UART

1. **UART:**

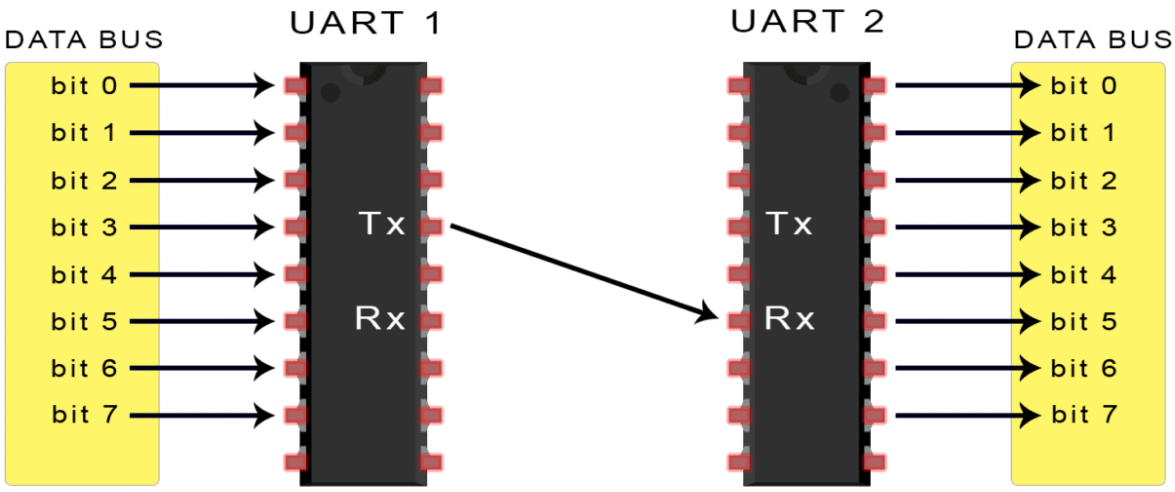
   Universal Asynchronous Receiver Transmitter

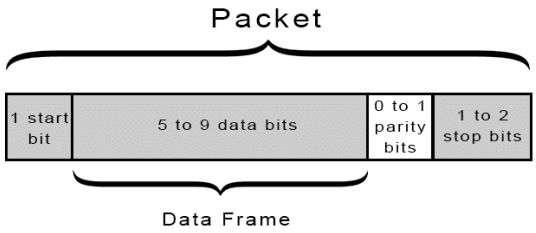
   used for serial communication

   UARTs transmit data *asynchronously*, which means there is no clock signal to synchronize the output of bits from the transmitting UART to the sampling of bits by the receiving UART. Instead of a clock signal, the transmitting UART adds start and stop bits to the data packet being transferred. These bits define the beginning and end of the data packet so the receiving UART knows when to start reading the bits.

   Since there is no clock signal, baud rate of transmitting device should be know and match with receiving one.

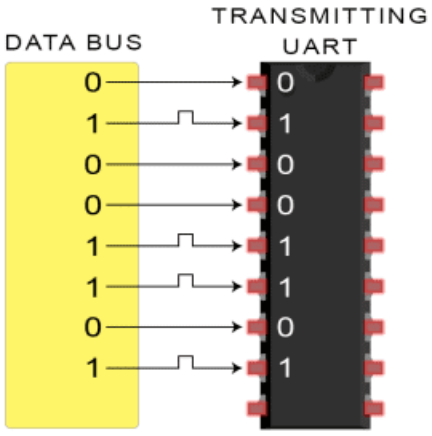
Wires Used	2
Maximum Speed	Any speed up to 115200 baud, usually 9600 baud
Synchronous or Asynchronous?	Asynchronous
Serial or Parallel?	Serial
Max # of Masters	1
Max # of Slaves	1



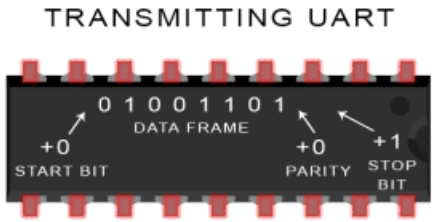


Steps of UART Transmission:

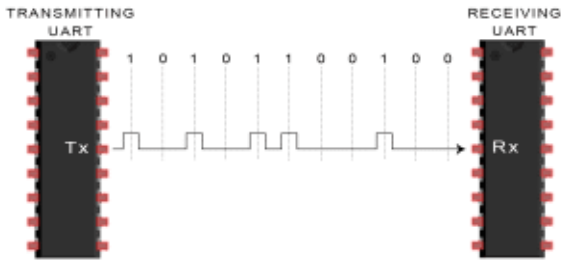
1. The transmitting UART receives data in parallel from the data bus:



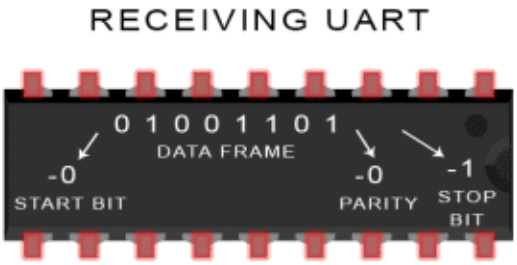
2. The transmitting UART adds the start bit, parity bit, and the stop bit(s) to the data frame:



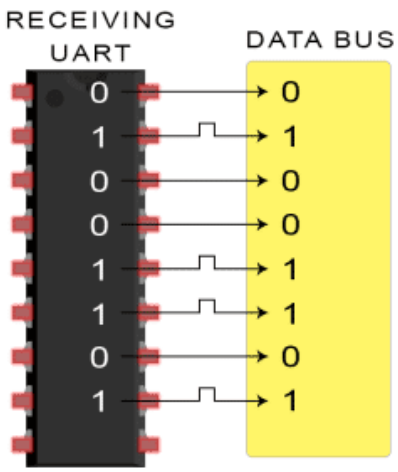
3. The entire packet is sent serially from the transmitting UART to the receiving UART. The receiving UART samples the data line at the pre-configured baud rate:



4. The receiving UART discards the start bit, parity bit, and stop bit from the data frame:



5. The receiving UART converts the serial data back into parallel and transfers it to the data bus on the receiving end:



Serial Control (SCON) SFR(Special Function Registers):

Bit	Name	Bit Address	Explanation of Function
7	SM0	9Fh	Serial port mode bit 0
6	SM1	9Eh	Serial port mode bit 1.
5	SM2	9Dh	Mutliprocessor Communications Enable (However, when SM2 is set the “RI” flag will only be triggered if the 9th bit received was a “1”. That is to say, if SM2 is set and a byte is received whose 9th bit is clear, the RI flag will never be set.)
4	REN	9Ch	Receiver Enable. This bit must be set in order to receive characters.
3	TB8	9Bh	Transmit bit 8. The 9th bit to transmit in mode 2 and 3.
2	RB8	9Ah	Receive bit 8. The 9th bit received in mode 2 and 3.
1	TI	99h	Transmit Flag. Set when a byte has been completely transmitted.
0	RI	98h	Receive Flag. Set when a byte has been completely received.

Additionally, it is necessary to define the function of SM0 and SM1 by an additional table:

SM0	SM1	Serial Mode	Explanation	Baud Rate
0	0	0	8-bit Shift Register	Oscillator / 12
0	1	1	8-bit UART	Set by Timer 1 (*)
1	0	2	9-bit UART	Oscillator / 64 (*)
1	1	3	9-bit UART	Set by Timer 1 (*)

Setting the Serial Port Baud Rate:

Once the Serial Port Mode has been configured, as explained above, the program must configure the serial ports baud rate. This only applies to Serial Port modes 1 and 3. The Baud Rate is determined based on the oscillators frequency when in mode 0 and 2. In mode 0, the baud rate is always the oscillator frequency divided by 12. This means if you are crystal is 11.059Mhz, mode 0 baud rate will always be 921,583 baud. In mode 2 the baud rate is always the oscillator frequency divided by 64, so a 11.059Mhz crystal speed will yield a baud rate of 172,797.

In modes 1 and 3, the baud rate is determined by how frequently timer 1 overflows. The more frequently timer 1 overflows, the higher the baud rate. There are many ways one can cause timer 1 to overflow at a rate that determines a baud rate, but the most common method is to put timer 1 in 8-bit auto-reload mode (timer mode 2) and set a reload value (TH1) that causes Timer 1 to overflow at a frequency appropriate to generate a baud rate.

To determine the value that must be placed in TH1 to generate a given baud rate, we may use the following equation (assuming PCON.7 is clear).

TH1 = 256 – ((Crystal / 384) / Baud)

If PCON.7 is set then the baud rate is effectively doubled, thus the equation becomes:

TH1 = 256 – ((Crystal / 192) / Baud)

For example, if we have an 11.059Mhz crystal and we want to configure the serial port to 19,200 baud we try plugging it in the first equation:

TH1 = 256 – ((Crystal / 384) / Baud)
TH1 = 256 – ((11059000 / 384) / 19200)
TH1 = 256 – ((28,799) / 19200)
TH1 = 256 – 1.5 = 254.5

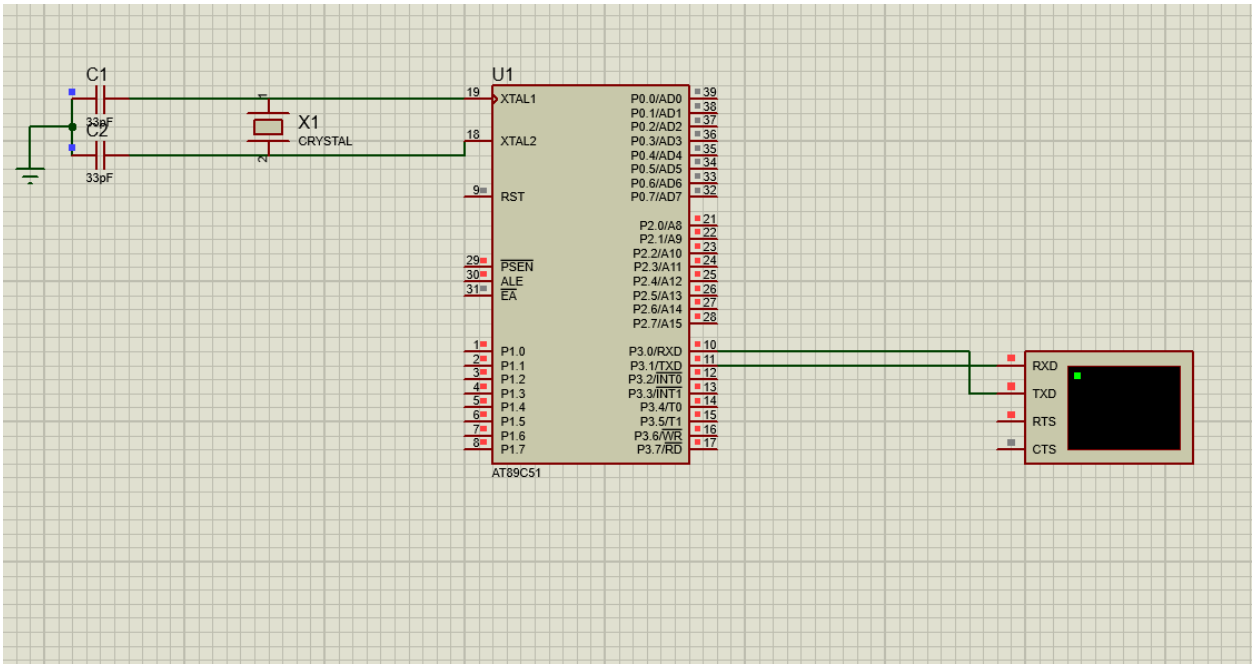
As you can see, to obtain 19,200 baud on a 11.059Mhz crystal wed have to set TH1 to 254.5. If we set it to 254 we will have achieved 14,400 baud and if we set it to 255 we will have achieved 28,800 baud. Thus were stuck…

But not quite… to achieve 19,200 baud we simply need to set PCON.7 (SMOD). When we do this we double the baud rate and utilize the second equation mentioned above. Thus we have:

TH1 = 256 – ((Crystal / 192) / Baud)
TH1 = 256 – ((11059000 / 192) / 19200)
TH1 = 256 – ((57699) / 19200)
TH1 = 256 – 3 = 253

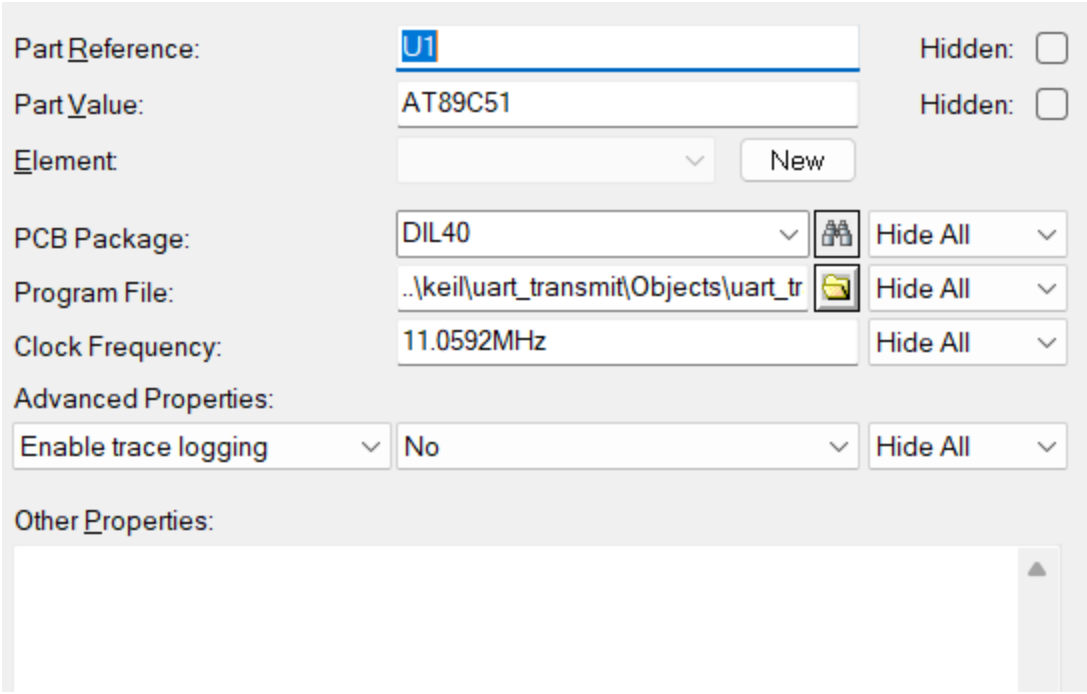
Circuit:

I have implemented by using 8051 controller in **proteus** software for simulation and **keil** for code compiling.

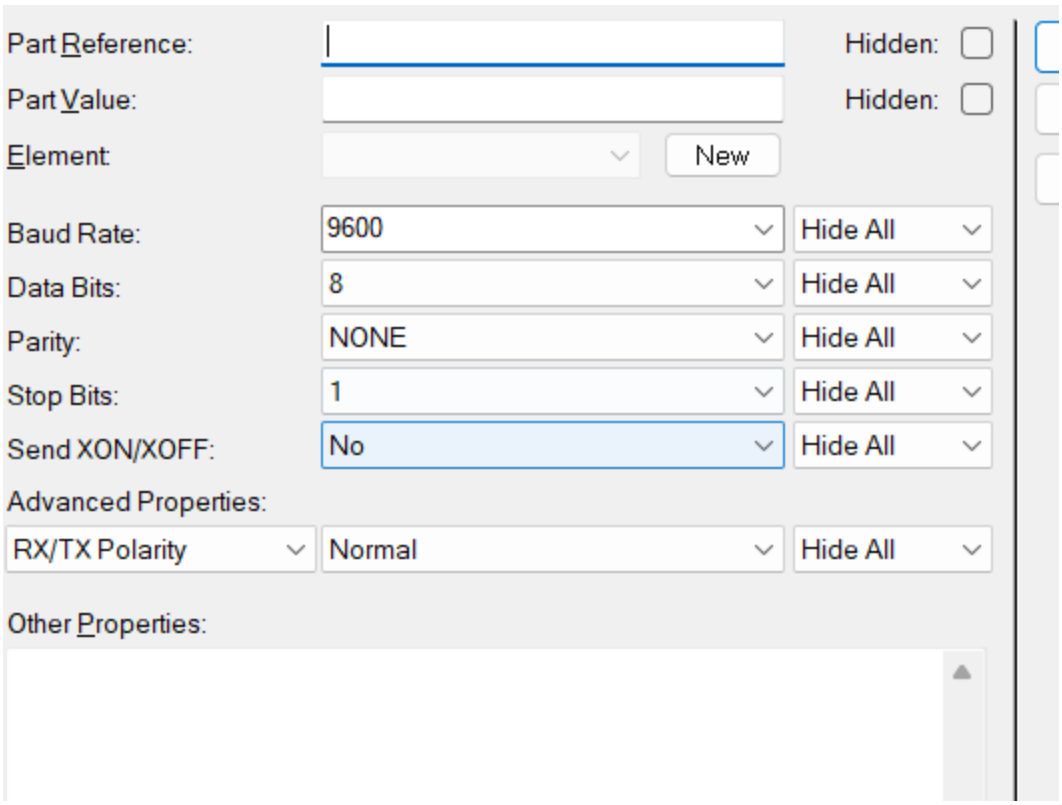


Here, I have used virtual terminal to see the output of transmitted data. Both the baud rate should be same by setting

1. correct clock frequency in controller



2. same baud rate in virtual terminal


Part Reference:
Hidden:
Part Value:
Hidden:
Element:
New
Baud Rate:
9600
Hide All
Data Bits:
8
Hide All
Parity:
NONE
Hide All
Stop Bits:
1
Hide All
Send XON/XOFF:
No
Hide All
Advanced Properties:
RX/TX Polarity
Normal
Hide All
Other Properties: