NEXYS A7 的串口使用

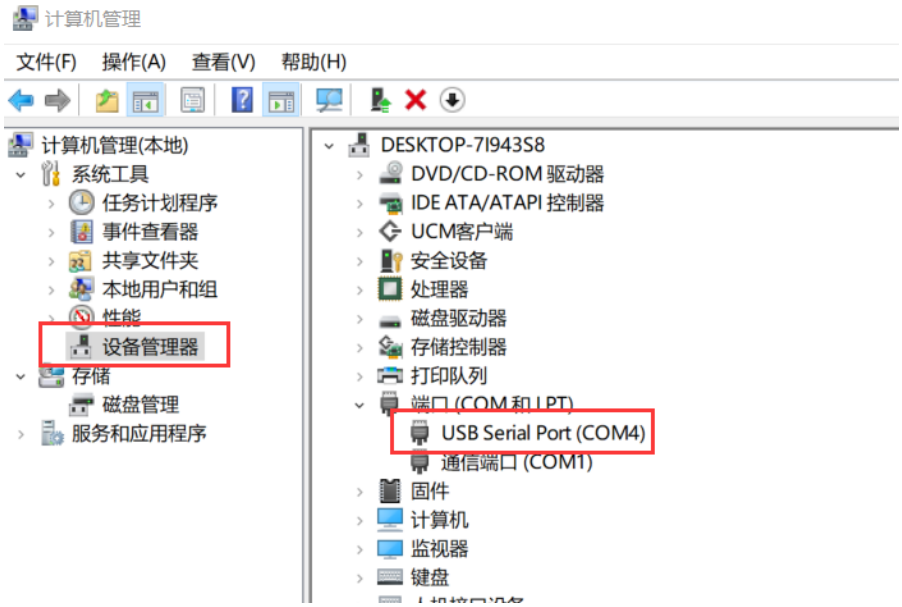
NEXYS A7提供了USB-UART的桥接，这使得我们可以直接通过烧写bitstream的数据线让开发板和电脑通信。事实上，电脑通过 COM 端口和开发板通信，为了方便起见，我们使用MobaXterm这个终端工具。MobaXterm可以在 https://mobaxterm.mobatek.net 免费下载。

NEXYS A7 的 USB-UART Bridge 可参考：

https://digilent.com/reference/programmable-logic/nexys-a7/reference-manual

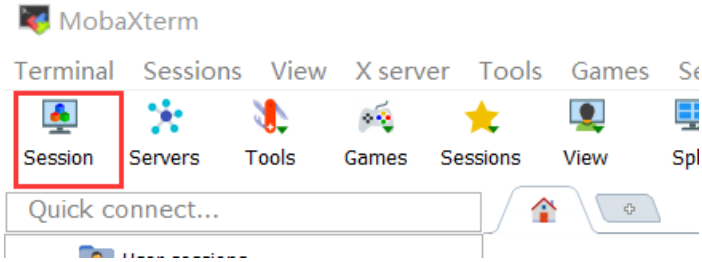
确认连接

和烧写bitstream一样，使用数据线将开发板连接到电脑。打开计算机管理中的设备管理器，等待一段时间后可以在 `端口(COM和LPT)` 看到USB Serial Port，请记住后面的端口号，在下面这张图里是 `COM4`。请注意，在你的设备上不一定是COM4，甚至也可能出现多个USB Serial Port，如果不确定是哪个的话就都试一次。

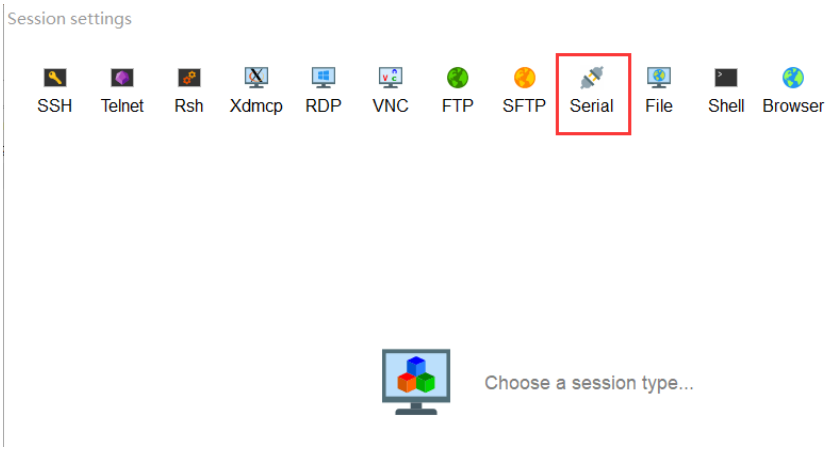


在MobaXterm建立Session

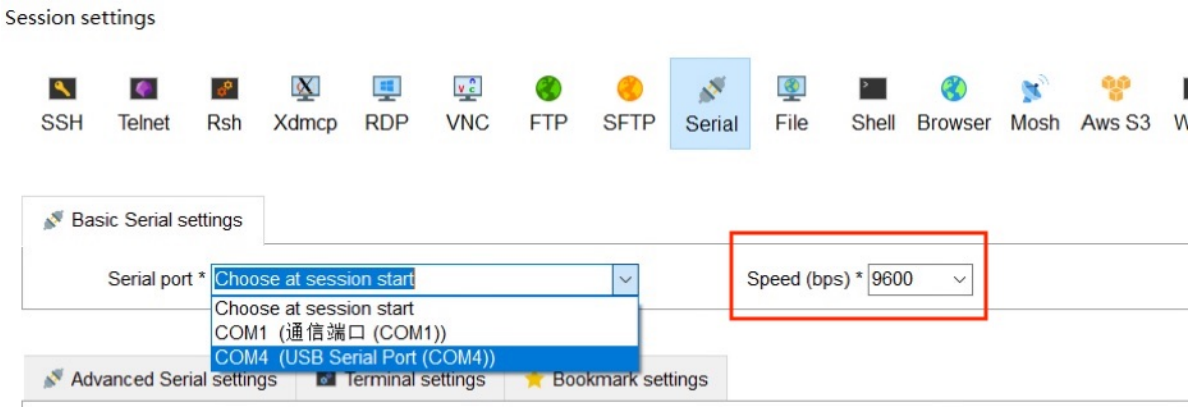
打开MobaXterm，点击Session以新建Session



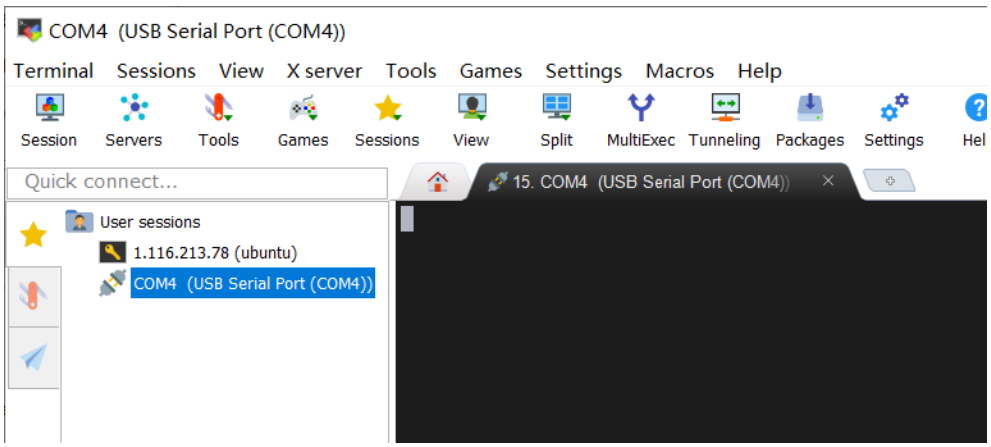
选择session类型为Serial，即串口



然后，配置Serial port为刚才在**设备管理器**中看到的port，在本例中为COM4，并配置Speed（即波特率）为9600bps，其他配置保持默认即可，点击`OK`

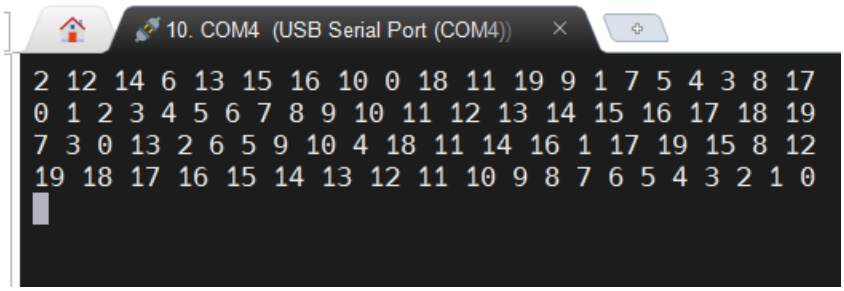


最后，双击左侧导航栏中刚刚建立的Session，即可建立连接



输出结果

现在已经成功建立了连接，那么如何显示测试的输出呢？假设已经完成了实验，直接将bitstream烧到板上，上面的窗口就会显示测试的输出：数组排序前后的值。



如果按了开发板的复位键（CPU reset），会再次输出，但都是经过排序的值。这是因为我们的“RAM”在复位的时候并不会重置，因此存储的还是上次计算的结果。所以如果要再次输出正确结果，就必须**重新烧写bitstream**。

一点提示：实际上，没添加分支预测器时CPU也能运行该测试，因此只要lab2实现正确，在按要求更新代码框架后，就可以运行测试程序并检查串口通信能否正确运作。

串口通信的简单原理

下面的介绍对两种开发板都适用：

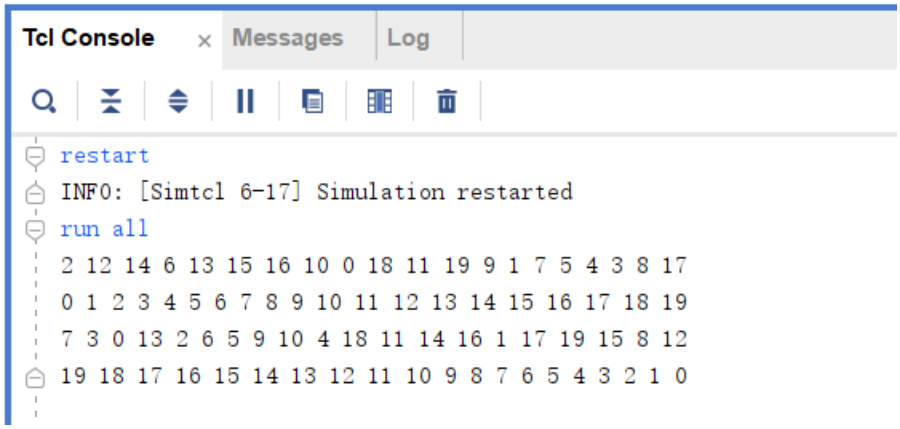
软件层面上，如果看过测试程序，会发现测试程序通过puti/puts两个函数输出数组的元素。通过反汇编代码 `ref/obj/sort-riscv32-cpu.txt` 可以看出，这两个函数实际上会把要输出的字符写入地址 0x10000000。

硬件层面上，按要求更新代码框架后不难发现，RV32core多了两个新的接口：`sim_uart_char_out` 和 `sim_uart_char_valid`。进一步的，这两个信号均从 data_ram 引出。从RAM_B的代码可以看出，在执行写操作（store）时，会判断地址是否为SIM_UART_ADDR（即0x10000000），如果是，则会将数据写入sim_uart_char_out，接下来外设的控制器就会根据sim_uart_char_out和sim_uart_char_valid来控制UART进行串口通信。

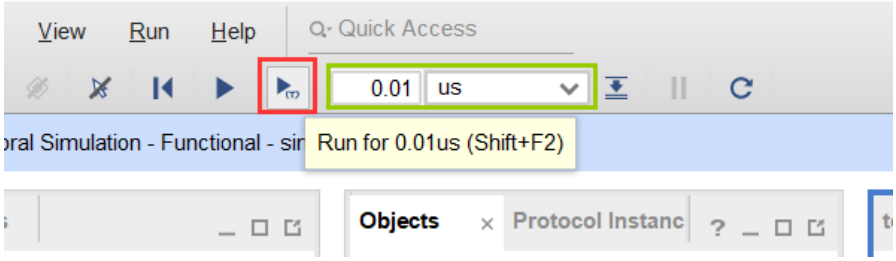
关于仿真调试的补充

如果你比较熟悉仿真的使用，这部分就不用看了

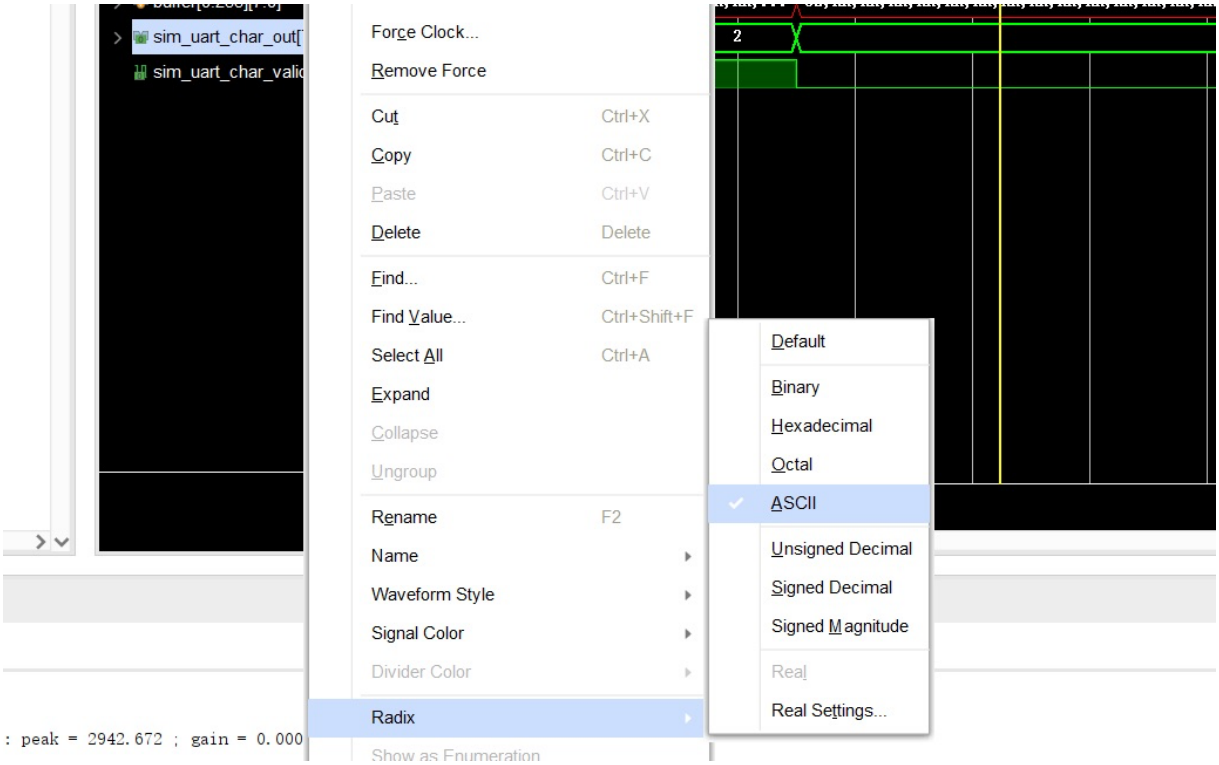
本次实验仿真时，测试程序的输出结果会显示在Tcl Console：



测试程序需要运行较多时钟周期数，因此点一次Run Simulation跑不完，我们可以通过控制下图框出的部分继续仿真。首先在绿框输入想要继续仿真的时间，然后点击红框的 `Run for` 按键就可以继续仿真对应的时间。例如，Run Simulation后仿真了1000ns，只跑了大约500条指令，下面的Tcl Console也没输出结果，显然测试还没跑完，那么我们可以把绿框的值调大（如1us）然后继续仿真。你也可以先点击 `Run for` 左侧的 `Restart` 按键，然后再点击 `Run All`，这样仿真就会一直跑下去，记得在Tcl Console输出结果后或者PC不再变化后暂停仿真。



上面已经介绍，测试程序的输出字符会传递给 `sim_uart_char_out` 这个信号，因此调试仿真的时候也可以通过它来确认测试程序输出的值。在波形图添加这个信号后会以16进制显示，为了更加直观，可以右键这个信号，在Radix中选择ASCII，这样显示的就是要输出的字符了。



如下图，sim_uart_char_valid被拉起，同时sim_uart_char_out的值为2，说明测试程序输出字符 '2' 。