Final Project Report
On
Modeling and Simulation of High-Speed Digital Subscriber Line Generation 2 (HDSL2) Modems Using Ptolemy

Sultan Ahmed
George Benavides
Khalid Islam

May 8, 1998

EE382C: Embedded Software Systems

Spring 1998

Prof. Brian L. Evans
Department of Electrical and Computer Engineering
The University of Texas at Austin

Abstract: The emerging HDSL2 standard promises reliable, low-cost, high-speed digital communications for telecommuting, home office transactions, enterprise computing, internet access, and video conferencing over the existing telephone network. This technology is slated to offer two-way data communication at a speed of 1.544 Mbps over a single twisted pair copper wire. This capacity far exceeds existing voice band maximum limit of 64 Kbps and ISDN limit of 144 Kbps and is comparable to that of T1 lines at a much lower cost.

We implemented a fully configurable HDSL2 modem in Ptolemy with the purpose of enabling tradeoff analysis. This will allow the HDSL2 standards committee members to vary the key parameters that significantly affect the complexity and memory requirement of the proposed standards. These requirements will help determine the feasibility of a single DSP chip solution.

1. Introduction

HDSL2 is a high-speed data transmission protocol using existing single-pair copper telephone lines. It is designed to transmit and receive data at speeds of over 1.5 Mbps. Our project goal was to model a fully configurable HDSL2 modem in Ptolemy. We chose the synchronous data flow (SDF) domain for, since it was adequate for modeling the HDSL2 modules.

The key value or innovation in this project is to allow key parameters of the computationally intensive modules, such as the Trellis decoder and the Tomlinson-Harashima precoder (THP), to be varied at run time. This feature is intended to allow dynamic analysis of the HDSL2 standards, which are continually changing. The use of this dynamic implementation helps determine efficiency and feasibility of the standards that ultimately will require a Digital Signal Processor (DSP) chip solution.

2. Modeling

The entire HDSL2 modem is a system, which requires significant signal processing. Therefore, the SDF domain is our choice for modeling. Some issues of possible BDF were considered since the modem does require retraining. This retraining requires a select BDF actor to switch between certain actors in the THP implementation (see section 3.4 for more details). However, if the standards committee requires periodic retraining (i.e. retrain the THP filter every certain period of time) then this behavior can be model as SDF. Other issues pertain to the few cases of multi-rate that are needed for the serial to parallel conversion, Trellis encoding and Trellis decoding. Although the cyclo-static SDF domain does well in reducing unneeded computation [1], these advantages are unnecessary for HDSL2. The SDF domain easily models these multi-rate

conversions with little overhead in memory and computation.

3. Implementation

3.1 Simple Modules

Figure 1 shows the block diagram of our HDSL2 implementation. There are several fairly simple blocks in this diagram that together require minimal computation. These are the framer, scrambler, serial to parallel converter, the bit to symbol mapper in the transmitter section, and their corresponding blocks in the receiver section. These blocks map directly to SDF stars. Programmable parameters in these blocks are not needed since the area of interest is in the coding schemes.

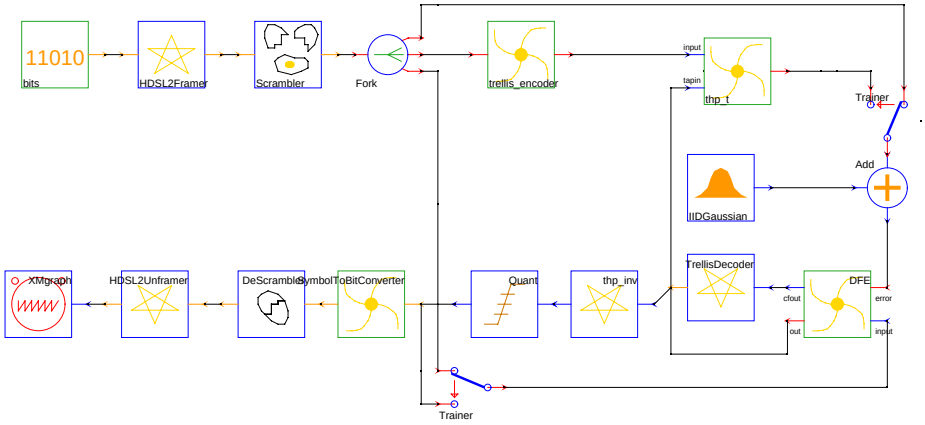

Figure 1. Our implementation in Ptolemy.

3.2 Trellis Encoder

Our Trellis encoder takes the least significant bit from a 3-bit value and produces two bits [2]. The two most significant bits of the 3-bit value are unaffected. Thus, the net effect of encoding is to produce a 4-bit quantity of which the least significant 2 bits are encoded. Our encoder is programmable. The user specifies the number of delays via parameters, which relates to the number of states, and the equations for each of the two

encoded bits. The equations are passed in as a one-dimensional array of bits with 0 and 1 corresponding respectively to the absence or presence of a particular delayed value.

3.3 Trellis Decoder

This decoder is the most computation and memory intensive module. It accounts for the majority of the total computation in the current HDSL2 proposal. Consequently, this algorithm is where most of the optimization effort via user programming is focused. We implemented a fully programmable Trellis decoder using the Viterbi algorithm. The number of delays and the equations for the outputs are specified by the user. The decoder implements a maximum-likelihood (ML) "soft" decoding scheme, which maximizes the probability of recovering the transmitted data by combining the quantization stage with the decoding [2].

The programmability is achieved by dynamically allocating the data arrays for storing the Trellis diagram information [2]. The size of the diagram grows exponentially (power of 2) with respect to the number of delays. Clearly, this is the key target for optimization since reducing delays by one reduces the complexity by a factor of 2. Figure 2 below shows how such a diagram is constructed for 2 states. The HDSL2 current proposal calls for a 512-state Trellis code [4]. Starting at a particular state (node) it is possible to transition to one of two states, depending on whether a 1 or a 0 bit input. Each transition has an associated output, which is compared to the actual value received by the decoder in order to compute the weight of the transition.

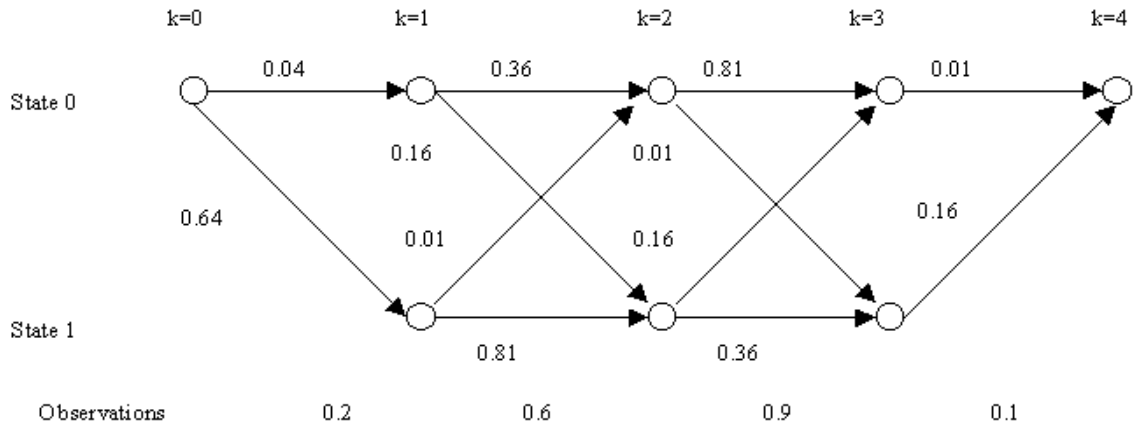

Figure 2: Trellis diagram for a two state Trellis code [1]

The accuracy of the decoder depends on another parameter called prune depth. To retrive the most likely decoded values, the Trellis diagram must be allowed to grow. For infinite input streams, this means an infinitely long Trellis diagram. The decoded values are output only after the last bit is received. In a practical implementation, the Trellis diagram is only allowed to grow to have a length equal to the prune depth. If the prune depth is sufficiently big, the decoder will perform very close to its maximal potential. The size of prune depth affects the memory requirement of the Trellis decoder linearly.

3.4 Tomlinson-Harashima Precoder

The THP is an innovative design, which pre-filters the effects of the transmitting channel before actual transmission [5,6]. The THP is also compatible with the Trellis encoder to gain higher signal to noise ratio (SNR). This fact is important, since the complexity of Trellis increases as SNR increases. The complexity of THP is much smaller and easy to implement using known structures or algorithms.

The structure of THP is similar to that of the Decision Feedback Equalizer (DFE).

The DFE transition to THP is a matter of placing the Feedback Filter (FBF) in the transmitter. Since the FBF is non-adaptive, we must transmit the appropriate coefficients to the transmitter after an execution of a training sequence. This training sequence is needed to find the optimal coefficients for the FBF and the Feedforward Filter (FFF), as shown in Figure 3. The optimal coefficients are found by using the DFE structure during the training sequence. The DFE structure will use a Least Mean Square (LMS) adaptive filter to find the needed coefficients for the FBF.

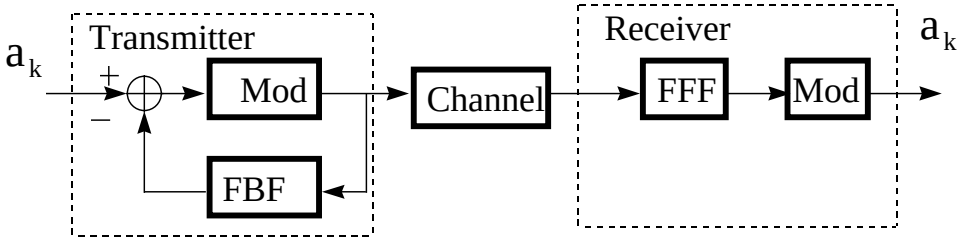

Figure 3: The THP structure consists of a FBF and a FFF with modulo operators at both the transmitter and receiver.

Implementing the THP in Ptolemy is a straightforward implementation once the modulo operators are understood. The modulo operator at the transmitter is not an actual arithmetic operation, but rather a technique to bound the output between the minimum and maximum symbol levels of the Pulse Amplitude Modulation (PAM). The modulo operation at the receiver, however, is an actual arithmetic modulo. In Ptolemy, galaxies include the transmitter precoder with the FBF and the adaptive equalizer at the receiver, which is a DFE during the training sequence and a single adaptive FFF during normal run-time (see Figure 4 and Figure 5.). This needed transformation requires a select BDF

star to implement retraining. However, for our implementation, we used a training SDF star, which switches inputs once after a predetermined period.

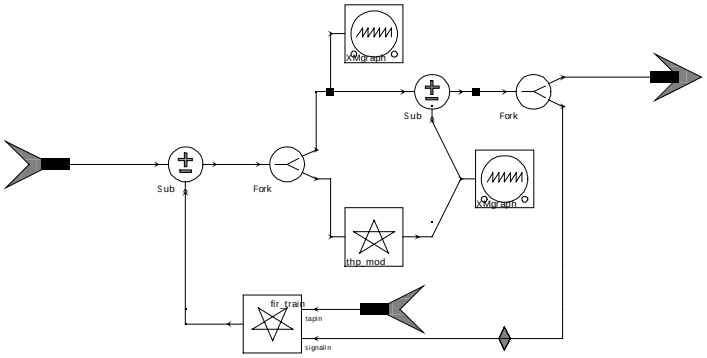

Figure 4. Adaptive equalizer in Ptolemy on the transmitter

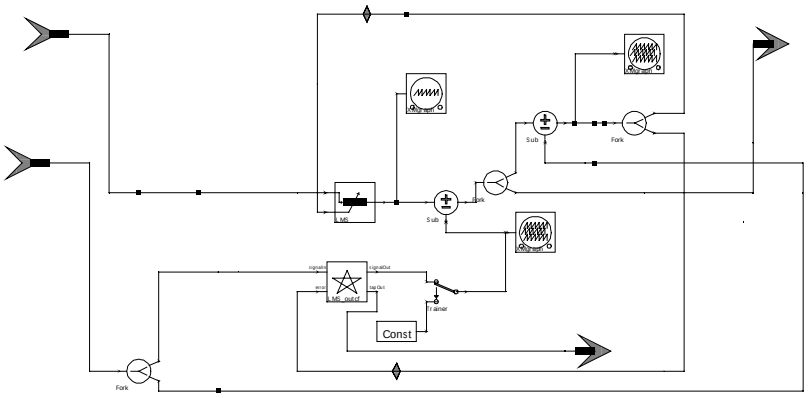

Figure 5. THP transmitter galaxy.

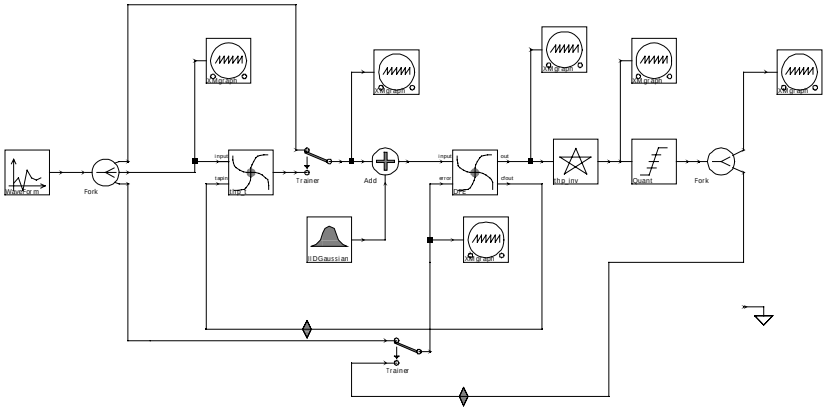

Figure 6. Testing model for the THP.

We tested the THP as a standalone implementation before including it in the overall model (see Figure 6). To test and debug, we connected an known input with additive nose and compared it with the output. Our channel model is non-existent at this point since we are interested in how the THP handles just noise. Using noise we can determine if the modulo operations are working in tandem with the adaptive equalizers. Since only additive noise is present in the channel the adaptive equalizer have nothing to model. Using a simple low-pass filter for the channel shows the adaptive behavior of the DFE during the training sequence. A more accurate model with correct wire-line attenuation and additive Near End Cross-talk (NEXT) is needed to fully understand how THP reacts in the real situation.

4. Innovation in the Trellis Decoder

4.1 Advantages of Soft Decoding over Hard Decoding

By combining the quantization step with the decoding step, the "soft" decoder avoids introducing errors into the system, which plague "hard" decoders that perform quantization before decoding. This happens because the latter have to make "hard" decisions when the received data falls on the boundary between two decision regions [2].

4.2 Tradeoff Analysis Made Possible by the Programmability of the Trellis Decoder

By implementing the Trellis coding and decoding as a programmable model, we allow the HDSL2 standards to be analyzed during the concept development phase. Primarily changing the number of delays in the code varies the complexity of the programmable coding. As indicated in section 3.3, the Trellis decoder memory requirement and execution time varies exponentially. Another parameter the designer can vary in our implementation is the prune depth. Memory depends linearly and

execution time less than linearly upon this parameter. The benefit of tradeoff analysis is that we can get an indication of what may be the simplest implementation that will satisfy the SNR margin required for HDSL2 reception.

5. Conclusion

Our HDSL2 modem model is intended to analyze the concept during the development phase of the HDSL2 protocol standards. Our implementation allows the standards committee to vary parameters, some of which drastically affect the algorithm complexity (i.e. the number of delays in the Trellis code) and the memory requirement. Our tool will hopefully allow the committee to rapidly verify whether simplifications to the Trellis and Tomlinson coding meet the SNR targets of HDSL2.

6. References

[1] T. Parks, J. Pino and E. Lee, "A comparison of Synchronous and Cyclo-Static Dataflow," Conference on Signals, Systems and Computers, October 1995.

[2] E. Lee and Messerschmidt, Digital data communications, Kluwer Pulishers Boston.

[3] Engineering Literature Survey On High-Speed Digital Subscriber Line Generation 2 (HDSL2) Modems.

[4] M. Tu and J. Liu, "A comparison of MS/TCM coding schemes for HDSL2," PairGain contribution, T1E1.4/97-299, Apr. 1997.

[5] M. Tomlinson, "New automatic equalizer employing modulo arithmetic," Electronics Letters, vol. 7, pp. 138-139, Mar. 1971.

[6] H. Harashima and M. Miyakawa, "Matched-transmission technique for channels with rational spectra," IEEE Transactions on Communications, vol. COM-20, pp. 774-780, August 1972.