Nouran Tarek Mahdy

USART

- It stands for **Universal Synchronous Asynchronous Receiver Transmitter**.
- It is sometimes called the Serial Communications Interface or SCI.
- USART module is one of the serial I/O modules for communication interfacing functions with other devices/units.
- It contains all shift registers, clock generators and data buffers needed for serial communication. –
- It can work in synchronous mode, or in asynchronous mode.
- The USART uses two I/O pins to transmit and receive serial data. Both transmission and reception can occur at the same time i.e. 'full duplex' operation.
- The USART can be configured as a full-duplex asynchronous system that can communicate with peripheral devices, such as CRT terminals and personal computers.
- It can also be configured as a half-duplex synchronous system that can communicate with peripheral devices, such as Analog-to-Digital (A/D) or Digital-to-Analog (D/A) integrated circuits, serial EPROM's, and so on

Modes of operations

- the synchronous mode requires that each end of an exchange respond in turn without initiating a new communication.
- Asynchronous operation means that a process operates independently of other processes.

Synchronous Mode vs Asynchronous mode

- It requires both data and a clock. Asynchronous mode requires only data.
- In synchronous mode, the data is transmitted at a fixed rate. In asynchronous mode, the data does not have to be transmitted at a fixed rate.
- Synchronous data is normally transmitted in the form of blocks, while asynchronous data is normally transmitted one byte at a time.
- Synchronous mode allows for a higher data transfer rate than asynchronous mode does, if all other factors are held constant.

Operation

It’s related to the various protocols:

- USARTs in synchronous mode transmits data in frames. In synchronous operation, characters must be provided on time until a frame is complete; if the controlling processor does not do so, this is an "*underrun error*," and transmission of the frame is aborted.
- It’s used either character-oriented or bit-oriented mode.
  - In character (STR and BSC) modes, the device relied on particular characters to define frame boundaries;
  - in bit (HDLC and SDLC) modes earlier devices relied on physical-layer signals.
- A synchronous line is never silent; when the modem is transmitting, data is flowing. When the physical layer indicates that the modem is active, a USART will send a steady stream of padding, either characters or bits as appropriate to the device and protocol.
- To send a byte, the application writes the byte to the transmit buffer, the UART stores received bytes in a buffer. Then the UART can generate an interrupt to notify the application or software can poll the port to find out if data has arrived.

USART Asynchronous Mode

Data transfer happens in the following way:

1. In idle state→

- Data line has logic high (1).
- Data transfer starts with a start bit, which is always a zero.
- Data word is transferred (8 or 9 bit), LSB is sent first.
- Each word ends with a stop bit, which is always high (1).
- Another byte can be sent directly after, and will start also with a start bit before data.

USART parts

- The USART peripheral consists of three main parts:
- Transmitter.
- Receiver.

- Baud generator.

Transmitter

1. The module is enabled by setting the TXEN bit.
2. Data to be sent should be written into the TXREG register.
3. When using 9bit, TX9D must be written before writing TXREG.
4. Byte will be immediately transferred to the shift register TSR after the STOP bit from the pervious load is sent.
5. From there, data will be clocked out onto the TX pin preceded by a START bit and followed by a STOP bit.

Receiver

1. The clock of the receiver is a multiple of the bit rate,
2. Every received bit is sampled at the middle of the bit's time period.
3. The USART can be configured to receive eight or nine bits by the RX9 bit in the RCSTA register.
4. After the detection of a START bit, eight or nine bits of serial data are shifted from the RX pin into the Receive Shift Register, one bit at a time.
5. After the last bit has been shifted in, the STOP bit is checked and the data is moved into the FIFO buffer.
6. Two bytes can be held in the FIFO buffer while a third is being received. 6. RCREG is the output of the two element FIFO buffer. A next start bit can be sent immediately after the stop bit v RCIF: indicates when data is available in the RCREG. - It is read only bit, cleared by hardware when the RCREG register has been read and is empty. - If two bytes have been received, the RCIF bit will remain set until all the data has been read from RCREG.

RECEIVER Errors

It happen during reception:

1. **Overrun error:** When the buffer is full, and a third byte is received in the RSR register. When sampling the STOP bit of this third byte, OERR (RCSTA<1>) will be set and this word will be lost and transfer from the RSR register to the RCREG register is inhibited. This bit is cleared in software.
2. **Framing error:** occurs when the receiver does not detect the STOP bit at the expected time. This happens because the receiver and the transmitter operates on different baud rates.

Baud Rate Generator

1. Each bit is received/transmitted using a clock that is a multiple of the bit rate (x16 or x64). - So, we have to get this clock frequency from the oscillator frequency (Fosc) using the baud rate generator.
2. It's done by dividing Fosc by a programmable timing device to synchronize the bit duration for both the transmitter and the receiver.
3. BAUD RATE GENERATOR supports both the Asynchronous and Synchronous modes.