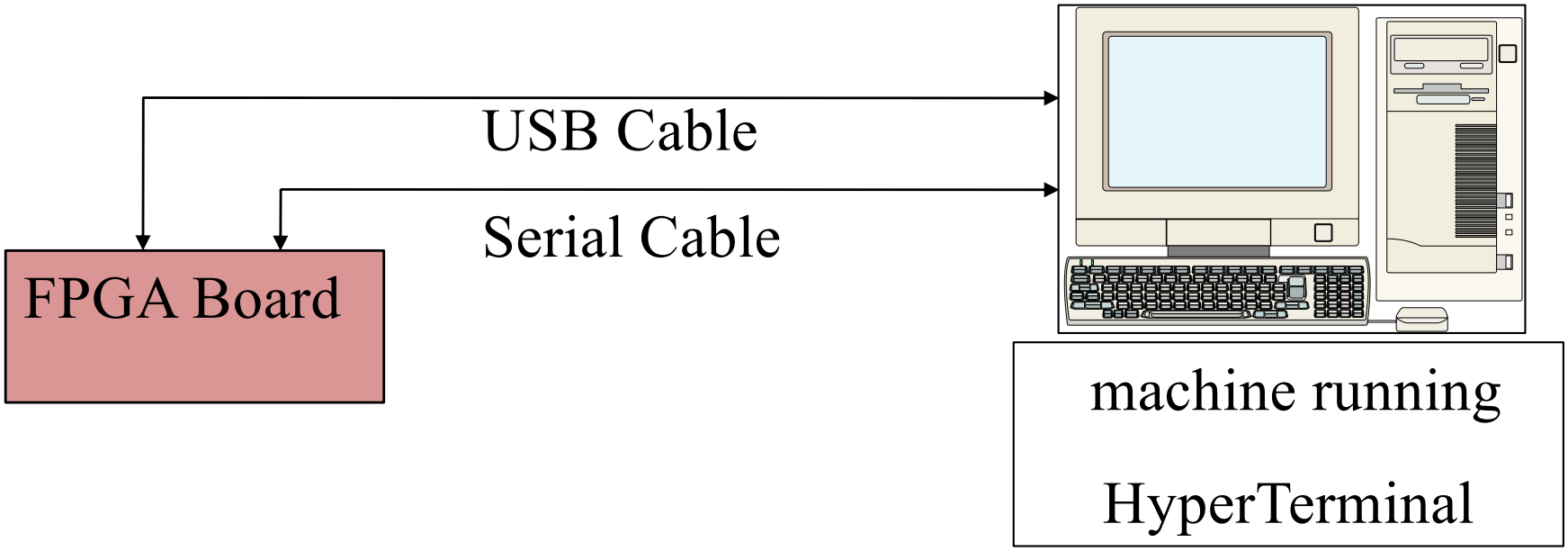
Mini-Lab 1 Overview

Lab Setup



USB port: Configuration download
Serial port: Mini Lab 1

Mini Lab 1

- Design a Special Purpose Asynchronous Receiver/Transmitter (SPART) and its testbench in Verilog
- Simulate the design to ensure correct performance
- Download the design and associated files and demonstrate correct functionality
- Prepare a report on your design

Mini Lab 1 Objectives

- To get familiar with the lab environment prior to the class project
- To get practice using HDL in your designs
- To provide example of basic I/O interface to the class project
- To get experience working with partners

SPART Interface

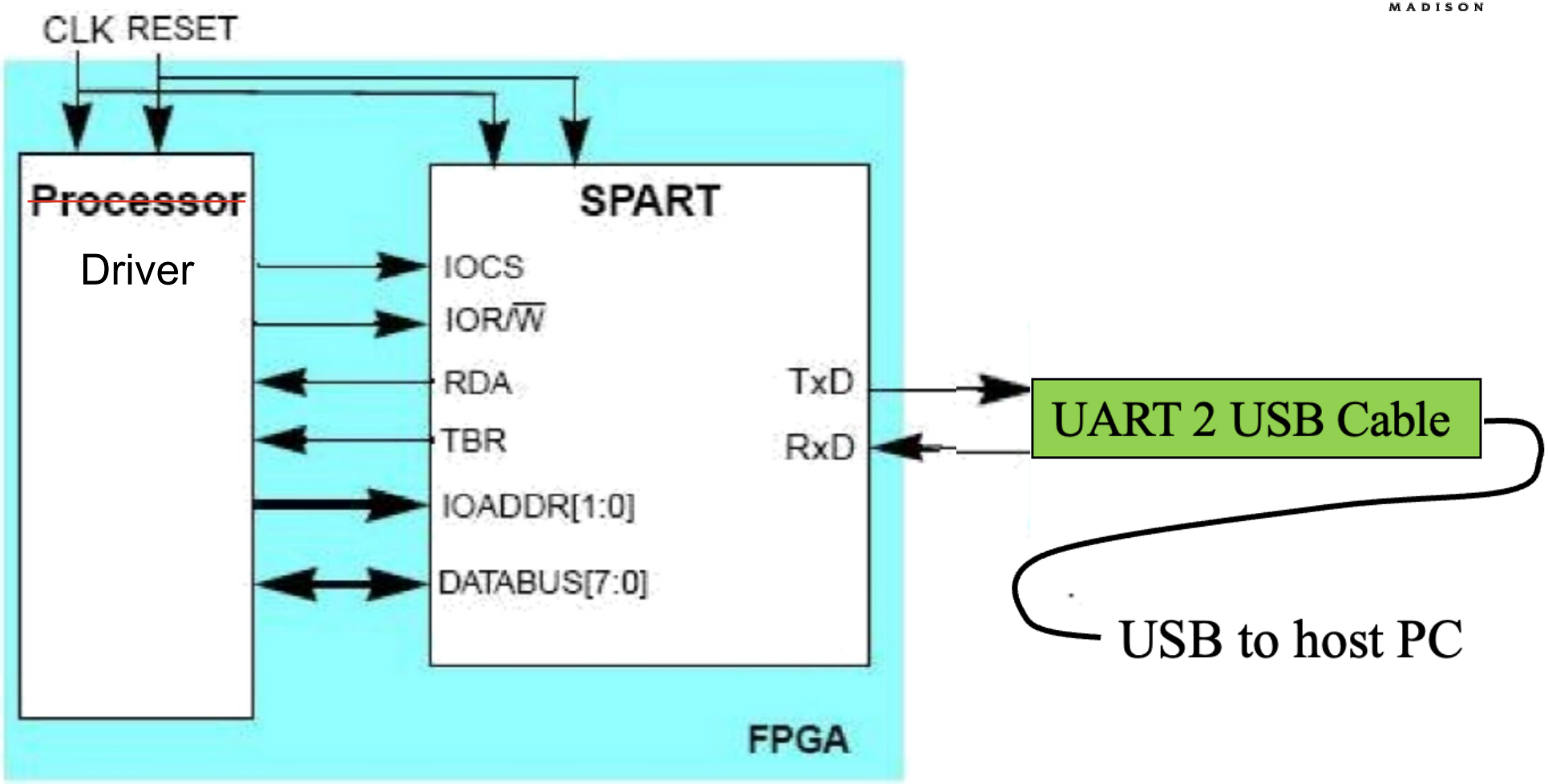
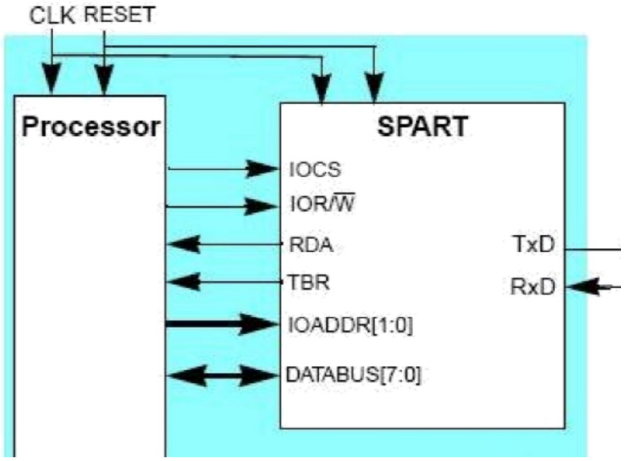


Figure 1: SPART Environment

Processor Interface

- Data is sent/received across the bidirectional data bus
- Handshaking (status) signals
  - TBR: Transmit Buffer Ready (Empty)
  - RDA: Receive Data Available
  - IOCS: Chip Select
  - IOR/W_: Read or Write Bar signal

SPART Interface

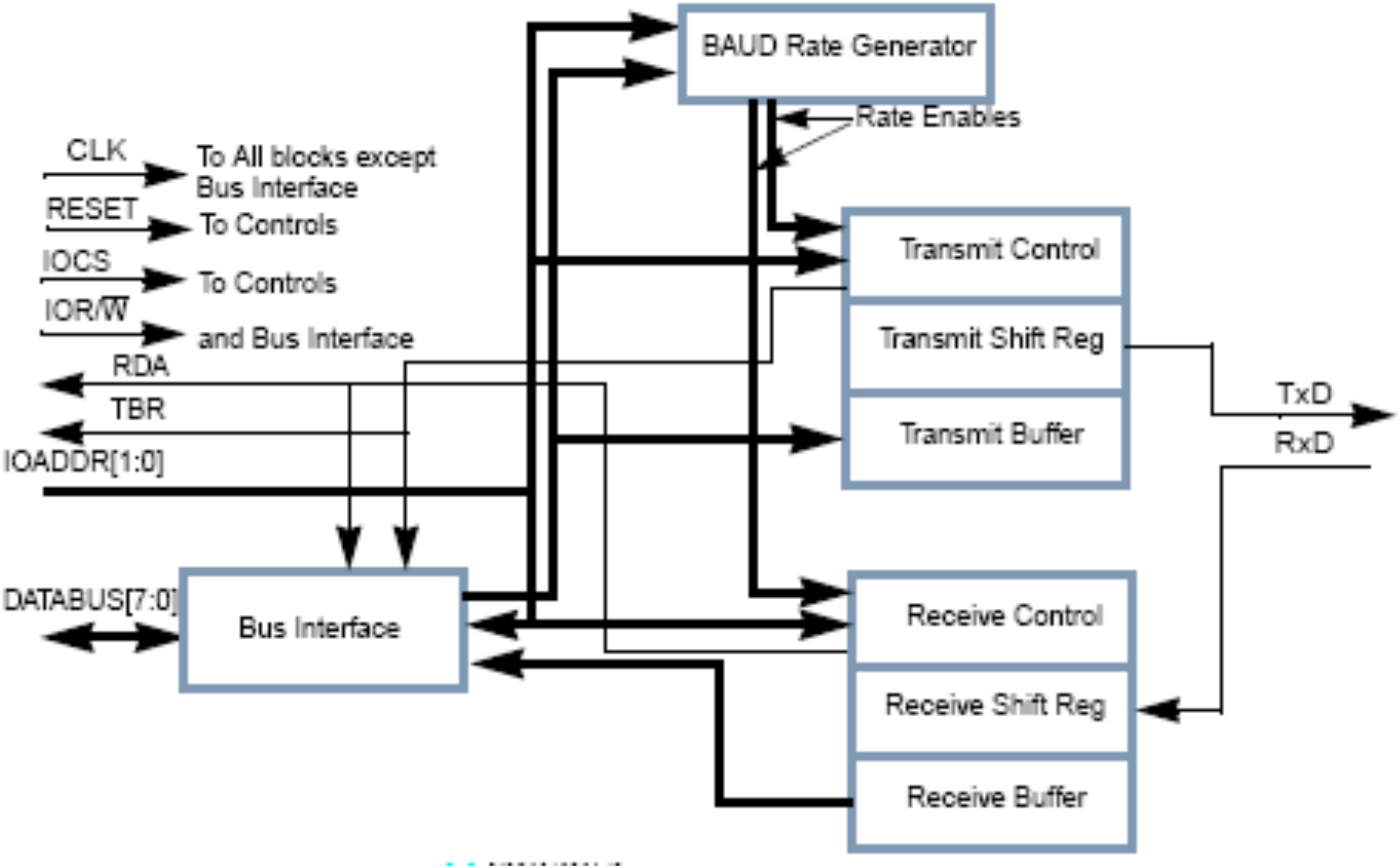


DATABUS[7:0]	An 8-bit, 3-state bidirectional bus used to transfer data and control information between the Processor and the SPART.
IOADDR[1:0]	A 2-bit address bus used to select the particular register that interacts with the DATABUS during an I/O operation.
IOR/W	Determines the direction of data transfer between the Processor and SPART. For a Read (IOR/W=1), data is transferred from the SPART to the Processor and for a Write (IOR/W=0), data is transferred from the processor to the SPART.
IOCS	I/O Chip Select
RDA	Receive Data Available - Indicates that a byte of data has been received and is ready to be read from the SPART to the Processor.
TBR	Transmit Buffer Ready - Indicates that the transmit buffer in the SPART is ready to accept a byte for transmission.

IOADDR	SPART Register
00	Transmit Buffer (IOR/W = 0); Receive Buffer (IOR/W = 1)
01	Status Register (IOR/W = 1)
10	DB(Low) Division Buffer
11	DB(High) Division Buffer

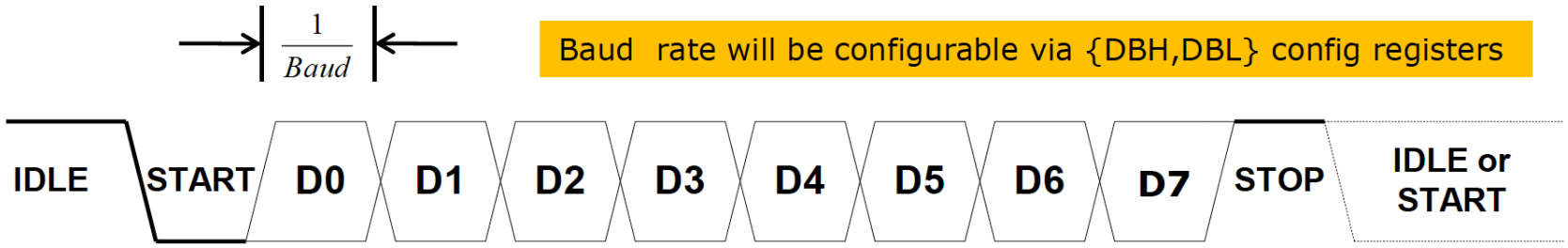
Table 1: Address Mappings

SPART Block Diagram



UART Serial Communication
(Universal Asynch Receiver Transmitter

- Idle
- Start bit
- Data (8-data for our project)
- Parity (no parity for our project)
- Stop bit – channel returns to idle condition
- Idle or Start next frame



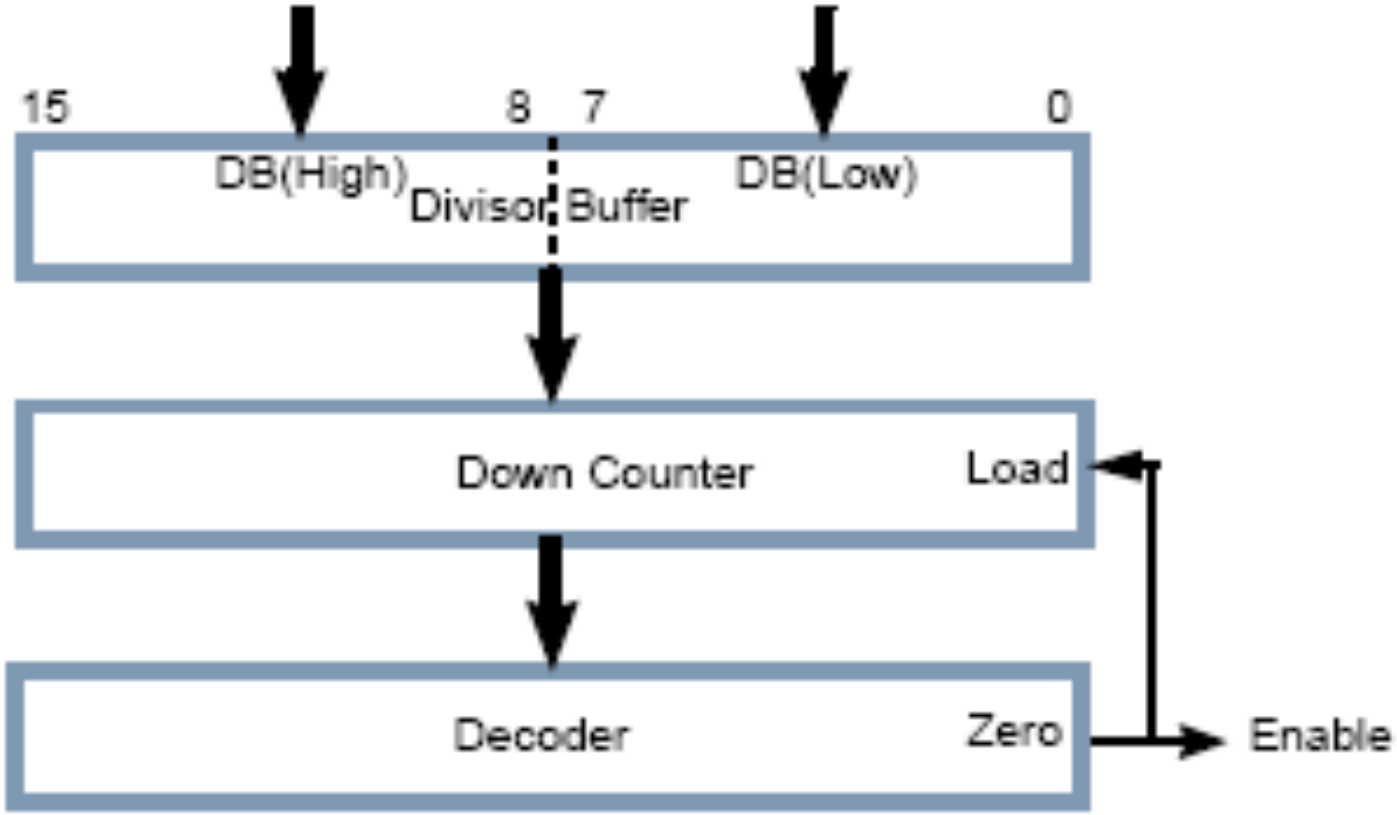
Transmitting

- Tx should be implemented and tested first.
- Tx shifts the “LSB” out from Tx buffer first.
- Tx sends “stop bit” when there is nothing to send.

Receiving

- Receiver samples the RxD to get the beginning of the “start bit”
- Data bits are sampled after start bit has been detected

Baud Rate Generator



Baud Rate and Sampling

- We want the transmission rate to be constant independent of the system clocks
- Example: Baud rates of 4800/9600/19200/38400 bit per second
- Sampling rate = x16 of the baud rate
- Divide the clock frequency to get the “Enable” signal (sampling rate) // see document formula and detailed examples

Testbench

- A mock processor implemented as a simple finite state machine
- Receive data on the RxD from keyboard and transmit (echos) back on the TxD back to the HyperTerminal
- Implement a simplified printf that can print character strings
- Load Baud Rate Generator with Arbitrary value
- Demonstrate ability to work at different Baud Rates using the BRG register (set using switches on the board)

Demonstration

- Show the ability to receive and transmit characters at 4800/9600/19200/38400 baud rates.
- Use the starter kit Lab1-SPART.zip posted on Canvas
- Also look at more detailed document linked on Canvas
- ~~Demo deadline is also end of lab on Thurs. Sep 19~~

Mini Lab 1 : What to submit?

- ~~Verilog code for your design with clear comments~~
- Report:
  - Include record of experiments conducted and how the design was tested
  - Problems encountered and solutions employed
- ~~Deadline (submit on Canvas) is end of lab on Thursday 9/19~~
- Submit once per group