IMPLEMENTATION OF A DPSK COMMUNICATION SYSTEM ON DIGITAL SIGNAL PROCESSOR TMS320C6711

[1]WASEEM KHAN, [2]M. ATIQUE-UR-REHMAN

[1,2]Department of Electrical Engineering, Air University, Islamabad
E-mail: [1]waseem.khan@mail.au.edu.pk, [2]atiq@mail.au.edu.pk

Abstract - This paper presents the implementation of a differential phase shift keying (DPSK) communication system on a DSP platform. Real-time issues of a typical DPSK system are main focus of this paper. Techniques for synchronization are described and have been implemented. White Gaussian noise has been generated and added by the transmitter part itself to simulate an AWGN channel. The consequent bit error rates were noted and compared with the theoretically calculated ones.

Keywords - DSP; C6711; Real-time BPSK

I. INTRODUCTION

Real-time implementation of a communication system has always been challenging. Especially, implementing a receiver involves numerous challenging tasks, a major one being the synchronization. Implementation of a real-time communication system is important for testing and evaluating the system complexity and performance. Due to the rapid growth in DSP technology in the current decade, DSPs family is the best choice for this purpose. C6000 is one of the famous families of DSPs manufactured by Texas Instruments. These DSPs are supported by a very user friendly integrated development environment. In this paper we describe the real-time implementation of DPSK communication system on TMS320C6711, a floating-point member of this family. Transmitter and receiver are implemented on separate DSP boards connected to each other by a wire, which carries analog signal between the two.

The rest of the paper is organized as follows. Section II presents system overview and describes the various blocks of the communication system. Section III describes the implementation on DSP. Section IV shows the results obtained by running DSPs in real-time and compares them with the analytically calculated ones. Finally we conclude the paper in Section V.

II. SYSTEM DESCRIPTION

Before describing the system in detail its useful to introduce the transmission strategy adopted by the system.

A. Transmission Strategy

In our case, the transmission strategy is dictated by the symbol synchronization technique being employed in the system. Symbol synchronization techniques can be broadly classified as data-aided and non-data-aided. In data-aided techniques, a special sequence, known to the transmitter, is sent. This sequence, termed as training sequence, is followed by the data sequence. The receiver should be able to detect the last bit of the training sequence or the first bit of data sequence. To ease this process, called as frame synchronization, usually a frame synchronization marker is inserted between the training and data sequences. During the training phase, the receiver keeps searching for the frame sync marker. The system being presented here employed the same strategy of three transmission phases named as TRG, SYNC and DATA. In these three phases, training sequence, sync marker and data sequence are sent respectively.

B. Transmitter

In this work, we have implemented a differential binary phase shift keying system. As discussed above, the transmitter functions as a three stage system. For differential phase modulation, differential encoding is used and root-raised cosine (RRC) filter has been used for baseband pulse shaping. Noise generator has been implemented within the transmitter DSP. Fig. 1 shows the functional overview of the complete system, where digital to analog converter (DAC) and analog to digital converter (ADC) are part of the DSP board employed in this work.

C. Receiver

1) Carrier Detection

In a wireless receiver, a basic function is to monitor whether there is a signal present at input. This function is called signal detection or carrier detection. The purpose of carrier detector is to determine whether a signal is present at the receiver input or just noise. It is necessary to ensure that the receiver must not try to extract bits from noise. The receiver does not process (demodulate, etc.) the received samples until it has detected a carrier at its input. Conventionally, in wireless receivers, there is a bandpass filter, before carrier detection stage, to stop all the out of band signals and noise. But in this work, it is not required due to a wired link between the transmitter-receiver pair.

For carrier detection we have used signal power estimation technique [1]. This technique calculates running estimate of received signal power. This power estimate can be formed by passing the squared ADC output samples through a low pass filter. When the

power estimate exceeds a predetermined threshold, for a period of time, a signal is declared to be present. Another function of carrier detector is to detect when the received signal stops. When the signal disappears, the receiver stops processing the input samples.

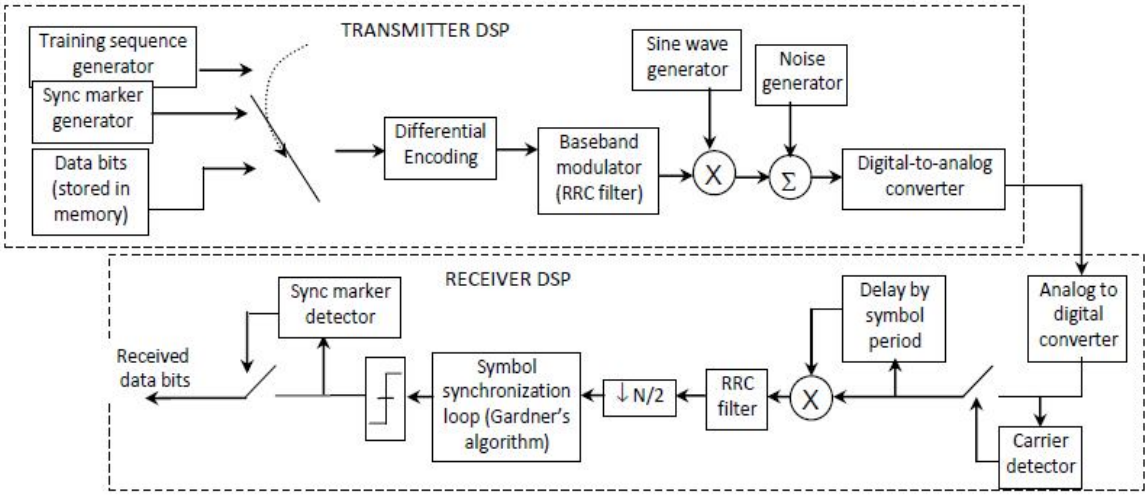

Figure 1: Functional Overview of the Complete System

2) Demodulation

DPSK signals can be demodulated easily by correlating the received signal with itself delayed by a symbol period. The output of this correlator is passed through a filter, matched to RRC filter at the transmitter side.

3) Symbol Synchronization

The nucleus of a symbol synchronizer is a timing-error detector (TED), which generates an error signal if the sampling points are not optimum. In this work, Gardner's TED[2] has been used which is a data-aided scheme. Gardner's TED takes two samples per symbol as input. Therefore, output of the matched filter (RRC) is decimated by N/2 to get two samples per symbol. Here N is the number of samples per symbol at the filter output. At any instant, Gardner's timing error is calculated as

$$e(k) = y(k - 1/2) - y(k - 1) \qquad (1)$$

where (k-½)th sample lies midway between kth and (k-1)th samples. This algorithm is more effective on symbol transitions or zero crossings, because for same consecutive polarities, the detector generates no information about the timing error. Therefore the training sequence used in this work comprises of alternate zeros and ones.

The two samples per symbol obtained at the decimator output may not be at the optimum points. To estimate the sample values at the optimum points, we have used Newton's backward interpolator [3]. Fig. 2 gives the complete functional diagram of symbol synchronizer. The timing error given by Gardner's TED is passed through a loop filter. Output of the loop filter is accumulated to form the interpolation parameter for the interpolator. This parameter may lie in the range [0, -1]. '0' and '-1' correspond to the current and previous samples respectively and any intermediate value corresponds to the intermediate sample. Newton's interpolator can take any number of samples as input, but for simplicity and moderate accuracy, we used three input samples, the current sample and two previous samples. Output of the synchronizer is down sampled by 2, to feed only one sample per symbol to the decision device.

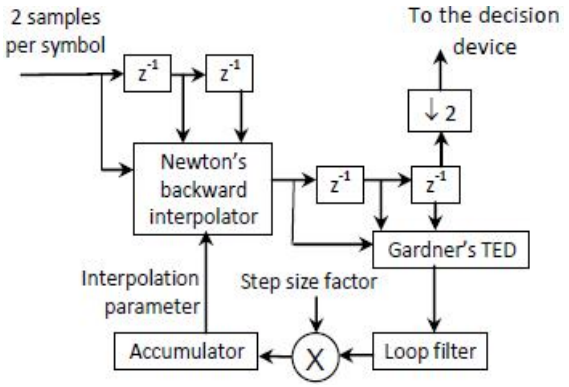

Figure 2: Functional Block Diagram of Symbol Synchronizer

4) Frame Synchronization

As discussed in Section I, training sequence is followed by a frame sync marker. The purpose of this special sequence is to make the receiver locate the starting bit of the data sequence. During the whole training sequence, the receiver keeps searching for the sync marker. The receiver keeps the incoming bits correlating with sync marker. Sync markers are specially designed sequences such that the autocorrelation main lobe is high and side lobes are small. But this is true when the symbols adjacent to the sync marker are all zeros. In a real transmission, the adjacent symbols are not zeros (e.g. +1, -1 in BPSK). And further, due to noise and consequent bit errors, main lobe falls down and side lobes rise. This situation may lead to false alarm or missed detection. The

probability of a miss for N bits long sequence where k or fewer errors are accepted, is given by [4]

$$P_m = \sum_{j=k+1}^{N} \binom{N}{j} p^j (1-p)^{N-j} \qquad (2)$$

where p is the probability of a bit error. The probability of a false alarm caused by N bits of random data is given by

$$P_{FA} = \sum_{j=0}^{k} \frac{\binom{N}{j}}{2^N} \qquad (3)$$

As obvious from these equations that P_m decreases by accepting more errors but at the same time P_{FA} increases. To minimize both simultaneously, N can be increased. In literature a number of synchronization sequences are available having a variety of lengths. Neuman and Hofman [5] have found sequences up to 100 bits long. Spilker [6] has listed sequences up to 24 bits long found by Neuman and Hofman. For this work we have selected a 23-bits long sequence from the list given by Spilker.

III. DSP IMPLEMENTATION

In this section, we present, first, the features of TMS320C6711 DSP board, Code Composer Studio Integrated Development Environment and finally the techniques used for DSP implementation of the system.

A. Overview of the Board

The DSP board DSK6711 (DSP kit C6711) is built around a 150 MHz TMS320C6711 Floating Point DSP manufactured by Texas Instruments. It is equipped with two multi-channel buffered serial ports and 16 interrupts. On board external memories include 16 Mbytes SDRAM and 128 Kbytes Flash ROM.

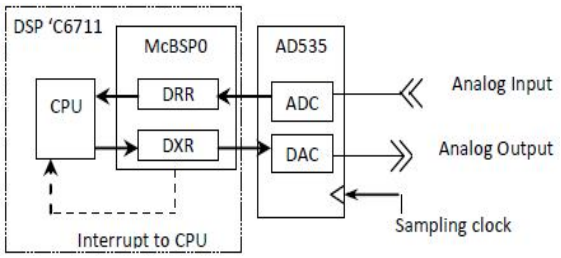

Figure3: DSP to Codec Connections

For audio signal processing, the DSP board is equipped with an on-board codec AD535. The codec converts the digital signal received from DSP to analog and vice versa. The codec is connected to the DSP as shown in Fig. 3. The digital data from codec is transferred to CPU through receive buffer (DRR) of McBSP0 and the data from CPU is transferred to codec through transmit buffer (DXR) of the serial port. When McBSP0 has data for CPU or it needs the next sample to transfer to codec, it interrupts the CPU.

B. Code Composer Studio

Code Composer Studio Integrated Development Environment (CCS IDE) provides all the necessary software tools for DSP code development. It supports code generation, real-time analysis, simulation, debugging, and data exchange between computer and DSK memory.

C. System Implementation

Two separate DSP boards have been configured as transmitter and receiver, while DAC output of the transmitter is connected to ADC input of the receiver board. For carrier detection, symbol delay, RRC filtering and correlation for sync marker detection, FIFO buffers have been implemented. All software development has been done in CCS IDE using C language. Key parameters of the system implemented in this work are following.

Sampling frequency: 96 kHz
Bit-rate: 4 kbits/s
Carrier frequency: 12 kHz
Root raised-cosine filter: Roll-off factor: 0.5
Pulse support time: 6 symbols

1) Symbol Synchronization

As discussed earlier that Gardner's TED works best on zero crossings, therefore the training sequence consists of alternate zeros and ones. Selection of the length of training sequence is a critical matter because if it is too short, the synchronizer may not be trained fully. Conversely if it is too long, it may be wastage of time. In this work, after a series of runs, 1000 bits long sequence has been found to be optimum. With this length, the synchronizer has been trained even at E_b/N_0 = 0 dB. If it is known that that noise is not so powerful, this length can be reduced. Step size factor, shown in Fig. 2, has been kept considerably high during the training phase; but once this phase is over, it is lowered to a very small value to compensate small deviations of optimum symbol time.

2) Transmission between DSPs and PC

The data bits to be transmitted are stored on computer hard disk. Before the system is run, the data bits are loaded to a C variable using the debug mode in CCS IDE. The loading of data bits is required for transmitter as well as receiver DSP. When the data bits have been loaded into DSP memories, the receiver DSP is run prior to the transmitter DSP. When the receiver gets the complete frame of data bits, it compares the received and expected data bits and displays the number of errors.

3) Noise Generation

To simulate an AWGN channel, the transmitter DSP itself generates white Gaussian noise and adds to the modulated signal before giving it to DAC. In C language, uniformly distributed random numbers can easily be generated using built-in functions. If U_1 and U_2 are uniform random positive integer variables, a Gaussian distributed random variable G, with zero mean and variance σ^2 can be generated using [1]

$$G = \sqrt{-2\sigma^2 \log_e(1-U_1)}\cos U_2$$

(4)

Variance of white noise is infinite, but in band-limited communication systems, due to filtering at the receiver, its variance is finite and given by $\sigma^2 = N_0/2$ [4]. Given that we have set a particular value of E_b/N_0 for simulation purpose, N_0 and σ^2 can be determined easily. Bit-energy E_b at the receiver side, in wireless systems is variable due to fading; but in this work a non-fading AWGN channel has been simulated, therefore E_b is fixed and deterministic.

IV. RESULTS

In this work, 10000 data bits have been transmitted between the two DSPs, with a range of values of E_b/N_o. Corresponding bit error rates have been noted and compared with probability of bit error, calculated analytically, and given by [7]

$$P_b = \frac{1}{2}\exp\left[-0.76\frac{E_b}{N_0}\right]$$

(4)

Fig. 4 plots the probability of bit error (P_b) and bit error rate (BER) for a range of E_b/N_o values. For high values of E_b/N_o, BER is slightly higher than P_b because of small number of transmitted bits. In fact the two quantities are different but tend to be same if the number of transmitted bits tends to be large.

CONCLUSION

In this paper, we present the implementation of a DPSK communication system based on TMS320C6711 DSP board. Real-time issues of synchronization have been described. AWGN channel has been simulated for various noise levels and corresponding bit error rates have been noted and compared with probability of bit error calculated analytically.

REFERENCES

[1] Tretter, S. A., Communication System Design Using DSP Algorithms: With Laboratory Experiments for the TMS320C6701 and TMS320C6711, Kluwer Academic / Plenum Publishers, 2002

[2] Gardner, F., M., "A BPSK/QPSK Timing Error Detector for Sampled Receivers", IEEE Trans. on Comm., vol. COM-34, pp. 423-429, May 1986

[3] Francis S., Theory and Problems of Numerical Analysis, Schaum's Outline Series, McGraw-Hill, 1989

[4] Sklar, B., Digital Communications-Fundamentals and Applications, 2nd ed., Pearson Education, Delhi, 2002

[5] Neuman, F. and Hofman, L., "New Pulse Sequences with Desirable Correlation Properties", Proc. Nat. Telemetry. Conf., pp. 272-282, Washington, DC, USA, Apr 1971

[6] Spilker, J. J., Digital Communications by Satellite, Prentice-Hall Inc., Englewood Cliffs, NJ., 1977

[7] Park, J. H., "On binary DPSK detection", IEEE Trans. Comm., COM 26, No. 4, Apr. 1978

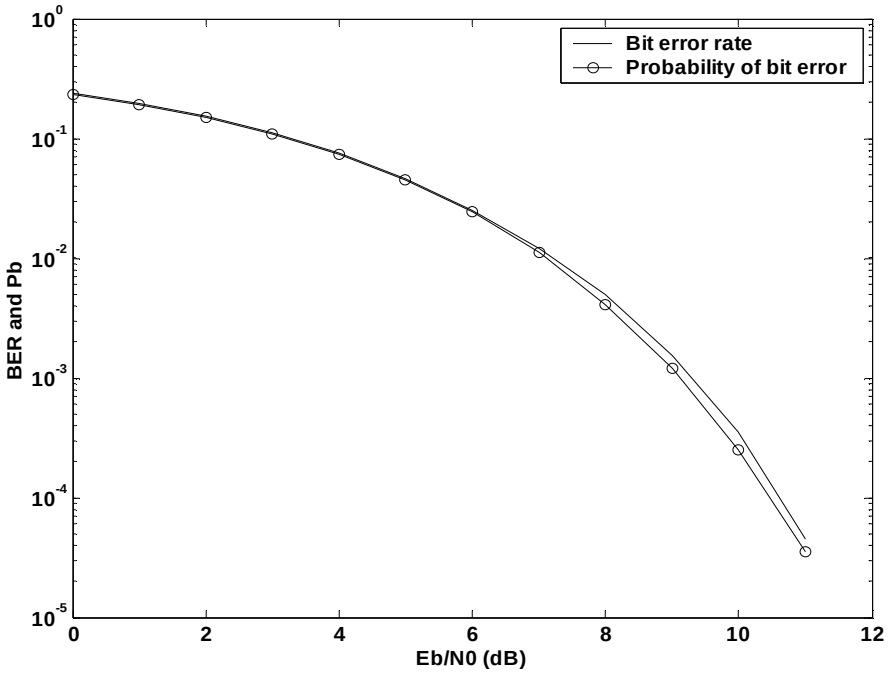

Figure 4: Comparison of Practical and Analytical Results

★★★