Measurement 2.: Logic Analyzer

(BME-MIT, Sz.P.)

The purpose of this measurement is to introduce the features of a logical analyzer and the basic theory behind it. Students have to get acquainted with the usage of the analyzer by executing some basic measurements. The labor is equipped with an Intronix LogicPort Logic Analyzer.

Preparation for the Measurements

1. Study this note and the analyzer software (http://www.pctestinstruments.com/downloads.htm)!
2. Answer the test questions at the end of this note!

Applied Instruments

PC	
Logical Analyzer	Intronix LogicPort
Power Supply of the FPGA test board	Spartan-3 Starter board
Configuration Software	Xilinx iMPACT

Logic analyzer

A logic analyzer is an electronic instrument that displays signals in a digital circuit that are too fast to be observed and presents it to a user so that the user can more easily check the operation of the digital system with precision. They are typically used for capturing data in systems that have too many channels to be examined with an oscilloscope. Software running on the logic analyzer can convert the captured data into timing diagrams, protocol decodes, state machine traces, assembly language, or correlate assembly with source-level software.

A logic analyzer can trigger on a complicated sequence of digital events, and then capture a large amount of digital data from the system under test (SUT). Once the probes are connected, the user programs the analyzer with the names of each signal, and can group several signals into groups for easier manipulation.

Next, a capture mode is chosen, either timing mode, where the input signals are sampled at regular intervals based on an internal clock source generated by the analyzer, or state mode, where the system clock of the SUT us used to capture data. Therefore, in state mode only stets of the SUT can be analyzed; to measure timing properties (e.g. the frequency of the SUT clock) timing mode has to be used. In timing mode, it is critical to select an appropriate sampling clock frequency in order to be able to accurately measure the SUT's signals.

After the mode is chosen, a trigger condition must be set. A trigger condition can range from simple (such as triggering on a rising or falling edge of a single signal), to the very complex. Recorded samples are stored in the internal memory of the analyzer. The trigger position divides this buffer into two parts: the pre-trigger buffer shows samples recorded before the trigger event, while the post-trigger buffer shows samples recorded after the trigger event. The size of the pre-trigger buffer can be configured in percentage of the whole buffer (e.g. if the interesting events happen after the trigger it is advised to lower the pre-trigger size).

Measurements

The logical analyzer is connected to the B1 extension port of the test board. In the course of the measurement the corresponding internal signals of the system (FPGA) have to be connected to the logic analyzer. This means that the configuration file of the FPGA has to be modified in order to able to wire the desired internal signals to those FPGA pins which are connected to the analyzer. The FPGA – Analyzer pin connections can be found in the UCF file which can be downloaded from the course web page.

1. "Second Counter" Analysis with a Logic Analyzer

First, a slightly modified version of the 2digit „second counter" is analyzed. This counter counts much faster in order to be able to store a whole wrap around cycle in the memory of the logical analyzer. (The Verilog source codes can be found on the homepage of the lab.) The following internal signals have to be routed to the FPGA pins:

- 50 MHz system clock (to CLK1 clock input of the logical analyzer)
- external reset
- count direction select signal
- clock divider output signal (*cy*)
- lower digit (*cntr_d0*)
- compare value output of the lower digit(*cntr_d0_eq0* és *cntr_d0_eq9*)
- upper digit (*cntr_d1*)
- compare value output of the upper digit (*cntr_d1_eq0* és *cntr_d1_eq5*)

The analyzer has 2 clock and 32 data inputs. It is recommended to create an upper level module which contains all of these signals whether or not they are used.

According to previous considerations the declaration of the upper level module is the following:

```
module wpbevtop1(
    input clk,btn0, sw0,
    output [6:0] q,
    output [33:0] LANAL
    );
```

The pin configuration of the analyzer can be found on the homepage of the lab (LANAL.UCF). The upper two bits of port LANAL (LANAL[33:32]) and the lower 32 bits are connected to the clock and data inputs of the analyzer respectively. This connection simply means the configuration of port LANAL. For instance:

```
module wpbevtop1(
    input clk,btn0, sw0,
    output [6:0] q,
    output [33:0] LANAL
    );

wire [3:0] cntr_d0;
wire [2:0] cntr_d1;
wire cntr_d0_eq9, cntr_d0_eq0;
wire cntr_d1_eq5, cntr_d1_eq0;
```

```
assign LANAL = {1'b0, clk,
                18'b0,
                rst, dir, ce,                 // 3
                cntr_d1_eq0, cntr_d1_eq5,     // 2
                cntr_d1,                      // 3
                cntr_d0_eq0, cntr_d0_eq9,     // 2
                cntr_d0};                     // 4
```

One of the clock inputs is connected to the system clock while the other one is connected to zero. In this particular case 14 data bits are analyzed and the remaining 18 data bits are connected to zero.

In order to be able to route the internal signals of the counter to port LANAL they have to be assigned in the module *count_sec* . This can be done by adding new ports to the module and routing them to the corresponding internal signals.

```
module count_sec(
   input clk, rst, ce, dir,
   output [6:0] q,
   output la_cntr_d0_eq0, la_cntr_d0_eq9,
   output la_cntr_d1_eq0, la_cntr_d1_eq5);

reg [3:0] cntr_d0;
wire cntr_d0_eq0, cntr_d0_eq9;

always @(posedge clk)
if (rst)
   cntr_d0 <= 0;
else if (ce)
   if (dir)            //DIR=1: count up
      if (cntr_d0_eq9)
         cntr_d0 <= 0; //overflow
      else
         cntr_d0 <= cntr_d0 + 1;
   else                 //DIR=0: count down
      if (cntr_d0_eq0)
         cntr_d0 <= 9;
      else
         cntr_d0 <= cntr_d0 - 1;

assign cntr_d0_eq0 = (cntr_d0 == 0);
assign cntr_d0_eq9 = (cntr_d0 == 9);

reg [2:0] cntr_d1;
wire cntr_d1_eq0, cntr_d1_eq5;

always @(posedge clk)
if (rst)
   cntr_d1 <= 0;
else if (ce)
   if (dir & cntr_d0_eq9)
      if (cntr_d1_eq5)
         cntr_d1 <= 0;
      else
         cntr_d1 <= cntr_d1 + 1;
   else if (~dir & cntr_d0_eq0)
      if (cntr_d1_eq0)
         cntr_d1 <= 5;
      else
```

```
            cntr_d1 <= cntr_d1 - 1;

    assign cntr_d1_eq0 = (cntr_d1 == 0);
    assign cntr_d1_eq5 = (cntr_d1 == 5);

    assign q = {cntr_d1, cntr_d0};

    assign la_cntr_d0_eq0 = cntr_d0_eq0;
    assign la_cntr_d0_eq9 = cntr_d0_eq9;
    assign la_cntr_d1_eq0 = cntr_d1_eq0;
    assign la_cntr_d1_eq5 = cntr_d1_eq5;

    endmodule
```

According to the port assignment instantiation of module count_sec is modified as follows:

```
    count_sec counter(
       .clk(clk),
       .rst(rst),
       .ce(ce),
       .dir(dir),
       .q({cntr_d1, cntr_d0}),
       .la_cntr_d0_eq0(cntr_d0_eq0),
       .la_cntr_d0_eq9(cntr_d0_eq9),
       .la_cntr_d1_eq0(cntr_d1_eq0),
       .la_cntr_d1_eq5(cntr_d1_eq5));
    assign q = {cntr_d1, cntr_d0};
```

After finishing the modifications above implement the project and generate the configuration files.

1.1 FPGA configuration

Download the generated file (.bit) to the FPGA by using LOGSYS GUI (do not forget to enable the power supply).

1.2. LogicWave logic analyzer Startup

Start the LogicPort logic analyzer.

- The logic analyzer is connected to the PC through the USB port. The LED on the analyzer indicates if it is connected to the PC properly.
- The software of the analyzer can be started either with the following command **`Start/Programs/LogicPort/LogicPort Application`** or by clicking on the desktop icon of LogicPort.
- Create a new project in the logic analyzer window (**`File/New`**) with the name of the measurement report by adding additional characters if it is necessary.

After that it is recommended to check out the tabs, commands and main features of the analyzer.

1.3 Configuration of the Analyzer Display

Add the active signals to the display. Bunch the data bits into buses and label them according to the Verilog source code. Display the values of these buses in hexadecimal format. (right click on the label of the signal select **`Data Format`**). Define the comparison level at the input of the analyzer. The power supply of the test board is 3.3 V. The output voltages are 3.3V CMOS compatible. Recommended value for the comparison level is ~1.65V. This can be set on the top of the **`Waveform`** view window at the **`Logic Threshold`** field. Finally, save the project to

the network drive.

1.4 Counter Analysis in State Mode

Set **State Mode** in **Sample Mode Setup** window with the the follow parameters:

- Sample data 2.5ns before the Rising edge of CLK1 (scroll down list box)

Do not set trigger condition (i.e. every condition triggers). This can be set in the **Trigger Setup** window in one of the following ways:

- **Trigger** Immediately when acquisition starts
- **Trigger** When level A is satisfied, and the **Level A conditions** fields are not ticked off.
- **Trigger** When level A is satisfied, the **Level A conditions:** Pattern A is True, and in the **Waveform** window choose **X** for the value of **Pattern A**, which means don't care with respect to the trigger condition.

Start acquisition **Acquisition Single**

After the trigger condition happened and the memory of the logic analyzer is full, a Ready text is displayed at the bottom left.

At the end of the acquisition the measured signals are display in the waveforms window. In case of **State Mode** analysis it is recommended to analyze the **State List** as well.

Analyze and evaluate the waveforms and the state list of the counter. Paste the waveform into the report.

1.4.1 Counter Analysis with Trigger conditions

First select a simple trigger condition:

- Sample data 2.5ns before the Rising edge of CLK1 (scroll down list box)
- the counter value is 15h and the *dir* value 1

Take a measurement with this trigger condition. Check the trigger condition out in the state list view. Evaluate the results.

Take a measurement appling the following two level trigger condition:

Change on the direction input (dir) while the counter is between 05h and 09h.

Answer the following questions:

- Can you detect hazards in State Mode?
- Applying the settings above can you estimate which edge of the *clk* signal triggers the counter?

1.5. Sequential Network Analysis in Timing Mode

The following network has to be analyzed in Timing Mode. Since the network and the signals are the same as about it is recommended to modify the previous project.

1.5.1 Settings of Timing Mode

Set the Timing Mode in the **Sample Mode Setup** window by selecting the Timing Mode tab. After that select a proper Sample Rate on the top of the Waveforms window.

1.5.2 Trigger conditions of Timing Mode

Study the possible trigger conditions of the LogicPort analyzer in Timing Mode. Then take a

measurement applying a simple trigger condition. For instance counter value = 0x00.

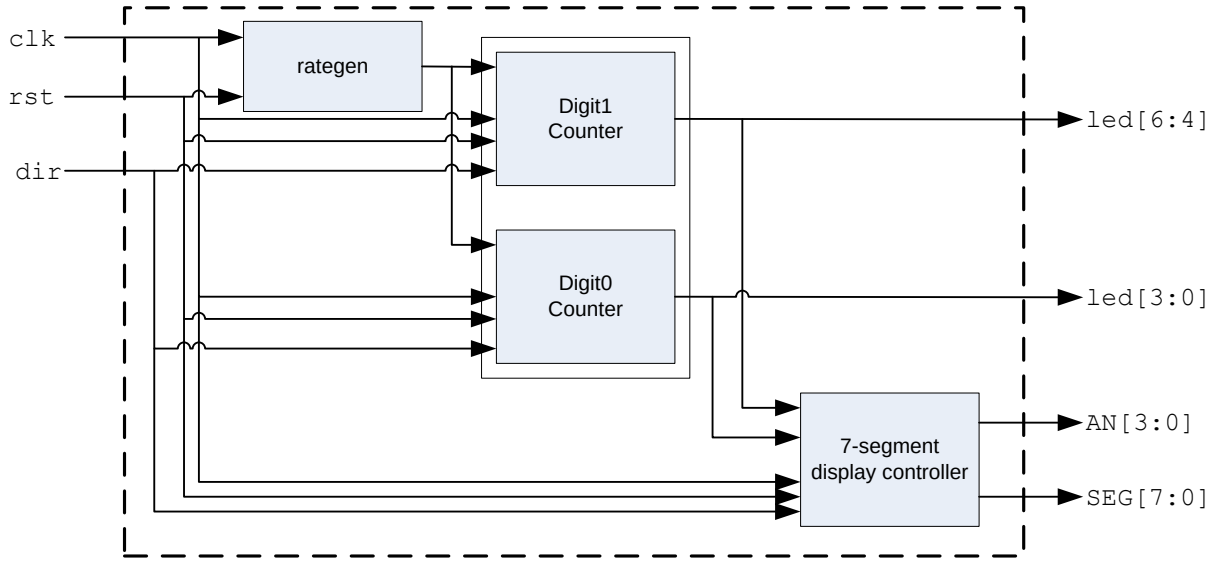
1.5.3 Time Interval Measurements with Logic Analyzer

Using the cursor lines measure the following metrics (document them with printscreens):

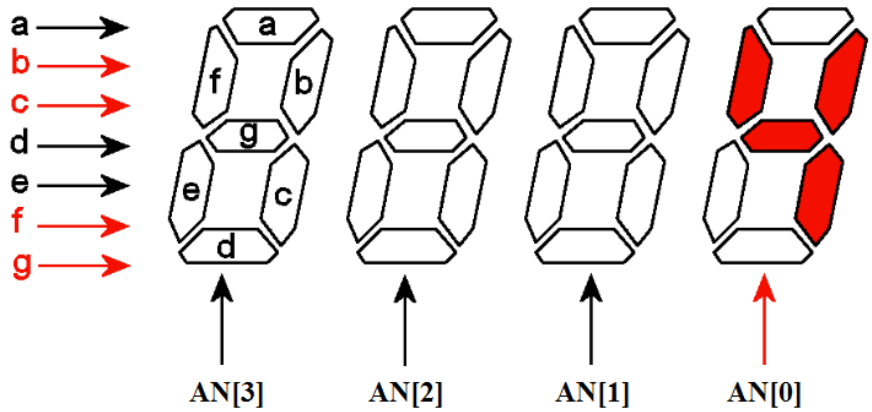
- period of the clock signal (clk)
- period of the enable signal
- wrap around cycle time (how much time does it to count from 0 to 59 or a full cycle)
- propagation time of the counter (the time difference between the clock and the corresponding counter outputs).

2. Implementing 4-digit 7-segment display interface

The next task is to extend the system design on the first measurement with a 7-segment display. That is, the modified schematic diagram is as shown below.

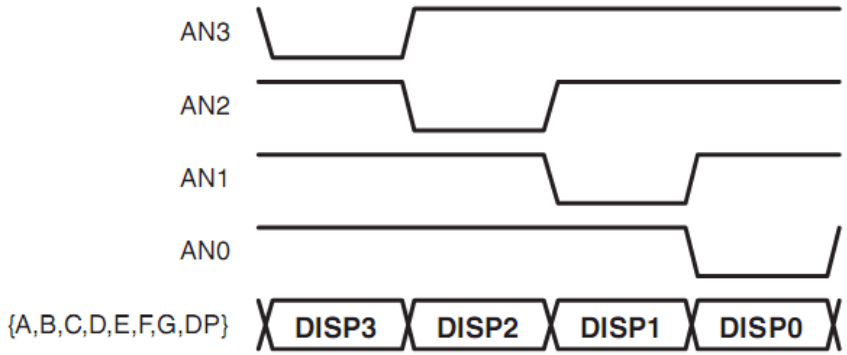


The display consists of 4 digits, each having 7 segments (plus the dot). The display is controlled by 8 segment signals and 4 so-called anode signals – the latter are the enable signals for each digit, as shown below. All signals are active low.



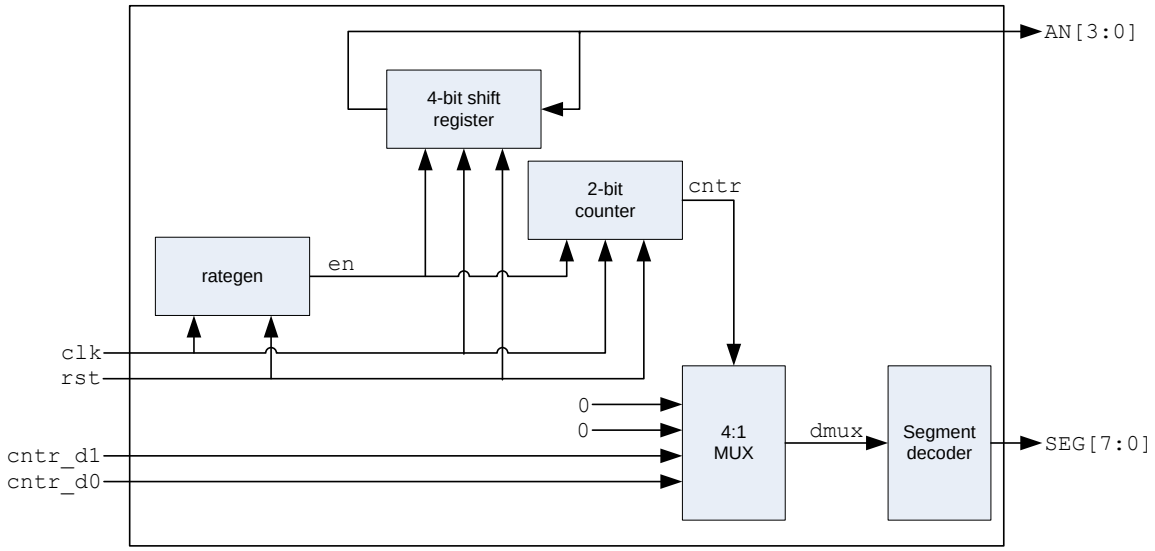
As the segment signals are shared between all 4 digits, the display should be controlled in a time-

divided manner. That is, if the first digit is enabled by driving AN[0] with '0', the segment lines should be driven with the value corresponding to the first digit; if AN[1] is enabled, then segments should be driven with the value corresponds to the second digit; and so on.



Circulating between the 4 digits should be slow enough to have enough time to turn on the LEDs, but should be fast enough for the human eye not to see blinking – the appropriate frequency is in the kHz range.

The following schematic diagram shows the display controller design.



Details:

- The rategen part generates a clock enable signal for the other parts in kHz range.
- The 4 bit shift register generates the anode signal by rotating a single '0' bit to generate the required waveform: 1110 → 1101 → 1011 → 0111 → 1110…… The reset value should be 1110, and shifting is enabled with the rategen's output.
- The 2-bit counter operates synchronously to the shift register, it always shows the position of the '0' bit. Therefore, this counter can be used as a select signal for the multiplexer which selects the data to be displayed ont he enabled digit.
- The multiplexer selects the appropriate input data: cntr_d0 and cntr_d1 for the first and second display digit, zero for the other two.
- The segment decoder converts the 4-bit binary data into 8-bit segment data.

Tasks:

- Download the Verilog module of the 7-segment controller complete it with the necessary Verilog code.
- Instantiate the module int he top level module, generate the FPGA configuration file and test the design.

Test Questions

1. What is the difference between the State and Timing Modes of the logic analyzer?
2. What is the purpose of trigger signals and trigger conditions of logic analyzers?
3. The states of the sequential network have to be determined. Which mode has to be chosen and what is the source of sampling signal in this case?
4. The propagation time (Td) of a network has to be determined with the logic analyzer. Which mode has to be chosen and what is the source and frequency of the sampling signal in this case?
5. Pulse widths has to be estimated. The sampling time of the analyzer is 200ns with a 0.001% uncertainty. The measured results are about 10us. How much is the measurement error of the pulse width caused by the sampling ("time quantization")? Provide results in percent!
6. A periodic square wave is analyzed with a logic analyzer. The frequency of the signal is between 5 kHz - 10 kHz and the duty cycle is between 20% - 50%. What is the minimum value of the sampling frequency if the time span of L and H values have to be derived with at most 5% uncertainty? Assume that only 1 period is captured with the analyzer and the uncertainty of the internal clock is negligible.