Frequency Meter
SLG47910

This application note shows how the ForgeFPGA can be used to build a frequency meter. Using a SLG47910V FPGA device, an SLG46722 GreenPAK device, and some additional components the frequency of an input signal can be determined and displayed.

Contents

Terms and Definitions

PLL	Phase Lock Loop
FPGA	Field Programmable Gate Array
FPGA Editor	Main FPGA design and simulation window
Go Configure Software Hub	Main window for device selection

References

For related documents and software, please visit:

ForgeFPGA Low-density FPGAs | Renesas

Download our free ForgeFPGA™ Designer software [1] to open the .ffpga design files [1] and view the proposed circuit design.

[1] ForgeFPGA Designer Software, Software Download and User Guide, Renesas Electronics
[2] AN-FG-006 Frequency Meter.ffpg, ForgeFPGA Design File, Renesas Electronics
[3] SLG47910, Preliminary Datasheet, Renesas Electronics

1. Introduction

This Application Note measures input frequency and outputs the measured value on a seven-segment display. The maximum input frequency value that can be measured is 999999 Hz. A frequency meter is implemented using the SLG47910V FPGA. An additional SLG46722 GreenPAK device is used as a driver for the seven-segment display. The Verilog code (AN-FG-006 Frequency Meter.ffpga) for the frequency meter is available on the ForgeFPGA webpage.

2. Ingredients

- ForgeFPGA Device SLG47910V (QFN-24L package)
- SLG46722V IC
- ForgeFPGA Development Board with USB cable and power supply
- Seven-segment display module, bread board and various components
- Latest Revision of ForgeFPGA Workshop software

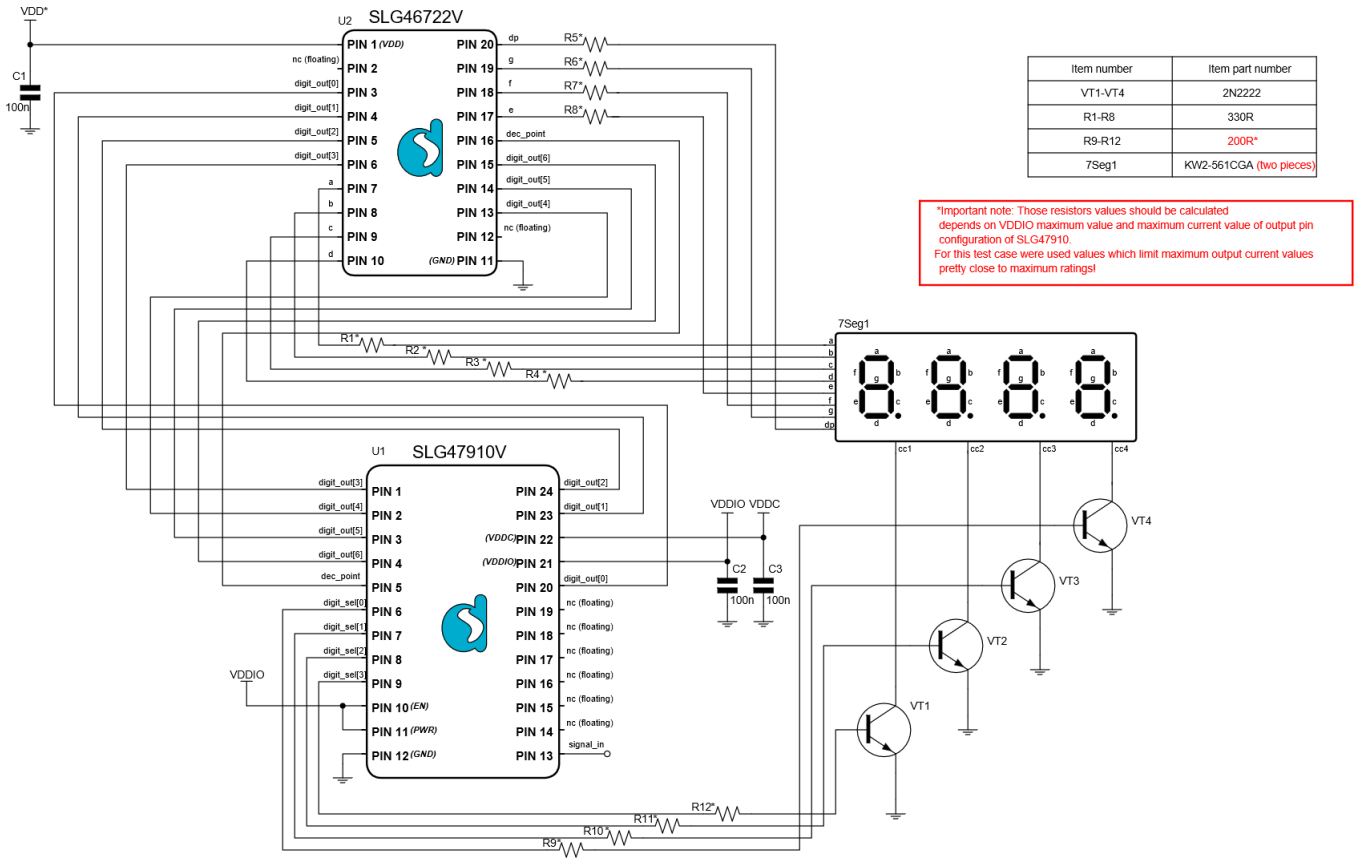

Figure 1: System Diagram

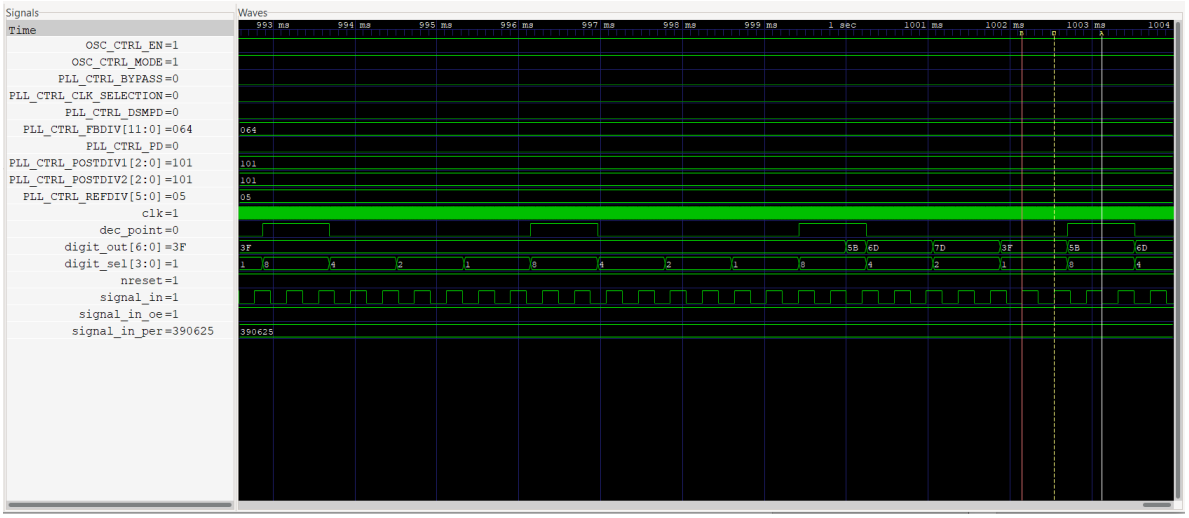

Figure 2: Frequency Meter Waveforms

3. Frequency Meter Operation

The complete frequency meter circuit is shown in . The FPGA samples input, signal_in on GPIO9. After one second, the FPGA outputs a HEX code on outputs digit_out[6:0] indicating the frequency of the input signal. The FPGA outputs, digit_out[6:0] and dec_point are routed to the SLG46722V device which conditions the signal to the seven-segment display. The FPGA also outputs control signals (digit_set[3:0]) that drive bi-polar transistors. To meet the current requirements of the bi-polar transistors set the GPIO pins for digit_set[3:0] in 2x Drive Push-Pull mode in the ForgeFPGA designer software.

The numbers on the seven-segment display before decimal point represents value in kHz, after decimal point value in Hz. A sample waveform for digit_out[6:0], digit_set[3:0] and dec_point is shown in Figure 2. The waveform values from 1000ms to 1002ms are summarized in Table 1.

Table 1: Design Outputs for signal_in = 2.560kHz

time	1000ms	1000.5ms	1001ms	1002ms
dec_point	0b1	0b0	0b0	0b0
digit_out[6:0]	0x5B	0x6D	0x7D	0x3F
Digit_set[3:0]	0x8	0x4	0x2	0x1

Using the Hexadecimal conversion table in Table 2, the seven-segment display will show a frequency of 2.560 kHz. The waveform shows that this sequence keeps repeating as long as the period does not change.

Table 2: Hexadecimal Conversion to Seven Segment Drive

Digit	Hex code	Seven Segment (gfedcba)
0	3F	1111110
1	06	0110000
2	5B	1101101
3	4F	1111001
4	66	0110011

Digit	Hex code	Seven Segment (gfedcba)
5	6D	1011011
6	7D	1011111
7	07	1110000
8	7F	1111111
9	6F	1111011

The Verilog code for the frequency meter uses the PLL as a sampling clock. The PLL settings are shown in the waveform. Refer to the SLG47910 data sheet for PLL control settings.

4. Frequency Meter Verilog Code

The Verilog code (AN-FG-006 Frequency Meter.ffpga) for the frequency meter can be downloaded from the ForgeFPGA webpage.

The design is configured by parameters:

BASE_FREQ - Determines base input frequency value which defined by configuration.

The BASE_FREQ cannot exceed the Timing Analysis Achievable Frequency value (44.94MHz).

COM_ANODE - Determines seven-segment display type, with common anode or common cathode.

Table 3: Frequency Meter Design Parameters

Design Parameters	Range	Default	Description
BASE_FREQ	[40000000:30000000]	40000000	Clock domain frequency
COM_ANODE	[1:0]	0	When one, display type is common anode, otherwise display type is common cathode.

The design has the following pins:

clk - input clock domain signal

nreset - input registers initialization signal

OSC_CTRL_EN - output OSC enable signal

OSC_CTRL_MODE - output OSC mode signal

PLL_CTRL_PD - output PLL enable signal

PLL_CTRL_DSMPD - output PLL enable signal

PLL_CTRL_CLK_SELECTION - output PLL clock select signal

PLL_CTRL_BYPASS - output PLL out path select signal

PLL_CTRL_REFDIV - output bus input PLL divider value

PLL_CTRL_FBDIV - output bus PLL Feedback divider value

PLL_CTRL_POSTDIV1 - output bus PLL output divider1 value

PLL_CTRL_POSTDIV2 - output bus PLL output divider2 value

signal_in - input frequency signal

signal_in_oe - output signal_in OE signal

dec_point - output decimal point signal

digit_out[0] - output A segment signal

digit_out[1] - output B segment signal

digit_out[2] - output C segment signal

digit_out[3] - output D segment signal

digit_out[4] - output E segment signal

digit_out[5] - output F segment signal

digit_out[6] - output G segment signal

digit_sel[0] - output fourth (from left to right) digit enable signal

digit_sel[1] - output third digit enable signal

digit_sel[2] - output second digit enable signal

digit_sel[3] - output first digit enable signal

The BASE_FREQ parameter is set so that it matches the PLL programmed value. The code below shows PLL and Oscillator configurations. The PLL is configured so the output is at 40Mhz. The PLL uses the Oscillator as the CLK input. The Oscillator should be trimmed before building the design. The number of detected rising edges on the signal_in input during measure period is equal to the input frequency value. The calculated frequency is converted to a seven-segment hexadecimal. The calculation is averaged over one second.

```
//OE
assign signal_in_oe = 1'd1;

//OSC settings
assign OSC_CTRL_EN = 1'd1;
assign OSC_CTRL_MODE = 1'd1;

//PLL settings
assign PLL_CTRL_PD = 1'd0;
assign PLL_CTRL_DSMPD = 1'd0;
assign PLL_CTRL_BYPASS = 1'd0;
assign PLL_CTRL_CLK_SELECTION = 1'd0;
assign PLL_CTRL_REFDIV = 6'd5;
assign PLL_CTRL_FBDIV = 12'd100;
```

assign PLL_CTRL_POSTDIV1 = 3'd5;

assign PLL_CTRL_POSTDIV2 = 3'd5;

Set the COM_ANODE parameter to meet the requirements of the seven-segment display. The configuration shown in . is for a common cathode configuration. The Verilog code support both types.

The freq_meter.ffpga file contains a testbench that can be used to validate the Verilog code. The output waveforms from the test bench are shown in Figure 2.

5. Implementation

The circuit shown in can be constructed as a design project. The steps below show how to build the circuit for evaluation.

1. Download the AN-FG-006 Frequency Meter.ffpga design from the ForgeFPGA website. Use the Go Configure Software to load the AN-FG-006 Frequency Meter.ffpga design.
2. Select the FPGA editor and configure the BASE_FREQ and COM_ANODE design parameters

Run the freq_meter_tb testbench to validate that the design functions correctly. Use the ForgeFPGA Development Board to program the SLG47910V device.

3. Review the driver requirements for seven-segment display and program the SLG46722 device to meet the requirements. See for notes on the selection of resistor values.
4. Use a breadboard to construct the circuit in .

6. Conclusion

This application note shows how the SLG47910V device can be used in a design project to build a circuit board that implements a frequency meter. The Verilog code for the frequency meter is provided. This testcase (AN-FG-006 Frequency Meter.ffpga) is available for download. If interested, please contact the ForgeFPGA Business Support Team.

7. Revision History

Revision	Date	Description
1.00	Mar 3, 2022	Initial release.

IMPORTANT NOTICE AND DISCLAIMER

RENESAS ELECTRONICS CORPORATION AND ITS SUBSIDIARIES (“RENESAS”) PROVIDES TECHNICAL SPECIFICATIONS AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for developers skilled in the art designing with Renesas products. You are solely responsible for (1) selecting the appropriate products for your application, (2) designing, validating, and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. Renesas grants you permission to use these resources only for development of an application that uses Renesas products. Other reproduction or use of these resources is strictly prohibited. No license is granted to any other Renesas intellectual property or to any third party intellectual property. Renesas disclaims responsibility for, and you will fully indemnify Renesas and its representatives against, any claims, damages, costs, losses, or liabilities arising out of your use of these resources. Renesas' products are provided only subject to Renesas' Terms and Conditions of Sale or other applicable terms agreed to in writing. No use o any Renesas resources expands or otherwise alters any applicable warranties or warranty disclaimers for these products.

(Disclaimer Rev.1.0 Mar 2020)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan

www.renesas.com

Contact Information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:

www.renesas.com/contact/

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.