Alternatives Evaluation

Introduction

In this section, design alternatives will be described. These alternatives are described in case the intended design fails to perform. More importantly, it provides a list of options in the design process. The design alternatives section describes the aforementioned alternatives for the overall project. It also details the general design for each alternative as well as offer a sketch of what it may look like. The client/user operation section describes how each design will be utilized. In the last section, the decision matrix, each design shall be critiqued via the design metrics outlined in the functional analysis. This matrix will highlight the pros and cons of each design in order to help establish which alternative is superior for the overall project.

Design Alternatives

There are three basic design alternatives. The design choices selected are either using a logic analyzer type design of logic placed directly onto the FPGA. The second option also involves using a portion of the FPGA to provide the necessary logic but instead of using a logic analyzer type setup instead it will use a simpler flip flop arrangement to catch the bit. The third option is to find an off the shelf component to perform the desired task. In this portion of the report each of these design alternatives will be described with some depth.

The design option that is being considered shall be referred to as the logic analyzer approach. This approach will take a portion of the FPGA to make a high frequency region within the FPGA which will have some simple and very parallel logic blocks that will serve as the event detector. A phase locked loop will be employed to raise the frequency within a portion of the FPGA. It is as of yet undetermined how high of a frequency this section will operate at. However, this portion of the board would need to

operate at a minimum of 400MHz. Prior experience indicates that this is an obtainable goal. The event detector would employ a series of shift registers to capture the input signal. These shift registers would then act as a demultiplexer to turn the input into multiple parallel outputs.

The number of outputs would depend on the the speed that this logic block is operating at. If the block operates at 400 MHz the shift register would have to be four registers deep and would turn the input signal into an output of four 100 MHz signals each of the four representing a different portion of the original signal. These signals would then be saved into a piece of instantiated RAM which could be read by the rest of the FPGA at 100 MHz. This system would require a 4 bit wide piece of RAM that is at this point of an undetermined depth. The address of the RAM that the system saves to would be controlled by a counter and event records would be overwritten with newer event records. This RAM would be clocked at 100MHz so that it could effectively interface with the rest of the FPGA.

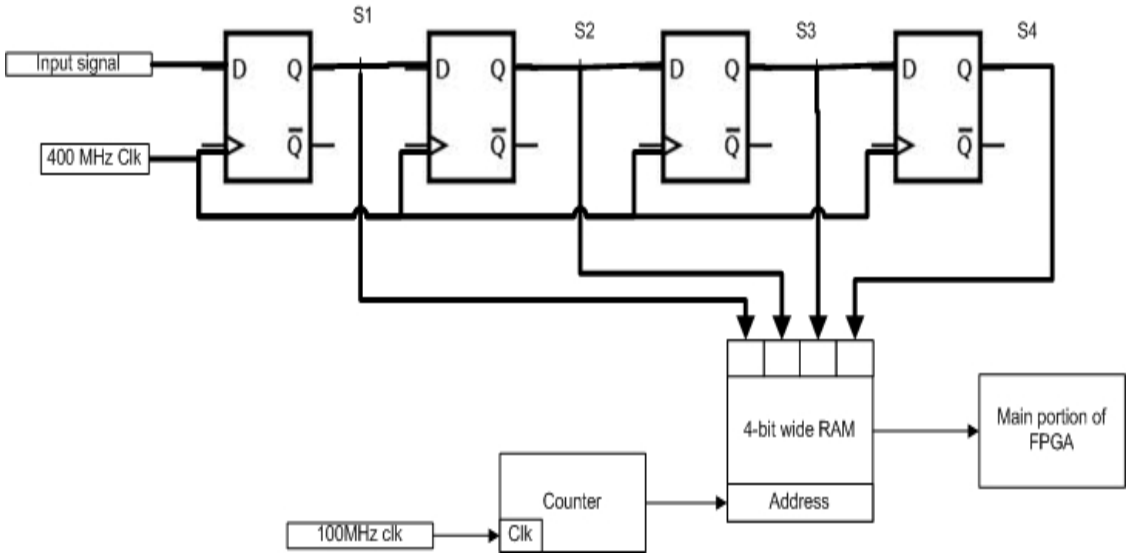

Figure 1: A diagram of the basic design for a logic analyzer circuit

This system has some advantages and disadvantages over other possible design. The main disadvantage of this system would be that at least four flip flops will be needed for every channel of the sensor. Since the sensor the event detector will interface with is a 32 channel sensor this means that this design will require 128 flip flops along with logic for a counter to control the address of the RAM block. Finally this method will require a piece of ram of as of yet undetermined size. All of this makes for a

reasonable design that could take up a fair amount of space on the FPGA.

The logic analyzer design has several advantages that could offset the disadvantage of it requiring a fair amount of logic. This design is a reasonably straight forward and should provide the desired functionality with relatively few issues that can cause design setbacks. Another advantage of this design is that it will be able to be implemented fully on the FPGA and will not require any extra equipment. The design can also be set up to allow for an as many stored pulse values as desired. All that is required is an additional 128 bits of memory for every extra 100MHz clock cycle that is to be recorded.

The second design idea will be referred to as the sticky bit approach. This approach is similar to the logic analyzer but much more condensed. The idea is to use a basic flip-flop to detect the strike. So unlike the logic analyzer approach this design will use only one flip-flop per channel. The delay circuit however will also need a latch that's attached to a second clock. This means that only 64 flip-flops will be needed for the entire design, shrinking the total logic necessary. It would also increase the speed due to less logic delay.

The basic design would be a D flip-flop with the D input tied to a logic one. The clock input would be supplied by the output of the sensor. The Q output would be retied to the D flip-flop as a reset after being passed through a delay circuit. The purpose of this delay is to regulate the frequency of the signal to be passed to the FPGA since the incoming signals from the sensor are too fast. So the signal is captured by the D-flip flop, stored, and delayed. This means that a second strike on the same input channel would cause the first strike to be lost.

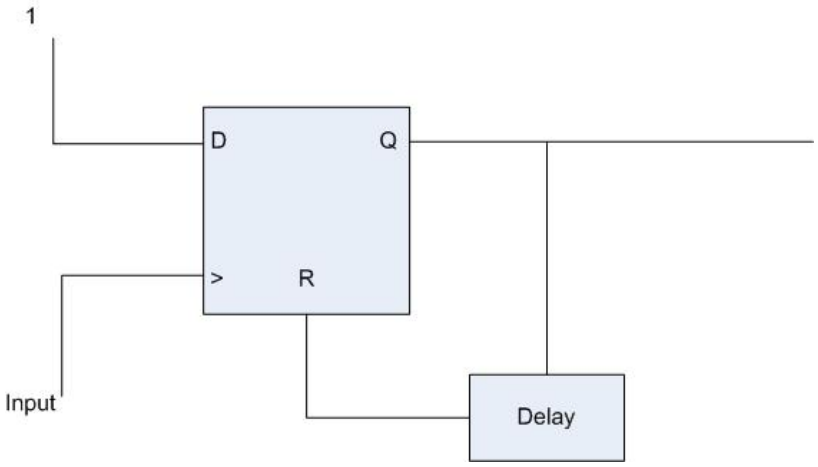

Figure 2: Basic Diagram of the Sticky Bit Circuit

The advantage of this approach is obviously size and cost. It can be implemented directly on the FPGA like the logic analyzer circuit, therefore no extra hardware is required. It doesn't use as much FPGA area as the logic analyzer does, therefore from a size perspective this approach is appealing. Also in terms of design time, the circuitry is much more simple, making it an appealing solution as well.

The main disadvantage of this approach is that the circuit is too simple to catch multiple strikes on the same channel. Thefore, the use of this design hinges on the assumption that no single channel of the sensor will experience back-to-back strikes. Also it would require the FPGA to interpret the data as its coming in, without storing the strikes in an easily accessible chunk of memory. This would make interfacing with the FPGA and a GUI more difficult.

The last alternative is off the shelf products. Based on the research online, we have found two series that we can use, the 105 or 106 series. Both series provide 32 channels which is what is needed for the system. The series can sense currents in the range 1-100mA, and they have a minimum duration of 0.1usec. The threshold resistance has the range 0.1-2000 ohms. Those series monitor the electrical resistance between 32 and 128 simple continuity loops, each loop consists of single or multiple specimens under test. Each test specimen is continuously monitored. The only difference between the two series is that the 105 has four 32-channel board and the 106 only has only one.

Features of the event detectors include programmable resistance threshold, selectable minimum event duration setting (0.1, 0.5, 1.0 microsecond), and selection of channel test currents. Also included with all event detectors is data collection/analysis software, RS232C serial port, capability to link with other event detectors, test input cables and a 1 year warranty. The 105/106 series have a maximum voltage of 2.5V, the maximum operating temperature is 35°C. The dimensions of this event detector is 11.5”H×18”W×15.5”D, with 51lbs weight.

The off the shelf event detectors will have the advantage of having a short design time, but the cost and the speed of operation might not be what we want. Since the cost is of big importance to the customer, we might have to reside to this as a last option.

Client/User Operation

This system is an internal component for a larger system as such the operation of the system is largely abstracted away from the user or client of the system. The general operation of the system requires the system to be able to record pulses as short as 5 ns and then to output the pulses to logic in an FPGA running at 100 MHz. The detector must be able to handle at least four consecutive pulses separated by as little as 2 ns. The signal from the sensor will enter the FPGA through the header pins if an FPGA based solution is chosen. The event detector would then have to be able to read the header pins that the signal comes in on. This should be fairly simple to allow the event detector to read the appropriate pins on the FPGA. If an off the shelf solution is chosen instead the off the shelf component must have sufficient and appropriate pins to read in the sensor information.

While those are the basic operation requirements for the system there are some user interactions involved in validating the systems functionality. To be able to demonstrate that this system does in fact work the detected pulses must be shown in some way. To accomplish this the recorded pulses will output to a graphical user interface and the recorded location of the pulses will be output onto a computer screen.

The FPGA that is being used has an RS232 port that will serve to get the information recorded by the even detector to the computer running the interface. This will require some additional logic on the FPGA which will be used just for demonstrating the functionality of the event detector and will not actually be part of the detector itself.

Overall the client and user operation for event detector is very minimal since the user will have no interaction with the detector itself. Ideally the detector would blend into the final system seamlessly. This would mean that the operator of the system would have no involvement in the continued operation of the event detector as long as the larger system which the event detector resides remains operational.

Decision Matrix

The design matrix is used to help evaluate different design ideas. It is useful because it helps to focus in on what aspects of the design are the most important. This allows for the different design alternative to be evaluated more objectively by simply comparing how the different designs satisfy the needs of the project in a quantitative manner instead of just going with a gut feeling for which design will work best. The most important design metric is the cost of the device. This is because there is a very small budget for additional equipment so our design must be low cost and take advantage of existing equipment. The second most important metric is design time. It is important that the chosen design scheme will be able to be designed in one semester. The third most important metric is FPGA area. Two of the three design alternatives will take up space on an existing FPGA so it is important that the design has a small footprint so as to leave room for other systems on the FPGA. The next metric is power usage. Power usage is directly tied to the amount of the FPGA used for the FPGA based designs and is a specification on the off the shelf design. A low power device is always attractive. The final metric is upside potential this serves as a catch all for the advantages that the different designs might have that are not necessarily required for the completion of the project but could be advantageous.

Alternative	Logic Analyzer	Sticky Bit	Off Shelf Components
Cost (9)	10	10	2
Design Time (8)	6	4	7
FPGA Area (7)	4	6	10
Power (5)	6	7	4
Upside Potential (2)	7	5	3
Totals:	**210**	**209**	**170**

Figure 3: Decision Matrix