SAINT LOUIS
UNIVERSITY™
EST. 1818

PARKS COLLEGE OF ENGINEERING, AVIATION AND TECHNOLOGY

Department of Electrical and Computer Engineering

3450 Lindell Blvd.
Suite 1017
St. Louis, MO 63103

P: 314-977-8300
F: 314-977-9510

www.slu.edu

ECE 3216

ISA LABS

ISA Expansion Port

Simulation Only Version

SAINT LOUIS
UNIVERSITY™
EST. 1818

PARKS COLLEGE OF
ENGINEERING,
AVIATION AND
TECHNOLOGY

*Department of Electrical
and Computer Engineering*

**3450 Lindell Blvd.
Suite 1017
St. Louis, MO 63103**

P: 314-977-8300
F: 314-977-9510

www.slu.edu

Kyle Mitchell, Ph.D.
March 2020 Revision

SAINT LOUIS
UNIVERSITY™
— EST. 1818 —

PARKS COLLEGE OF
ENGINEERING,
AVIATION AND
TECHNOLOGY

*Department of Electrical
and Computer Engineering*

3450 Lindell Blvd.
Suite 1017
St. Louis, MO 63103

P: 314-977-8300
F: 314-977-9510

www.slu.edu

Table of Contents

ISA Write

Use simulated digital logic for an ISA card to generate a multibit clock at a programmable frequency. Use the supplied BUS cycle generator to send bus commands to your design to set you clock rate. The clock should be an 8 bit counter that has a least significant bit that runs at the programmed frequency. The simulated board has a master clock frequency of 50MHz.

Background:
Discussion of creating a writable port on an ISA bus
In this lab your are creating a ISA attached x86 IO register. our design needs to monitor the ISA to determine if an ISA IO write is being performed to your address. If it is then you need to latch the data into your register.

For this experiment you need to create a 16 bit divisor for the master clock. In x86 ISA 16 bit quantities can be design and accessed in two different methods. In method one the 16 bits are transfered across the bus at the same time. In method two each of the 8 bit bytes are transfered separately. You design does not have to support both methods, but which ever method you choose needs to match in the VHDL design and the BUS cycle generator.

Discussion of
-Programmable clock frequencies
-Handle the rate conversion in C
-Handle the rate conversion in Hardware

Creating a programmable frequency clock requires creating a counter to divided the rate of a faster clock by some predetermined number. That is count the number of input ticks from the master clock and produce a tick on the slave clock every count ticks of the master. This will create a 50% duty cycle clock with a frequency of F_{master}/(count*2). This essentially means this design can only divided by even numbers. To calculate the required devisor one must calculate which of the possible frequencies is closest to the desired frequency, load the appropriate values into the counter to generate this frequency. This calculation is relative ease to handle in C by dividing the master clock by the desired frequency and rounding the result to the nearest even number. Division in hardware can consume large numbers of gates.

Discussion of
-Multi-bit counters
-Jitter
-Propagation Delay

SAINT LOUIS
UNIVERSITY™
— EST. 1818 —

PARKS COLLEGE OF
ENGINEERING,
AVIATION AND
TECHNOLOGY

Department of Electrical
and Computer Engineering

3450 Lindell Blvd.
Suite 1017
St. Louis, MO 63103

P: 314-977-8300
F: 314-977-9510

www.slu.edu

-Synchronizing
Multi-bit counters generated from ripple carry adders suffer from jitter on their counts. As the addition ripples through the adder the count bits are not updated synchronously. Flip-flops can be added to the output of the adder and triggered such that the count is produced in a synchronous manner. The maximum speed a counter can achieve is a function of the longest delay path through the counter. For a ripple carry style counter this is the path from one of the inputs through the carry chain and finally out through the setup time of the flip-flop. The time delay through this chain limits the speed at which the counter can be clocked.

Procedure:
In either VHDL or in a schematic create an ISA device that implements an 8 bit variable frequency 50% duty cycle clock. This clock is to use the 50MHz clock on the MESA 4i38 card as a master clock and is to produce divided version of this clock. Your design should be capable of producing 50MHz, 25MHz, 12.5MHz ... 1525.878Hz clocks.

Your clock is to be settable by writing a divisor into IO space of the simulated ISA card.

Along with your hardware design you are to supply a DOS executable capable of requesting a clock frequency and setting the hardware to the requested frequency.

Questions:
How did you handle allowing the master clock to be seen on the output, that is how did you handle dividing the master clock by 1.

Estimate the propagation delay of the longest delay path through your counter. The output files from the Xilinx compiler may help with this endeavor.

ISA Read

Use simulated digital logic for an FPGA based ISA card to drive an simulated A/D converter and capture samples into an x86 IO space mapped buffer. Update the BUS cycle generator to read the analog samples out of the IO space buffer and print them to the console.

Background:
Discussion of creating a readable port on an ISA bus
In this lab your are creating additional ISA attached x86 IO registers. This design will use the programmable clock from the previous lab to drive the sample rate of the A/D converter. It will add a status register and a value register to handle data movement.

The A/D converter will signal when a conversation is complete. Along with the conversion complete the A/D will present data. Your status register should capture this information. When the BUS cycle generator reads data from the data register, that read should clear the status register. In this manner reads to the status register will indicate if new data is ready in the data register.

You will need to modify the BUS cycle generator to generator continous reads of your status register, followed by a read of the data register once the status register becomes true. The data value should be printed to the console and the process should repeat.

Discussion of
-A/D converters
ADCs use one of several methods to create a bit pattern representing where a voltage lies with respect to an upper and lower bound. Typically these bounds are ground and V_{ref} or $-V_{ref}$ and $+V_{ref}$. A bit pattern of all 0s represents a voltage equal to the lower bound. A voltage of all 1s represents a voltage of the (upper bound-(upper bound-lower bound)/(2^bits)). That is the output is only calibrated to just below the upper bound and a voltage input equal to the upper bound can produce erratic results.

Procedure:
Use the simulated x86 IO space and the supplied A/D model to build a simulated A/D system. Use your designed system to exercise the A/D and print the values to the console. Capture these values to a file that can be read by Matlab. Use Matlab to generate plots of the waveforms. The A/D model has a 2 bit waveform input. It is capable of generating 4 different waveforms. You are to determine the following properties of each waveform: Shape, Frequency, Dynamic Range (Max(amplitude) - Min(amplitude))

The A/D model closely matches that of an ADC0808. Design a hardware to facilitate reading samples from an ADC0808 and storing these in a buffer that the BUS cycle logic can read from across the ISA bus. Design the ISA bus interface required to allow this buffer to be read from in the x86 IO space.

Questions:
How do you set the sample rate on an ADC0808?

What voltage ranges can an ADC0808 read?

What changes would be required to be able to read wall voltage?