**Journal of Computing and Electronic Information Management**
ISSN: 2413-1660 | Vol. 15, No. 1, 2024

# QPSK circuit optimization Based on Model-Based Design

**Yang Nie**

School of Physics and Electronic Information Engineering Jining Normal University, Ulanqab, China
nieyangwork@163.com

**Abstract:** In order to address the difficult problem of hardware implementation of complex communication algorithm, this paper takes digital phase modulation as an example, and uses model-based design approach to quickly complete the algorithm modeling, simulation and automatic generation of HDL code. Model-based design is a fast and effective design process for algorithm modeling, simulation, testing and automatic code generation. The simulation results show that the model-based design method cannot only quickly accomplish the modeling and testing of the algorithm, but also improve the design accuracy.

**Keywords:** Model-Based Design; FPGA; Wireless Communication.

## 1. Introduction

To reduce development costs for a wireless communication system while speeding up development and maintaining reliability, the industry has moved towards reconfigurable architecture that can adapt to future mission requirements and handle system failures. In communication systems, all kinds of complex algorithms, such as modulation and demodulation, are required to complete the hardware implementation. Currently, these algorithms are run on hardware by ASIC, DSP and FPGA. FPGA used widely in high-speed, parallel to achieve aspects of digital signal processing algorithms. With the continuous improvement of semiconductor process technology, FPGA is widely used to implement high-performance DSP system. Owing to the lack of efficient design methods, many are familiar with C language programming signal processing engineers, which cannot effectively use FPGA. Therefore, an efficient design method is needed to help them complete the hardware design.

Model-Based Design (MBD) is innovating the way for engineers and scientists working. In Model-Based Design, a system model is in the center of the development process, from requirements development, through design, implementation, and testing [1]. Model-based design gives us a complete product from idea to generate the code development process. In MBD, a system model is right in the center around the development process, from requirement's progress, through design, implementation, and testing [2,3]. This workflow is elaborated to create hardware and software partitioning, automatically create hardware and software implementation code, and verify the hardware and software implementations in the context of the complete system. It is not just overcoming the defects of low efficiency and difficulty of meeting the requirements in traditional methods of progress, but also to meet time-to-market and cost objectives.

This paper takes the digital phase shift keying as an example, and completes the modeling, simulation and automatic generation of HDL code using the MBD method. The paper is organized as follows: In Section II, the difference between the traditional design process and the model-based design process is introduced. Section III discusses the basic principle of QPSK, which are most trustworthy and efficient for hardware implementation. The main contribution of this work is described in section IV, where the QPSK model of the transmitter is optimized and verified using MBD. Finally, Section V summarizes the main conclusions of this work.

## 2. Model-Based Design

### 2.1. Traditional system development process

The traditional system development process is characterized by using of paper specification generated from requirements gathered from various sources building physical prototypes for design. Writing code manually to implement control algorithms and complex logic. Dependence on a test system for verification and validation. Traditional system development process is shown in Fig 1, where design, implementation, and test is done by individual team each suffering from the barrier of communication with its supplier and customer.

A significant problem with this process is that errors in the specification are often not discovered until the final validation, which leads to specification change, redesign, re-implementation, and re-validation. Consequently, cost of fixing errors is high. Because of the many system configurations in a commercial system, complete test coverage of the design is practically impossible using a test system. As a result, errors may be leaked into products, leading to field repairs.

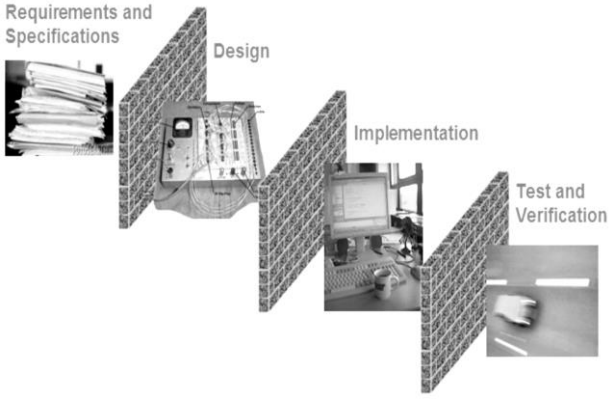


**Fig.1** Traditional system development process

## 2.2. Development process with Model-Based Design

In Model-Based Design, the development process centers around a system model from requirements capture and design to implementation and test. This system model is the heart of an executable specification which is used and elaborated throughout the design flow. The executable specification can also include inputs and expected outputs or acceptance criteria, and the application environment, as well as links or references to the requirements. The goal of the executable specification is to unambiguously communicate the design goals, as well as to allow feasibility and compatibility analysis of the requirements via simulation.

In the MBD, a system model is at the center of the development process, from requirements development, through design, implementation, and testing [4, 5]. MBD workflow is illustrated in figure 2. MBD emerged from the need to reduce development cost and improve product quality at the same time, while the complexity of the system. A model of the system being developed serves as the common thread throughout the development process from requirements capture to final validation. As an executable specification, this system model is refined throughout the development process. Simulation is used at each process step to verify whether the design meets the requirements. Code generation is used for implementing control algorithms and complex logic, thus eliminating errors introduced by writing source code by hand.

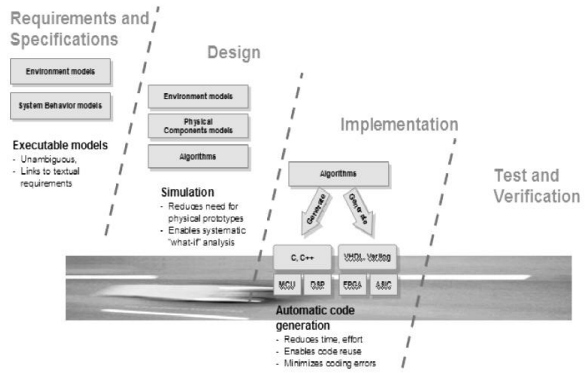


**Fig.2** Model-Based Design development process

# 3. The principle of QPSK

In digital modulation techniques, a set of basis functions is chosen for a particular modulation scheme. Generally, the basis functions are orthogonal to each other [6,7]. Once the basis functions are chosen, any vector in the signal space can be represented as a linear combination of the basis functions.

Quaternary-Phase-Shift-Keying (QPSK) is using coherent detection. As with binary PSK, information about the message symbols in QPSK is contained in the carrier phase. In particular, the phase of the carrier takes on one of four equally spaced values, such as $\pi/4$ , $3\pi/4$ , $5\pi/4$ and $7\pi/4$. For this set of values, we may define the transmitted signal as

$$s_i(t)=\begin{cases}\sqrt{\dfrac{2E}{T}}\cos\left[2\pi f_c t+(2i-1)\dfrac{\pi}{4}\right] & \begin{matrix}0\le t\le T\\ i=1,2,3,4\end{matrix}\\ 0 & elsewhere\end{cases} \quad (1)$$

Where E is the transmitted signal energy per symbol and T is the symbol duration. The carrier frequency $f_c$ . Each possible value of the phase corresponds to a unique dibit. Thus, for example, we may choose the foregoing set of phase values to represent the Gray-encoded set of dibits, 10, 00, 01, and11, where only a single bit is changed from one dibit to the next.

Using a well-known trigonometric identity, we may expand (1) to redefine the transmitted signal in the canonical form:

$$s_i(t)=\sqrt{\frac{2E}{T}}\cos\left[(2i-1)\frac{\pi}{4}\right]\cos(2\pi f_c t)-\sqrt{\frac{2E}{T}}\sin\left[(2i-1)\frac{\pi}{4}\right]\sin(2\pi f_c t) \quad (2)$$

Where i = 1, 2, 3, 4. Based on this representation. There are two orthonormal basis functions, defined by a pair of quadrature carriers:

$$\phi_1(t)=\sqrt{\frac{2}{T}}\cos(2\pi f_c t) \qquad 0\le t\le T \quad (3)$$

$$\phi_2(t)=\sqrt{\frac{2}{T}}\sin(2\pi f_c t) \qquad 0\le t\le T \quad (4)$$

There are four message points, defined by the two-dimensional signal vector

$$s_i=\begin{bmatrix}\sqrt{E}\cos\left((2i-1)\dfrac{\pi}{4}\right)\\ -\sqrt{E}\sin\left((2i-1)\dfrac{\pi}{4}\right)\end{bmatrix} \qquad i=1,2,3,4 \quad (5)$$

Elements of the signal vectors, namely $s_{i1}$ and $s_{i2}$ , have their values summarized in Table 1; the first two columns give the associated dibit and phase of the QPSK signal.

**Table 1.** Signal-space characterization of QPSK

| Gray-encoded input dibit | Phase of QPSK signal (radians) | Coordinates of message points | |
|---|---|---|---|
| | | $s_{i1}$ | $s_{i1}$ |
| 11 | $\pi/4$ | $+\sqrt{E/2}$ | $+\sqrt{E/2}$ |
| 01 | $3\pi/4$ | $-\sqrt{E/2}$ | $+\sqrt{E/2}$ |
| 00 | $5\pi/4$ | $-\sqrt{E/2}$ | $-\sqrt{E/2}$ |
| 10 | $7\pi/4$ | $+\sqrt{E/2}$ | $-\sqrt{E/2}$ |

Accordingly, a QPSK signal has a two-dimensional signal constellation (i.e., N = 2) and four message points whose phase angles increase in a counterclockwise direction, as illustrated in Fig. 3. As with binary PSK, the QPSK signal has minimum average energy.

We may build on (2) to (4) to construct the QPSK transmitter shown in Fig.4. A distinguishing feature of the QPSK transmitter is the block labeled demultiplexer. The function of the demultiplexer is to divide the binary wave produced by the polar NRZ-level encoder into two separate binary waves, one of which represents the odd-numbered dibits in the incoming binary sequence and the other represents the even-numbered dibits. The QPSK transmitter may be viewed as two binary PSK generators that work in parallel, each at a bit rate equal to one-half the bit rate of the original binary sequence at the QPSK transmitter input.

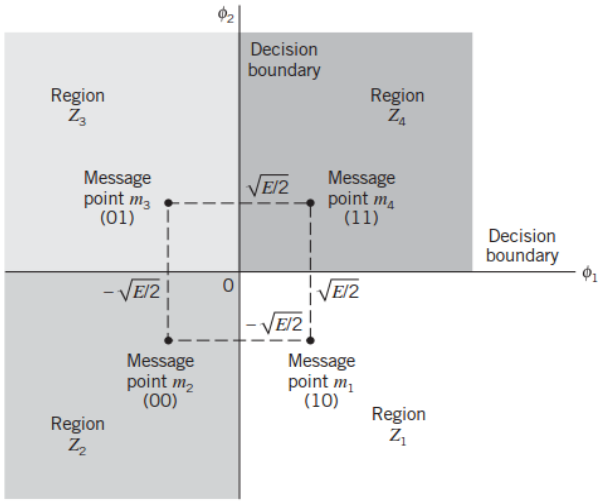


**Fig.3** Signal-space diagram of QPSK system

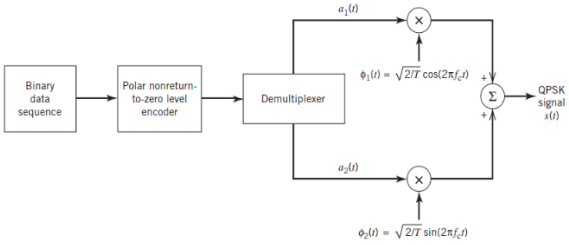


**Fig.4** Signal-space diagram of QQPSK system

## 4. Optimization of QPSK transmitter model

This section introduces the QPSK transmitter model, and the QPSK transmitter model is optimized based on MBD [8,9]. The HDL code is automatically generated and the simulation test is completed. The transmitter subsystem consists of three parts: bit generation, QPSK modulator and raised cosine filter, which is shown in Fig 5. Bit generation generates the bits for each frame. QPSK modulator modulates the bits into QPSK symbols. Raised cosine transmit tilter uses a rolloff factor of 0.5, and upsamples the QPSK symbols by two.

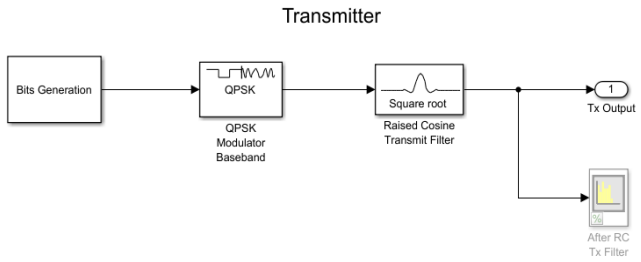


**Fig.5** QPSK transmitter subsystem

QPSK transmitter subsystem generates the in-phase and quadrature channels representing a QPSK symbol mapping. This is sent in as a complex valued vector of the root raised cosine transmit pulse shaping filter. The output of the subsystem therefore consists of the pulse shaped and upsampled symbols. Optimized QPSK transmitter subsystem consists of three parts: data generation and packetization, symbol mapping and pulse shaping, as showed in Fig 6. Pipeline registers are inserted between the subsystems in order to minimize the combinatorial path delay between components and therefore maximize the achievable clock frequency (i.e. minimizing the critical path delay).

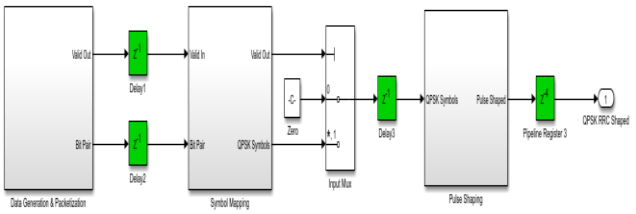

**Fig.6** QPSK transmitter optimized subsystem

The most important model in data generation and packetization subsystem is data source, whose structure is shown in Fig 7. The data source component contains two LUTs, containing the preamble and data bits. The preamble bits are invariant between packets, and so the counter used to address the preamble LUT is reset after the loading of each preamble. The counter used to address the data LUT is allowed to wrap around after reaching the maximum address value, which corresponds to the final character of the packet body. Each of the LUTs is zero padded in order to ensure that the number of data entries is of a power of two lengths. In addition to enabling the counter for the data LUT, the load data input is used to control when the data scrambler component should be enabled, and to control whether the preamble or data bits should be passed to the output via the 2:1 Preamble Data Mux.

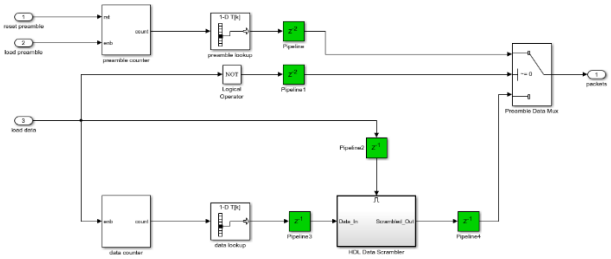

**Fig.7** QPSK transmitter optimized subsystem

Symbol mapper completes the mapping of data bits to symbols, as showed in Fig 8.The symbol mapper uses the QPSK Modulator Baseband block fin order to map the integer input value [0,1,2,3] onto the appropriate complex valued symbol [1 + 1i, -1 + 1i, 1 - 1i, -1 - 1i]. Mapping symbols in this manner allows for a 2 bit values to be used to represent each symbol. This allows the wordlengths in the RRC transmit filter for the product and accumulator datapaths to be optimized. The block uses a gray mapping scheme.

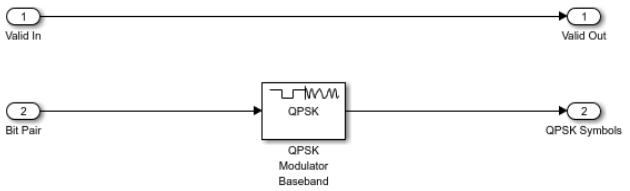


**Fig.8** QPSK transmitter optimized subsystem

Pulse shaping converts the mapped symbol into the baseband signal specified in the protocol, and its structure is shown in Fig 9.The Pulse shaping component uses an FIR interpolation filter, featuring an upsampling factor of four, and a Root Raised Cosine impulse response.The filter is pipelined in order to ensure that the combinatorial delay is minimized throughout the design.

The Error Vector Magnitude (EVM) is the calculated magnitude of the error vector between the ideal constellation point and the constellation point obtained after transmit and receive pulse shape filtering has been performed. The

simulation results are shown in Fig 10,which shows a snapshot from the constellation diagram scope output. The EVM gives a figure of merit regarding the distortion introduced by the transmit and receive filtering. In this case, the EVM allows the system designer to determine if the distortion introduced by the pulse shaping filters due to the effects of fixed point quantization and non-ideal (truncated) impulse responses is acceptable or otherwise.

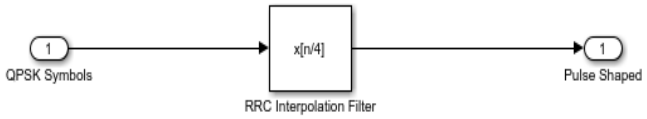


**Fig.9** QPSK transmitter optimized subsystem

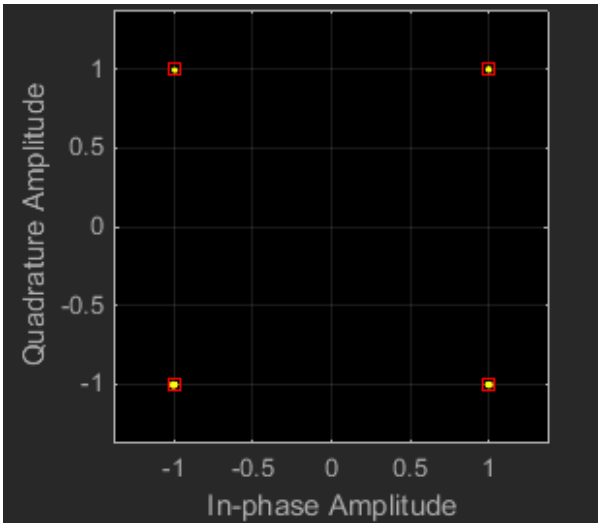


**Fig.10** The output comparison of the constellation diagram

The HDL code generated has been synthesized using Xilinx® ISE on a Virtex6 (XC6VLX240T) FPGA, and ran at around 300 MHz. When operating the RRC transmit filter at maximum rate, the QPSK input symbols are generated at 75M symbols/second. The resulting theoretical maximum data rate is therefore 150M samples/second, taking into account the data transmitted over both the I and the Q channels.

## 5. Conclusion

In this paper, the MBD workflow is used to complete the QPSK transmitter model. The whole design workflow starts from algorithm to realization can be done without written any hardware and software code. Through the MBD process, we can quickly establish the system's modeling, simulation and automatic code generation. Compared with the traditional design process, the MBD design process not only can quickly and efficiently build the system model, but also to perform the simulation in each stage and the automatic generation of code for the hardware implementation. This innovative design ideas and development process is not only the development of the new trend of complex system design process, but also improve the efficiency and security of the code generation. The proposed solution is particularly suitable for hardware development for the complex system. It is believed that MBD will become the new trend of development in the FPGA development. Therefore, the implementation of DSP algorithms using MBD workflow will be widely used in the future.

## References

[1] Wang S, Shin K G. Task construction for model-based design of embedded control software[J]. IEEE Transactions on Software Engineering, 2006, 32(4):254-264.

[2] Nie Y, Ge H, Jing L. Model-Based Design Methodology for Digital Up and Down Conversion of Software Defined Radio[J]. International Journal of Multimedia and Ubiquitous Engineering, 2016, 11(4): 27-36.

[3] Sánchez-Solano S, Brox M, del Toro E, et al. Model-based design methodology for rapid development of fuzzy controllers on FPGAs[J]. IEEE Transactions on Industrial Informatics, 2013, 9(3): 1361-1370.

[4] Penner R R, Steinmetz E S. Model-Based Automation of the Design of User Interfaces to Digital Control Systems [J]. IEEE Transactions on Systems Man and Cybernetics - Part A Systems and Humans, 2002, 32(1):41-49.

[5] Sharma S, Chen W. Using Model-Based Design to Accelerate FPGA Development for Automotive Applications[J]. SAE International Journal of Passenger Cars - Electronic and Electrical Systems, 2009, 2(1):150-158.

[6] Gronemeyer S /, Mcbride A L. MSK and offset QPSK modulation[C]// IEEE Transactions on Communications. 1976:809-820.

[7] Gardner F. A BPSK/QPSK timing-error detector for sampled receivers [J]. IEEE Transactions on communications, 1986, 34(5): 423-429.

[8] Rodriguez A S, Mensinger M C, Ahn I S, et al. Model-based software-defined radio(SDR) design using FPGA[C]// IEEE International Conference on Electro/information Technology. IEEE, 2011:1-6.

[9] Lotze J, Fahmy S A, Noguera J, et al. A Model-Based Approach to Cognitive Radio Design[J]. IEEE Journal on Selected Areas in Communications, 2011, 29(2):455-468.