Lecture 18

Input and Output

Purpose:

- 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.

Types of I/O

There are several key parameters that are used to describe types of I/O.

Pin Vs. Port

- Pin usually refers to a single wire
- Port usually refers to two or more wires

Direction

- Unidirectional – data flows in one direction on a wire
- Bidirectional – data may flow in either direction on a wire

Full-Duplex vs. Half Duplex

- Full-Duplex – data in both direction simultaneously
- Half-duplex – data in both directions must alternate

Asynchronous vs. Synchronous

- Asynchronous – no clock signal is used
- Synchronous – clock signal is used

Accessing I/O

I/O is often accessed by a program just much like accessing memory addresses. Processors differ in when the memory addresses are located, and there are two main approaches.

Memory-Mapped I/O

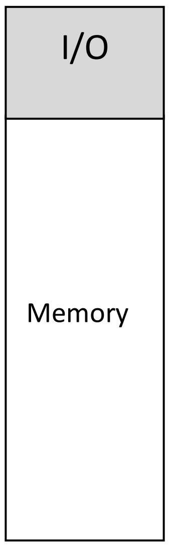
Regular memory addresses correspond to

I/O ports, standard Load/Stores are used

Adv – Easy, just need address of pot

Dis – Must reserve block of addresses in

Main memory area



Port-Mapped I/O

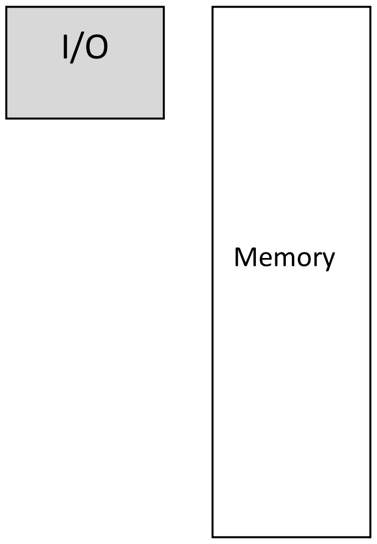
Second memory space is used just for I/O

Adv – Special I/O instructions are used,

Easy to see when I/O is occurring

Dis – 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. Different processors in the family have a subset of these ports. Most of these ports can be used for a specific function, such as the RS-232 port that communicates with the PCs in lab, of they can be used as **general-purpose I/O**.

We will talk briefly about some of the special functions in a later lecture. Today, we will limit the discussion to the ports used as general purpose I/O on the Dragon12+ board.

General Purpose Data for Ports B, H, and P

These three ports are all have 8 pines (as with many of the S12's ports). Each port has a corresponding memory address that shows the values of the 8 pins.

PortB = $0001

PTH = $0260

PTP = $0258

When used as input ports, the value is 1 (i.e. true) if the voltage at the pin is high, and 0 (i.e. false) 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, program store values into the same locations, and the hardware commands the voltages as above.

This should lead to the following question: How do we determine if a port is being used for input or output?

Data Direction Registers

When being used as general purpose I/O, a port that is bidirectional must have a configuration method to select wither input or output. For the S12, these are called data direction registers. Each port has its own register, and the pins of each port are configured separately.

DDRB = $0003

DDRH = $0262

DDRP = $025A

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

1: use the pin as an output

0: use the pin as an input

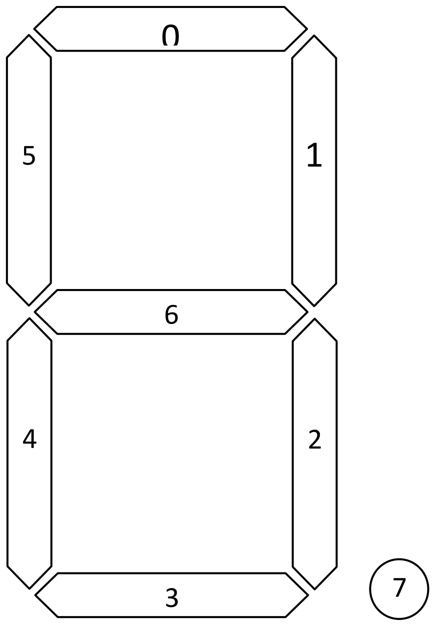
When a pin is configured for an input, storing a value to its data bit is ignored. When configured as an output, the voltage at the physical pin is ignored (as far as we are concerned in Micros I).

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

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

PortB

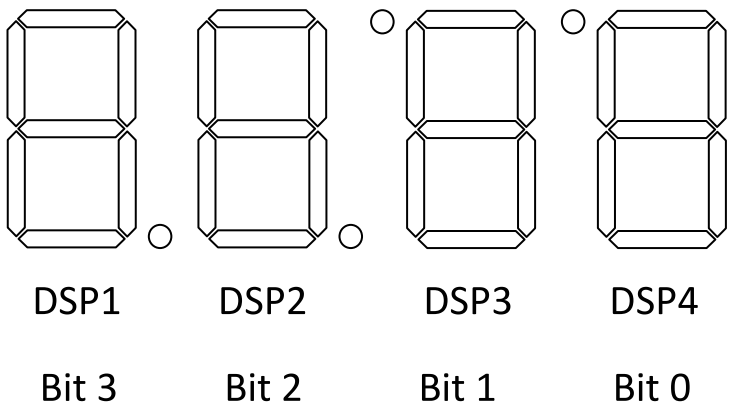
This port supplies the values to the 7-segment digits. Each digit actually has 8 LEDs including the decimal point. The diagram below shows which bit of PortB controls each LED.



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.

Port P

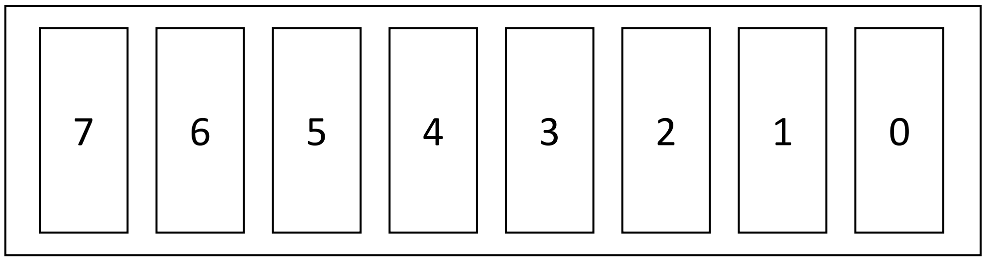
This port is used to select which of the four 7-segment LED digits are enabled. Any digit that is enabled will display the pattern commanded by Port B. Digits that aren't enabled will have all LEDs off. The diagram below shows which bit in PTP controls each 7-segment display. Note that only 4 bits are used in this port.



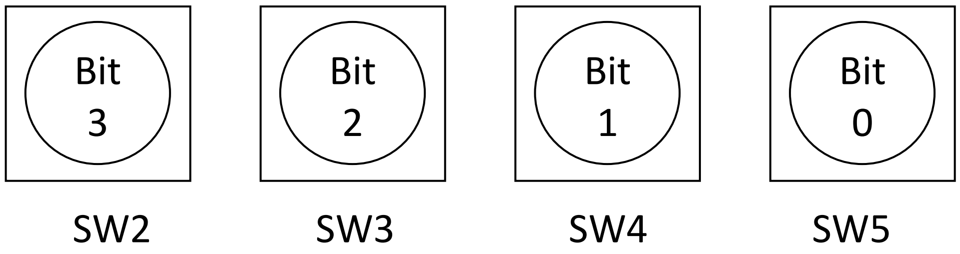
To enable a digit with the Dragon12+ board's hardware, PTP outputs a 0. To disable a digit, PTP must output a 1.

Port H

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



DIP Switches



Push Buttons

Example Program

Write a program that turns on one LED segments of digit 2 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
NOTPUSH     BRSET   PTH,%00000010, NOTPUSH
            LDX     #$6000
DEBOUNCE    DEX
            BNE     DEBOUNCE
            BRSET   PTH,%00000010, NOTPUSH
            LSL     PORTB
            BNE     PUSHED
            MOVB    #%00000001,PORTB
PUSHED      BRCLR   PTH,%00000010, PUSHED
            LDX     #$6000
DEBOUNCE2   DEX
            BNE     DEBOUNCE2
            BRCLR   PTH,%00000010, NOTPUSH
            BRA     NOTPUSH
```