EMBEDDED HARDWARE DESIGN

Experiment 1 **Asynchronous Serial Port** 11/08/2010

This experiment will address the operational features of the programmable serial interface 8251A, often referred to as a USART (Universal Synchronous/Asynchronous Receive/Transmit interface). This chip is used extensively for providing an asynchronous serial port, but not so much as a synchronous interface. In this experiment, therefore, we will concentrate only on the asynchronous port behaviour.

A. Mode, Command and Status bytes

The nomenclature of these bytes and explanation of the bits used in this experiment are given below.

(a) **Mode Byte**, written with C/D' = 1 after Reset:

Number of STOP bits	Parity	Parity Enable	Bits per Character	TxC Divider
00 – Invalid, 01 – 1,	0 – Odd	0 – Disabled	00 – 5, 01 – 6,	00 – Sync, 01 – 1x
10 – 1.5, 11 – 2	1 – Even	1 – Enabled	10 – 7, 11 – 8	10 – 16x, 11 – 64x

Total number of bits in a frame = 1 (START bit) + Bits per character + Parity bit (if enabled) + STOP bits.

(c) **Command Byte**, written with C/D' = 1 after Mode has been configured:

Enter Hunt (EH)	Internal Reset (IR)	Set RTS (RTS)	Error Reset (ER)	Send Break (SBRK)	Enable Rx (RxE)	Set DTR (DTR)	Enable Tx (TxEn)

EH and SBRK are used for synchronous operation. IR, RTS and DTR will not be used in this experiment. ER bit is used to Reset all the three error bits (FE, OE, PE) in the status byte to '0' whenever desired. TxEn and RxE bits are used to enable transmission and reception respectively.

(d) **Status Byte**, read with C/D' = 1:

DSR'	SYNDET/ Break	Framing Error (FE)	Overrun Error (OE)	Parity Error (PE)	TxEmpty	RxRdy	TxRdy

TxRdy ← 0 after data is written into the Tx Buffer through the Data Bus and
TxRdy ← 1 and TxEmpty ← 0 after Tx Buffer contents are loaded into the Parallel-to-Serial Converter.
TxEmpty ← 1 after all the bits are shifted out of the Parallel-to-Serial Converter.
RxRdy ← 1 after received data is loaded into the Rx Buffer from the Serial-to-Parallel Converter.
RxRdy ← 0 after received data is read from the Rx Buffer through the Data Bus.

B. Tri-State Buffer (TSB) Connections

1. Set up an 8-bit Tri-State Buffer (TSB) using two 74LS125 ICs, with the TSB input connected to Input Switches IP1-IP8, a common control $C' = C_3' = C_2' = C_1' = C_0'$ for both the chips applied through Input Switch IP9, and the TSB output connected to the LED Display Points in the order shown in Fig. 1.1. The reason for connecting the LEDs in this strange manner at this stage is that DP7 and DP8 will later be disconnected from the TSB output, leaving only DP1-DP6 connected to the lower 6 bits.

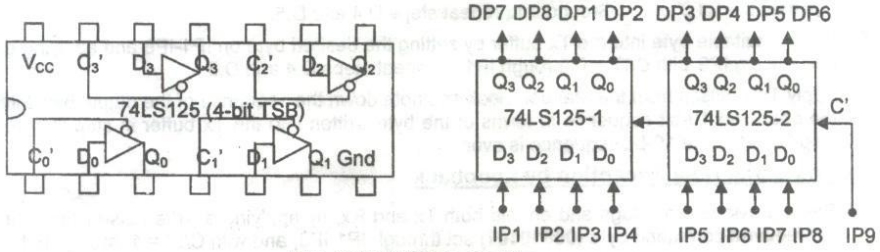

Fig. 1.1 Tri-State Buffer

2. Verify that the TSB output is the same as the TSB input when C' is LOW, and that the TSB output remains high-impedance, with none of the LEDs glowing, when C' is HIGH.
3. Now remove the connections from the TSB to DP7 and DP8, so that these two LEDs can be used to display other outputs of interest during the experiment.

C. 8251A (USART) Connections

1. Connect the **TxC'** and **RxC'** inputs (transmission and reception clocks of 8251A) together to the Manual Clock output CLK-M, and the **CLK** input of the 8251A to the output of the Function Generator, with its ***output mode set as TTL*** and its frequency set at 1 MHz.
2. Connect the **WR'** input of the 8251A to the common control input **C'** of the TSB, and the **RD'**, **C/D'** and **Reset** inputs of the 8251A to IP10, IP11 and IP12 respectively. Connect the **CS'** (chip select) input to Ground. The **Reset** input has to be kept LOW for normal operation.
3. Connect the $D_{7\text{-}0}$ bits (Data Bus) of the 8251A to the TSB output in the order – $D_{7\text{-}4}$ to $Q_{3\text{-}0}$ of 74LS125-1 and $D_{3\text{-}0}$ to $Q_{3\text{-}0}$ of 74LS125-2. Thus either the **Mode** or the **Command** byte can be written into the 8251A from IP1-IP8 through the data bus by making ***C' and WR' simultaneously LOW*** using IP9. Note that only the lower 6 bits of the STATUS byte are relevant to us and they will be displayed on DP1-DP6 when ***RD' is made LOW with C' HIGH*** using IP10 and IP9.
4. Connect **TxD** to DP7 to display the transmitted data bit by bit, and also to **RxD** to make a LOOPBACK arrangement, where the data written into the Tx Buffer gets transmitted serially out of TxD and into RxD, and then gets assembled in the Rx Buffer and can be read back.

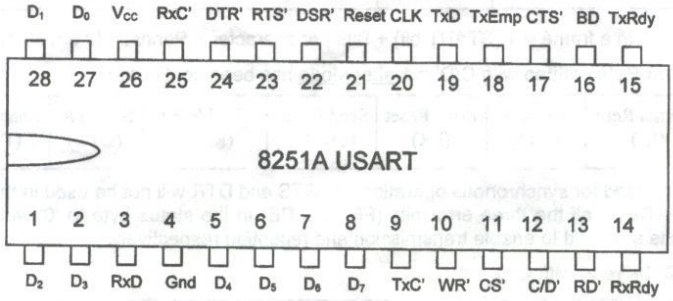

Fig. 1.2 8251A Pin Connections

D. Testing Serial Data Transmission

1. Connect the CTS' pin to V_{CC} so as to disable transmission. Reset the 8251A by applying a positive pulse (0→1→0) to the Reset pin through IP12. Connect the TxRdy pin to DP8.
2. Program the 8251A for asynchronous data transmission with 2 STOP bits, even PARITY, 6 BITS/CHARACTER and 1x TxC Divider by applying a write pulse (WR' = 1→0→1) using IP9, with the required mode byte **(11110101)** set through IP1-IP8 and with C/D' = 1 through IP11.
3. Reset previous error flags and enable Tx only, by applying a write pulse using IP9, with the required command byte **(00010001)** set through IP1-IP8, and with C/D' = 1 through IP11.
4. Read the Status byte (lower 6 bits only) by applying a read pulse (RD' = 1→0→1) using IP10, with C/D' = 1 through IP11. Note down the Status bits and interpret their observed values.
5. Compare the observed value of the TxRdy bit with the logic level at the TxRdy pin (= TxRdy bit•CTS•TxEn), observed through DP8.
6. Connect the CTS' pin to Ground and repeat steps D.4 and D.5.
7. Write a suitable byte into the Tx buffer by setting the desired byte on IP1-IP8 and applying a write pulse using IP9 with C/D' = 0 through IP11. Repeat steps D.4 and D.5.
8. Apply TxC pulses from the Manual Clock and note down the sequence of the output bits. Interpret the observed 10-bit sequence in terms of the byte written into the Tx buffer in step D.7. Repeat step D.4 after the 10-bit sequence is over.

E. Testing Serial Data Reception by Loopback

1. Reset previous error flags and enable both Tx and Rx, by applying a write pulse using IP9, with the required command byte **(00010101)** set through IP1-IP8, and with C/D' = 1 through IP11.
2. Disconnect TxRdy and connect RxRdy pin to DP8 and repeat steps D.7 and D.8, noting RxRdy pin through DP8 after every Manual Clock pulse. Interpret the observed values of the Status bits.
3. Read Rx Buffer, applying a read pulse using IP10, with C/D' = 0 through IP11. Repeat step D.4.
4. Repeat steps E.1, E.2 and E.3, inserting wrong STOP/DATA/PARITY bits through RxD, disconnecting RxD from TxD, and also allowing Rx Buffer to be overwritten by a second data before the previous one is read. Verify whether the error flags indicate the errors correctly or not.