University of California at Berkeley
College of Engineering
Department of Electrical Engineering and Computer Sciences

CS 150 J. Wawrzynek
Fall 1996 N. Weaver

Checkpoint 2
UART

1 Objectives

For this checkpoint, you will

1. Learn how to operate a serial port
2. Gain more practice in digital design
3. Implement the low level serial interface between the computer and your Xilinx board
4. Test your interface.

2 Prelab

You will need to wire wrap the connections for a Maxim 233 CPP adapter chip, which converts TTL level signals (which the Xilinx uses) to RS232 serial port signaling (which uses +/- 12 volts). You should also design both portions of your UART and the test circuit before coming to class.

This checkpoint is where you build all the circuits you need to interface with a computer through a serial port.

3 The Maxim DC-DC converter

The computer uses an RS232 Serial Port, which uses a $+/-10$ volt signaling scheme. Since the Xilinx operates on a 0 and 5 volt scheme, we need to convert between the two voltages. The Maxim 233 provides a convenient, single package way to convert between the two. The R_{in} pins take RS232 signals as input, and output them on the corresponding R_{out} pins. Similarly, the T_{in} pins take a TTL signal (from the Xilinx) and output on the T_{out} pins an RS232 signal.

In order to wire up your Maxim chip, you should connect a T_{in} pin to pin 8 on your Xilinx. One of the R_{out} pins should be connected to pin 6 on your Xilinx. The two C2+ pins should be wired together. The two C2- pins should be wired together. The two V- pins should be wired together. And Vcc and ground should be hooked up normally. The corresponding T_{out} and R_{in} pins will hook up to the serial port through jumper cables.

4 UART

A UART (Universal Asynchronous Receiver/Transmitter) is a device for communicating on a serial port such as RS232. Normally, when there is no data to be sent, the serial line is high. When data is sent, it is transmitted 8 bytes at a time. What first happens is the line goes down for one clock cycle. Then, on each successive clock cycle the data is sent out 1 bit at a time, starting with the least significant bit. Finally, the data line goes high for at least one clock cycle, and stays high until the next byte is sent. Graphically, this is shown in Figure 1.

To send data over a serial line such as this is comparatively easy, one just builds a parallel to serial converter as seen early on in class. The only minor complications are insuring that you first shift out

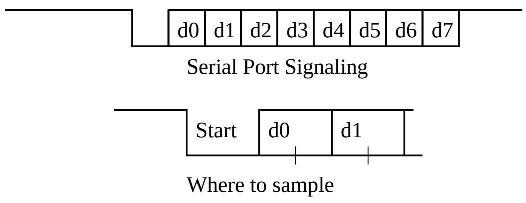

Figure 1: Bits on a serial line

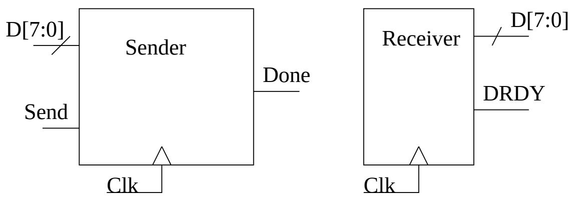

Figure 2: Components for your UART

a 0 to start, and afterwards that you keep outputting 1s. The shift registers in the Xilinx library are a useful primitive.

Receiving data is a little more difficult. The receiver needs to run at 8 times the bit rate of the serial line. It constantly monitors the serial line, and when the line goes low for 4 clock cycles, it assumes that a byte is being sent. Then it records each bit. Since the receiver is operating at 8 times the clock rate of the serial port, it should try to capture the middle of each bit[1]. In order to capture the bits themselves, you can use a different sort of shift register from the Xilinx library. Once all 8 bits are captured, a DRDY signal should be output for a single clock cycle.

The bit rate for the serial port is 115200 bits per second, or 115200 hertz. So, the receiver should operate at 8 times that, or .9216 MHz. Also, to keep things reasonable, the sender should accept a .9216MHz clock, and should adapt internally to output the data at 115.2 kHz.

The receiver component should take a .9216 MHz clock as input, and should have an 8 bit data bus as output and a DRDY signal as output. Internally, it should be connected to the serial port pin. When it receives new data, it should shift it in a bit at a time and, when complete, assert DRDY for a single clock cycle.

The sender component should take a .9216 MHz clock, an 8 bit data bus, and a send signal as input. You might want to have it produce a done signal as output, but you don't have to. When send goes high for 1 clock cycle, the sender captures the data on it's input and then shifts it out to the computer.

For both of these components, most of your logic should consist of shift registers and counters, with some small additional control circuitry. You should think CAREFULLY about your design before implementing it!

In addition, you need to tie these components together in a test rig. The test rig accepts a 14.7456 MHz clock from your crystal occilator (you may need a new crystal on your board. Check with your TA when you are in the lab). It will need to produce a .9216 MHz clock[2] for the rest of your components. When a byte comes in from the computer, it should be displayed on the 7 segment displays by hooking the output of the receiver to the displays. The data input of the sender should be the DIP switches, and when the spare button is pressed, it should send the contents of the dip switches over the serial port to the computer. A small test program will be provided which runs on the computer and monitors the serial port. Instructions on this program will be provided in the newsgroup.

[1] a technique called super-sampling
[2] You can use a counter to divide the clock.

Name: ____________________ Name: ____________________

Lab Section (Check one)

M: ☐11 ☐12 T: ☐13 W: ☐14 ☐15 Th: ☐16

5 Checkoffs

1. MAXIM wire wrapped TA: ____________ (20%)

2. Sender working TA: ____________ (20%)

3. Receiver working TA: ____________ (50%)

4. Turned in on time TA: ____________ (10%)

.... Cut Here ....

Maxim 233

$R2_{out}$	$T2_{in}$
$R2_{in}$	$T1_{in}$
$T2_{out}$	$R1_{out}$
V-	$R1_{in}$
C2-	$T1_{out}$
C2+	GND
V+	VCC
C1-	C1+
V-	GND
C2+	C2-

Maxim 233

$R2_{out}$	$T2_{in}$
$R2_{in}$	$T1_{in}$
$T2_{out}$	$R1_{out}$
V-	$R1_{in}$
C2-	$T1_{out}$
C2+	GND
V+	VCC
C1-	C1+
V-	GND
C2+	C2-

Maxim 233

$R2_{out}$	$T2_{in}$
$R2_{in}$	$T1_{in}$
$T2_{out}$	$R1_{out}$
V-	$R1_{in}$
C2-	$T1_{out}$
C2+	GND
V+	VCC
C1-	C1+
V-	GND
C2+	C2-

Maxim 233

$R2_{out}$	$T2_{in}$
$R2_{in}$	$T1_{in}$
$T2_{out}$	$R1_{out}$
V-	$R1_{in}$
C2-	$T1_{out}$
C2+	GND
V+	VCC
C1-	C1+
V-	GND
C2+	C2-

Maxim 233

$R2_{out}$	$T2_{in}$
$R2_{in}$	$T1_{in}$
$T2_{out}$	$R1_{out}$
V-	$R1_{in}$
C2-	$T1_{out}$
C2+	GND
V+	VCC
C1-	C1+
V-	GND
C2+	C2-