Basic Input/Output Operations

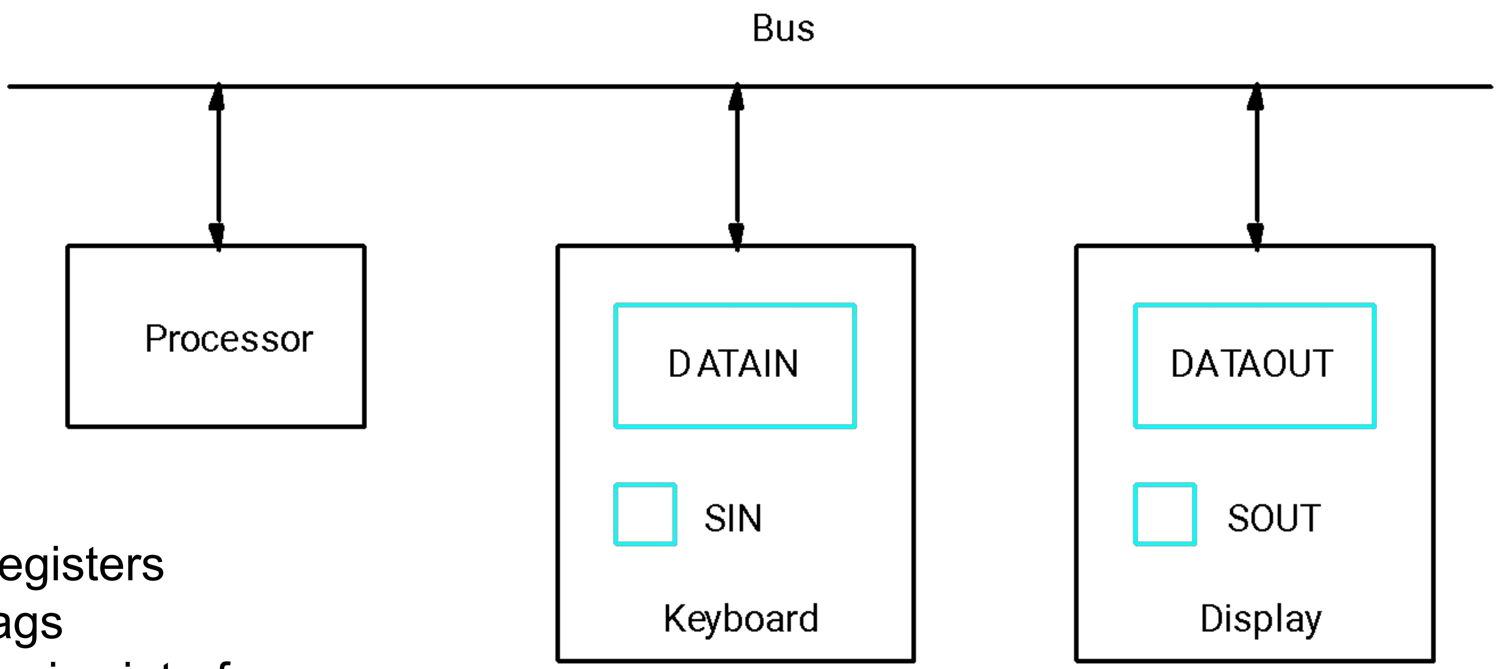
I/O

- The data on which the instructions operate are not necessarily already stored in memory.
- Data need to be transferred between processor and outside world (disk, keyboard, etc.)
- I/O operations are essential, the way they are performed can have a significant effect on the performance of the computer.

Program-Controlled I/O Example

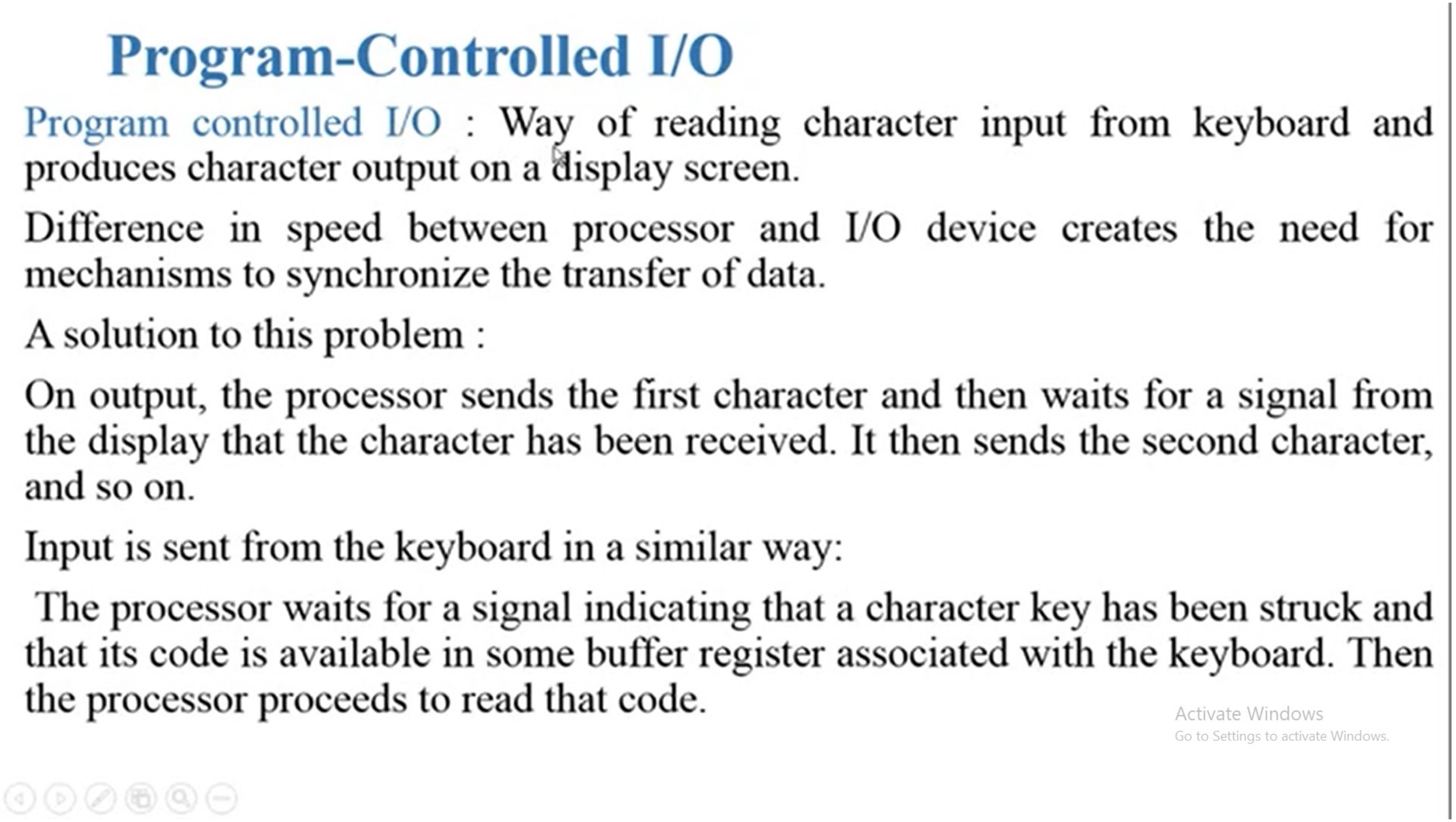
- Read in character input from a keyboard and produce character output on a display screen.
- Rate of data transfer (keyboard, display, processor)
- Difference in speed between processor and I/O device creates the need for mechanisms to synchronize the transfer of data.
- A solution: on output, the processor sends the first character and then waits for a signal from the display that the character has been received. It then sends the second character. Input is sent from the keyboard in a similar way.

Program-Controlled I/O Example



- Registers
- Flags
- Device interface

Figure 2.19 Bus connection for processor, keyboard, and display



Program-Controlled I/O

Program controlled I/O : Way of reading character input from keyboard and produces character output on a display screen.

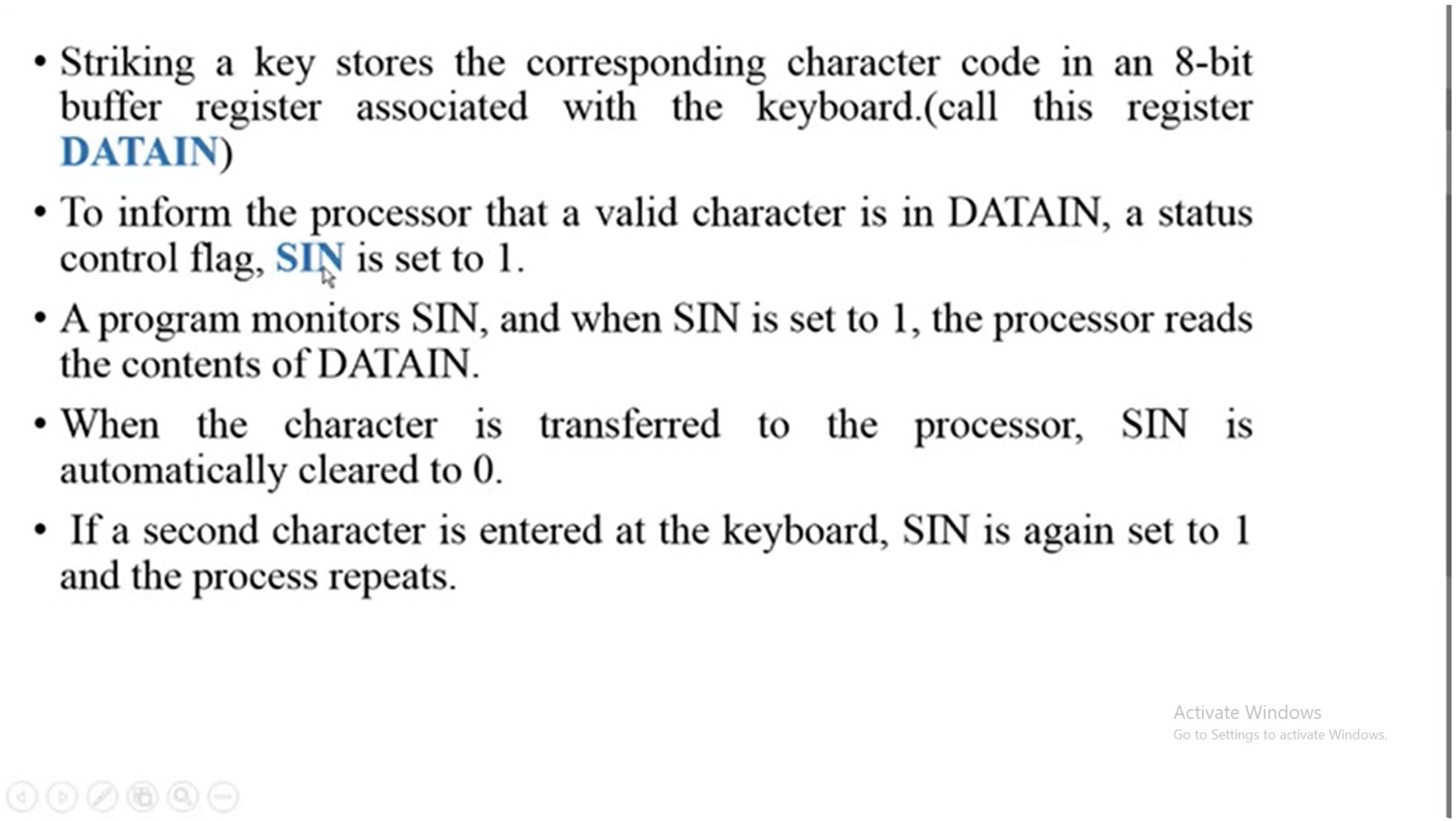
Difference in speed between processor and I/O device creates the need for mechanisms to synchronize the transfer of data.

A solution to this problem :

On output, the processor sends the first character and then waits for a signal from the display that the character has been received. It then sends the second character, and so on.

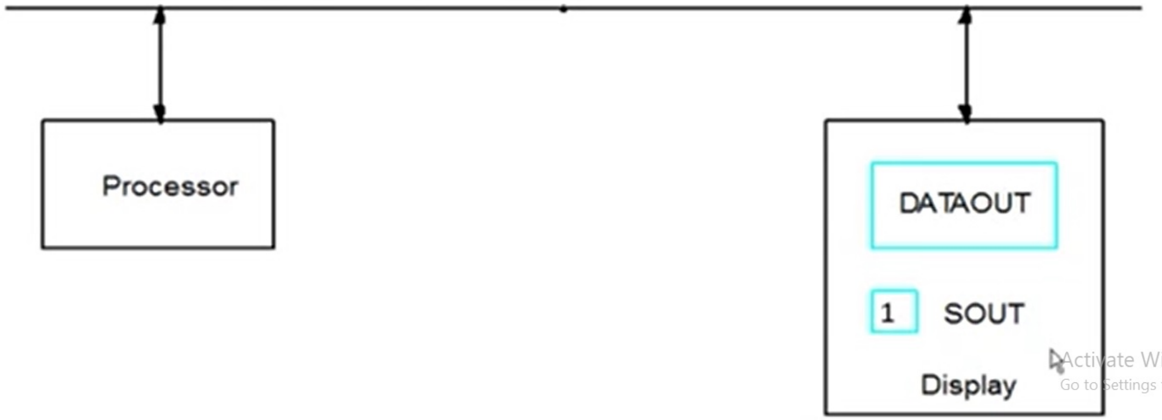
Input is sent from the keyboard in a similar way:

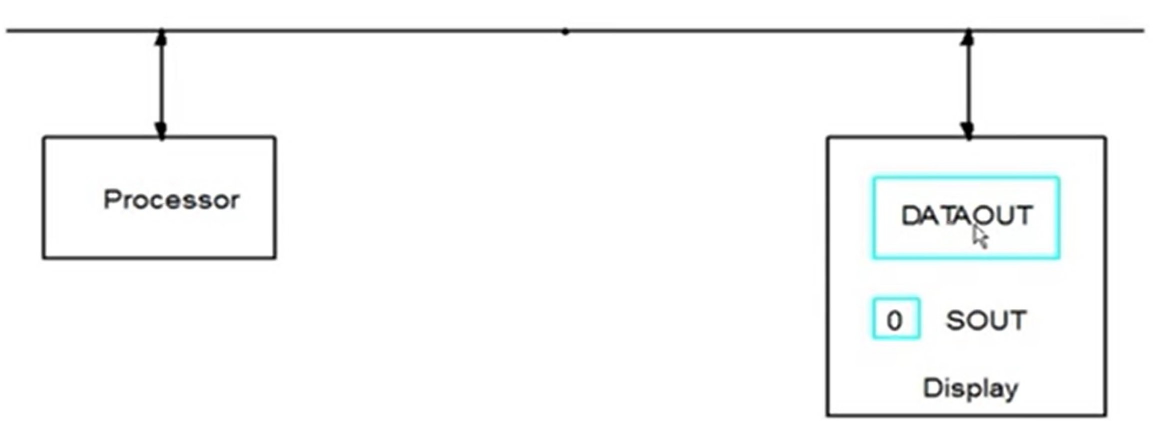
The processor waits for a signal indicating that a character key has been struck and that its code is available in some buffer register associated with the keyboard. Then the processor proceeds to read that code.



- Striking a key stores the corresponding character code in an 8-bit buffer register associated with the keyboard.(call this register **DATAIN**)
- To inform the processor that a valid character is in DATAIN, a status control flag, **SIN** is set to 1.
- A program monitors SIN, and when SIN is set to 1, the processor reads the contents of DATAIN.
- When the character is transferred to the processor, SIN is automatically cleared to 0.
- If a second character is entered at the keyboard, SIN is again set to 1 and the process repeats.

- When characters are transferred from the processor to the display ,A buffer register , DATAOUT, and a status control flag , SOUT, are used .
- When SOUT equals 1, the display is ready to receive a character. Under program control, the processor monitors SOUT, and when SOUT is set to 1, the processor transfers a character code to DATAOUT.
- The transfer of a character to DATAOUT clears SOUT to 0; when the display device is ready to receive a second character, SOUT is again set to 1.





- The buffer registers **DATAIN** and **DATAOUT** and the status flags **SIN** and **SOUT** are part of circuitry and known as a ***device interface.***

Program-Controlled I/O Example

- Machine instructions that can check the state of the status flags and transfer data:

```
READWAIT  Branch to READWAIT if SIN = 0
          Input from DATAIN to R1

WRITEWAIT Branch to WRITEWAIT if SOUT = 0
          Output from R1 to DATAOUT
```