Example – Synchronous bus transfer from a master device to a slave device -- a read operation from memory to a CPU.

Assume the bus transfer are coordinated by a by a 100MHz clock. This means that the clock goes through 100 million cycles per second (= 10^6 cycles per second), which in turn means each cycle must last $\frac{1}{10^6}$seconds = $\frac{10}{10^7}$seconds, or 10 nanoseconds. Assume further that a memory read operation requires 15 nsecs from the time the read address value is available until one can assume a stable output value is available.

The overall specifications for the synchronous transfer protocol that we will use are as follows:

- The rising edge of the clock pulse that represents the start of cycle T_1 is the "go" signal for establishing the correct values on the address lines. We will assume that the proper values for the address have been reached after a delay of time $\mathbf{T_{AD}}$.

- While the clock pulse in T_1 is still high (or even earlier for that matter) signals indicating the type of data transfer to take place (a read signal, RD, in this case) and the device involved (a memory request signal, MREQ, for this example) can be activated. The actual realization of these signals will not commence however until the falling edge of the clock pulse for T1 arrives, and then we still experience delays T_{RL} (for the RD signal) and T_M (for the MREQ signal) until the signal values stabilize on their respective lines. Note, some memory designs may also require that the address values be established for a certain time T_{ML} before the memory enable signal is activated. If this is the case, this constraint must be accounted for as well. Such a delay may be necessary, for example so that the address decoders inside main memory have a chance to get the proper address before the enable signal for memory, is activated.

- Although the data to be transferred to the master device (in this example a CPU) may be available earlier, the actual transfer of the data will not commence until the falling edge of the clock pulse of the cycle in which the data values became stable on the data lines. Furthermore we will require that this stability of data values signals must have been achieved at least for time T_{DS} prior to the falling edge.

We now become more specific about the sequences of activities involved in performing a memory read to a CPU over a synchronous bus and their timings.

1. At the "beginning" of a clock cycle T_1, the CPU places an address on the bus' address lines (ADDR)
   - Clock pulses do not change immediately so we will take the midpoint of the rising edge of the clock pulse to represent the beginning point of a clock pulse. Likewise the midpoint of the falling edge will be regarded as the end of the clock pulse. We also assume it takes 1 nsec for the clock pulse to change.
   - The address is not immediately stable on the address lines. There is an "address output" delay which represents how long one must wait until the address signals are stable on the address lines. We denoted this delay above as T_{AD}. If we suppose T_{AD} is at most 4 nsecs, then 4nsecs after the beginning of T_1 we can assume the address lines contain the correct address.
2. The CPU indicates the address on the address lines is a memory address by activating the MREQ line and the RD control lines. The bars over the lines in their specification means activation requires a low (0) signal.
   - Before dropping the signal on the MREQ line from 1 to 0 we shall agree to wait for $\mathbf{T_{ML}}$ nsecs after the address has stabilized.
   - The actual change in signals for MREQ and RD will be triggered by the falling edge of the clock pulse. In order to give the MREQ and RD times to stabilize, however, we must wait for a period of time denoted by T_M and T_{RL}, respectively. For the purposes of this example we shall assume this delay is no more than 3 nsecs for each.
   - In some buses there may be special lines such as MEMR (memory read), MEMW (memory write), IOR (IO device read), IOW (IO device write), etc. for this task that allows one signal line to be used instead of two.
3. Ideally the speed of memory is such that it can have the data available for the CPU, and the CPU can start to store it internally, shortly the falling edge of the next clock cycle T_2 (meaning a new bus operation could commence as early as cycle T_3). This ideal circumstance may not be realizable, however, and the CPU must then be told that the data will not be available during the next cycle and so should not be allowed to store the data on the data lines during that cycle. This signaling is done by activating (with a low value) a $\overline{WAIT}$ signal.
   - In our specific example, by the time all of the setup times and delay times are accommodated, we are close to the rising edge of the next clock pulse T_2. Our assumption is that reading from memory requires 15 nsec from the time

the address values become stable. Since our clock period is 10 nsecs this means the next clock cycle, T_2, will not afford us enough time to complete this memory operation; consequently we need a way to signal the CPU that it will have to wait at least another cycle for the data to be available. This is done by having control signal WAIT drop to 0 from its normal value 1 at the start of T_2. Since the additional clock cycle will afford us more than enough time for memory to place the correct value on the data lines and for the CPU to store it, we can reverse the value of WAIT at the start of T_3 so the CPU knows it can store the value on the data lines during cycle T_3.

4. Memory (the slave device) reads data from the appropriate word and places it on the bus' data lines so the CPU can access it and store it. The timing specifications are that the CPU can commence storing the value on the falling edge of the pulse beginning cycle T3.
    - Given what we know about our memory's operating times, we also know that the read operation will complete sometime during the first half of T_3.since the read operation commenced prior to the start of T_2. If the timing specification requires that data is to be allowed to enter the CPU on the falling edge of T_3 then we have to know the setup time for the data to stabilize on the bus before the falling edge of T_3. Let us call this time T_{DS} and assume it must be at least 2 nsec. This means that we have to be assured that the data read must be completed 2 nsec or more before the falling edge of the clock pulse in order for this to happen
5. Once the memory data has been read into a CPU register the CPU (master) can deactivate the read and memory request lines.
    - Usually one requires waiting for times T_{MH} and T_{RH} after the time the data has started loading into the CPU before deactivating the READ and MREQ lines. Likewise there should be a delay of T_{DH} seconds after the RD signal has been deactivated before the values on the data lines can be changed again.