Metropolia

Serial communication with UART

- Data is transmitted over a single data line
- Bits are transmitted one after the other in a timed sequence
- Asynchronous transfer does not require a clock signal
- Timing is based on start and stop bits → some overhead in the transmission
- Typically the overhead is 2 bits per transmitted byte
    - To send 8 bits of data we need to transfer 10 bits → data rate calculation is easy for a typical case – just divided bit rate with to get number of bytes per second
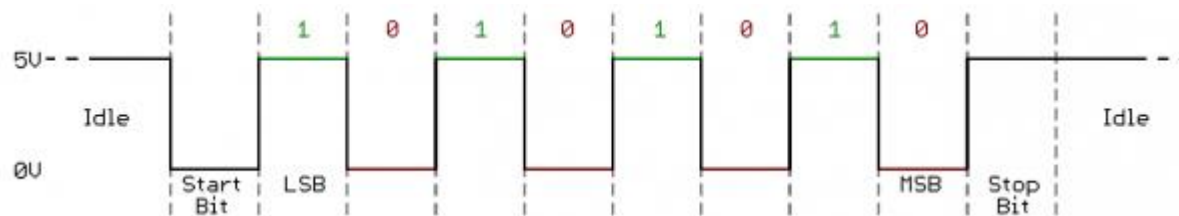
Interface to your favorite OS

- Modern operating systems have standard interfaces for receiving and transmitting data over a serial line
  - Windows calls them COM-ports (COMxx, where xx is a number between 1-255)
  - OS X and linux see them as files (/dev/tty....)
- OS has UART drivers which create a standard interface that is independent of the underlying hardware
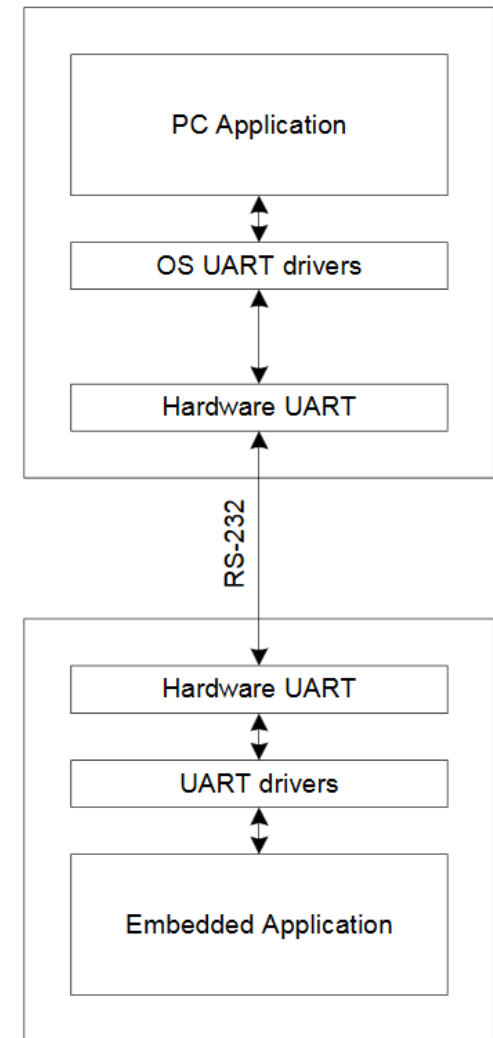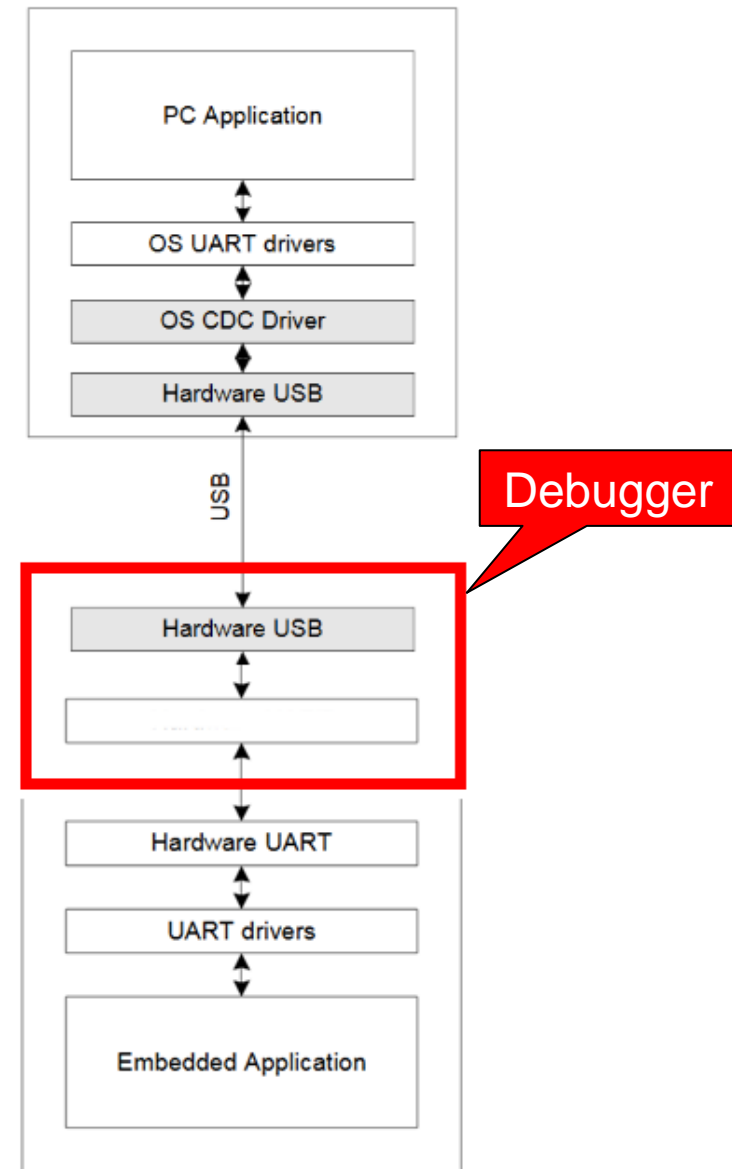
Migrating to USB

- UART interface is ofter replaced with a virtual UART interface where (slow) RS-232 interface has replaced with USB

- Simplicity of UART communication makes it still a viable choice for many embedded applications

- Accessing data through standard interface enables changing the transport method with no changes to the application

- mDraw uses OS UART drivers and does not know (or care) how data transfer take place in the hardware level.

Metropolia
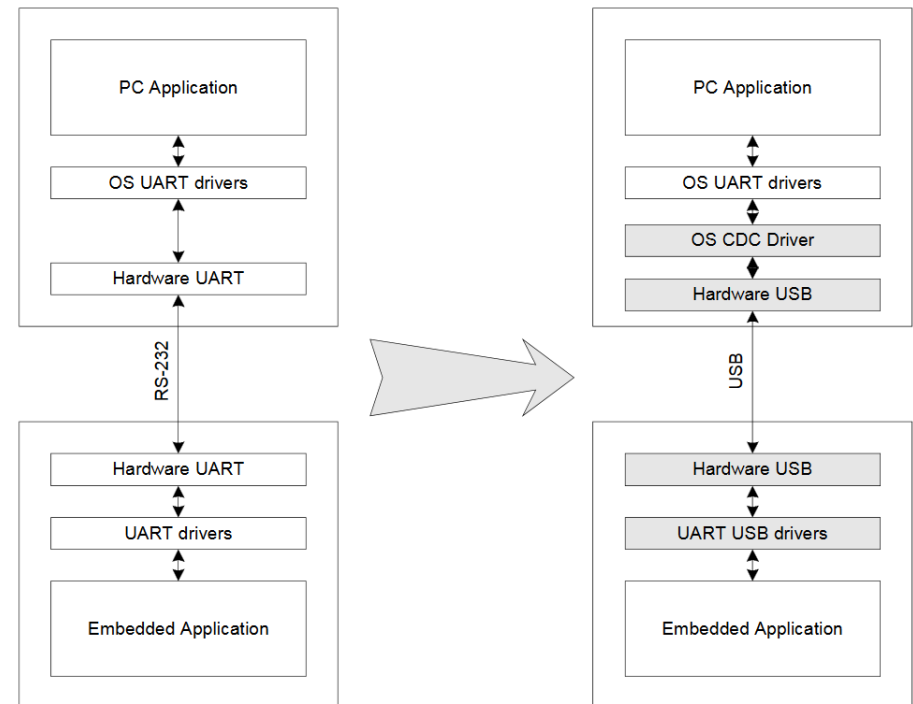
LPCXpresso debug UART

- LPCXpresso debug UART uses LPC1549 built in UART

- Debugger receives the data and sends it to PC using the debugger USB hardware

- MCUXpresso debugger must be running for this to work

- LPC1549 has a "real" USB port that does not require debugger (see previous page for block diagram)

Example block diagram

mDraw
OS UART drivers
USB CDC drivers
USB HW

Something to process G-codes and to control steppers etc.	
Your receiver Task	
	CDC task
FreeRTOS	USB CDC drivers
	USB HW