## **Vivado Frequency Meter**

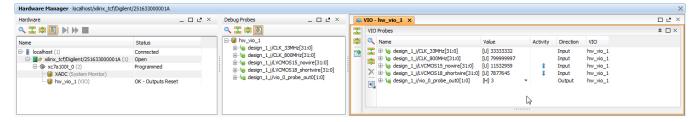
Frequency meter? So simple you may say. Yes, sure. But doing FPGA based frequency meter that can be instantly used by thousands of engineers? Hm.. once long time ago I did a JTAG controlled FPGA Frequency meter IP Core and PC GUI application for it also. I think I wanted to have some pictures and links to Xilinx and yes I did got at least semi-official "green" from them to-do so. The GUI application talked over LPT Based JTAG adapter and did some clever tricks to measure the frequency on some FPGA pin without the use of any known reference clock in the FPGA. As part of that work I did get from semi-official channels schematic of original Xilinx Parallel Cable IV. In order to support it I did create a Coolrunner JEDEC to VHDL conversion tool. I also had a ideas about integration with Chipscope IP Cores, so I did some reverse engineering on the Chipscope ICON.

This was long time ago.

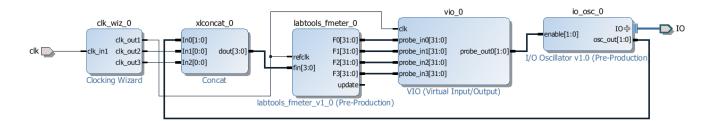
LPT Ports are long past history, so bringing back in live that old design makes no sense. Maybe the FPGA based simple Frequency counter is already done? There are options of course, Altium has on-chip instrumentation that includes Frequency meter (if I recall it correctly). But Altium FPGA support is something that is maybe fun for educational purposes not much usable for real life.

Vivado Labtools do not have Frequency meter.

## Who said that? They do now!



Here, TE0710 Art ix module as hardware test bench. Frequency Counter IP Core configured for 4 channels. Reference clock is from 100 MHz oscillator. Just for testing 33.3333 MHz and 800MHz signals from FPGA Clock PLL are connected to channels 0 and 1. Channels 2 and 3 are connected to I/O Oscillator IP Core. Channel 2 I/O pin is "empty" FPGA ball from DDR3 bank with 1.5V I/O Voltage. Channel 3 I/O pin is in 1.8V bank and is connected to on-board Ufl coaxial connector with PCB trace of about 12 mm length.



This the top level design.

umentation 🛅 IP Location												
IP Symbol Resource	Component Name desi	gn_1_dk_wiz_0_	_0									
now disabled ports	Board Clocking	Options Out	put Clock	IS MMCM S	Settings Por	Renaming	Summa	ry				
Í	The phase is calculat			put clock.								
	Output Clock	Output Freq Requested	(MHz)	Actual		hase (deo Requested		Actual	Duty Cycle Requested		Actual	Drives
	✓ ck_out1	100.000	8	100.000		.000		0.000	50.000	8	50.0	BUFG
	✓ ck_out2	800.000	8	800.000	0	.000	8	0.000	50.000	0	50.0	No buffer
	✓ dk_out3	33.000	8	33.333	C	.000	8	0.000	50.000	0	50.0	No buffer
	ck_out4	100.000		N/A	C	.000		N/A	50.000		N/A	BUFG
	dk_out5	100.000		N/A	C	.000		N/A	50.000		N/A	BUFG
	clk_out6	100.000		N/A	C	.000		N/A	50.000		N/A	BUFG
	dk_out7	100.000		N/A	0	.000		N/A	50.000		N/A	BUFG
clk_out1 = =-clk_in1 clk_out2 =-		QUENCING	Number		ocking Feedbac Source			Signaling				
clk_out3 <b>-</b>	ck out1	1			Automa			③ Single				
	clk_out2	1			Automa	tic Control C	off-Chip	Difference	ential			
	ck_out3	1			O User-Co	ntrolled On	-Chip					
	ck_out4	1			O User-Co	ntrolled Off	-Chip					
	ck_out5	1			0							
	ck_out6	1										
	ck_out7	1										
			input.	_dk_stopped		Type Active High Active Low						

Clock Wizard is used to configure the FPGA Clocking, I have selected 33.3333Mhz and 800MHz, please notice that "measurement" channels are configured with no BUFG.

🖵 Re-customize IP			×
labtools_fmeter_v1_0 (1.0)			4
Show disabled ports     F0[31:0]     F1[31:0]     F1[31:0]     F3[31:0]     update     vupdate	Component Name Number of channels Number of bits	design_1_labtools_fmeter_0_0         4       (116)         32       4       (848)	OK Cancel

Frequency meter IP Core configured for 4 channels.

E Re-customize IP	
I/O Oscillator v1.0 (1.0)	
W Documentation 🛅 IP Location	
Show disabled ports IO + enable[1:0] osc_out[1:0]	Component Name design_1_io_osc_0_0 I/O Count 2 [11024]
	OK Cancel

I/O Oscillator configured for 2 channels. The default oscillator implementation uses no fabric logic only I/O Cell is used. This minimizes the impact of fabric delays and routing to the oscillation frequency.

FF LUT Memory LUT	1588 798	126800	1.25
	798	62400	
Memory LUT		63400	1.26
	24	19000	0.13
I/O	3	210	1.43
DSP48	5	240	2.08
BUFG		32	9.38
MMCM	1	6	16.67

Utilization report: Frequency counter core uses no clock buffers (assuming the reference clock has a global buffer already). Frequency counter uses 1 DSP48E per channel, and 1 DSP48E for the reference clock. All clock measurement nets are driving exactly one load, the clock input of the DSP48E primitive, there is no other connections on that net. So it is irrelevant if the net is driven by some buffer or fabric local routing, the frequency performance of the channel is the same.

⊡ <u>Clock Region X0Y2</u>		Global Buffers @ Y=2	2	① Image: Clock Region X1Y2
	Site		Cell	
	S BI	JFGCTRL_X0Y31		
		JFGCTRL_X0Y30		
		JFGCTRL_X0Y29		
	S BI	JFGCTRL_X0Y28		
	S BI	JFGCTRL_X0Y27		
	: 🛽 🖉 Bl	JFGCTRL_X0Y26		
	S BI	JFGCTRL_X0Y25		
		JFGCTRL_X0Y24		1
		JFGCTRL_X0Y23		
		JFGCTRL_X0Y22		
		UFGCTRL_X0Y21		
		JFGCTRL_X0Y20		
		UFGCTRL_X0Y19		
		JFGCTRL_X0Y18	🖂 alle hue (DUEC)	
		JFGCTRL_X0Y17 JFGCTRL_X0Y16	<pre>clkf_buf (BUFG) clkout1_buf (BUFG)</pre>	
😑 Bottom Half				
Clock Region X0Y1	🕞 Global Buffers @ Y=1	L	🕀 📼 <u>Clock Reg</u>	<u>iion X1Y1</u>
	Site	Cell		
	BUFGCTRL_X0Y15			
	BUFGCTRL_X0Y14			
	BUFGCTRL_X0Y13			
	BUFGCTRL_X0Y12			
	BUFGCTRL_X0Y11			
	BUFGCTRL_X0Y10			
	BUFGCTRL_X0Y9			
	BUFGCTRL_X0Y8			
	BUFGCTRL_X0Y7			
	BUFGCTRL_X0Y6			
	BUFGCTRL_X0Y5			
	BUFGCTRL_X0Y4 BUFGCTRL_X0Y3			
	BUFGCTRL_X0Y2			
	BUFGCTRL_X0Y1			
	BUFGCTRL_X0Y0	💷 u_bufg_icon (B	UFG)	
⊡    ⊡ <u>Clock Region X0Y0</u>	Global Buffers @ Y=0	)		jion X1Y0

Clock resource allocation, clearly visible that all BUFG that are used are for the MMCM or Vivado ICON IP Core, none of the BUFG in the design is needed for the Frequency meter IP Core.