MPM_VPX570

Development Accelerator "Multi-Path Modulation" Software Package



Key Features

- Loopback pipeline ADC-DAC
- Quadrature Generator
- Multi-Path with N independent branches
- N independent variable delays and gain
- N independent Phase and Amplitude modulators
- Combiner for N branches
- MATLAB® model included

Benefits

- Accelerated development
- Flexibility covers a maximum of application cases
- Reference design with VHDL source code included
- MATLAB model allows user to start development prior to hardware supply
- AS9100 and ISO9001 certified company





vadatech

MPM_VPX570

The MPM_VPX570 Multi-Path Modulator Software package from VadaTech provides additional FPGA image and source code for the VPX570, intended to accelerate the development of a customer's end application.

The standard VPX570 is delivered by default with a BSP reference design loaded that allows a customer to test all of the board hardware features and acts as a start point for custom code development. The additional Multi-Path Modulator Software is a separate package implementing a pipeline between the ADC and the DAC using a subset of the available hardware functions.

The loopback pipeline between the ADC and the DAC integrates the following features:

- Quadrature Generator: Hilbert Transform to generate the complex representation of the real-valued ADC output
- Splitter: Splits the Quadrature generator's complex output to N independent branches
- Delay: Delays the input signal by a variable delay value
- Phase and Amplitude Modulator: Modulates the complex signal by a complex exponential, and applies gain
- Combiner: Sum the N Phase and Amplitude Modulator outputs
- (Optional) Controller: AXI4-Lite control of the Delay, Phase and Amplitude modulator parameters

A Matlab® model of the loopback pipeline is provided. This model does the following:

- Bit/cycle accurate simulation of each block of the loopback pipeline
- Performance measurement and comparison with Matlab® floating point equivalent functions
- Test of loopback parameters (delay, frequency, phase, gain)
- Generation of input/output files, for Xilinx simulator

Please note that VadaTech does not provide licenses for MATLAB® or third-party tools and these will have to be purchased separately.

MPM_VPX570 license conditions restrict use to implementation on VadaTech hardware only.



Figure 1: VPX570

Quadrature Generator

The Quadrature Generator transforms the real input signal to a complex output. To do so, a Hilbert transform is used, in order to remove the negative frequency component of the real input signal.

Delay

Each branch has its own independent delay block. Delay value can be updated at each clock cycle. The coarse delay blocks use FPGA UltraRAM memory. The minimum coarse delay is 6 clocks cycles (due to URAM minimum latency), and the maximum coarse delay depends on how many URAM are implemented. This can be modified based on the number of signal branches in the system.

Phase and Amplitude Modulator

Each branch has its own modulator. The DDS Frequency/Phase as well as the gain can be updated at each clock cycle. The complete modulator pipeline operates in full precision. The input parameters DDS Frequency (phase increment) and DDS Phase (phase offset) are the inputs to a 32-bit Phase Accumulator, followed by a Phase Offset adder. The final phase is used as an address for a -SINE/COSINE lookup table. DDS Frequency and DDS Phase parameters can be changed every clock cycle to create a complex DDS output waveform.

Combiner

The combiner sums the N branch modulator outputs (48 bits), and rounds/saturates the sum output to 16-bit. The combiner output error, due to the quantization of the full scale 48-bit input to 16-bit output, stays below 0.5 LSB.

Optional AXI4-Lite Controller

For each branch, the Delay and the Phase and Amplitude Modulator have the following control signals:

- Delay Coarse
- Delay Fine
- DDS Frequency
- DDS Phase
- Gain

These control signals can be directly controlled by the user application, or routed to an AXI4-Lite Controller to control it via Microblaze, PCIe, etc.

Parameter	Minimum	Maximum	Unit
Delay coarse	6	32768 (typical)	ADC sampling clock/16
Delay fine	0	15	ADC sampling clock
DDS frequency	0	4294967295	0-360°
DDS phase	0	4294967295	0-360°
Gain	0	8191	UFix13_12

Table 1: Loop Parameters Range

Matlab® Model

A Matlab model of the loopback pipeline is provided. This model does the following:

- Bit/cycle accurate simulation of each block of the loopback pipeline
- Performance measurement and comparison with Matlab floating point equivalent functions
- Test of Loopback parameters (delay, frequency, phase, gain)
- Generation of input/output files, for Xilinx simulator

Block Diagram



Figure 2: Multipath Modulator Signal Processing pipeline

Commands

VPX570 Tool v1.0 Usage: <crr> - i - i - i - aGHz sampling clock from front panel CLK ADC and DAC both @3GSPS dac mode rpb rpw - Set DAC analog parameters mode: nrz, nrz rtz, rf rpb: rpb0, rpb1, rpb2, rpb3, rpb4 rpw: rpw0, rpw1, rpw2, rpw3, rpw4 dac_gain gain - Set DAC internal gain (0 to 1023, default 512) status - cOOPBACK CONTROL </crr>				
Usage: <cmd> [<opts>] i -Initialize the card 3GHz sampling clock from front panel CLK ADC and DAC both @3GSPS dac mode rpb rpw -Set DAC analog parameters mode: nrz, nrtz rtz, rf rpb: rpb0, rpb1, rpb2, rpb3, rpb4 rpw: rpw0, rpw1, rpw2, rpw3, rpw4 dac_gain gain - Set DAC analog parameters mode: nrz, nrtz rtz, rf LOOPBACK CONTROL - set_loop idx dl_corse dl_fine ph freq gain - Set Ioopback parameters idx : sub range index (1 to number of subrange) dl_coarse: sub range delay, from 0 to 32767 (unit 187.5Mhz period) dl_fine set_loop idx dl_corse dl_fine ph freq gain - Set Ioopback parameters idx : sub range modulator phase, from 0 to 32767 (unit 187.5Mhz period) dl_fine sub range modulator frequency (unit 187.5Mhz period) ph gain - Set Ioopback parameters idx : sub range modulator frequency (unit 142) gain : sub range modulator frequency (unit 142) gain get_loop - Get loopback parameters adc_source source - Set ADC output: 0= ADC samples, 1=fixed Fs/4 sine wave, 2=zeros </opts></cmd>	VPX570 Tool v1.0			
i - Initialize the card i - Initialize the card 3GHz sampling clock from front panel CLK ADC and DAC both @3GSPS dac mode rpb rpw - Set DAC analog parameters mode: nrz, nrz rtz, rf rpb: rpb0, rpb1, rpb2, rpb3, rpb4 rpw: rpw0, rpw1, rpw2, rpw3, rpw4 dac_gain gain - Set DAC internal gain (0 to 1023, default 512) status - Get ADC/DAC sync status LOOPBACK CONTROL set_loop idx dl_corse dl_fine ph freq gain - Set loopback parameters idx : sub range index, from 6 to 32767 (unit 187.5Mhz period) dl_fine : sub range modulator frequency (unit H2) gain : sub range modulator frequency (unit H2) gain : sub range gain, from 0 to 4095 (4095 0dB gain) get_loop - Get loopback parameters adc_source source - Set ADC output: 0= ADC samples, 1=fixed Fs/4 sine wave, 2=zeros	Usage: <cmd> [<opts>]</opts></cmd>			
i - Initialize the card 3GHz sampling clock from front panel CLK ADC and DAC both @3GSPS dac mode rpb rpw - Set DAC analog parameters mode: nrz, nrtz rtz, rf rpb: rpb0, rpb1, rpb2, rpb3, rpb4 rpw: rpw0, rpw1, rpw2, rpw3, rpw4 dac_gain gain - Set DAC internal gain (0 to 1023, default 512) status - Get ADC/DAC sync status LOOPBACK CONTROL 	HARWDARE CO	NTROL		
3GHz sampling clock from front panel CLK ADC and DAC both @3GSPS dac mode rpb rpw - Set DAC analog parameters mode: nrz, nrtz rtz, rf rpb: rpb0, rpb1, rpb2, rpb3, rpb4 dac_gain gain - Set DAC internal gain (0 to 1023, default 512) status - Get ADC/DAC sync status	i	- Initialize the card		
ADC and DAC both (g3GSPS) dac mode rpb rpw - Set DAC analog parameters mode: nrz, nrtz rtz, rf rpb: rpb0, rpb1, rpb2, rpb3, rpb4 rpb: rpb0, rpb1, rpb2, rpb3, rpb4 rpw: rpw0, rpw1, rpw2, rpw3, rpw4 dac_gain gain - Set DAC internal gain (0 to 1023, default 512) status - Get ADC/DAC sync status		3GHz sampling clock from front panel CLK		
dac mode rpb rpw - Set DAC analog parameters mode: nrz, nrz rtz, rf rpb: rpb0, rpb1, rpb2, rpb3, rpb4 rpw: rpw0, rpw1, rpw2, rpw3, rpw4 dac_gain gain - Set DAC internal gain (0 to 1023, default 512) status - Get ADC/DAC sync status	data manda mehanusi	ADC and DAC both @3GSPS		
Indue: niz, niz nz, ni rpb: rpb0, rpb1, rpb2, rpb3, rpb4 rpw: rpw0, rpw1, rpw2, rpw3, rpw4 dac_gain gain - Set DAC internal gain (0 to 1023, default 512) status - Get ADC/DAC sync status	dac mode rpb rpw	pw - Set DAC analog parameters		
input: nput: nput		rnb: $rnb0$: $rnb1$: $rnb2$: $rnb3$: $rnb4$		
dac_gain gain - Set DAC internal gain (0 to 1023, default 512) status - Get ADC/DAC sync status		rpw: rpw0 rpw1 rpw2 rpw3 rpw4		
status - Get ADC/DAC sync status LOOPBACK CONTROL	dac gain gain	- Set DAC internal gain (0 to 1023, default 512)		
LOOPBACK CONTROL set_loop idx dl_corse dl_fine ph freq gain - Set loopback parameters idx : sub range index (1 to number of subrange) dl_fine : sub range delay, from 6 to 32767 (unit 187.5Mhz period) dl_fine : sub range delay, from 0 to 15 (unit 3GHz period) ph : sub range modulator phase, from 0 to 360 deg freq : sub range modulator frequency (unit Hz) gain : sub range gain, from 0 to 4095 (4095 0dB gain) eft-loop - Get loopback parameters adc_source source - Set ADC output: 0= ADC samples, 1=fixed Fs/4 sine wave, 2=zeros DEBUG COMMAND - d <g> - Dump FPGA register contents w <addr> <val> - Write value at FPGA address r <addr> - Dump device's register contents wd <ald> - Write value to the device's address rd <ald> - Write value to the device's address rd <ald> - Read value from the device's address sa <clr><dowdar <="" td=""> - Read value from the device's address sa <clr><dowdar <="" td=""> - Set PE43713 attenuators in dB Device Key: g = GPIO a = EV12DS460AZP</dowdar></dowdar></ald></ald></ald></addr></val></addr></g>	status	- Get ADC/DAC sync status		
LOOPBACK CONTROL set_loop idx dl_corse dl_fine ph freq gain - Set loopback parameters idx : sub range index (1 to number of subrange) dl_coarse: sub range delay, from 6 to 32767 (unit 187.5Mhz period) dl_fine : sub range delay, from 0 to 15 (unit 3GHz period) ph : sub range modulator phase, from 0 to 300 deg freq : sub range modulator frequency (unit Hz) gain : sub range gain, from 0 to 4095 (4095 0dB gain) get_loop - Get loopback parameters adc_source - Set ADC output: 0= ADC samples, 1=fixed Fs/4 sine wave, 2=zeros				
set_loop idx di_corse di_inine primed gain - Set loopback parameters idx : sub range index (1 to number of subrange) dl_coarse: sub range delay, from 0 to 32767 (unit 187.5Mhz period) dl_fine : sub range delay, from 0 to 15 (unit 3GHz period) ph : sub range modulator phase, from 0 to 360 deg freq : sub range modulator phase, from 0 to 360 deg gain : sub range modulator frequency (unit Hz) gain : sub range gain, from 0 to 4095 (4095 0dB gain) get_loop - Get loopback parameters adc_source - Set ADC output: 0= ADC samples, 1=fixed Fs/4 sine wave, 2=zeros	LOOPBACK CON	VTROL		
idx : set notpback parameters idx : sub range index (1 to number of subrange) dl_coarse: sub range delay, from 0 to 32767 (unit 187.5Mhz period) dl_fine : sub range delay, from 0 to 15 (unit 3GHz period) ph : sub range modulator phase, from 0 to 360 deg freq : sub range modulator phase, from 0 to 360 deg gain : sub range modulator phase, from 0 to 360 deg gain : sub range modulator frequency (unit Hz) gain : sub range gain, from 0 to 4095 (4095 0dB gain) get_loop - Get loopback parameters adc_source - Set ADC output: 0= ADC samples, 1=fixed Fs/4 sine wave, 2=zeros	set_loop idx di_corse di_fine ph freq gain			
dl_coarse: sub range delay, from 6 to 320767 (unit 187.5Mhz period) dl_fine : sub range delay, from 0 to 15 (unit 3GHz period) dl_fine : sub range modulator phase, from 0 to 360 deg ph : sub range modulator phase, from 0 to 360 deg freq : sub range modulator phase, from 0 to 360 deg gain : sub range gain, from 0 to 4095 (4095 0dB gain) get_loop - Get loopback parameters adc_source - Set ADC output: 0= ADC samples, 1=fixed Fs/4 sine wave, 2=zeros		idy		
dl_fine : sub range delay, from 0 to 15 (unit 3GHz period) ph : sub range modulator phase, from 0 to 360 deg freq : sub range modulator phase, from 0 to 360 deg gain : sub range gain, from 0 to 4095 (4095 0dB gain) get_loop - Get loopback parameters adc_source source - Set ADC output: 0= ADC samples, 1=fixed Fs/4 sine wave, 2=zeros		dl coarse: sub range delay, from 6 to 32767 (unit 187 5Mbz period)		
ph : sub range modulator phase, from 0 to 360 deg freq : sub range modulator frequency (unit Hz) gain : sub range gain, from 0 to 4095 (4095 0dB gain) get_loop - Get loopback parameters adc_source source - Set ADC output: 0= ADC samples, 1=fixed Fs/4 sine wave, 2=zeros		d fine : sub range delay, from 0 to 15 (unit 3GHz period)		
$\begin{array}{rcl} & & \mbox{freq} & : \mbox{sub range modulator frequency (unit Hz)} \\ & \mbox{gain} & : \mbox{sub range gain, from 0 to 4095 (4095 0dB gain)} \\ & \mbox{get_loop} & - \mbox{Get loopback parameters} \\ & \mbox{adc_source source} & - \mbox{Set ADC output: } 0 = \mbox{ADC samples, 1=fixed Fs/4 sine wave, 2=zeros} \\ \hline & \mbox{DEBUG COMMAND} \\ \hline & \mbox{ddr} & - \mbox{Dump FPGA register contents} \\ & \mbox{w < addr} & - \mbox{Dump FPGA register contents} \\ & \mbox{w < addr} & - \mbox{Num evalue at FPGA address} \\ & \mbox{r < addr} & - \mbox{Read back at FPGA address} \\ & \mbox{dd} & \mbox{dd} & - \mbox{Dump device's register contents} \\ & \mbox{wd < ald} & \mbox{-} \mbox{Vurit value to the device's address} \\ & \mbox{rd < ald} & \mbox{-} \mbox{Read value from the device's address} \\ & \mbox{rd < ald} & \mbox{-} \mbox{Read value from the device's address} \\ & \mbox{sa < c[r > 0-31.75)} & \mbox{-} \mbox{Set PE43713 attenuators in dB} \\ \hline & \mbox{Device Key:} \\ & \mbox{g} & = \mbox{GPIO} \\ & \mbox{a} & = \mbox{EV12DS460AZP} \\ \hline & \mbox{dd} & \mbox{-} \mbox{Add} & $		ph : sub range modulator phase, from 0 to 360 deg		
gain : sub range gain, from 0 to 4095 (4095 0dB gain) . Get loopback parameters . Get loopback parameters adc_source source . Set ADC output: 0= ADC samples, 1=fixed Fs/4 sine wave, 2=zeros		freq : sub range modulator frequency (unit Hz)		
get_loop - Get loopback parameters adc_source source - Set ADC output: 0= ADC samples, 1=fixed Fs/4 sine wave, 2=zeros		gain : sub range gain, from 0 to 4095 (4095 0dB gain)		
adc_source source - Set ADC output: 0- ADC samples, 1-lixed FS/4 sine wave, 2-zeros	get_loop	- Get loopback parameters		
DEBUG COMMAND d <g> - Dump FPGA register contents w <addr> <val> - Write value at FPGA address r <addr> - Read back at FPGA address dd <a d> - Dump device's register contents wd <a d> <addr> <val> - Write value to the device's address rd <a d> <addr> <val> - Write value to the device's address sa << r> <0-31.75> - Set PE43713 attenuators in dB Device Key: g = GPIO a = EV12AS350A d = EV12DS460AZP</val></addr></a d></val></addr></a d></a d></addr></val></addr></g>	adc_source source	- Set ADC output. 0- ADC samples, 1-lixed Fs/4 sine wave, 2-zeros		
d <g> - Dump FPGA register contents w <addr> <val> - Write value at FPGA address r <addr> - Read back at FPGA address dd <ald> - Dump device's register contents wd <ald>< addr> <val> - Write value to the device's address rd <ald> - Write value from the device's address sa <clr> rd <ald>> - Set PE43713 attenuators in dB Device Key: g = GPIO a = EV12AS350A d d = EV12DS460AZP</ald></clr></ald></val></ald></ald></addr></val></addr></g>	DEBUG COMMAN	ND		
w <addr> <val> - Write value at FPGA address r <addr> - Read back at FPGA address dd <ald> - Dump device's register contents wd <ald><addr> <val> - Write value to the device's address rd <ald><addr> - Read value from the device's address sa <c r><0~31.75> - Set PE43713 attenuators in dB Device Key: g g = GPIO a = EV12AS350A d = EV12DS460AZP</c r></addr></ald></val></addr></ald></ald></addr></val></addr>	d <g> - Dump FPGA register contents</g>			
r <addr>- Read back at FPGA addressdd <a d>- Dump device's register contentswd <a d> <addr> <val>- Write value to the device's addressrd <a d> <addr>- Read value from the device's addresssa <c r><0~31.75>- Set PE43713 attenuators in dBDevice Key:$g = GPIO$a = EV12AS350A$d = EV12DS460AZP$</c r></addr></a d></val></addr></a d></a d></addr>	w <addr> <val> - Write value at FPGA address</val></addr>			
$ \begin{array}{llllllllllllllllllllllllllllllllllll$	r <addr> - Read back at FPGA address</addr>			
$ \begin{array}{llllllllllllllllllllllllllllllllllll$	dd <a d> - Dump device's register contents</a d>			
$a_{A}(2 < a_{A}(2 < a_{A$	wd <ald> <addr> <val> - vvr</val></addr></ald>	Ite value to the device's address		
Device Key: g = GPIO a = EV12AS350A d = EV12DS460AZP	sa <c r=""></c>	at Value from the device's address		
Device Key: g = GPIO a = EV12AS350A d = EV12DS460AZP				
g = GPIO a = EV12AS350A d = EV12DS460AZP	Device Key:			
a = EV12AS350A d = EV12DS460AZP	g = GPIO			
d = EV12US460AZP	a = EV12AS350A			
a m Atter DE 42742 for front panel (211/ nort				
r = Attn = FE43712 for front panel REE CLK port				

Figure 3: Console Command Tool

Related Products

•

VPX004





VTX870



- Unified 1 GHz quad-core CPU for, Shelf Manager, and Fabric management
- Automatic fail-over with redundant VPX004
- 1GbE base switch with dual 100/1000/10G uplink
- 3U ADC 12-bit @ 5.4 GSPS and DAC 12-bit @ 6 GSPS, Virtex UltraScale+
- PCIe Gen3 x 16 (or x8 or x4) or No-PCIe SERDES option (SRIO, XAUI, Aurora)
- Direct RF clock or on-board PLL option
- Open VPX benchtop development platform
- Dedicated Switch/management slot
- Up to five 3U VPX payload slots

Contact

VadaTech Corporate Office

198 N. Gibson Road, Henderson, NV 89014 Phone: +1 702 896-3337 | Fax: +1 702 896-0332

Asia Pacific Sales Office

7 Floor, No. 2, Wenhu Street, Neihu District, Taipei 114, Taiwan Phone: +886-2-2627-7655 | Fax: +886-2-2627-7792

VadaTech European Sales Office

VadaTech House, Bulls Copse Road, Southampton, SO40 9LR Phone: +44 2380 016403

info@vadatech.com | www.vadatech.com

Choose VadaTech

We are technology leaders

- · First-to-market silicon
- · Constant innovation
- Open systems expertise

We commit to our customers

- · Partnerships power innovation
- · Collaborative approach
- Mutual success

We deliver complexity

- · Complete signal chain
- System management
- Configurable solutions

We manufacture in-house

- Agile production
- · Accelerated deployment
- AS9100 accredited



Trademarks and Disclaimer

The VadaTech logo is a registered trademark of VadaTech, Inc. Other registered trademarks are the property of their respective owners. AdvancedTCA[™] and the AdvancedMC[™] logo are trademarks of the PCI Industrial Computers Manufacturers Group. All rights reserved. Specification subject to change without notice.

> © 2019 VadaTech Incorporated. All rights reserved. DOC NO. 4FM737-12 REV 01 | VERSION 1.1 – MAR/19

