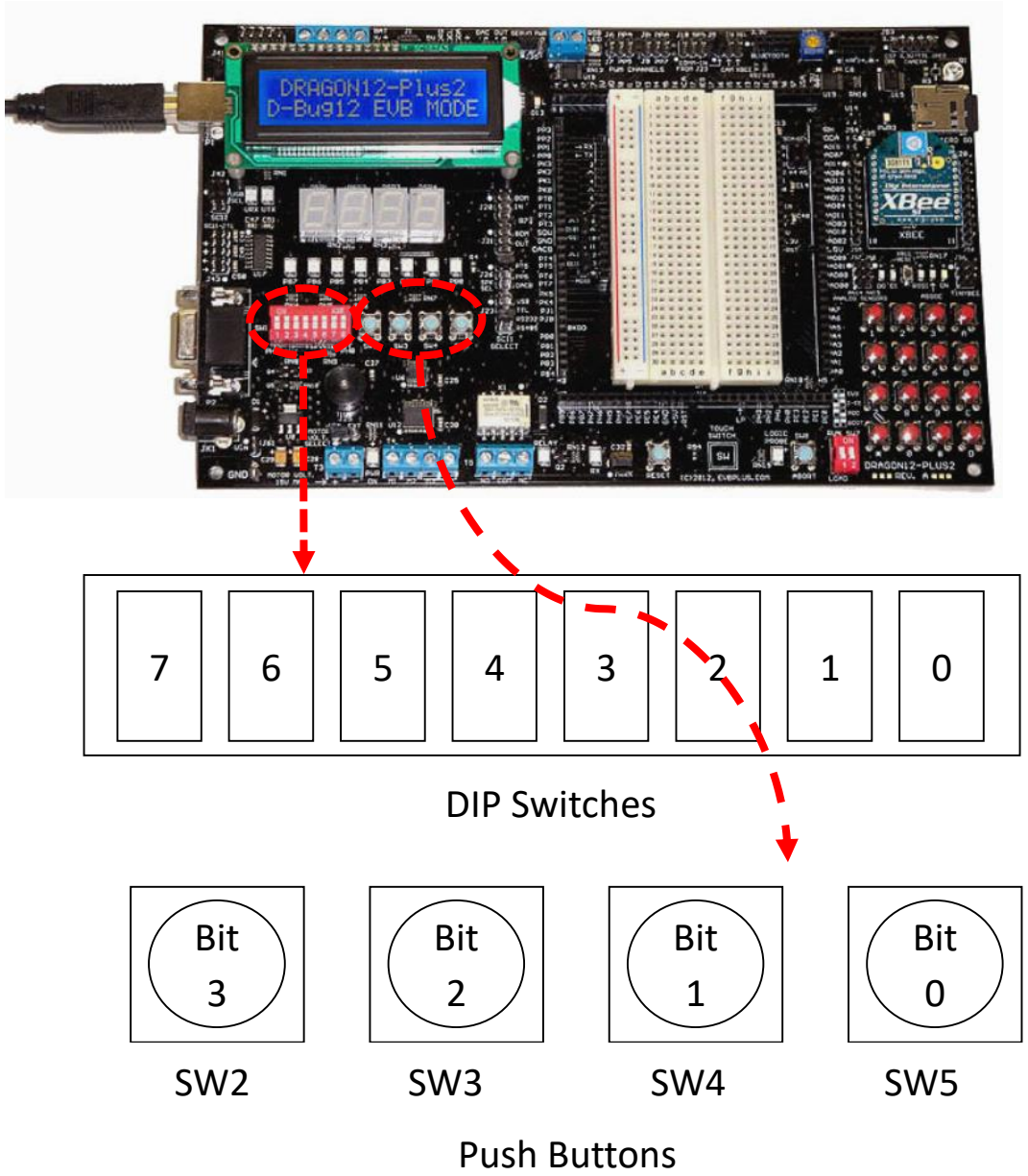
Port H

- The port is used to read the 8-DIP switches and the 4 push buttons as shown here in the diagram.

- There are two things to note.

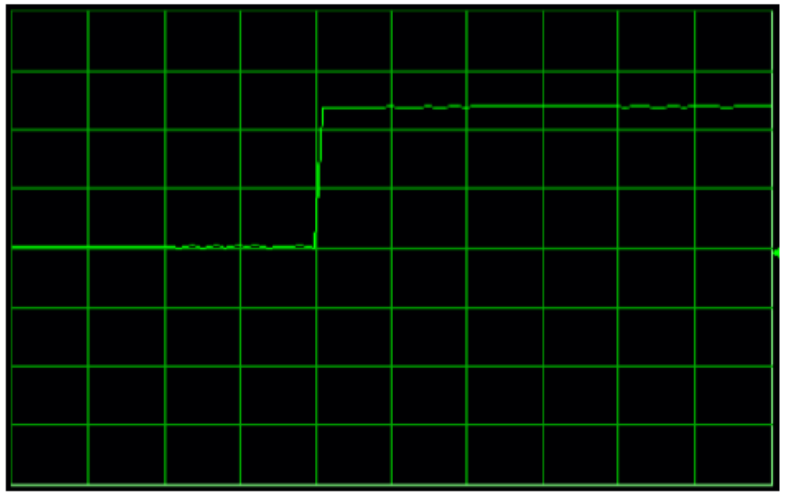
  - **First**, there are four pins that monitor both a switch and a push button.
    - There is no way to distinguish which is being pressed.

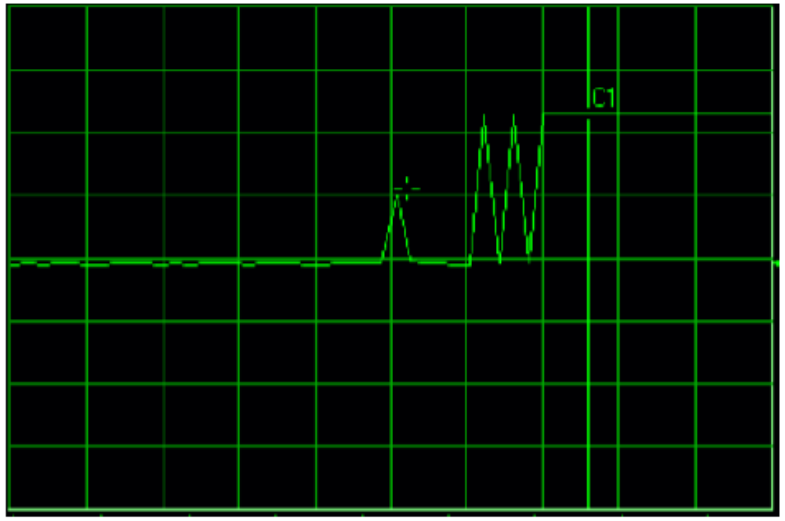
  - **Second**:
    - Pressing a button/flipping a switch pulls the input voltage low, so the processor would read a 0 in the data register.
    - A 1 appears in the register when the switches/buttons are not being asserted.



Switch Bounce

- When a switch is asserted, we expect a signal something like the top right picture.
- However, signals has a transient period.
- When a switch (or button) is asserted (or pressed), the actual signal can be the bottom right figure.
  - For a short period of time, the switch signal is bouncing.
- That is why the program detects multiple buttons.

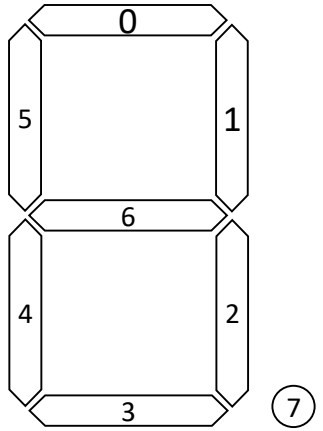




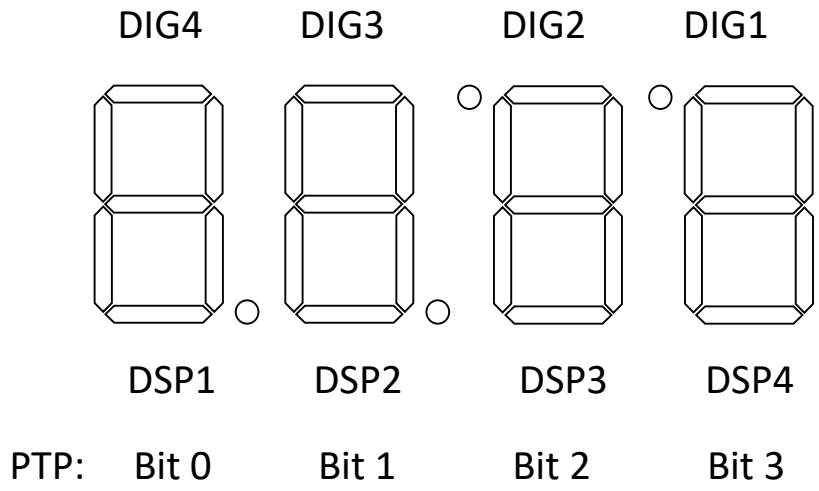
Homework Example

Write a program that

- Turns on one LED segment of display 2 (DISP2) at a time.
- When the program begins, only segment 0 should be on.
- Every time SW4 is pressed, the current LED should turn off and the next one (by number) should be turned on.




```
PORTB   EQU     $0001
DDRB    EQU     $0003
PTH     EQU     $0260
DDRH    EQU     $0262
PTP     EQU     $0258
DDRP    EQU     $025A
```



```
                ORG     $C000
                MOVB    #%11111111, DDRB
                MOVB    #%11111111, DDRP
                MOVB    #%00000000, DDRH
                MOVB    #%00000001, PORTB
                MOVB    #%11111101, PTP
NOTPUSHED       BRSET   PTH,%00000010, NOTPUSHED
                LDX     #$6000
DEBOUNCE        DEX
                BNE     DEBOUNCE
                BRSET   PTH,%00000010, NOTPUSHED
                LSL     PORTB
                BNE     PUSHED
                MOVB    #%00000001,PORTB
PUSHED          BRCLR   PTH,%00000010, PUSHED
                LDX     #$6000
DEBOUNCE2       DEX
                BNE     DEBOUNCE2
                BRCLR   PTH,%00000010, PUSHED
                BRA     NOTPUSHED
```