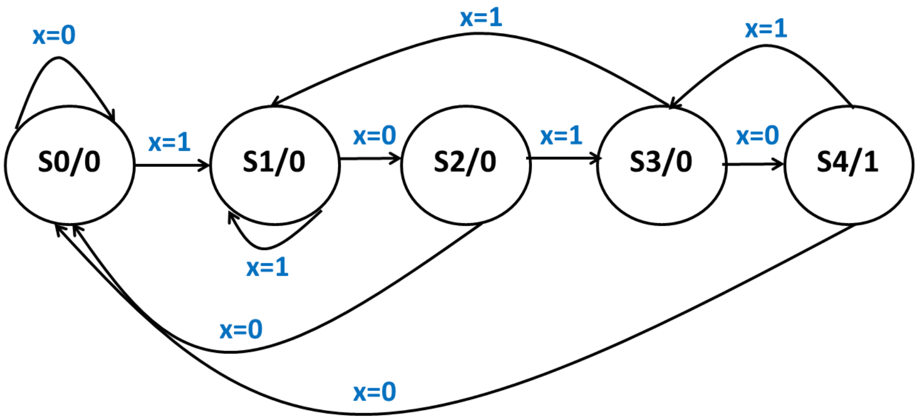
ECE485/585 Homework No. 1 Solution

Problem No. 1

(a) We need 5 states to detect the input pattern. S0 is the initial state. S4 is the state which is reached when the input pattern “1010” is detected. S1, S2 and S3 are intermediate states which indicate that a subsequence of the “1010” has been detected so far.

(b) The state transition diagram for the FSM is as follows. The variable “x” represents the input pattern bits. The output “z” becomes “1” when the state machine is in state S4 (pattern “1010” is detected).



(c) To represent 5 states, we need at least three state variables. Let us use state variables y_0, y_1 and y_2 to express each state as a 3-bit number $y_2\,y_1\,y_0$. We use the following state assignment: S0 = 000, S1 = 001, S2 = 010, S3 = 011 and S4 = 100. The resultant state-assigned state table is as follows:

Present State	**Next State**		**Output**
	x = 0	x = 1	**z**
$y_2\,y_1\,y_0$	$y_2^*\,y_1^*\,y_0^*$	$y_2^*\,y_1^*\,y_0^*$	
000	000	001	0
001	010	001	0
010	000	011	0
011	100	001	0
100	000	011	1

The logic expression for output “z” is as follows:

$$z = y_2\, y_1'\, y_0'$$

The logic expressions for next state logic are as follows:

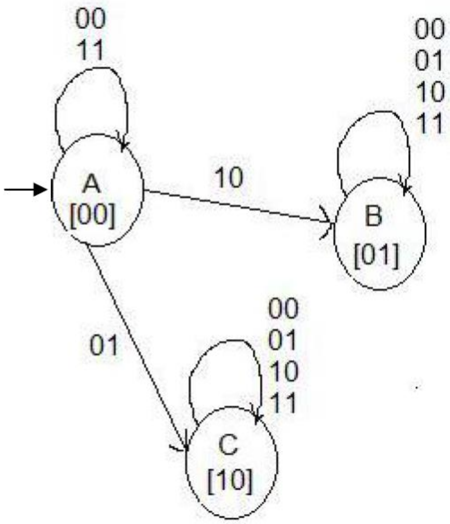
$$y_2^* = x'\, y_2'\, y_1\, y_0$$

$$y_1^* = x'\, y_2'\, y_1'\, y_0 + x\, (y_2'\, y_1\, y_0' + y_2\, y_1'\, y_0') = x'\, y_2'\, y_1'\, y_0 + x\, y_0'\, (y_2 \text{ XOR } y_1)$$

$$y_0^* = x$$

Problem No. 2

The state diagram is as follows:



Problem No. 3

(a) Using the same pins for both address and data reduced the total pin count, which in turn reduced the package cost.

(b) In double handshake, the receiver needs to inform the sender that it is ready to receive data, before the sender is allowed to transmit data. This permission is not needed in single handshake; the receiver only confirms that it has received the data.

(c) Interrupt-driven I/O is more efficient from the CPU’s perspective because it enables the CPU to perform useful work instead of being in a busy-wait loop, while waiting for the peripheral device to be ready.

(d) The efficiency gap between polled I/O and interrupt-driven I/O is higher when the CPU is communicating with a slower I/O device, since more cycles are wasted by polled I/O when waiting for the slower device. In the given example, the efficiency gap will be higher for the HDD as compared to the SSD.

(e) To recover processor state to what it was before the ISR was called. This ensures correct resumption of the interrupted process.

(f) *BHE* stands for “Byte High Enable”. This pin enables the CPU to specify whether the data being transferred on the high order byte data lines (D8 – D15) should be considered valid or discarded.

Problem No. 4

Without a DMA Controller: The device would interrupt the CPU for each I/O transaction.
No. of I/O transactions = Size of data block / Data transferred per each I/O transaction = (8192 * 8) / 32 = 2048
Therefore, number of interrupts = 2048

With a DMA controller: The DMA controller will be in charge of the data transfer. It will generate an interrupt only when the data transfer is complete.

Therefore, number of interrupts = 1