Lab 4 - Serial Output Port

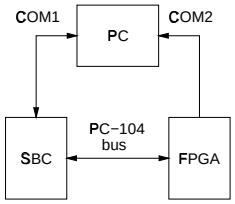
Introduction

In this lab you will design a serial output interface (a "serial port") capable of transmitting 8-bit characters at 9600 bps. The interface has an 8-bit data output port and an 8-bit status input port that are accessible over the PC-104 bus.

The design consists of 7 parts:

- two address decoders
- an 8-bit transmit data register
- an 8-bit status port
- a data bit multiplexer
- a 9600 Hz clock generator
- an 11-state controller

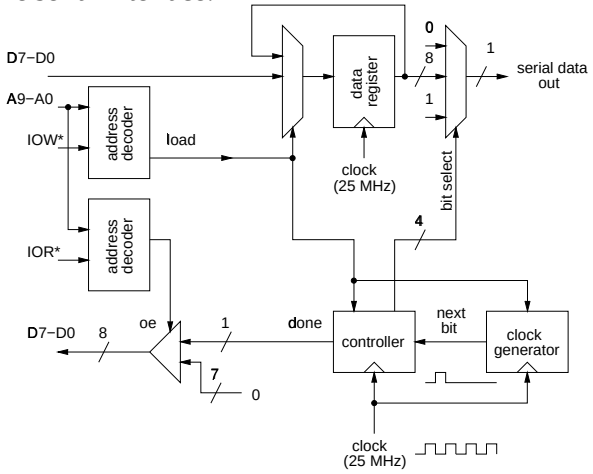
You will also write an 8086 assembly-language program to output a string consisting of your name and student number over this serial port. You will run this program on the lab SBC. Your program will output the string over your serial port and, if everything works correctly, you will see the string displayed on a second terminal emulator program monitoring the lab PC's second (COM2) serial port:



You may be able to re-use the code for the output port, the input port and possibly the clock generator from previous labs. You need not make the different parts of the design into separate components.

Hardware Description

The diagram below shows the internal structure of the serial interface:



All registers are loaded synchronously with the 25 MHz system clock, SYSCLK (see Lab 2).

Address Decoders and Status Port

When IOW* is asserted and the address bus contains 220H the `load` signal is asserted.

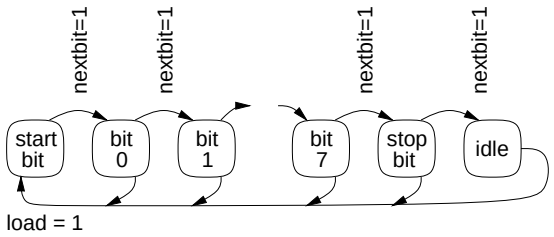
When IOR* is asserted and the address bus contains 221H the `done` signal from the controller is placed on the LS bit of the data bus and the MS 7 bits are set to zero. Otherwise the tri-state bus driver is left in high-impendance mode.

Transmit Data Register

This 8-bit register is loaded with the contents of the data bus when the `load` signal is asserted. Note that this register is loaded synchronously with the 25 MHz system clock.

Controller

The diagram below shows the state transition diagram for the controller and the value placed on the serial output during each state:



The controller's state is set to the "start bit" state when the `load` signal is asserted. The controller state advances to the next state when the `nextbit` signal is asserted.

The `bitselect` signal is set according to the controller state and selects one of the multiplexer inputs. The `done` signal is asserted when the controller is in the idle state.

Pick a state encoding that represents the idle state as an all-zero state register contents to avoid the need for additional reset logic.

Bit Clock Generator

The bit clock generator creates a 9600 Hz signal that is used to advance the controller state. To generate a 9600 Hz signal this circuit counts from $25,175,000/9,600-1=2,621$ down to 0. Note that the `nextbit` output should be asserted on the *last* clock period (i.e. when the count is 0).

The clock generator's counter is reset (to 2621) by the `load` signal in order to synchronize the bit clock to the start of the character transmission.

Serial Data Multiplexer

This 10-to-1 multiplexer sets the serial data output to be a start bit (space, H, VHDL '1'), one of the eight data bits in the transmit data register, or a stop bit or idle level (mark, L, VHDL '0'),

The following values are placed on the serial output:

- a '1' start bit
- one of the eight data bits in order from LS to MS bit, and
- a '0' stop bit

depending on the value of `bitselect`. Note that the RS-232 standard requires that a logic 0 data bit should be output as a high logic level (space, H, VHDL '1'). When no character is being sent the output should be at the idle (mark, L, VHDL '0') level.

The serial output should be connected to FPGA pin 109 (see Lab 2 for location).

Pre-Lab Assignment

Before the lab you must write, assemble and test an 8086 assembly-language program that does the following:

1. initializes a pointer to the first character of an ASCII string consisting of your name and student number
2. returns control to DOS if the current character is zero
3. waits until the status bit indicates the buffer is empty
4. writes the current character to the output port
5. increments the pointer to the next character
6. loops back to step 2

You should come up with a way to test as much of your program as possible. For example, substitute calls to the DOS character input and output functions in place of the port input and output.

You must also design the circuit and test it by simulating its operation. Modify your code to assume the clock frequency is 19200 Hz rather than 25.175 MHz when simulating the operation of your circuit. Your simulation should demonstrate a status read that returns "done" status, a data write, and a second status read returning "not-done". This should be followed by at least 11 clock cycles (after the first write) to view the start bit, the data bits and the stop bit being transmitted for the first value written. Then show a status read that returns "done" status again. Verify that the polarity and bit order are correct.

You will be asked to hand in your VHDL code and assembly listings at the start of the lab.

Lab Procedure

Connect the PC-104 address bus, data bus, IOR* and IOW* to the FPGA pins on the interconnect board

as described in the previous labs. Connect the serial interface cable to the PC's second serial interface (on the rear of the PC) using the cable supplied by the TA. The serial interface connector cables are wired as follows:

Colour Colour	DB-25 pin	Signal	Direction on DTE
Red	2	TxD	Output
White	3	RxD	Input
Black	7	Common	Ground

For this lab you should use the black (ground) and white (signal) lines. Double-check your connections and turn on the power.

Compile your VHDL code if you haven't already done so, and configure the FPGA as described in the previous lab.

Assemble and link your assembly code if you haven't already done so. All of your files should be stored in a subdirectory of the `c:\max2work` directory.

Run two copies of the Windows Hyperterm program. The first should connect to the SBC as usual and be used to download and run your program. The second terminal emulator program should be set for a direct connection to COM2 and a baud rate of 9600 bps so that you can monitor the output of your program. When you run your program you should see your name and student number displayed on the second terminal emulator window.

When your device is working properly ask the TA to check your work. He will make sure your device works as required and ask you one or two questions to verify your understanding of the material.

Report

Submit a short report with a written description of your circuit. Include a block diagram showing the connections between the PC-104 bus, the FPGA and the serial output, a listing of your assembly-language program, the VHDL code and a printout of the simulation waveforms that demonstrate correct operation of your device.

Optional Features

To get bonus marks you can improve the design by adding a control register (at address 221H) that selects the following options:

- different bit rates (e.g. 57600, 38400, ... 300 bps)
- different word lengths (e.g. 5, 6 ... 8 data bits)
- an optional parity bit
- an optional second stop bit