MAXIMUM LIKELIHOOD CARRIER PHASE SYNCHRONIZATION IN FPGA-BASED SOFTWARE DEFINED RADIOS

Michael Rice
Brigham Young University
Provo, Utah

Chris Dick
Xilinx, Inc.
San Jose, California

fred harris, Signal Processing Chair
San Diego State University
San Diego, California

ABSTRACT

Digital signal processing techniques are applied to maximum likelihood carrier phase synchronization for QPSK and QAM in an all-digital sampled data receiver. To achieve the flexibility required by modern Software Defined Radios (SDR's), this task must either be performed in a DSP processor (reconfigurable software) or in an FPGA (reconfigurable hardware). This paper describes the design process for an FPGA-based design and summarizes the FPGA resources required for QPSK carrier phase synchronization.

1. INTRODUCTION

The last two-decades has borne witness to a steady trend of migrating many radio functions, traditionally performed by analog processing tasks, to DSP based implementations. In conjunction with this DSP insertion, we have seen the boundary between the analog and digital segments in the signal conditioning chain move inexorably towards the antenna. The implementation of these high-performance digital communication systems has been made possible by advances in semiconductor process technology in the form of application specific standard parts (ASSPs), full custom silicon chips, instruction set based digital signal processors (DSPs) and high performance general-purpose processors (GPP). In current generation systems, the hardware solution is often best provided using a heterogeneous computing approach, using the silicon appropriate to each particular function. In the early 1990's field programmable gate arrays (FPGAs) also played a role in digital communication hardware where they were often applied as glue logic to support bus interfacing, complex state machines and memory controller tasks. In recent years, FPGA technology has undergone revolutionary changes. Gate densities and clock speeds of recent generation FPGAs provide the communication system architect with a highly configurable logic fabric that can be used for realizing sophisticated real-time signal processing functions.

The ever-increasing demand for mobile and portable communication forces two competing requirements on system design: 1) the requirement for high-performance systems employing advanced signal processing techniques to allow operation with very small implementation losses, and 2) the requirement to respond to market and fiscal pressures to easily accommodate evolving and fluid standards. Software defined radios (SDRs) are emerging as a viable solution for meeting the conflicting demands in this arena. SDRs support multimode and multi-band modes of operation and allow service providers an economic means of future-proofing these increasingly complex and costly systems.

One of the challenging tasks in a communication system is carrier and symbol timing recovery. A large amount of time is spent solving these problems, and frequently a large amount of hardware and software in a SDR is dedicated to synchronization [1,2]. This paper examines techniques for developing, modeling and generating FPGA implementations of carrier synchronization loops for QPSK and QAM modulations.

2. CARRIER RECOVERY IN AN SDR

The basic operations required by an all digital QPSK or QAM receiver are illustrated in Figure 1. QPSK and QAM signals carrier information on the amplitudes of the quadrature carriers. The quadrature down-conversion and matched filter operations produce estimates of the quadrature amplitudes which are the basis of the data decisions. The role of carrier phase synchronization is to perform the quadrature down-conversion using phase coherent replicas of the quadrature carriers.

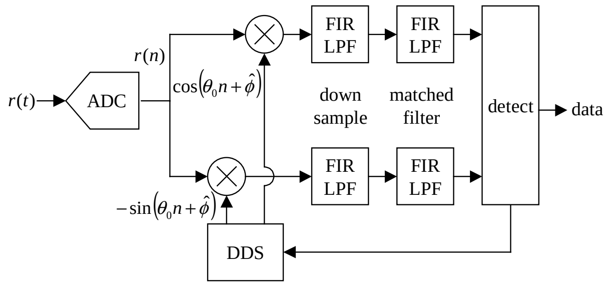


Fig. 1. Block diagram of basic QPSK/QAM digital receiver.

There are many options for implementing carrier phase and frequency synchronization in a digital communication system. At the heart of all synchronizers is the phase-locked loop (PLL). An all-digital receiver can be implemented with a digital phase-locked loop (DPLL) as shown in Figure 2. This DPLL employs a proportional plus integral loop filter formed by a scaled digital integrator and a scaled direct path. The filter coefficients K_P and K_I control the PLL bandwidth and damping factor. In the digital implementation, the VCO takes the form of a direct digital synthesizer (DDS). The phase detector is implemented using the arctangent operation suggested by the ATAN block.

The most complex component in the loop is the phase detector. Since the phase of QPSK or QAM signals is data dependent, the phase detector must strip the modulation from the received signal and produce a signal proportional to the phase difference between the local generated quadrature carriers and those of the received signal. The complexity of the phase detector can be reduced by computing a signal proportional to the sine of the phase difference $\Delta\phi = \phi - \hat{\phi}$. Note that $\sin(\Delta\phi)$ is monotonic with $\Delta\phi$ for $-\pi/2 \le \Delta\phi \le \pi/2$ and is a good phase estimator in that interval. The periodicity of the sine function produces the $\pi/2$ phase ambiguity associated with the phase detector. For small $\Delta\phi$, $\sin(\Delta\phi) \approx \Delta\phi$ so that the sine function approximates the ideal phase detector for small $\Delta\phi$.

The reduced complexity phase detector for QPSK is illustrated in Figure 3. The phase error is computed by comparing the phase difference between the received signal $X(n)+jY(n)$ and the closest constellation point $\hat{I}(n)+j\hat{Q}(n)$ as illustrated in Figure 3. For this constellation, the four constellation points are located in the center of the four quadrants so that the nearest constellation point corresponding to the received signal is easily computed by taking the sign of the inphase and quadrature amplitudes. The sine of the phase difference may be expressed as

$$\sin(\Delta\phi(n)) = \frac{\hat{I}(n)Y(n)-\hat{Q}(n)X(n)}{\sqrt{\hat{I}^2(n)+\hat{Q}^2(n)}\sqrt{X^2(n)+Y^2(n)}} \quad (1)$$

which shows that a phase error signal proportional to the sine of the phase difference can be generated by forming the difference of the cross product as illustrated.

The phase detector for a received signal with a denser modulation constellation, such as 16-QAM, must conduct additional comparisons (other than the sign) to bracket the location of the received two-tuples. This function is incorporated into the phase detector using a 2-dimensional slicer as illustrated in Figure 4. The phase difference is still given by Equation (1) so that the cross product difference is still computed. Note that detected points in the 16-QAM constellation reside on one of three contours indicating distance from the origin. The larger radii contours represent signals with higher signal to noise ratio. Phase measurements for data points with larger radii contain less noise and the PLL should take advantage of this side information and rely more heavily on measurements with high SNR. The block labeled "DET GAIN" in the phase detector forms the SNR dependent gain by assigning different scalar gains as a function of the radius of the detected two-tuple. An example of the gain assignment is gains of (1/4, 1/2, and 1) assigned to detected two-tuples of radii (R_1, R_2, and R_3) respectively.

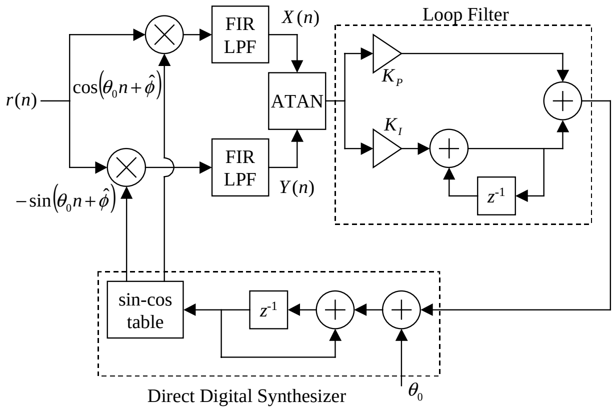
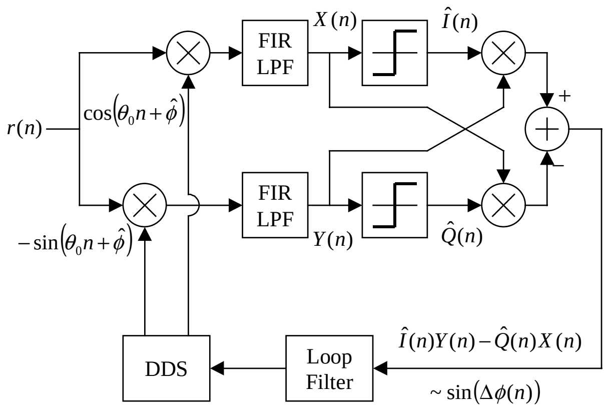


Fig. 2. Digital phase locked loop for carrier phase synchronization in an all-digital QPSK/QAM receiver.

3. QPSK PLL IMPLEMENTATION

To produce a fixed-point arithmetic realization of the QPSK PLL in Figure 3 a combination of Matlab [3] and Simulink [4] were used. A floating-point model was developed and simulated in Simulink to verify satisfactory loop operation. After the Simulink floating-point model was completed and verified, approximately 30 minutes was required to generate the fixed-point, or quantized, solution. (This approach also provided a simple mechanism to compare corresponding values in the floating-point and fixed-point models.) Figure 5 illustrates the fixed-point arithmetic model. After the quantized model was verified using simulations in Simulink, a conventional FPGA implementation flow using VHDL and the Xilinx Core Generator are used to produce the final design.

Operation of the PLL was verified by simulating the carrier phase PLL performance in the presence of a Doppler shift and unknown phase offset. The transmitter generated a pseudo random complex sequence that was shaped by a multirate transmit filter (square-root raised cosine with an excess bandwidth α= 0.25) and an interpolation factor of 1-to-8. To simulate Doppler shift, the model introduced a small frequency translation of the carrier. The simulation results are summarized in Figures 6 where constellation diagrams are illustrated during the periods of start-up, acquisition, and lock. Observe that during startup, the constellation rotates at the Doppler shift as illustrated in Figure 6 (a). As the PLL pulls the frequency and phase of the DDS outputs toward the frequency and phase of the received carrier, the constellation stops rotating as shown in Figure 6 (b). During lock, the constellation points are locked to the four possible QPSK constellation points as shown in Figure 6 (c). Figure 7 (a) is a plot of the received carrier phase and PLL phase as a function of time. The slope of the phase is the Doppler shift. Observe that the loop attains lock after a few thousand samples (a few hundred symbols). The difference between the two-phase trajectories, or phase error, is plotted in Figure 7 (b).



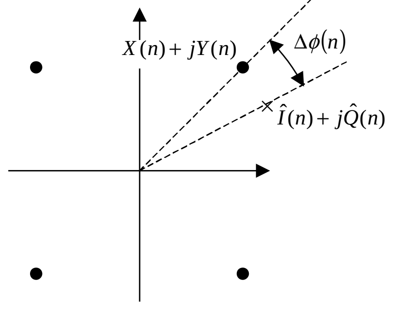
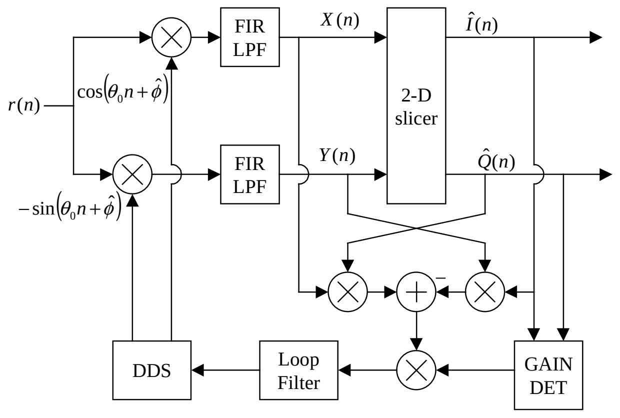


Fig. 3. QPSK carrier phase PLL using the cross product phase detector and the constellation. The difference of the cross product is proportional to the sine of the phase error.



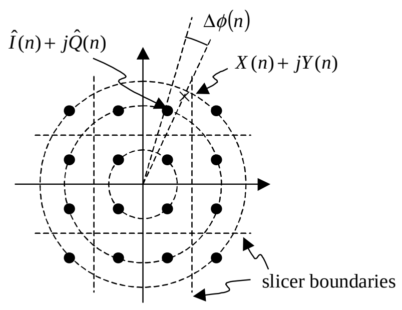


Fig. 4. 16-QAM carrier phase PLL with SNR dependent gain and constellation with decision boundaries.

4. AN FPGA IMPLEMENTATION

Several functional units are required to implement the carrier recovery loop. The complex heterodyne required to down-convert the input signal is a complex multiplier. There are two matched filters, one for each of the I and Q arms. The phase detector is straightforward, consisting of two 1-bit slicers (sign detector), two 2's complementers and a subtractor. The second-order loop filter is realized using two multipliers, an integrator and an adder. The DDS requires an accumulator, and adder and a sine/cosine look-up table. The complex multiplier is implemented using 4 real multipliers and two real adders. The ADC samples are represented using 8-bits, while the heterodyning signal employs 12-bit samples. Each 8 x 12 multiplier occupies approximately 81 logic slices. The complete multiplier occupies 344 slices. The recursive nature of the PLL requires the use of purely combinatorial multipliers and adders. Two matched filters are required. One for each of the I and Q processing arms. These filters are 97-tap symmetrical FIR filters with 12-bit coefficients and 9-bit input samples. The filters were generated using the Xilinx Core Generator [6] filter compiler. The implementation employs serial distributed arithmetic. Taking advantage of the symmetrical coefficient data, each filter occupies 248 Virtex FPGA [5] logic slices. Using 9-bit input samples, the filter requires 10 clock cycles to compute a new output. The bit-clock for the filter is a function of the FPGA speed grade. Typical values are between 100 and 150 MHz. This translates to a sample throughput of 10 to 15 M-Samples/sec. The multipliers in the loop filter occupy most of the logic resources for this sub-system. The coefficient parameters are represented using 16-bits while the input samples are carried with 9-bit precision. The complete loop filter occupies 80 slices. The DDS was implemented using a simple phase truncation architecture [7]. Using the quantized Simulink model, a 1024-point sin/cos look-up table (LUT) was formed with 12-bit precision samples, a precision adequate for this application. Using quarter cycle symmetry [7], the sin/cos LUT requires only a single Virtex block RAM (BRAM) [5]. The dual-port nature of the BRAM permits simultaneous computation of I and Q samples. The DDS phase accumulator consists of a 28-bit adder and register. These components occupy a modest 14 logic slices. The complete QPSK PLL occupies approximately 1000 logic slices.

5. CONCLUSION

An overview of how carrier phase synchronization in QPSK and QAM systems can be realized using FPGA technology. The design example illustrates how sampled-data receivers can be designed with FPGA's and that these designs can achieve the performance required by future generation digital communications standards. The software in a SDR defines the system personality, but currently, the implementation is often a mix of analog hardware, ASICs, FPGAs and DSP software. The rapid uptake of state-of-the-art semiconductor process technology by FPGA manufacturers is opening-up new opportunities for the effective insertion of FPGAs in the SDR signal conditioning chain. Functions frequently performed by ASICs and DSP processors can now be done by configurable logic.

6. REFERENCES

[1] H. Meyr, M. Moeneclaey and S. A. Fechtal, *Digital Communication Receivers*, John Wiley & Sons Inc., New York, 1998.

[2] f.j harris and C.H. Dick, "On Structure and Implementation of Algorithms for Carrier and Symbol Synchronization in Software Defined Radios", EUSIPCO-2000, "Efficient Algorithms for Hardware Implementation of DSP Systems", Tempere, Finland, 5-8 Sept. 2000.

[3] The Mathworks Inc, Matlab, *Getting Started with Matlab*, Natick, Massachusetts, U.S.A, 1999.

[4] The Mathworks Inc, Simulink, *Dynamic System Simulation for Matlab, Using Simulink*, Natick, Massachusetts, U.S.A, 1999.

[5] Xilinx Inc., *The Programmable Logic Data Book*, 1999.

[6] Xilinx Core Generator System, http://www.xilinx.com/products/logicore/coregen/index.htm

[7] C. H. Dick and f. j. harris, "Direct Digital Synthesis - Some Options for FPGA Implementation," SPIE International Symposium On Voice Video and Data Communication: Reconfigurable Technology: FPGAs for Computing and Applications Stream. Boston, MA, USA, pp. 2-10, September 20-21 1999.

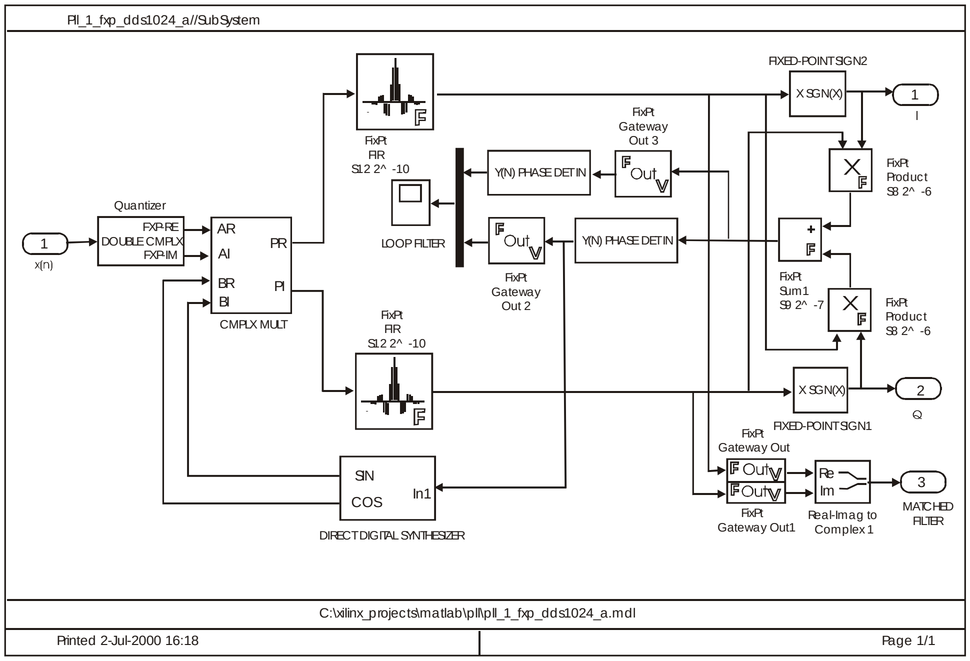


Fig. 5. Simulink model of the QPSK PLL using fixed point arithmetic.

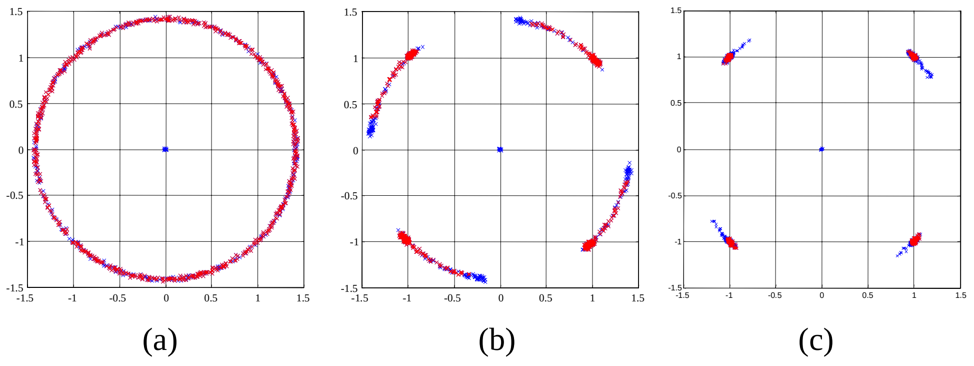
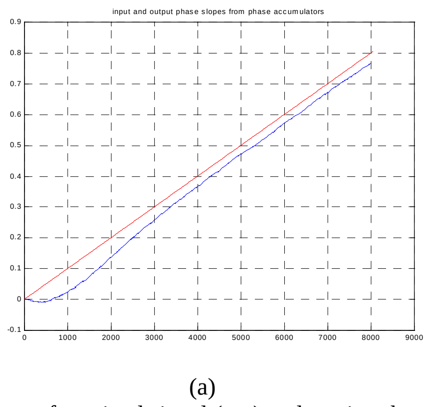


Fig. 6. QPSK constellation diagrams during PLL simulation: (a) during the initial phase of acquisition; (b) during the transition from acquisition to phase lock; (c) during phase lock.



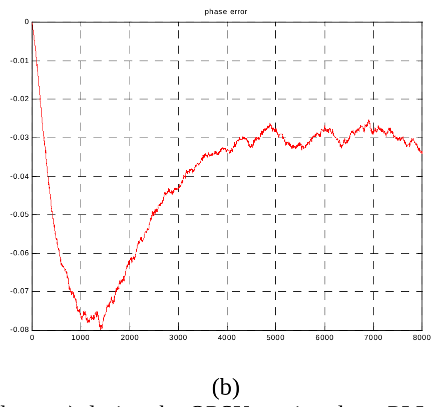


Fig. 7. (a) Phase vs. time of received signal (top) and carrier phase PLL (bottom) during the QPSK carrier phase PLL simulation. (b) Phase error vs. time during the QPSK carrier phase PLL simulation.