

Teaching MODEM Concepts and Design Procedure with MATLAB Simulations

fred harris
San Diego State University
fred.harris@sdsu.edu

Abstract:

MATLAB simulation is used as the primary tool to illustrate concepts, to validate MODEM designs, and to verify operation of the subsystems employed in DSP based transmitters and receivers presented in a pair of classes on *MODEM Design* and *Digital Receiver Design*. The whole gamut of subsystems found in conventional and experimental modem designs are simulated and assembled to form a full end-to-end simulation of an operating MODEM. This paper describes the philosophy used to guide class involvement and assess the experience and the learning value to student participants.

Introduction:

The EC&E Department at San Diego State University has been using MATLAB based instruction and work assignments in our undergraduate program since 1990 when Huseyin Abut, a colleague in the department, incorporated it in our EE-410, *Linear Systems* course. MATLAB is now thoroughly woven into the undergraduate curriculum where it is introduced in the second year *Statistics* course, EE-300, after which the students benefit from its high utilization in our various communication systems and signal processing courses. Relying on the MATLAB experience and proficiency developed by the students in their undergraduate program, it was natural to make widespread use of MATLAB in our graduate program. In particular, we make extensive use of MATLAB simulations to demonstrate concepts, to illustrate process, and to validate designs employed in DSP based MODEMS and the various DSP based subsystems found in modern transmitters and receivers. These subsystems include digital and analog segments of the transmitter, the analog channel with its various impairments, and the analog and digital segments of the receiver.

In addition to the design and the programming tasks involved in the simulation, the students also learn to extract, display, and interpret the various observable indicators of MODEM operation and thus learn to gauge the health and operating boundaries of proper and faulty MODEM operation. Probe and display options include time series and spectral plots, as well as eye-diagrams, transition diagrams, and constellation diagrams at various points in the signal conditioning, signal processing, and signal transformation chain. Other displays include transient and steady state time series representing oscillator phase profiles and phase error profiles of phase locked loops during carrier and timing acquisition. Still others include time series representing transient and steady state errors formed by the receiver equalizer and decision processes. The system simulation can be performed with channel impairments disabled during initial design and debugging and similarly with all arithmetic operations performed with floating point precision prior to invoking fixed point processing. In many cases, while manipulating reality to sustain clear waters for the benefit of the learning process, we summon one other minor fiction. We conduct all simulations with complex base band signals and channel models. We are careful, of course, to explain and illustrate the transparency of the modulator up-conversion and the demodulator down-conversion to the modulation, acquisition, and demodulation process. We are similarly careful to identify those aspects of the missing conversions that are not simulated by the complex base band model, namely distortion terms due to non-linear power amplifiers at the transmitter and nonlinear mixing with strong adjacent channels in the analog mixers at the receiver.

The benefit to the student of performing both the design and programming of the simulation is manifold. First and foremost, it reinforces the student's understanding of abstract concepts by invoking right-brain learning processes. Images of time series and spectra obtained from various monitoring points in a block diagram form strong associations between the mathematics and internal comprehension. Second it exposes the student to a common but complex integrated system that compel the student to organize and order her understanding of the many (often heretofore disconnected) abstract concepts of functions into a set of interacting task implementations. Many a "Ah-ha...., I've got it!" threshold switches of understanding are toggled (on) during simulation programming. A third benefit of the simulation is that it entices and invites the student to play. It enables the path for the student to explore and ask, and to then answer, questions that begin with, "I wonder what would happen if.....?". After all, that is our business, isn't it? How goes the saying? "Our task is not to teach the student to learn the right answer to the asked question, but rather to learn the right questions to ask!"

Modem Model:

The simulation evolves during the journey through the course material. The material starts with an overview of the communication system viewed from the physical layer. We present a hierarchical description of the modulator, channel, and demodulator starting at the functional block diagram level. Descriptions of the modulator and demodulator blocks are enhanced through successively less abstract, more detailed levels till the description includes virtual registers and algorithm structures closely resembling the architecture of a programmable DSP or ASIC engine implementation.

We describe here a design and simulation of a V.22bis phone-line Modem conducted in our Spring 2000 Modem Design class. This modem operates at 600 symbols per second, with two bits per symbol (1200 bps) D-QPSK or with four bits per symbol (2400 bps) D-16QAM. The modem can transmit or receive a quadrature signal modulated to 1200 Hz or 2400 Hz center frequency.

A block diagram of the modulator structure is shown in figure 1. As can be seen, the modulator design and simulation consists of a *random binary data generator*, a *mapping to constellation points*, *Square-Root Nyquist Shaping* and *1-to-16 Up-Sampling filters*, *quadrature sine wave generation*, *I-Q modulation*, an *equalizer filter to compensate for DAC induced $\sin(x)/x$ distortion*, a *1-to-10 interpolating filter*, an *8-bit DAC (zero-order hold)*, and a *2-nd order analog smoothing filter*.

A block diagram of the demodulator structure is shown in figure 2. Here we see that the demodulator design and simulation consists of a *2-nd order analog anti-aliasing filter*, an *8-bit ADC*, an *AGC loop*, a *digital anti-alias filter with 10-to-1 down-sampling*, an *8-to-1 down-sampling matched-filter and derivative matched-filter with phase controlled by a timing recovery loop*, a *complex de-rotator*, a *quadrature DDS controlled by the loop filter of a carrier recovery loop*, a *20-tap fractionally spaced equalizer controlled by a decision directed LMS controller*, a *slicer*, an *arc-tangent phase error detector*, and a *constellation point to binary data mapper*. The AGC loop is composed of a *VGA*, a *Hilbert Transform filter*, and a *log-gain LMS controller*.

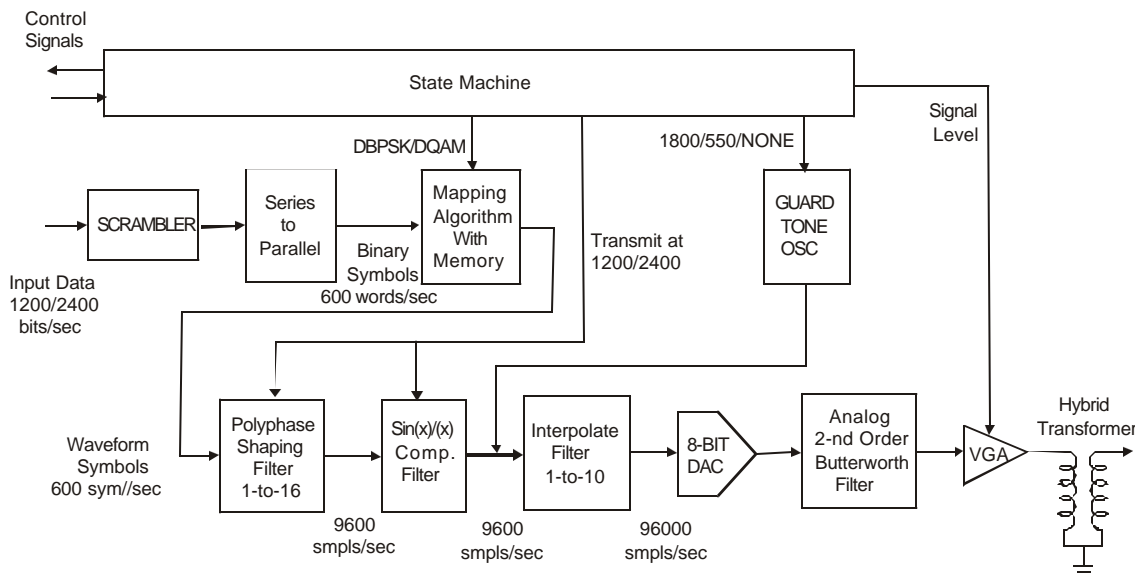


Figure 1. Model of Modulator Segment of V_22 bis Modem

Simulation Output

A simulation program offers the student to a chance to examine a design at all levels of detail. The student can “kick the tires” or can “take her out for a spin to see what she can do”. At the finest detail she can probe the signal conditioning paths at any location in the chain and see more detail than can be found in an actual hardware modem. As indicated earlier the probe can present plots of a short segment of time series, a windowed spectrum, an eye diagram, a transition diagram, or constellation diagram. Additional time series plots can be extracted to capture transients formed by the AGC, timing, carrier, and equalizer control loops in the receiver. Typical plots from probe point at various points in the transmitter and receiver models are shown in figures 3 and 4 respectively.

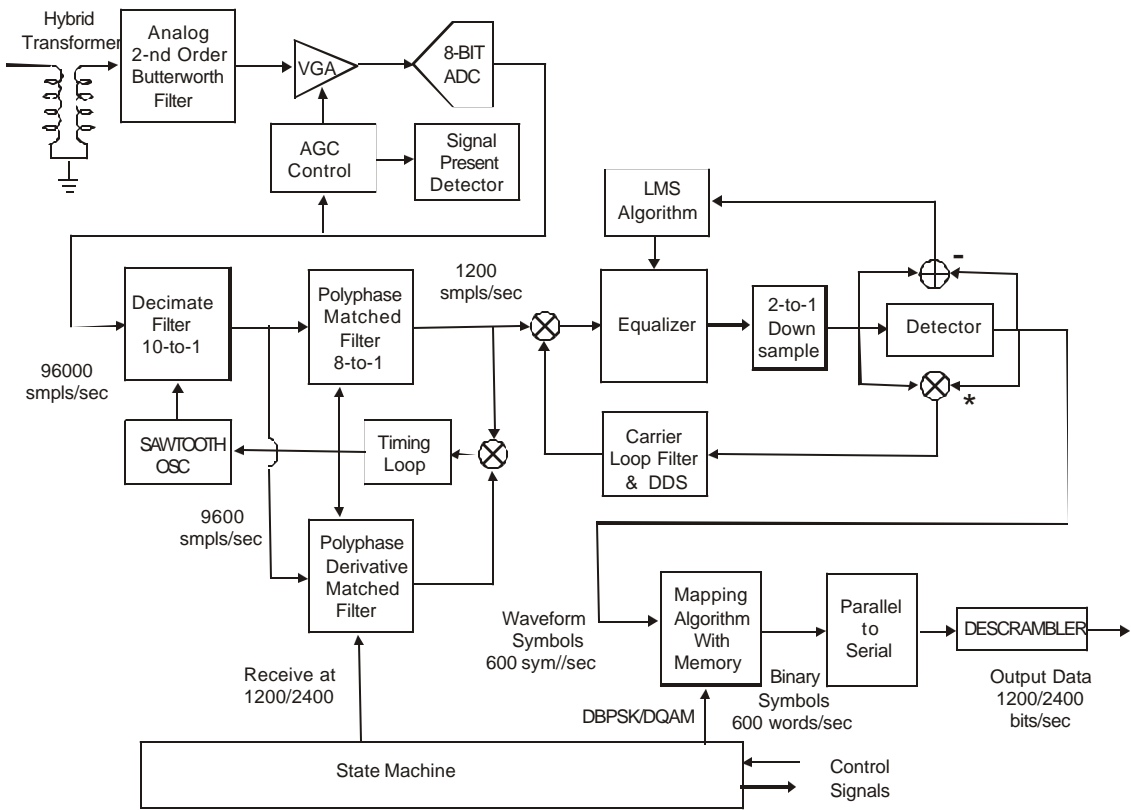


Figure 2. Model of Demodulator Segment of V_22 bis Modem

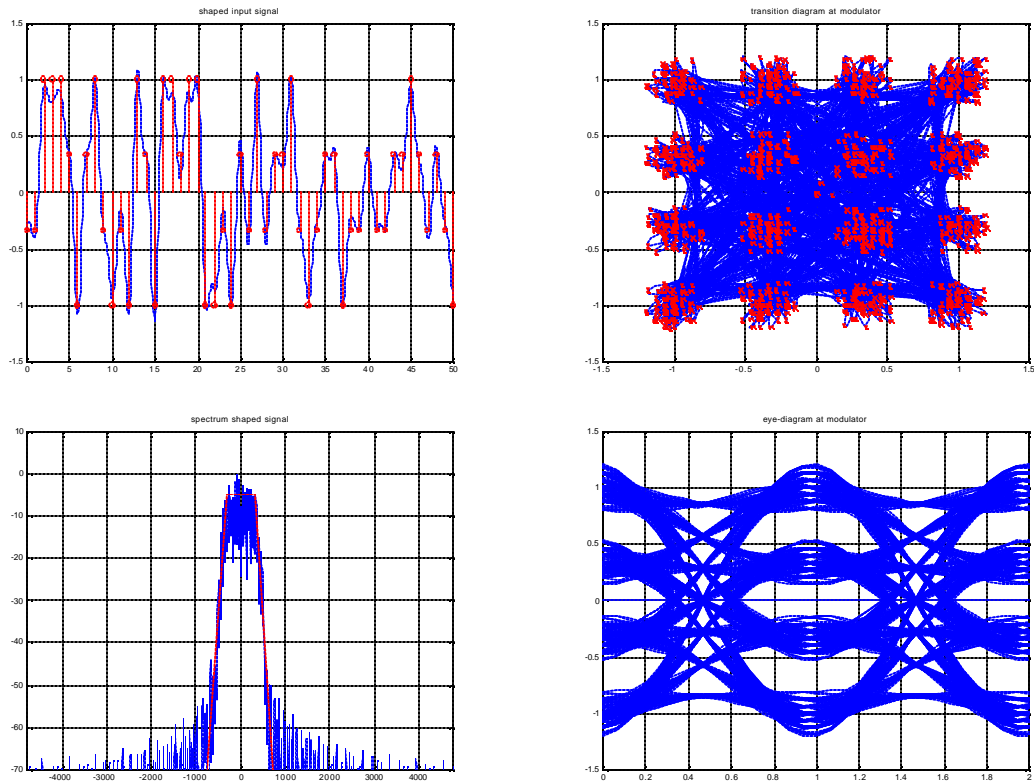


Figure 3. Time Series, Spectrum, Transition Diagram, and Eye Diagram at Modulator

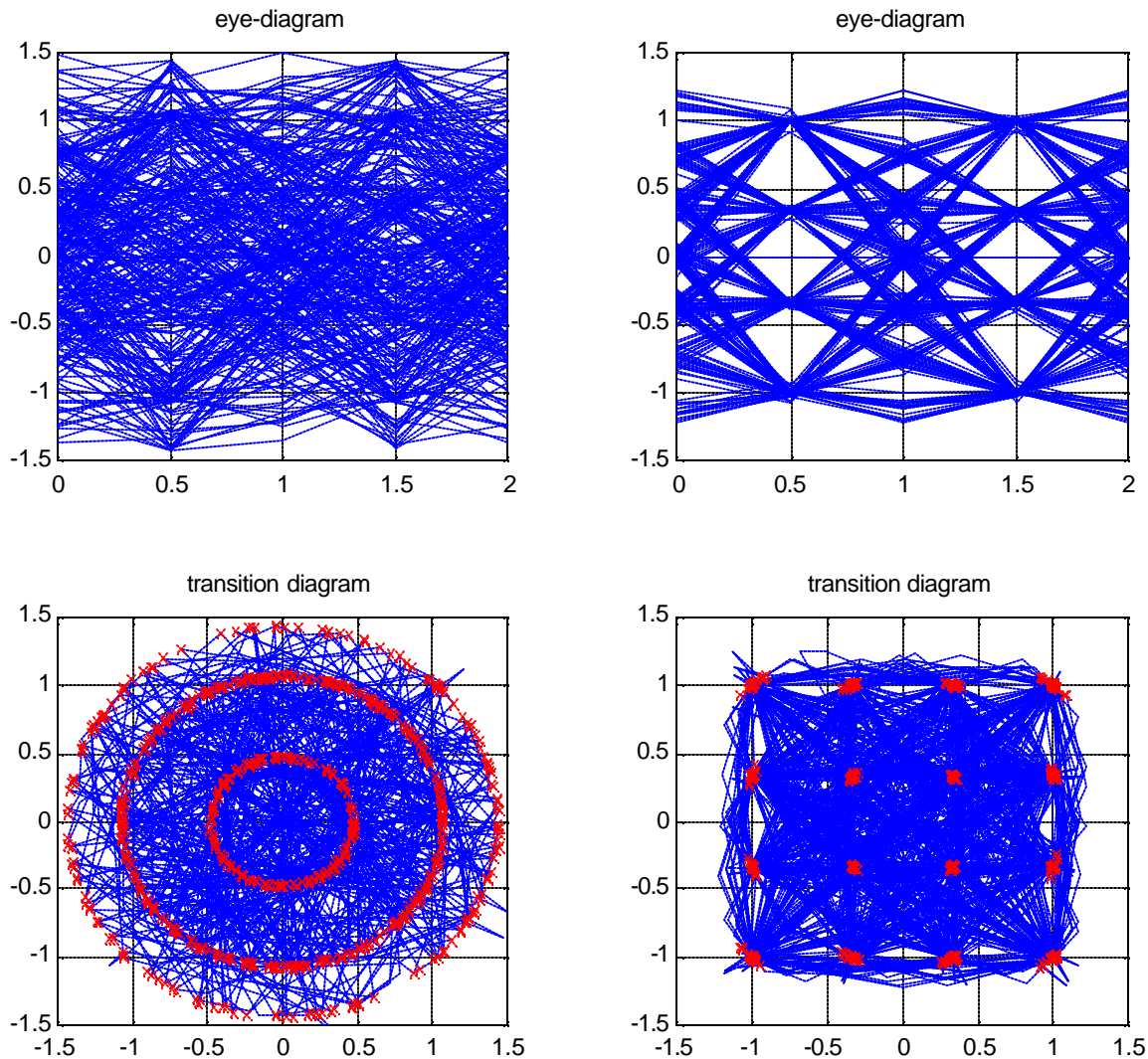


Figure 4. Eye Diagram and Transition and Constellation Diagrams after Timing Recovery and after Carrier recovery at Receiver

When writing the code for these simulations, the student is required to code each plot formed for the output displays. This includes the time series, the spectra, the eye-diagrams and the constellation diagrams. Actively involving the student in generating the figures, and in particular the time, frequency, and amplitude scales makes it more likely that the student understands the relationships between the sample rates and symbol rates, and between spectral span and symbol bandwidth observed at each probe point. These relationships are different at various points in the system and by requiring the student to keep tabs of the changes (through axis scaling) the student can more easily track the changes to help understand the reasons and advantages of the multirate design.

The simulation process also reinforces the perspective we prefer the students to have when dealing with multirate processing. In many design and simulation environments we are comfortable in the use of normalized frequency [standard f/f_{sample} or MATLAB's $f/(0.5*f_{\text{sample}})$] and normalized time (t/T_{sample}). For instance we understand that a normalized band-edge frequency of 0.2 is equivalent to frequency $0.2*f_{\text{sample}}$. In multirate systems, the standard reference, f_{sample} , changes. In a sense, we have an elastic ruler, a ruler of different length at different points in the signal processing chain, one which when used as a reference, leads to significant confusion. Our preference, which we think avoids the confusion while developing insight for multirate filters, is to use the signal symbol-rate as the reference. We have students use multiples of the normalized or non-normalized frequency when scaling the frequency axis. For instance, the frequency axis

used to display spectrum after the 1-to-16 up sampling and shaping filter would be of the form shown below.

```
Plot((-0.5:1/1024:0.5-1/1024)*16, fftshift(20*log10(abs(fft(x_out(1,1024))))))
```

Here the normalized symbol rate is unity. The non-normalized symbol rate would be 600 symbols/sec and the plot call would be changed as indicated below. This code was used to generate the base-band modulated spectrum shown in figure 3.

```
Plot((-0.5:1/1024:0.5-1/1024)*16*600, fftshift(20*log10(abs(fft(x_out(1,1024))))))
```

Similarly, having the student code the eye diagrams and constellation keeps the student aware that samples of data at the symbol rate are a correctly phased and down-sampled subset of the complex valued over-sampled I-Q data samples. An example of this coupling is the code required to plot an eye diagram. This code is demonstrated below for the signal array `xflt`, a 1-to-16 over-sampled time series. This code formed the two-symbol span eye diagram shown in figure 3. Note the time axis of the plot command `axis` acknowledges the 16 samples per symbol required to span one symbol (increments of 1/16 over the span of a unit interval), as well as the 33 samples required to span two symbol intervals bounded by three successive symbol points. Here time is normalized by symbol time duration ($t_{\text{norm}} = t/t_{\text{symbol}}$).

```
plot([-1 1],[0 0])
hold on
for n1=1:32:length(xflt)
    plot(1:1/16:1,xflt(n1:n1+32))
end
hold off
grid
axis([-1 1 -1.5 1.5])
```

Student Assessment

Feedback from the students in the Modem Design Class was positive. Many students reported that their understanding crossed a threshold. The concept ideas of shaping, of matched filtering, of timing recovery, of carrier recovery, of equalization, and of various signal diagnostics have moved from abstract isolated ideas to a well grounded, comfortable, level of system comprehension. We found that additional probing of the simulation data supplied very high marginal increase in student awareness and understanding. For instance we formed amplitude density plots at the timing recovery sample points for different signal to noise ratios and compared expected detection error rates with measured rates. The students could see broadening of the (sample) conditional density function and the increase in threshold level crossings of the density function tails due to increased noise or ISI.

References

Chris Dick, Michael rice, and fred harris, “*Synchronization in Software Radios: Carrier and Timing Recovery Using FPGAs*”, IEEE Symposium On Field-Programmable Custom Computing Machines, Napa Valley CA, April 16-19, 2000

Chris Dick and fred harris, “*FPGA Signal Processing Using Sigma-Delta Modulation*”, IEEE Special Issue Magazine on Industrial Signal Processing Magazine, January 2000

fred harris, “*Band Edge Filtering and Processing for Timing and Timing Recovery*”, COMCON-7, Athens Greece, 28-June, 2-July 1999