CDMA_01 - a simulation of a simplified CDMA system

Günther Sinapius, gsinapius@onlinehome.de

Original Version 05/2002
Adaptation to Windows XP 01/2007

Adaptation to Windows XP **January 2007**

The program also runs under Windows XP. However, some default settings like buttons or default fonts seem to differ from those of Windows 98. Therefore the layout had to be revised. In addition minor amendments were made to the program.

Original Version **May 2002**

I. Introduction

This program simulates the transmission and reception in a simplified CDMA system. Before the simulation is started, several parameters of the system can be choosen. Some parameters can also be modified later on. When a transmission/reception cycles has been finished, various displays can be obtained, e.g. signal amplitude vs. time and the operation of a Viterbi decoder. In the following the CDMA system, the input parameters, the outputs and the program are briefly described. The reason for writing this program was twofold:

- to gain a deeper understanding of CDMA systems,
- to get an idea of object-orientated programming in C++.

In their imperfection, both, the CDMA system and itsimplementation in C++ reflect this starting point.

In the CDMA system the main emphasis is put on the results of the coding and modulation and the corresponding demodulation and decoding stages. The signals which result from each of theses stages can be visualized in the form of plots.

The following issues are not treated consistently or not included and could be improved

- the normalisation after the analog modulation stages,
- the channel, i.e. Rayleigh fading, delay spread. frequency selective fading and the Doppler effect,

- the RAKE receiver,
- the exchange of data between the different C++ classes (see also sect. VI. below),
- superfluous "include" commands,
- the visual design.

II. How to run the program

The program has been written with the Borland C++ Builder 5, for Windows 2000/98/95/NT and should therefore run under these operating systems. To run this program, the following files have to be in the sane directory

- CDMA_01.exe (the program) and
- borlndmm.dll, vcl50.bpl , cc3250mt.dll

The screen resolution has to be set to 1024 x 768 pixels. To start the program, double-click CDMA_01.exe.

When started, the program comes up with a start-up menue, which allows some running paramters to be set. After setting the parameters click "setup" to proceed to the run-time menue appears which allows you to

- modify some of the running parameters;
- modify the time scale for the display of the signals;
- start the simulation of a transmission/reception cycle.

When the signals for such a cycle have been calculated, the run-time menue presents various choices for inspecting the signals or to obtain information on elements of the system.

III. The CDMA system

The CDMA system is made up from seven logical stages which comprise respective modulators/encoders and corresponding demodulators/decoders, cf. Appendix 1. Stage 0 is a dummy to allow the input of data bytes which subsequently are converted to data bits. Stages 1-5 perform the digital signal processing and in stage 6 the analog processing takes place. For testing purposes the system can be simplified by selectively bypassing/switching off any of stages >= 2 and the subequent higher stages. In stage 6, the Q branch, the IF and the RF de/modulation and four optional bandpass filters can be selectively de/activated. These options do not reflect realistic choices for real CDMA systems.

IV. Input parameters

In the start menue of the program four buttons to view/change running parameters of the program are shown.They are relate to (for details cf. appendix 2):

- start parameters
  can beviewed/modified at this stage only, because they influence the memory allocation for the program.

- analog modulation parameters
- bandpass filter parameters
- noise/distortion parameters
  parameters to be viewed/modified for every new transmission/reception cycle

The analog modulation, bandpass filter and noise/distortion parameters can also be set in the run-time menue.

V. Outputs

Once a transmission/reception cycle is completed, the following can be displayed

1	a	by clicking the blue "?" on the diagram of the system: the signal amplitudes vs. time after each of the stages indicated in Appendix 1 ("?" should appear only after those stages activated by the corresponding input parameters)
1	b	after an amplitude spectrum has been chosen for display, corresponding FFT frequency spectrum can be obtained by clicking the "FFT" button the time axis (and as consequence thereof the frequency axis of the frequency spectra) can be changed in two ways • the width of the time window displayed (in steps of powers of 2) • the time zero of the display (in steps of the width of the time window). When the time axis is changed, the amplitued vs. time plots (configured as MDI children) are updated immediately whereas the FFT plots are erased and have to be chosen again if desired.
2		an eye pattern of the Nyquist pulses (in the I or Q branch) either • after the DAC of the transmitter or • after the IF stage, before the ADC in the receiver.
3		the operation of the r=1/2, K=3 encoder and of the Viterbi decoder. After the selection of one of either one, its operation is displayed on a bit by bit basis for the 24 first encoded and decoded bits by repeatedly clicking the "proceed" button.
3	a	For the encoder its input and output, the output of the XOR gates and the state of the respective memory cells is shown. When the encoder is in a given state, the value of the next input bit determines the next state of the encoder (this can be better seen in the trellis for the Viterbi decoder).
3	b	For the Viterbi decoder the corresponding trellis for two subsequent decoding steps is shown. The rows of the trellis represent the different states (00, 01, 10 and 11) the encoder, every new input bit to the encoder gives rise to a subsequent column of the trellis. Transitions between subsequent encoder states depend on the current state of the encoder and the value of the data bit (0 or 1), the possible transitions are shown as full and dashed lines for 0 and 1 respectively.

		The output of the Viterbi decoder is that path connecting subsequent decoder states for which the sum of squared deviations (between the two received signals and the nominal values for a given transition) over all elements, i.e. transitions, of the path is minimal. The evolving path is drawn in bold, its color is chosen according to the most probable values of the corresponding pair of received signals (blue, green, velvet and red for 00, 01, 10 and 11 respectively), the value of the corresponding data bit is indicated by the style of the path (full line or dashed lines for 0 or 1). Underneath the trellis plots the following values are displayed • the value (0 or 1) of the respective encoded bit, • the value (0 or 1) of the respective decoded bit, which results in state 00, • the amplitude of the two received bits which are decoded, • the accumulated squared deviations from the theoretical values of the received data along the path ending in the states 00, 01, 10 and 11.
4		the following properties of the Walsh functions/states of PN generators (U, I, Q) implemented
4	a	two selected functions/states, their product and the integral of this product over the length of the Walsh functions/PN generator,
4	b	the correlation of a selected Walsh function/state of a PN generator with • all other functions in case of Walsh functions • a selected number of states (< the maximum number of available states) of the PN generator in the case of states of PN generators
4	c	for PN generators only: the arrangement and parameters for two (by default subsequent) states of the PN generator together with a)
4	d	for PN generators only: the time required to generate a predetermined number of states of the PN generator.
5		the operation of bandpass filters of the type used in this program for two kinds of functions the frequencies of which have the following charactersistics • freqency = f(t) • frequency = sum over contributions with different frequencies.

VI. Programming notes

VI.a. The main elements of the program are

- the file CDMA_01.cpp - the WINDOWS program
- the file main_screen.cpp - the form TMain_screen
- the file main_program.cpp - the main program to control the program flow

VI.b. Logics

- I/O frames with one instance, which contain all their programs and get data via "extern":
  eye_patterns
  spreading_fcts

- I/O frames with one instance, which contain all their programs and transfer data via transfer routines, which transfer resp. addresses

set_times
parameter_menue

- output frames with more than one instance, which contain no programs or paramters
  signal_plot:
  calls on paint plot_data((TSignal_plot*)((TPaintBox*)Sender)->Parent)
  fourier_plot:
  calls on paint plot_fft((TPaintBox *)Sender)
  pn_generator:
  made visible by spreading_ftcs and filled by Plot_gen which is called by spreading_ftcs

VII. References

Backgrounds on the following topics can be found in the books and articles indicated below. This selection is rather arbitrary and reflects essentially which texts I was familiar with and which were at my disposition when I dealt with particular aspects of the program.

CDMA, en/decoding	Andrew J. Viterbi "Principles of Spread Spectrum Communication", Addison-Wesley, 1995
Analog modulation	Leon W. Couch II "Digital and Analog Communication System", Prentice Hall, 1997
CDMA systems	Irving Kalet "CDMA-The Physical Interface-Third Generation WCDMA", course at the EPO Munich, 2000, CEI Europe
CDMA systems	patent US-A-5 103 459 (Gilhousenet al.), 1992
Walsh decoding	patent US-A-5 442 627 (Viterbi et al.), 1995
PN generators	patent applicat. EP-A-0 955 733 (Sony), 1999
FFT, bandpass filter	William H. Press et al. "Numerical Recipes in C", Cambridge university Press, 1988

Appendix 1:

Stage/ step	***action***	***input (bits/chips)***	***output (bits/chips)***	***output freq. (MHz)***
0	vocoder	n 8-bit chars	n*8	0,0144
1	convolutional encoder; r=1/2,K=3	n*8	2*input	0,0288
2	block interleaver	n*2*8	=input	0,0288
3	Walsh modulator	n*2*8	(64/6)*input	0,3072
4	PN-U spreading	n*2*8*64/6	4*input	1,2288
5	PN I, Q spreading	n*2*8*64*4/6	=input	1,2280
6a	DAC (Nyquist pulses)	n*2*8*64*4/6	baseband	1,2288
6b	IF-mixers	I, Q	I*sin wt, Q*cos wt (IF)	
7	first bandpass			
8	RF mixer	I*sin, Q*cos	* sin wt (RF)	
9	second bandpass		s	
10	distortions	s	s'	
11	RF mixer	s'	s'*sin wt (RF)	
-10	third bandpass			
-9	IF mixers	I, Q	I*sin wt, Q*cos wt (IF)	
-8	fourth bandpass			
-7	DAC	I, Q	av. n/chip	1,2288
-6				
-5	PN I, Q despread	n*2*8*64*4/6	=input	1,2288
-4	PN-U despread (average over 4 chips)	n*2*8*64*4/6	input/4	0,3072
-3	Hadamard transform / I Q Walsh demodulator	n*2*8*64/6	input*6/64	0,0288
-2	block deinterleave	n*2*8	=input	0,0288
-1	Viterbi decoder	n*2*8	n*8	0,0144

Appendix 2:

The meaning of the running parameters is as follows

start parameters:	• logical stages of the system = 1,...6; • nr of data bytes to be transmitted: 3, 9 or 36 bytes (at a data rate of 14.4 khz, 36 bytes correspond to a frame of 20 ms);
analog modulation:	• number of neighbouring Nyquist pulses to integrate in the DA conversion; • roll off factor of the Nyquist pulses"; • number of samples per chip to be taken and averaged in the AD conversion; • number of I Q branches, 1 == only I, 2 == I and Q; • IF frequency (Mhz); • IF modulation/demodulation state: 0 == not performed, 1 == performed; • RF frequency (Mhz); • RF modulation/demodulation state: 0 == not performed, 1 == performed;
bandpass filters:	for the four bandpass filters • state (0 == off, 1 == active); • center frequency (Mhz); • width(Mhz);
noise/distortion:	• amplitude of an unmodulated RF noise signal relative to the amplitude of modulating RF signal (noise to signal); • relative frequency and phase shift of the demodulating IF signal (%) • relative frequency and phase shift of the demodulating RF signal (%).