FPGA Implementation of QAM Transmitter and Receiver

D.Dayakara Reddy, S.Karunakar Reddy
VLSI SYSTEM DESIGN
Aurora's Technological & Research Institute Hyderabad, India
Assoc. Professor Aurora's Technological & Research Institute Hyderabad, India

Abstract—The FPGA technology has been playing a considerable role in Rportable and mobile communication. This is due to the features of flexibility, accuracy and configurability in designing and Implementation. The project presents a complete design for QAM Transmitter and Receiver based on the Spartan 3e FPGA kit. The implemented system can be used in typical wimax system and any other communication systems. The carrier synchronization and timing synchronization issues are covered in the implementation. The transmitter of QAM consists of data sampler, phase accumulator, symbol mapper, NCO. The receiver of QAM consists of phase locked loop, symbol demapper, clock distributer. Modelsim Xilinx edition(MXE) will be used for simulation and functional verification. Xilinx ISE will be used for synthesis, and bit file generation. Xilinx FPGA board will be used for testing and demonstration of the implemented system.

keywords: FPGA design, Carrier Synchronization, Symbol Recovery, QAM modulation;

I INDRODUCTION

Quadrature Amplitude Modulation or QAM is a form of modulation which is widely used for modulating data signals onto a carrier used for radio communications. It is widely used because it offers advantages over other forms of data modulation such as PSK, although many forms of data modulation operate along side each other**.**

Quadrature Amplitude Modulation, QAM is a signal in which two carriers shifted in phase by 90 degrees are modulated and the resultant output consists of both amplitude and phase variations. In view of the fact that both amplitude and phase variations are present it may also be considered as a mixture of amplitude and phase modulation.

I Analogue and Digital QAM

Quadrate amplitude modulation, QAM may exist in what may be termed either analogue or digital formats. The analogue versions of QAM are typically used to allow multiple analogue signals to be carried on a single carrier. For example it is used in PAL and NTSC television systems, where the different channels provided by QAM enable it to carry the components of chroma or colour information. In radio applications a system known as C-QUAM is used for AM stereo radio. Here the different channels enable the two channels required for stereo to be carried on the single carrier.Digital formats of QAM are often referred to as "Quantised QAM" and they are being increasingly used for data communications often within radio communications systems. Radio communications systems ranging from cellular technology through wireless systems including WiMAX, and Wi-Fi 802.11 use a variety of forms of QAM, and the use of QAM will only increase within the field of radio communications.

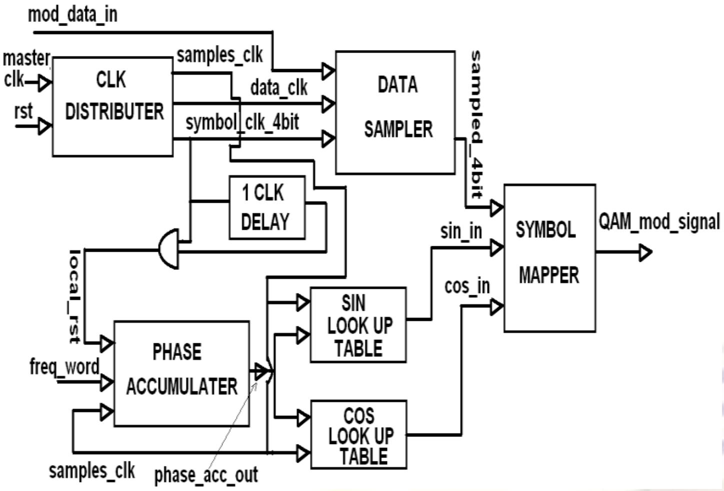
A. Digital / Quantised QAM basics

Quadrature amplitude modulation, QAM, when used for digital transmission for radio communications applications is able to carry higher data rates than ordinary amplitude modulated schemes and phase modulated schemes. As with phase shift keying, etc, the number of points at which the signal can rest, i.e. the number of points on the constellation is indicated in the modulation format description, e.g. 16QAM uses a 16 point constellation.

When using QAM, the constellation points are normally arranged in a square grid with equal vertical and horizontal spacing and as a result the most common forms of QAM use a constellation with the number of points equal to a power of 2 i.e. 2, 4, 8, 16 . . . . By using higher order modulation formats, i.e. more points on the constellation, it is possible to transmit more bits per symbol. However the points are closer together and they are therefore more susceptible to noise and data errors.

II QAM TRANSMITTER DESIGN

Block diagram of QAM modulator is given above every block is explained below in detail. Clk_distributor the inputs for the clock generation module is reset and master clock of 50Mhz and the outputs are sample_clk, data_clk and symbol_clk. The sample_clk is same as the master_clk. The clock generation module consists of 10 bit counter. In the counter 8th bit value gives the output of data_clk and the 10th gives the output of symbol_clk . The data_clock will be at active low for 256 clock pluses and for other 256 clock pluses will be in active high to complete one cycle. Symbol_clk will be at active low for 1024 clock pluses and for other 1024 clock pluses will be in active high to complete one cycle



Rst is used to clear or reset module, data_clk will be four times faster then symbol_clk_4bit When rst is active high then left shift register will be cleared, if rst is active low with raising edge of data_clk then data_in will be forced on to the LSB bit of left shift register. Left shift register will be shifter one bit. Raising edge of symbol_clk_4bit then left shift register will be reflected on the output sampled_4bit.

Clk is used for synchronication and rst is used to clear two registers that are phase increment registrer and phase register. The phase increment register stores the instantaneous phase_inc_word that is fed to a 8 bit adder as one of its input**.** The other input for adder is phase register output. The phase register holds the instantaneous phase for each clock pulse (i.e adder output). The accumulated phase also is also 8 bits, which limits the maximum phase by 11111111, and addition by 1 to maximum value causes the phase to become 00000000

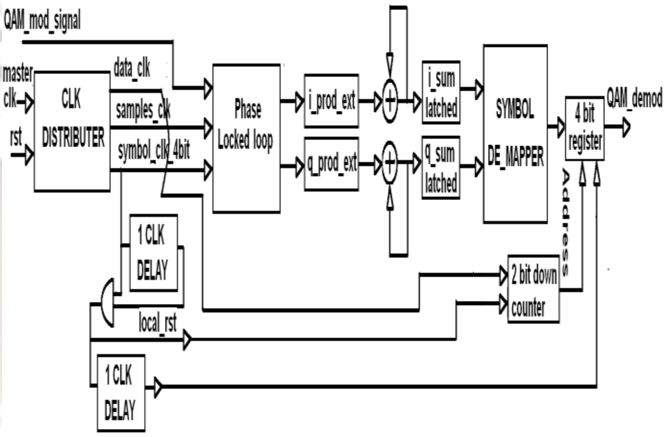
SIN AND COS LUT'S

According to the address given the predefined data stored in 8 bit rom will reflected on the output pin Amp_Bits. This is expected and desired since the Look Up Tables are programmed to consider 255 as highest phase value and phase increment by one results next cycle of waveform. Since 8 bits are used to represent the 0^{O} to 360^{O} the increment in digital phase value by one causes effective increment of 1.40625^{O} (results by dividng 360^{O} with 256 maximum possible combinations of 8 bits) **.** This also implies that outputs can’t have more that 256 samples for one cycle.

SYMBOL MAPPER

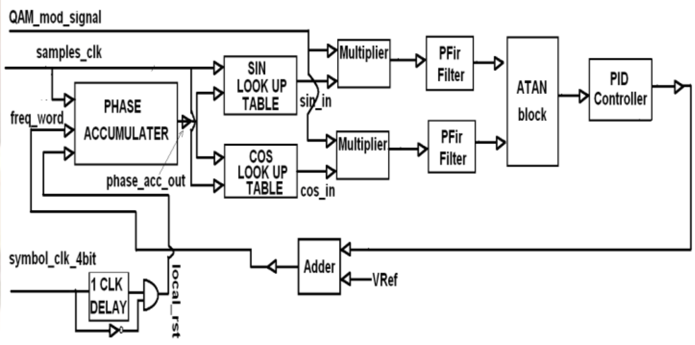
This incremented phase is multipled with sin and cos for sin_lut and cos_lut respectively. The resultant amplitude will be saved in the 8 bit Rom block. So according to address that will give ampltude of sin or cos signal by using this LUT’s Sampled_4bit data is loaded in to a tempary register there 0th bit and 1st bit are assumed as I bits and 2nd and 3rd bit are asumed as Q bits, these I bits and Qbits are assumed as address to 3 bit Rom block’s in this according to Rom address data in that address will be given to the next block called as multipler’s. I bits are multipled with cos_in bits and Q bits are multipled with sin_in bits than resultant output’s of the multipler section will be added which generates the final output as QAM_mod_signal.

III QAM RECEIVER DESIGN



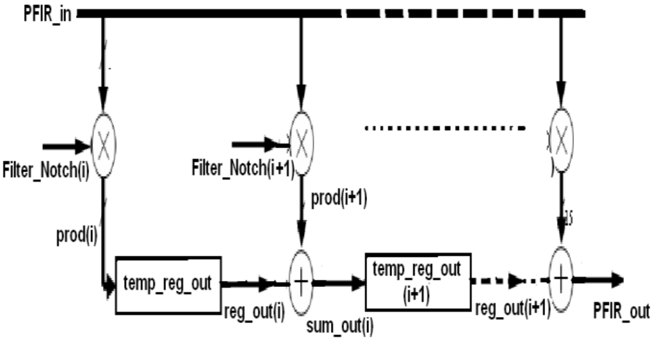
The system generator library provides a demo for QAM baseband demodulator which consists of two main blocks: Carrier recovery and Symbol demapper. Fast computing techniques using CORDIC is employed to correct the phase error. The adaptive algorithm does work to minimize errors. However, because of these features, this design cannot be applied to deep fading channels phenomenon.

PHASE LOCKED LOOP



Phase locked loop is used to correct the signal at demodulator it’s plays a key role in demodulation. The carrier recovery can be done by using phase locked loop. It consists of phase accumulator. Phase accumulator is the combination of phase register and phase increment register. The phase increment register stores the instantaneous phase_inc_word that is fed to a 8 bit adder as one of its input. The other input for adder is phase register output. The phase register holds the instantaneous phase for each clock pulse (i.e adder output). The accumulated phase also is also 8 bits, which limits the maximum phase by 11111111, and addition by 1 to maximum value causes the phase to become 00000000.

PFIR module:



The package created which consist of the filter coefficients will be utilized by calling the package in to program; PFIR structure is shown in above figure. Input is taken as PFIR_in and is given to a multiplier in a loop and another input to multiplier is the filter coefficient (i) stored in the PFIR package. Result generated by the multiplier will be added with another multiplier in the loop (i+1). Loop will be running till last coefficient is multiplied and result is reflected on to output signal PFIR_out .

Pid controller:

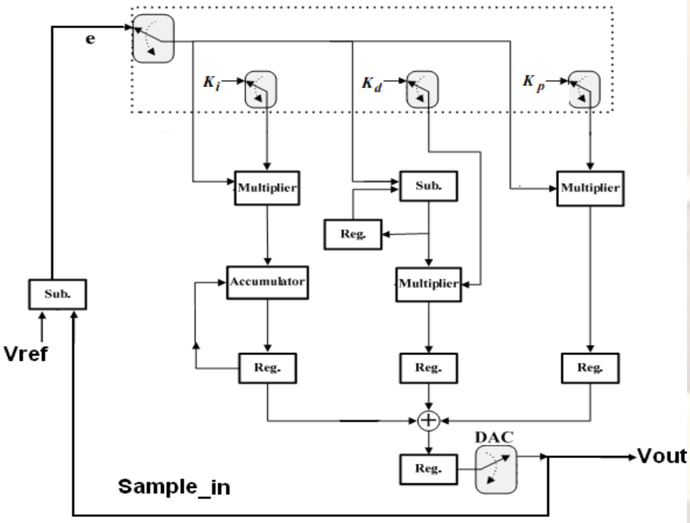


Figure 3.3 Block diagram of PID controller

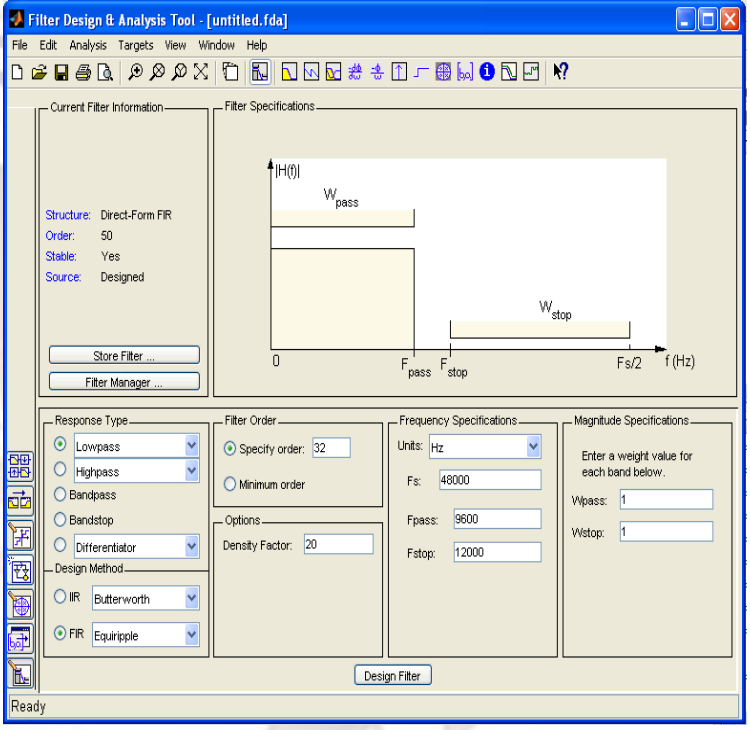
The output of the system is taken as input (i.e sample_in) of pid controller as that is compared with vref to generate error signal. Pid control has three section that is integration section, differentiation section and proportional section. In Proportional section gain K_p is multiplied by error and produces K_p_prod signal but in integration section accumulation(i.e past error with present error) of the error is multiplied by gain K_i to produce K_i _prod and the subtraction of the past error from present error generated result is multiplied by K_d_prod. Addition of K_i_prod, K_d _prod and K_p_prod generates vout which is given to the system to control

Coefficient Generation for low Pass Noise Removal Filter**:**

Low Pass filter.

The filter coefficients are generated by mat lab by given process below

Start → Mat lab → toolboxes → filter design HDL coder → filter design and analysis tool (FDA tool). Open FDA tool then select as shown



Structure: Direct-Form FIR
Order: 50
Stable: Yes
Source: Designed

Response Type: ● Lowpass ○ Highpass ○ Bandpass ○ Bandstop ○ Differentiator

Design Method: ○ IIR Butterworth ● FIR Equiripple

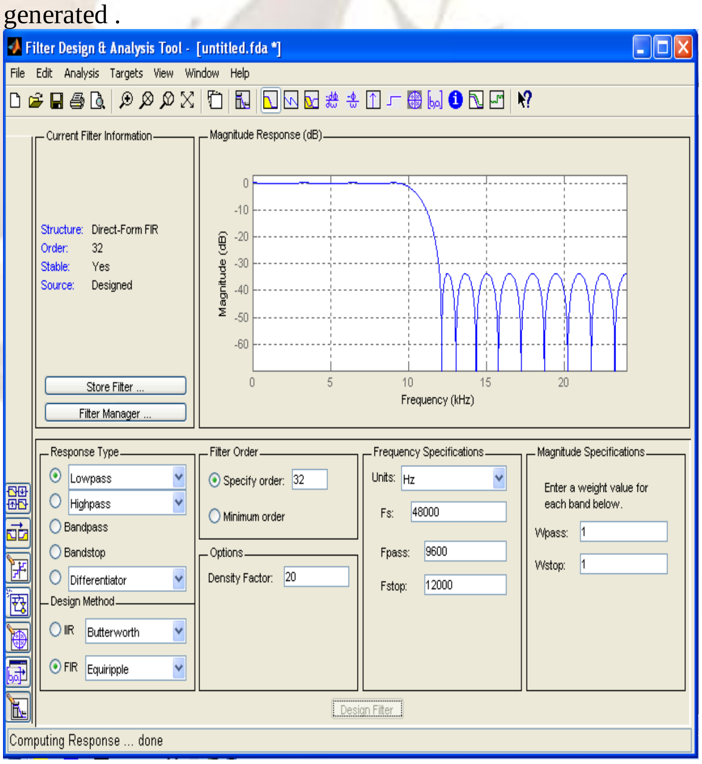
Filter Order: ● Specify order: 32 ○ Minimum order

Options: Density Factor: 20

Frequency Specifications: Units: Hz; Fs: 48000; Fpass: 9600; Fstop: 12000

Magnitude Specifications: Enter a weight value for each band below. Wpass: 1; Wstop: 1

Click design filter. Then the coefficients will be generated .


Filter Design & Analysis Tool - [untitled.fda *]
File Edit Analysis Targets View Window Help
Current Filter Information
Magnitude Response (dB)
0
-10
-20
-30
-40
-50
-60
Magnitude (dB)
0
5
10
15
20
Frequency (kHz)
Store Filter ...
Filter Manager ...
Design Filter
Computing Response ... done

Structure: Direct-Form FIR
Order: 32
Stable: Yes
Source: Designed

Response Type: ● Lowpass ○ Highpass ○ Bandpass ○ Bandstop ○ Differentiator

Design Method: ○ IIR Butterworth ● FIR Equiripple

Filter Order: ● Specify order: 32 ○ Minimum order

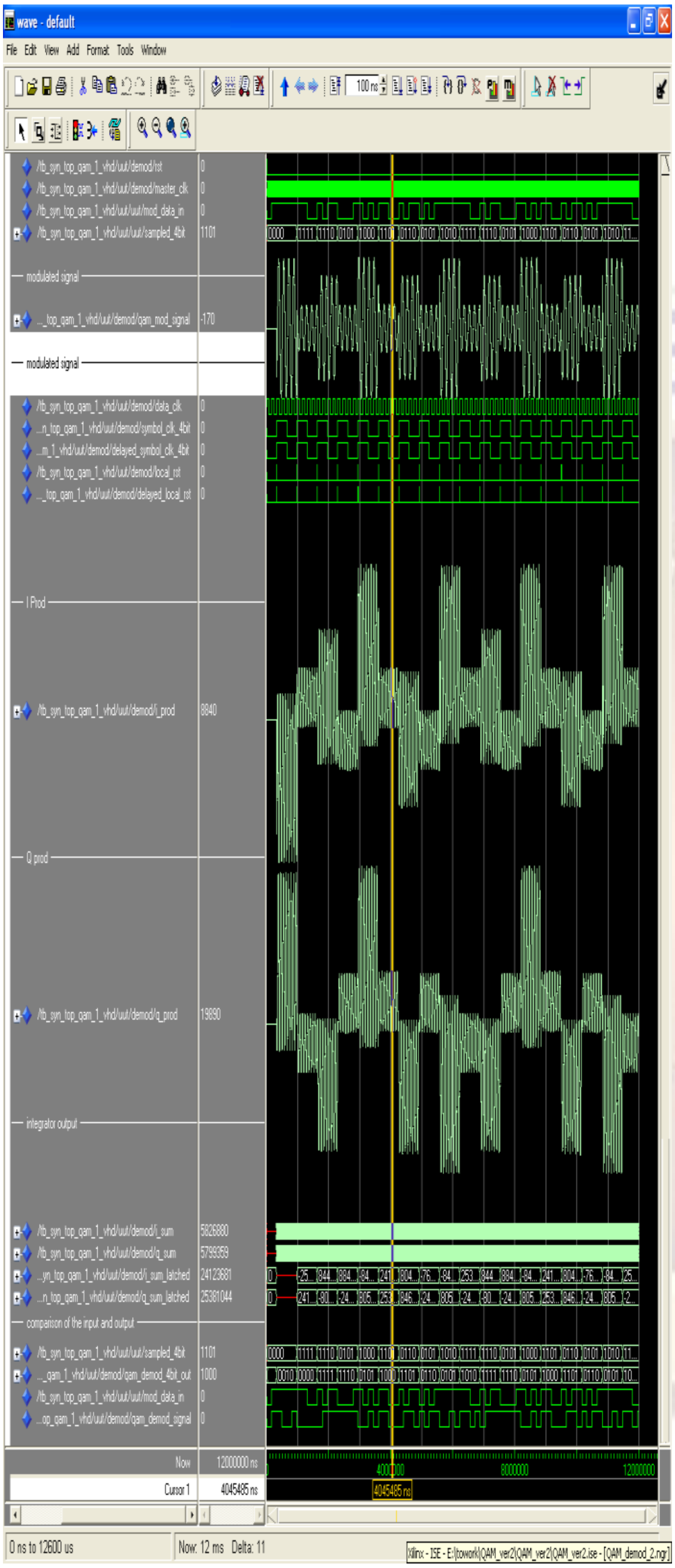
Options: Density Factor: 20

Frequency Specifications: Units: Hz; Fs: 48000; Fpass: 9600; Fstop: 12000

Magnitude Specifications: Enter a weight value for each band below. Wpass: 1; Wstop: 1

IV . RESULTS:

Simulation Waveform



/tb_syn_top_qam_1_vhd/uut/demod/rst	0
/tb_syn_top_qam_1_vhd/uut/demod/master_clk	0
/tb_syn_top_qam_1_vhd/uut/uut/mod_data_in	0
/tb_syn_top_qam_1_vhd/uut/uut/sampled_4bit	1101
modulated signal	
..._top_qam_1_vhd/uut/demod/qam_mod_signal	-170
modulated signal	
/tb_syn_top_qam_1_vhd/uut/demod/data_clk	0
...n_top_qam_1_vhd/uut/demod/symbol_clk_4bit	0
...m_1_vhd/uut/demod/delayed_symbol_clk_4bit	0
/tb_syn_top_qam_1_vhd/uut/demod/local_rst	0
..._top_qam_1_vhd/uut/demod/delayed_local_rst	0
I Prod	
/tb_syn_top_qam_1_vhd/uut/demod/i_prod	8840
Q prod	
/tb_syn_top_qam_1_vhd/uut/demod/q_prod	19890
integrator output	
/tb_syn_top_qam_1_vhd/uut/demod/i_sum	5826880
/tb_syn_top_qam_1_vhd/uut/demod/q_sum	5799359
...yn_top_qam_1_vhd/uut/demod/i_sum_latched	24123681
...n_top_qam_1_vhd/uut/demod/q_sum_latched	25381044
comparison of the input and output	
/tb_syn_top_qam_1_vhd/uut/uut/sampled_4bit	1101
..._qam_1_vhd/uut/demod/qam_demod_4bit_out	1000
/tb_syn_top_qam_1_vhd/uut/uut/mod_data_in	0
...op_qam_1_vhd/uut/demod/qam_demod_signal	0
Now	12000000 ns
Cursor 1	4045485 ns

0 ns to 12600 us Now: 12 ms Delta: 11

4000000
8000000
12000000
4045485 ns
Xilinx - ISE - E:\towork\QAM_ver2\QAM_ver2\QAM_ver2.ise - [QAM_demod_2.ngr]

V. CONCLUSIONS

A computationally efficient algorithm is developed for a low power and versatile ASIC design of QAM transmission system. Utilization of this algorithm in the development of mobile and portable communications requires not only high performance of hardware systems but also affectivity and flexibility in design and implementation. Simulation(Modelsim6.2c), synthesis(xilinx ISE), implementation(spartan 3e) and verification of design after implementation is successfully completed.

REFERENCES

[1] 16 QAM transmitter and receiver design based on FPGA IEEE 2010, Xuan-thang Vu, Nguyen Anh Duc, Trinh Anh Vu

[2] C. Dick, F.Harris, M.Rice, *Synchronization in Software Radios-Carrier and Timing Recovery* Using FPGAs, Proceeding of 2000 IEEE Symposium on Field-Programmable Custom Computing Machines.

[3] C.Dick, F.Harris, M.Rice, *FPGA Implementation of Carrier Synchronization for QAM Receivers*, Journal of VLSI Signal Processing, Vol 36, p 57-71,2004.

[4] Joaquin Garcia, Rene Cumplido, *On the design of an FPGA-Based OFDM modulator* for IEEE 802.11a, Proceeding of ICEEE, September 7-9, 2005, Mexico.

[5] John G. Proakis, *Digital Communication*, McGraw-Hill 1993.

[6] *VLSI for Wireless communication*, Bosco Luen Pearson Education and VLSI series

[7] http://www.xilinx.com