DEVICE DRIVERS AND TERRUPTS SERVICE MECHANISM–

Lesson-12: Device Driver Programming

Writing the software for the driver

- Information about how the device communicates.
- Information about device three sets of device registers─ (i) data register (s) or buffer (s), (ii) control register (s) and (iii) status register (s).
- Information of other registers and common addresses to a device register.

Device Registers

Each physical device has the

- Control Register,
- Status Register,
- Receive Buffer [Register(s) or Memory] and
- Transmit Buffer [Register or Memory]

Control-register bits

- Control all actions of the device.
- A control bit can even control that which address corresponds to which data register at an instance. For example, at the instance when the DLAB control bit if set, then at that instant 0x2F8 corresponds to the divisor- latch lower byte.

Writing the software for the driver ISR

- A device *initializes* (also means configures, registers, attaches) by setting the control register bits.
- A device *closes* (also means resets, de-registers, detaches) by resetting the control register bits.

Writing the software for the driver ISR

- A device *initializes* (also means configures, registers, attaches) by setting the control register bits.
- A device *closes* (also means resets, de-registers, detaches) by resetting the control register bits.

Parameters to know for writing a device driver ISR

(i) Distinct as well as common addresses for each device register,

(ii) Purpose of each bit at the control register when reset or set

Parameters to know for writing a device driver ISR

(iii) Device status for the reset and set states of each of the status flag and which flag sets on which action. Is that flag resets itself on execution of the ISR after the action?

(iv) Whether control bits and status flags are at the same address (processor instruction then auto-distinguishes these)

Parameters to know for writing a device driver ISR

(v) Whether control bits and status flags are at the same register, few bits are used as control bits and few as the flags.

(vi) Whether the status flag, which sets on a device- interrupt, auto-resets on executing the ISR or if the ISR needs to reset it.

Parameters to know for writing a device-driver ISR

(vii) Whether **control bits need** to be changed, reset or set again before return to the interrupted process.

(viii) List of actions at the data buffers, control registers and status registers required by the driver ISR.

Parameters to know for writing a device driver

(ix) Device register address modifications on changing of the glue circuit (interface circuit with processor and memory)

Device Register Special Types

- One common device register address
- For example, transmit register and receive registers at same address.
- An Out or In (Store or Load) processor-instruction to device will make the device to select appropriate one, receiver buffer or transmit buffer.

Status-register

- Status-register bits are the flags for the statuses of the device at an instant and change after the device performing the actions as per device driver. Each status flag bit reflects the present status of a device-function or action.

Four sets of the codes in ‘C’ or assembly

(i) Configuring (initializing) or activating by write into control registers. Also called opening or attaching)

(ii) and (iii) Read or write as per intended function (s) using an ISR(s), and

(iv) Resetting (also called deactivating or closing or detaching).

Set of addresses for Registers

- Processor uses the addresses for appropriate read-write or other operations using the registers.
- A device driver ISR coding is for at least four sets of the codes in ‘C’ or assembly for a device.
- The driver also defines a descriptor for the device.

Example of Device registers-

- SBUF (Serial Buffer Register) in 8051 Microcontroller Serial Port,
- RBR and THR the Receiver data-Buffer and Transmitter Holding Registers in UART 8250, used at a serial line device in IBM PC's COM ports.

Serial line device

- For example, Transmitter empty flag reflects in serial line device an instance between finishing the transmission of bits from the THR buffer register and obtaining the new bits for next transmission. As soon as THR gets new byte for serial transmission, the flag resets. As soon as THR empties after transmission, it sets.

System initiation of call for executing an ISR

- Either setting of status flag (software call) or hardware call by a signal, the system *initiates* a call for executing an ISR. [An ISR (also called Interrupt Handler Routine) executes if **(i)** it is enabled (not masked at the system) and **(ii)** the interrupt system itself is also enabled.]

Serial line device UART 8250

- Addresses are 0x2F8-0x2FF for COM1 and 0x3F8-0x2FF for COM1 in IBM PC . Embedded system UART 8250 will have different glue circuit, hence different addresses.

Serial Line Device Driver for 8250

Definitions in Pre- Processor

1) Device initialization and activation codes to use the 8250; define the Output port and input port addresses and COM2 registers

Serial Line Device Driver

2) Declare Eight Character data structure for data at the device registers
3) Initialize Normal Serial Data Output functions for driving the output using the function outportb ( )
4) Initialize Normal Serial Data Input functions for driving the input at port using the function inportb ( )

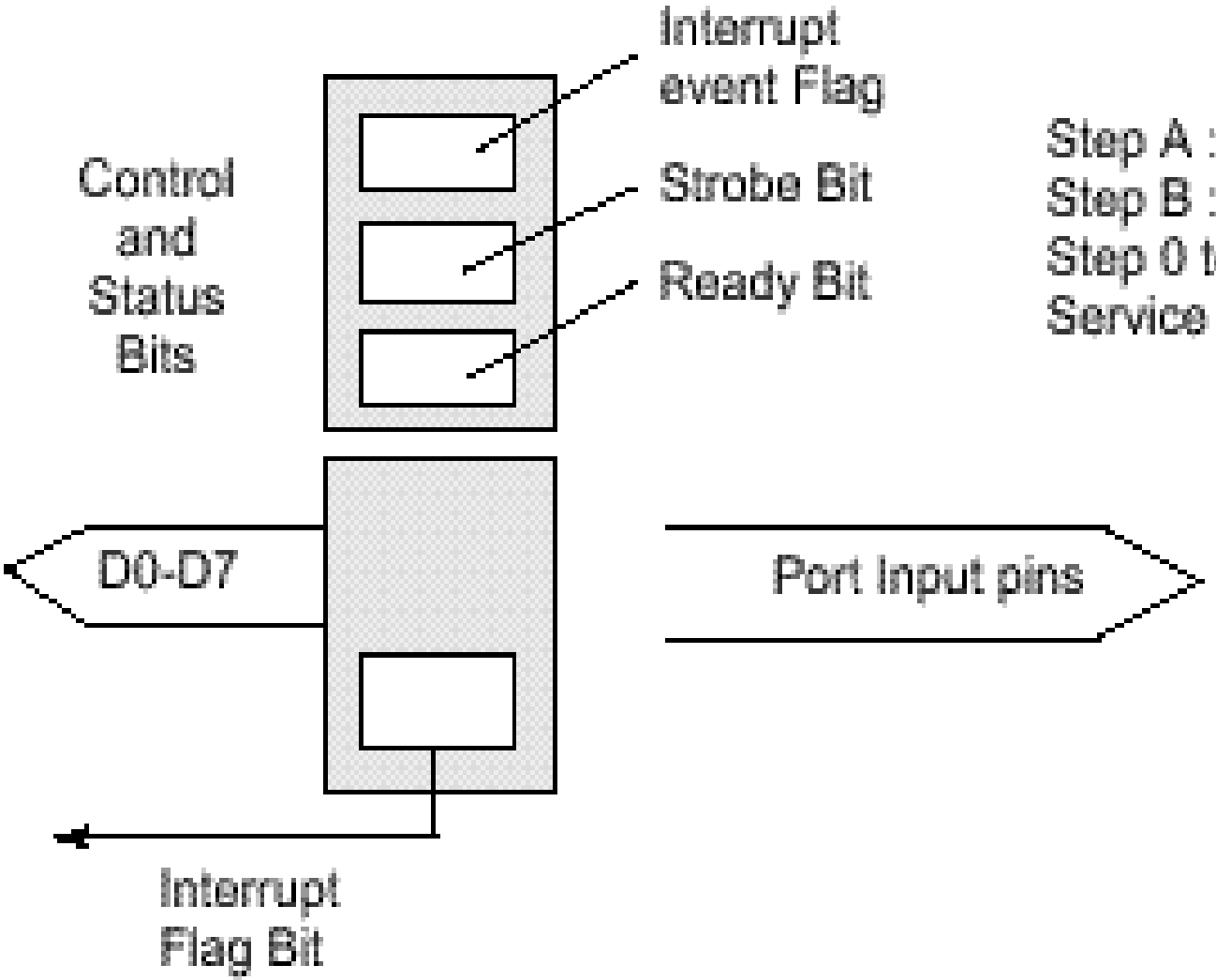
Serial Line Device Driver

5) Device driver code for the transfer of a character to Serial Line Device by writing to serial port at 0x2F8.
6) Device driver code for the transfer of a character from Serial Line Device by reading from serial port at 0x2F8.

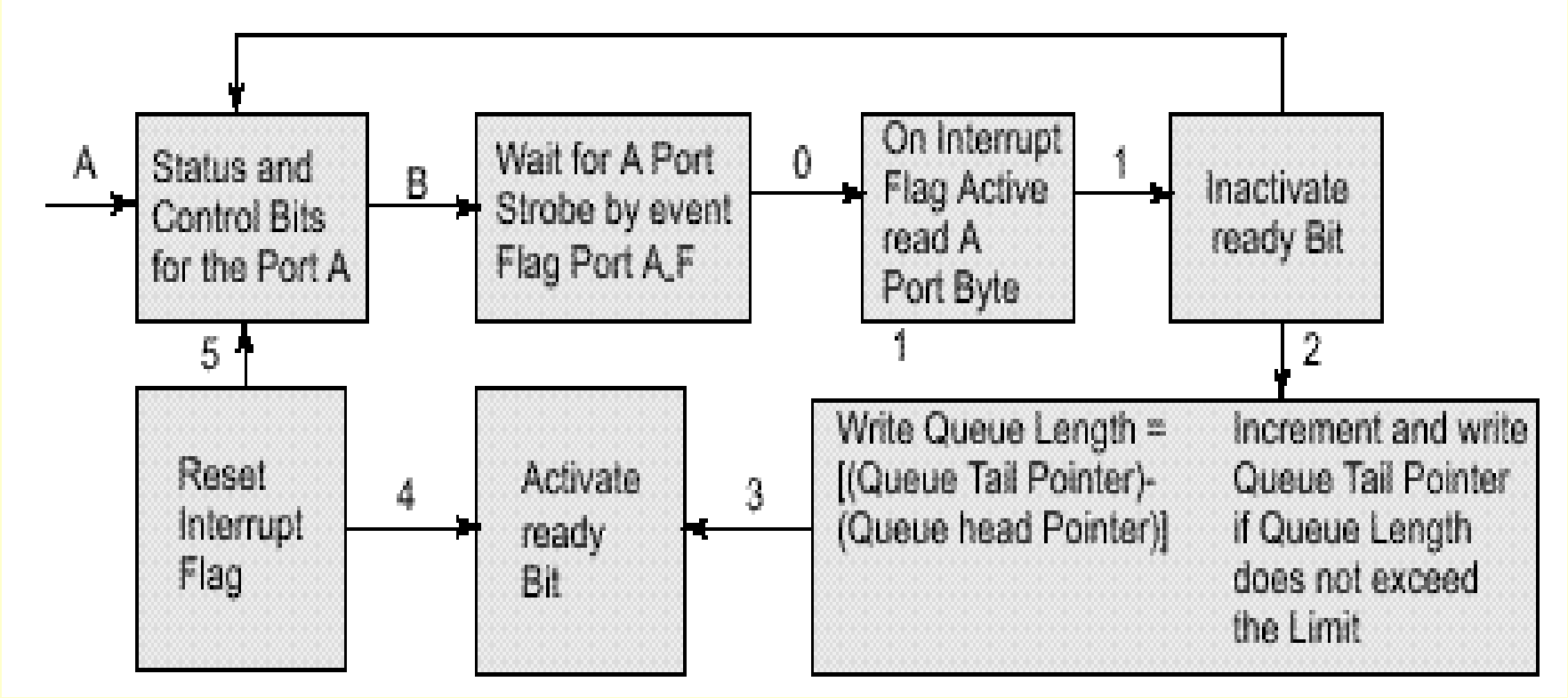
Example of a Parallel Port Device Driver

Driving a parallel port needs three modules:

1) For initializing by placing appropriate bits at the control register
2) For calling on interrupt whenever a status flag sets at the status register
3) For interrupt servicing (device driver) programming



Control bits and status flags used and the port interface with the system data bus



Steps A, B, and 0 to 5 for reading the port input

Summary

We learnt

- Device configures and initializes by driver using the control bits at its control register(s).
- Device status registers reflect device status to the driver.

We learnt

- A device driving ISR is designed using the device addresses and three sets of device registers- data register (s) or buffer(s), control register (s) and status register (s).
- A device function or action is configured and controlled by a control bit. Each control bits control a distinct function

We learnt

- The driver initiates and executes on status flag change.

End of Lesson 12 of Chapter 6
on
Device driver Programming