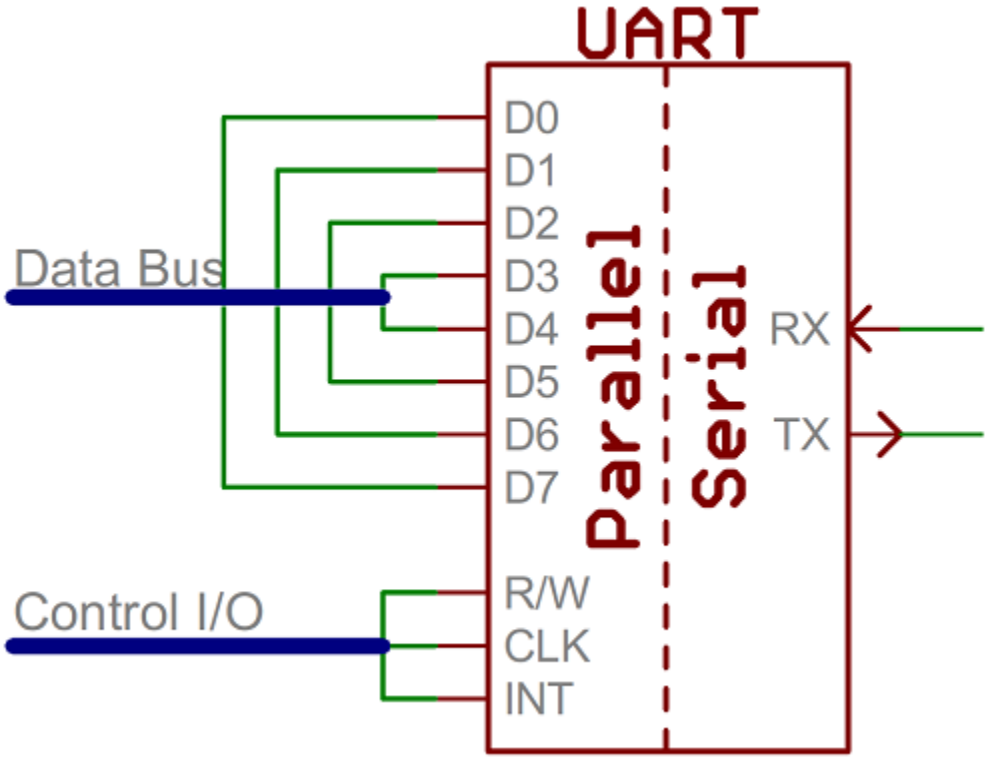
UARTS

The final piece to this serial puzzle is finding something to both create the serial packets and control those physical hardware lines. Enter the UART.

A universal asynchronous receiver/transmitter (UART) is a block of circuitry responsible for implementing serial communication. Essentially, the UART acts as an intermediary between parallel and serial interfaces. On one end of the UART is a bus of eight-or-so data lines (plus some control pins), on the other is the two serial wires - RX and TX.

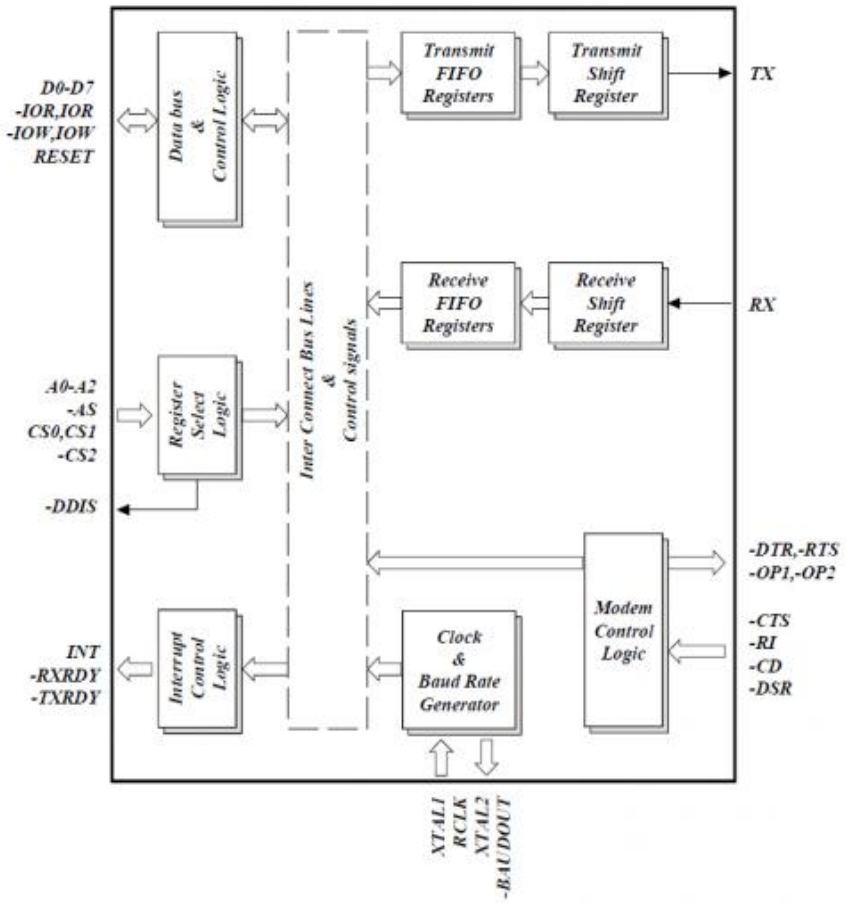


Super-simplified UART interface. Parallel on one end, serial on the other.

UARTs do exist as stand-alone ICs, but they're more commonly found inside microcontrollers. You'll have to check your microcontroller's datasheet to see if it has any UARTs. Some have none, some have one, some have many. For example, the Arduino Uno - based on the "old faithful" ATmega328 - has just a single UART, while the Arduino Mega - built on an ATmega2560 - has a whopping four UARTs.

As the *R* and *T* in the acronym dictate, UARTs are responsible for both sending and receiving serial data. On the transmit side, a UART must create the data packet - appending sync and parity bits - and send that packet out the TX line with precise timing (according to the set baud rate). On the receive end, the UART has to sample the RX line at rates according to the expected baud rate, pick out the sync bits, and spit out the data.

More advanced UARTs may throw their received data into a **buffer**, where it can stay until the microcontroller comes to get it. UARTs will usually release their buffered data on a first-in-first-out (FIFO) basis. Buffers can be as small as a few bits, or as large as thousands of bytes.



Internal UART block diagram (courtesy of the Exar ST16C550 datasheet)

Software UARTs

If a microcontroller doesn't have a UART (or doesn't have enough), the serial interface can be **bit-banged** - directly controlled by the processor. This is the approach Arduino libraries like Software Serial take. Bit-banging is processor-intensive, and not usually as precise as a UART, but it works in a pinch!

Source: https://learn.sparkfun.com/tutorials/serial-communication