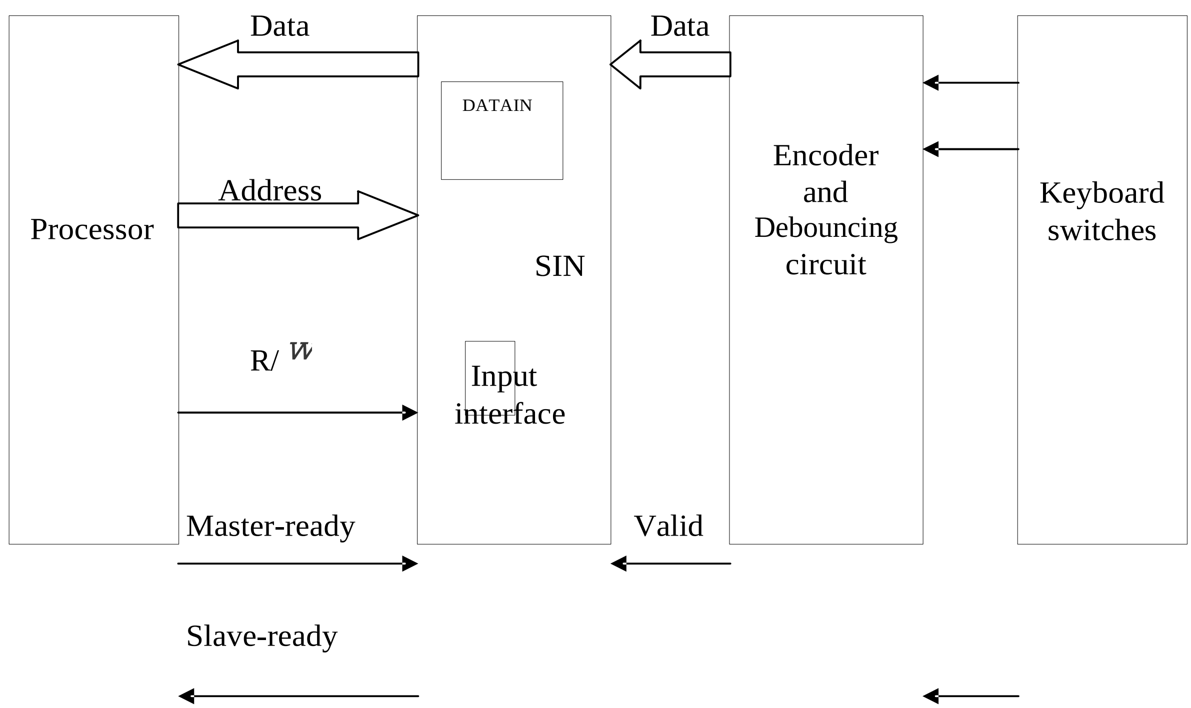
INTERFACE CIRCUITS - I

Parallel port

The hardware components needed for connecting a keyboard to a processor. A typical keyboard consists of mechanical switches that are normally open. When a key is pressed, its switch closes and establishes a path for an electrical signal. This signal is detected by an encoder circuit that generates the ASCII code for the corresponding character.

Figure 11 Keyboard to processor connection.



The output of the encoder consists of the bits that represent the encoded character and one control signal called Valid, which indicates that a key is being pressed. This information is sent to the interface circuit, which contains a data register, DATAIN, and a status flag, SIN. When a key is pressed, the Valid signal changes from 0 to 1, causing the ASCII code to be loaded into DATAIN and SIN to be set to 1. The status flag SIN is cleared to 0 when the processor reads the contents of the DATAIN register. The interface circuit is connected to an asynchronous bus on which transfers are controlled using the handshake signals Master-ready and Slave-ready, as indicated in figure 11. The third control line, R/ W distinguishes read and write transfers.

Figure 12 shows a suitable circuit for an input interface. The output lines of the DATAIN register are connected to the data lines of the bus by means of three-state drivers, which are turned on when the processor issues a read instruction with the address that selects this register. The SIN signal is generated by a status flag circuit. This signal is also sent to the bus through a three-state driver. It is connected to bit D0, which means it will appear as bit 0 of the status register. Other bits of this register do not contain valid information. An address decoder is used to select the input interface when the high-order 31 bits of an address correspond to any of the addresses assigned to this interface. Address bit A0 determines whether the status or the data registers is to be read when the Master-ready signal is active. The control handshake is accomplished by activating the Slave-ready signal when either Read-status or Read-data is equal to 1.

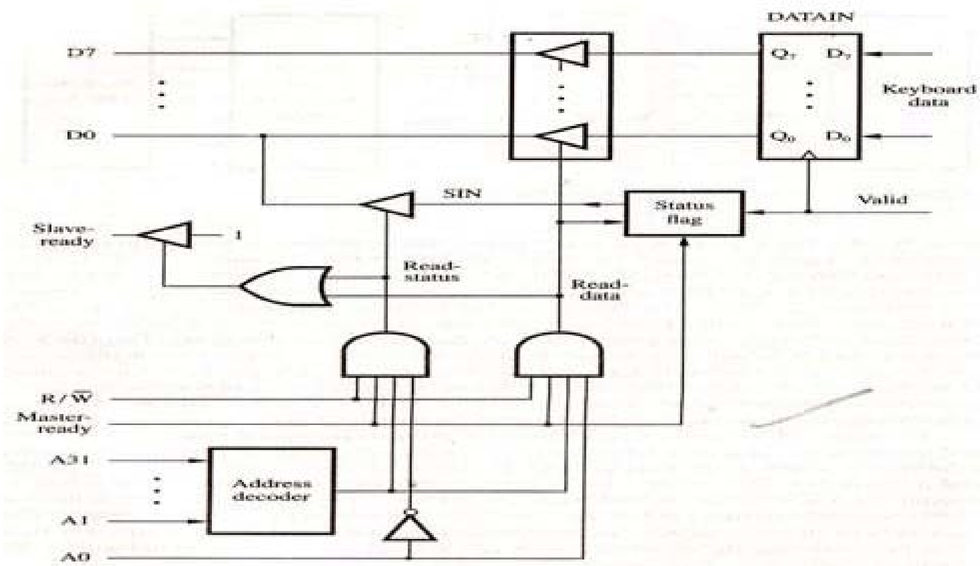


Fig 12 Input interface circuit

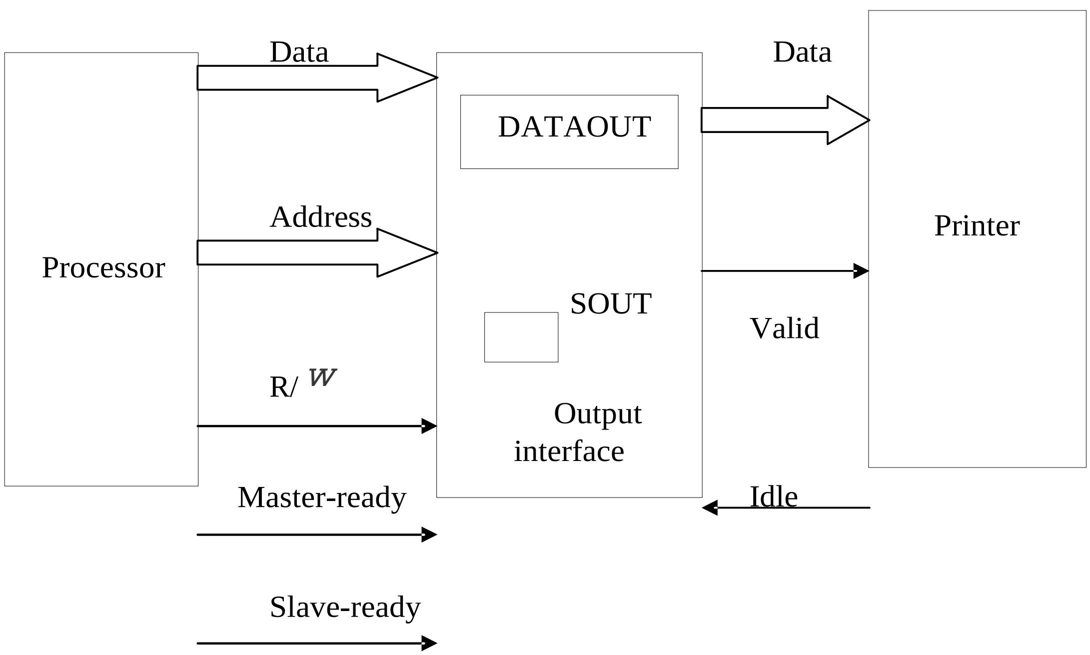


Fig 13 Printer to processor connection

Let us now consider an output interface that can be used to connect an output device, such as a printer, to a processor, as shown in figure 13. The printer operates under control of the handshake signals Valid and Idle in a manner similar to the handshake used on the bus with the Master-ready and Slave-ready signals. When it is ready to accept a character, the printer asserts its Idle signal. The interface circuit can then place a new character on the data lines and activate the Valid signal. In response, the printer starts printing the new character and negates the Idle signal, which in turn causes the interface to deactivate the Valid signal.

The circuit in figure 16 has separate input and output data lines for connection to an I/O device. A more flexible parallel port is created if the data lines to I/O devices are bidirectional. Figure 17 shows a general-purpose parallel interface circuit that can be configured in a variety of ways. Data lines P7 through P0 can be used for either input or output purposes. For increased flexibility, the circuit makes it possible for some lines to serve as inputs and some lines to serve as outputs, under program control. The DATAOUT register is connected to these lines via three-state drivers that are controlled by a data direction register, DDR. The processor can write any 8-bit pattern into DDR. For a given bit, if the DDR value is 1, the corresponding data line acts as an output line; otherwise, it acts as an input line.

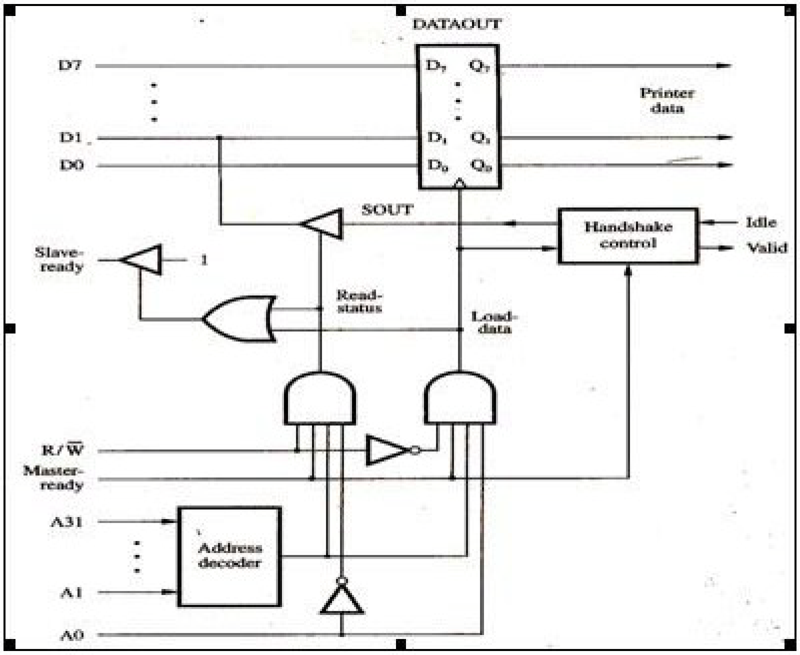


Fig14 Output interface circuit

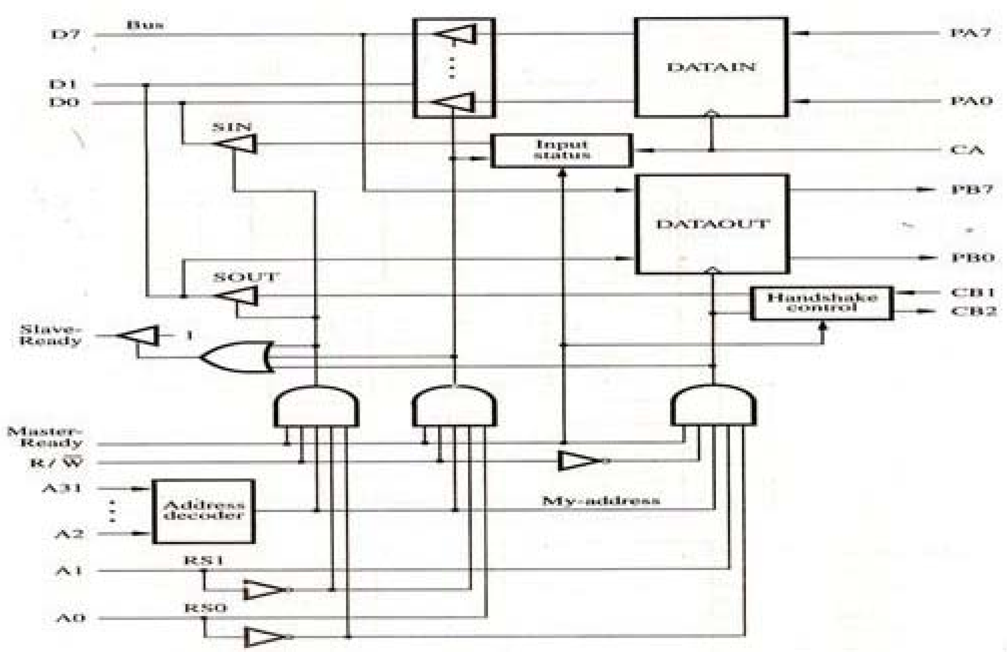


Fig 15 combined input/output interface circuit

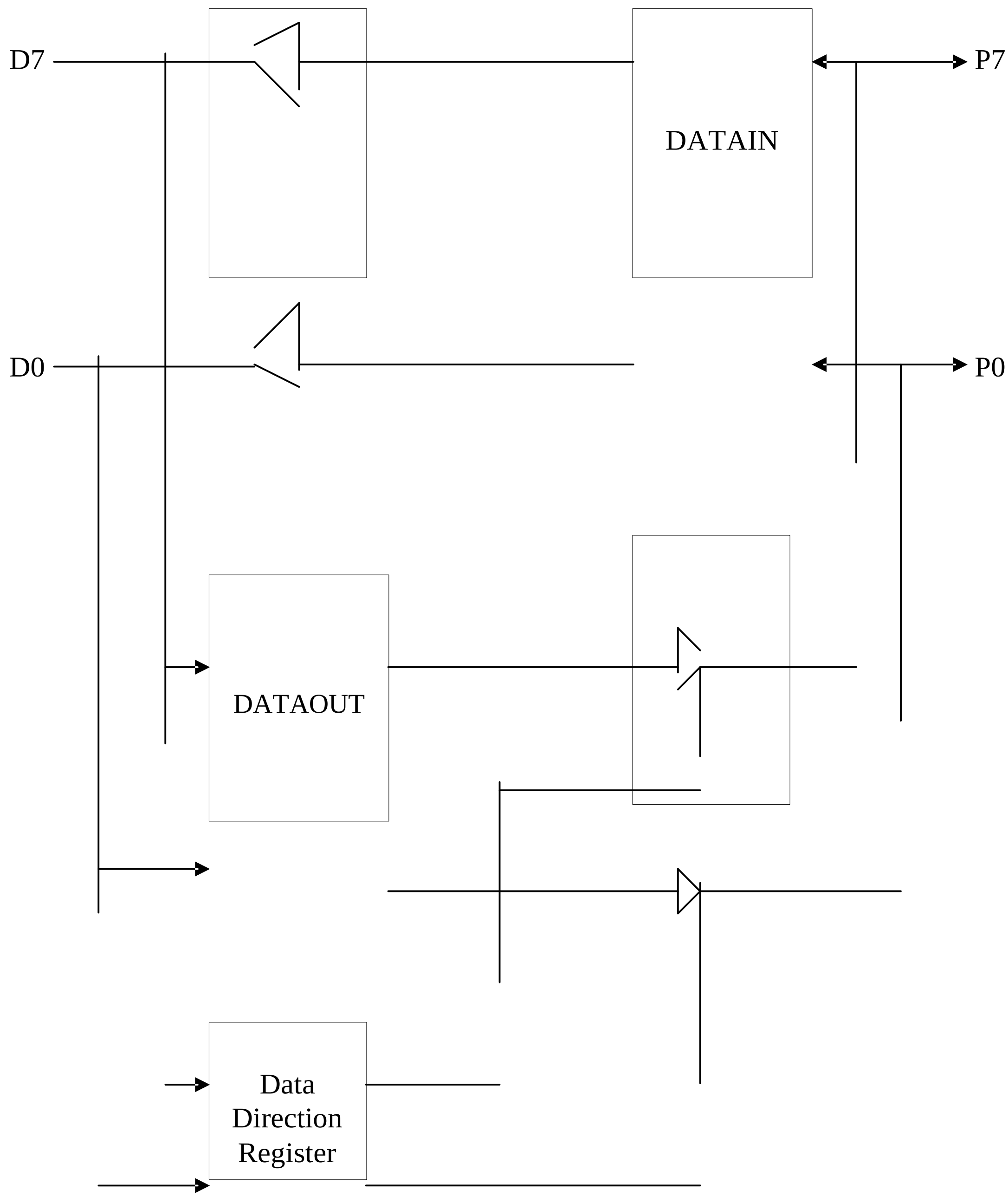


Fig 16 A general 8-bit interface

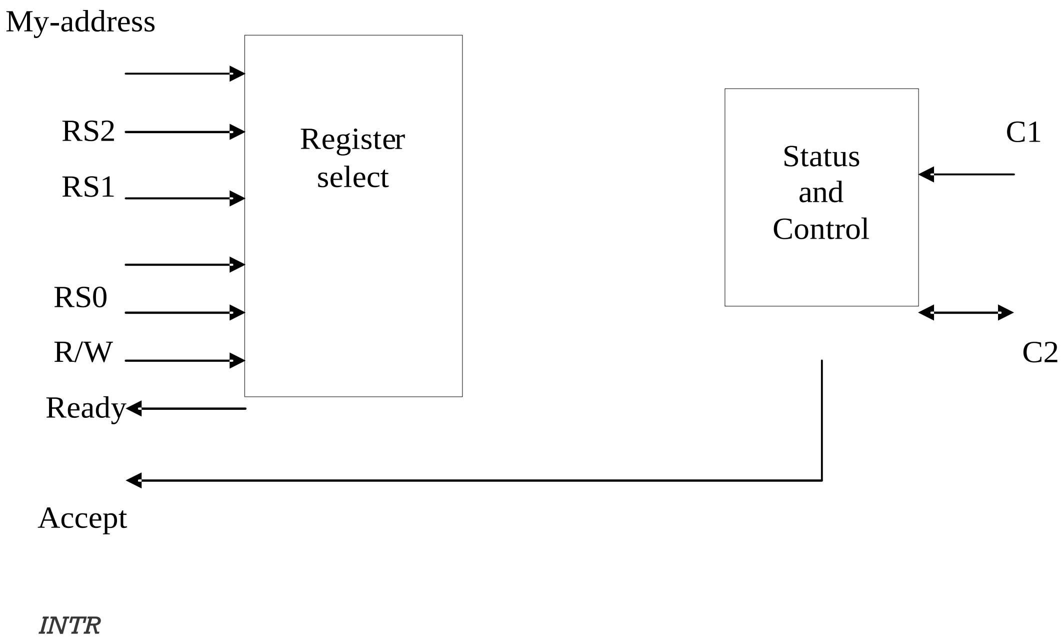


Fig 16 A general 8-bit parallel interface

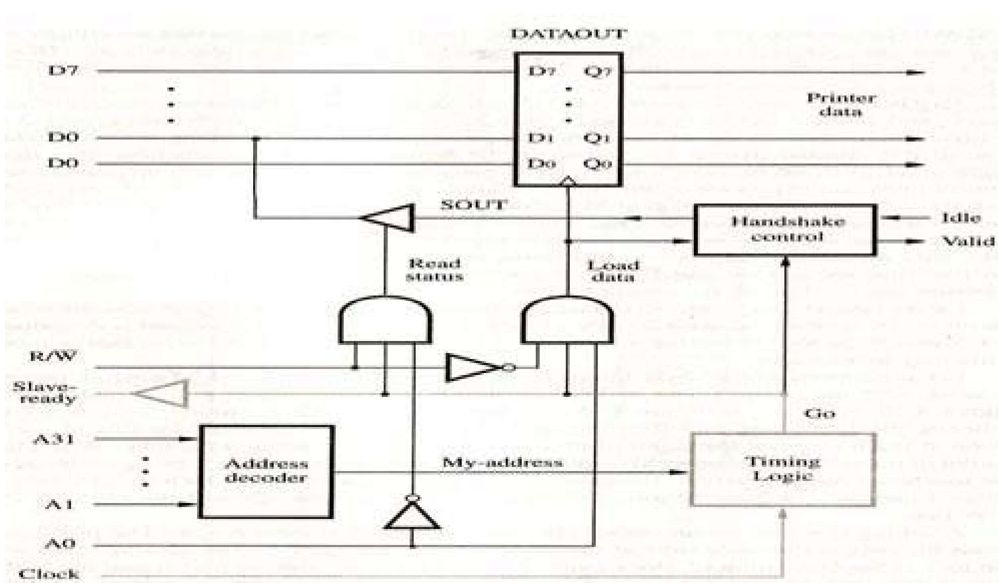


Fig 17 A parallel port interface for the bus

Fig 18 State diagram for the timing logic.

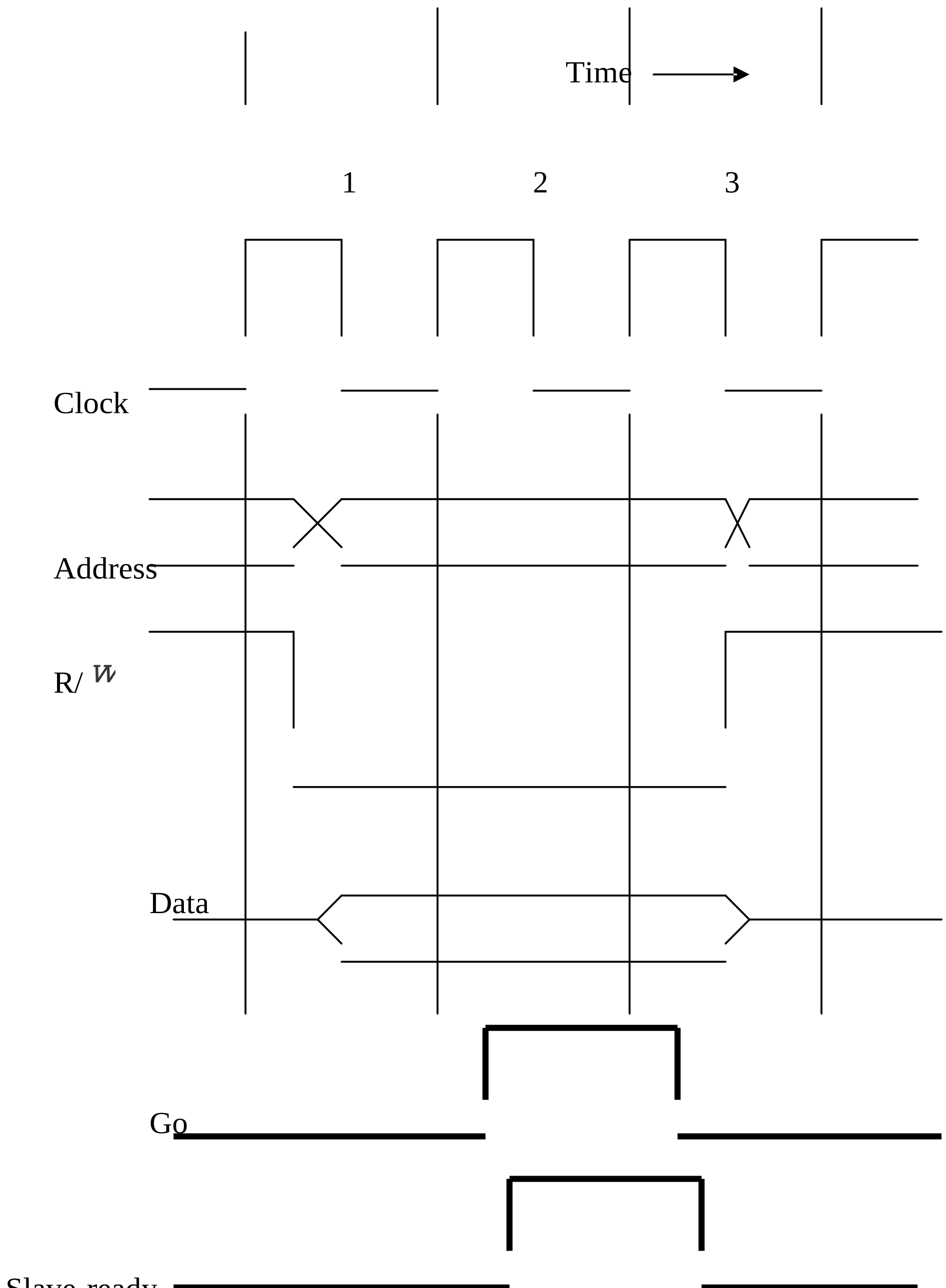


Figure19 Timing for output interface

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