Computer Organisation and Architecture

CS31007
Assignment-2 Report

Ashwani Kumar Kamal (20CS10011)

October 31, 2022

---

1 Problem Statement

1. Input/output is an important aspect of most computation tasks I/O must be done via programming the CPU.
2. What is the architectural view of an i/o device for a programmer?
3. How is the architectural view supported organisationally? Explain with the help of a diagram.
4. Consider any simple i/o task (such as reading from a keyboard, reading or writing a block of data to the disk) and explain how that would be achieved in your framework.

1.1 Marking Guidelines

For each of the items listed below full marks are awarded if the feature is satisfied, otherwise none (0 marks).

Table 1:

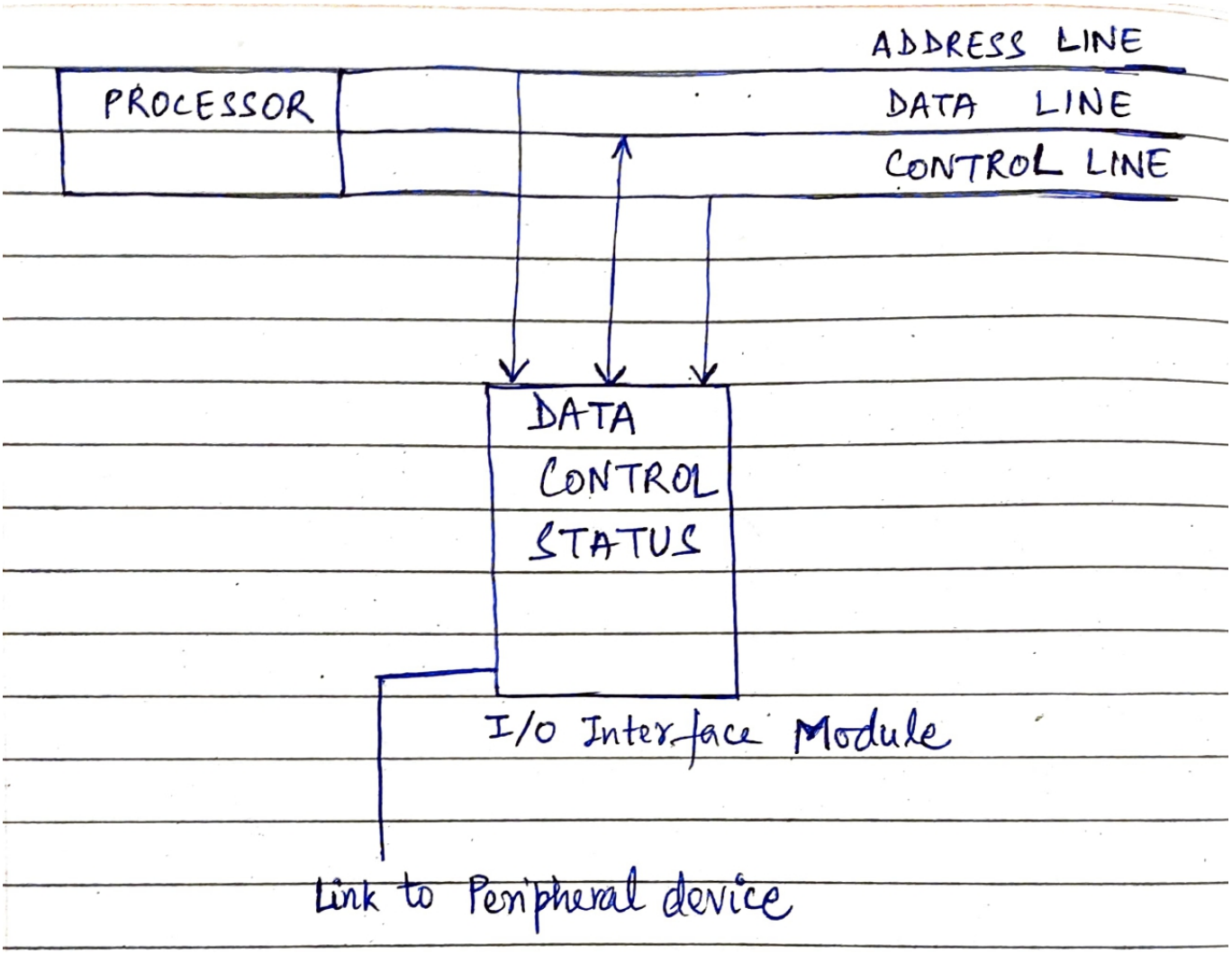
Action	Marks
Architectural view	5
Organisational support	10
Example	15
Total Marks	30

2 Solution

2.1 Architectural view of i/o device for a programmer

Architecturally a programmer can use software interrupts to manually invoke control signals to peripheral devices for their functioning. Interrupts are very similar to procedural calls. Interface modules have an address associated with them which makes it possible to programmatically change the corresponding registers and invoke the functions. The processor polls the status register of the interface module and correspondingly gives control signal to the peripheral device.
Programmer's view: Accessing registers are to be done using memory locations. Each register (data, status and control, in a contiguous manner) is assigned a specific address that is recognized by the interface module. These are the addresses of the data registers. This makes all addresses word-aligned in a 32-bit word computer, which is usually done in practice.

2.2 How is the architectural view supported organisationally?



- There are three lines to the processor namely address line, data line and control line.
- Transmission happens from processor to I/O interface module in case of address and control line. Data line is bidirectional as the peripheral device can accept / produce data to and from processor.
- I/O interface module has three registers namely DATA, CONTROL and STATUS.
  - DATA- contains the data value to be written or read by the processor on device
  - STATUS- contains the flag value for read and write data indicator bit
  - CONTROL- receives the control signals sent by processor to the device
- Peripheral devices are mapped to address ranges. If I/O interface is at address A, DATA is at A, CONTROL at A+4 and STATUS at A+8.
- The I/O interface module is connected to the peripheral device and controls it using theses three registers.

2.3 Explanation of I/O peripheral working- reading from a keyboard

Whenever a key is pressed on the keyboard, the circuit places the ASCII encoded character into the DATA register. At the same time, INPUT flag is set to 1. When it detects that INPUT is set to 1, it transfers the contents of DATA into a processor register. Afterwards, INPUT flag is set to 0.

Algorithm 1 Psuedo code to transfer data read from keyboard to processor's R6 register

```
READWAIT        Read INPUT flag
                Jump to READWAIT if INPUT = 0
                Transfer data from DATA to R6
```

- In processors that use memory-mapped I/O, in which some addresses are used to refer to registers in I/O interfaces, data can be transferred between these registers and the processor using instructions such as Load, Store, and Move.

```
LoadByte R5, KBD_DATA
StoreByte R5, DISP_DATA
```

- LoadByte loads the value of KBD_DATA into R5 register
- StoreByte stores the values of R5 register into DISP_DATA

Implementation-

```
READWAIT:       LoadByte R4, KBD_STATUS
                And R4, R4, #2
                Branch_if_[R4]=0 READWAIT
                LoadByte R5, KBD_DATA
```

- KBD_DATA = DATA register of I/O interface for keyboard.
- KBD_STATUS = STATUS register of I/O interface for keyboard.
- KIN = INPUT flag for keyboard. It is a part of the 8 bit register KBD_STATUS.
- The And instruction is used to test the KIN flag, which is bit b_1 of KBD_STATUS in R4.
- As long as $b_1 = 0$, the result of the AND operation leaves the value in R4 equal to zero, and the READWAIT loop continues to be executed.