BUSES IN COMPUTER ARCHITECTURE

The processor, main memory, and I/O devices can be interconnected by means of a common bus whose primary function is to provide a communication path for the transfer of data. The bus includes the lines needed to support interrupts and arbitration. In this section, we discuss the main features of the bus protocols used for transferring data. A bus protocol is the set of rules that govern the behavior of various devices connected to the bus as to when to place information on the bus, assert control signals, and so on. After describing bus protocols, we will present examples of interface circuits that use these protocols.

Synchronous Bus:-

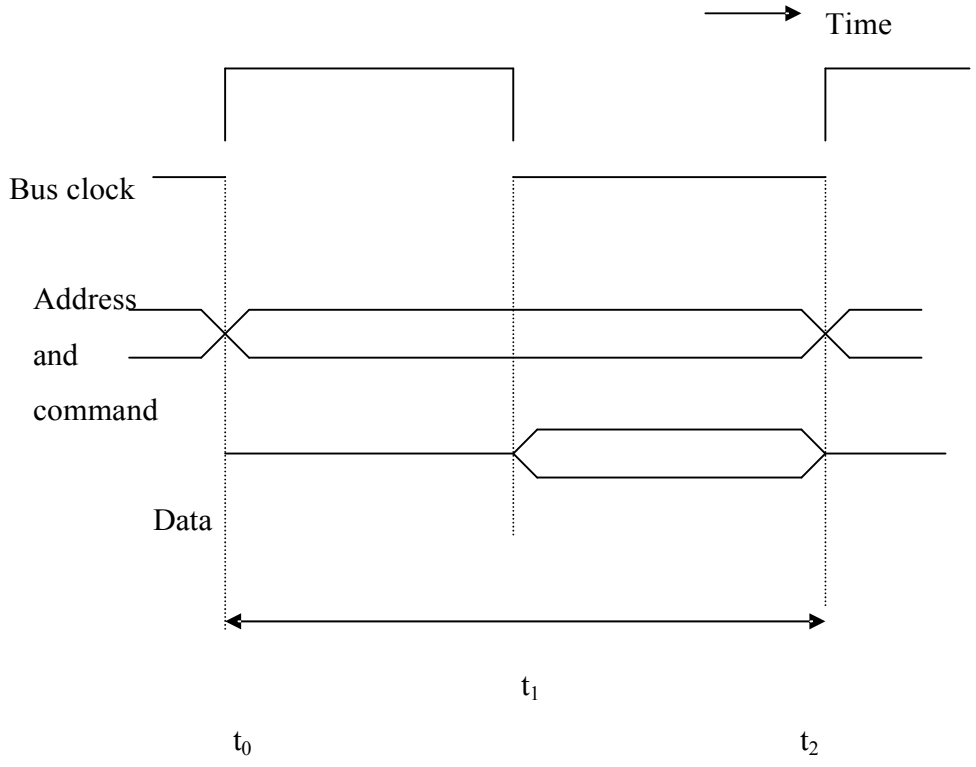
In a synchronous bus, all devices derive timing information from a common clock line. Equally spaced pulses on this line define equal time intervals. In the simplest form of a synchronous bus, each of these intervals constitutes a bus cycle during which one data transfer can take place. Such a scheme is illustrated in figure 7 The address and data lines in this and subsequent figures are shown as high and low at the same time. This is a common convention indicating that some lines are high and some low, depending on the particular address or data pattern being transmitted. The crossing points indicate the times at which these patterns change. A signal line in an indeterminate or high impedance state is represented by an intermediate level half-way between the low and high signal levels.

Let us consider the sequence of events during an input (read) operation. At time t_0, the master places the device address on the address lines and sends an appropriate command on the control lines. In this case, the command will indicate an input operation and specify the length of the operand to be read, if necessary. Information travels over the

bus at a speed determined by its physical and electrical characteristics. The clock pulse width, $t_1 - t_0$, must be longer than the maximum propagation delay between two devices connected to the bus. It also has to be long enough to allow all devices to decode the address and control signals so that the addressed device (the slave) can respond at time t_1. It is important that slaves take no action or place any data on the bus before t_1. The information on the bus is unreliable during the period t_0 to t_1 because signals are changing state. The addressed slave places the requested input data on the data lines at time t_1.

At the end of the clock cycle, at time t2, the master strobes the data on the data lines into its input buffer. In this context, "strobe" means to capture the values of the .

Figure 7 Timing of an input transfer on a synchronous bus.



Bus cycle

Data of a given instant and store them into a buffer. For data to be loaded correctly into any storage device, such as a register built with flip-flops, the data must be available at the input of that device for a period greater than the setup time of the device. Hence, the

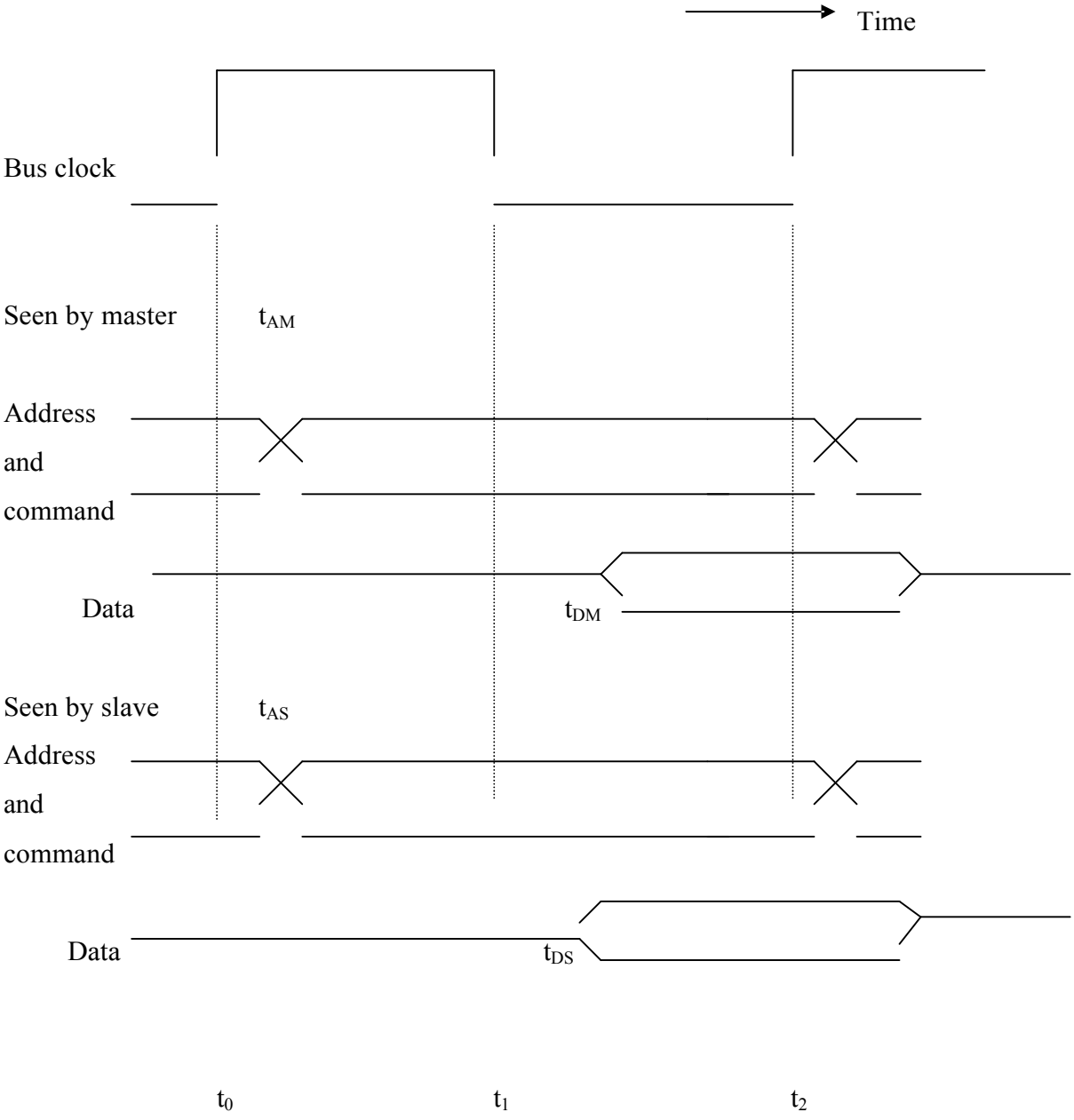
period t_2 - t_1 must be greater than the maximum propagation time on the bus plus the setup time of the input buffer register of the master.

A similar procedure is followed for an output operation. The master places the output data on the data lines when it transmits the address and command information at time t_2, the addressed device strobes the data lines and loads the data into its data buffer.

The timing diagram in figure 7 is an idealized representation of the actions that take place on the bus lines. The exact times at which signals actually change state are somewhat different from those shown because of propagation delays on bus wires and in the circuits of the devices. Figure 4.24 gives a more realistic picture of what happens in practice. It shows two views of each signal, except the clock. Because signals take time to travel from one device to another, a given signal transition is seen by different devices at different times. One view shows the signal as seen by the master and the other as seen by the slave.

The master sends the address and command signals on the rising edge at the beginning of clock period 1 (t_0). However, these signals do not actually appear on the bus until f_{AM}, largely due to the delay in the bus driver circuit. A while later, at t_{AS}, the signals reach the slave. The slave decodes the address and at t1 sends the requested data. Here again, the data signals do not appear on the bus until t_{DS}. They travel toward the master and arrive at t_{DM}. At t2, the master loads the data into its input buffer. Hence the period t_2-t_{DM} is the setup time for the master's input buffer. The data must continue to be valid after t_2 for a period equal to the hold time of that buffer.

Figure 8 A detailed timing diagram for the input transfer of figure 7



Multiple-Cycle transfers:-

The scheme described above results in a simple design for the device interface, however, it has some limitations. Because a transfer has to be completed within one clock cycle, the clock period, t_2-t_0, must be chosen to accommodate the longest delays on the

bus and the lowest device interface. This forces all devices to operate at the speed of the slowest device.

Also, the processor has no way of determining whether the addressed device has actually responded. It simply assumes that, at t_2, the output data have been received by the I/O device or the input data are available on the data lines. If, because of a malfunction, the device does not respond, the error will not be detected.

To overcome these limitations, most buses incorporate control signals that represent a response from the device. These signals inform the master that the slave has recognized its address and that it is ready to participate in a data-transfer operation. They also make it possible to adjust the duration of the data-transfer period to suit the needs of the participating devices. To simplify this process, a high-frequency clock signal is used such that a complete data transfer cycle would span several clock cycles. Then, the number of clock cycles involved can vary from one device to another.

An example of this approach is shown in figure 4.25. during clock cycle 1, the master sends address and command information on the bus, requesting a read operation. The slave receives this information and decodes it. On the following active edge of the clock, that is, at the beginning of clock cycle 2, it makes a decision to respond and begins to access the requested data. We have assumed that some delay is involved in getting the data, and hence the slave cannot respond immediately. The data become ready and are placed on the bus in clock cycle 3. At the same time, the slave asserts a control signal called Slave-ready.

The Slave-ready signal is an acknowledgment from the slave to the master, confirming that valid data have been sent. In the example in figure 9, the slave responds in cycle 3. Another device may respond sooner or later. The Slave-ready signal allows the duration of a bus transfer to change from one device to another. If the addressed device does not respond at all, the master waits for some predefined maximum number of clock

cycles, then aborts the operation. This could be the result of an incorrect address or a device malfunction.

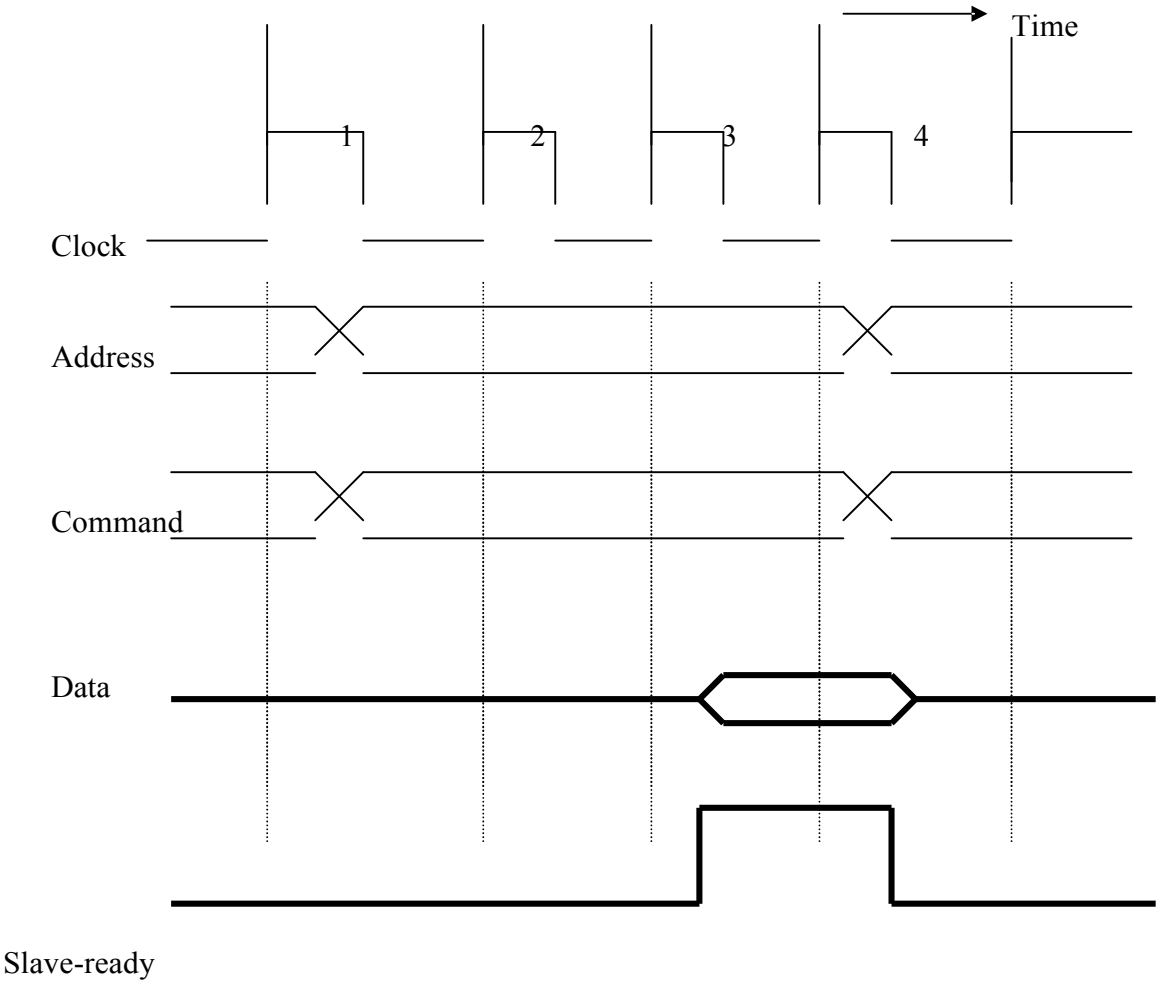


Figure 9 An input transfer using multiple clock cycles.

Source : http://elearningatria.files.wordpress.com/2013/10/cse-iv-computer-organization-10cs46-notes.pdf