Inquiry: The University of Arkansas Undergraduate Research Journal

Volume 2 Article 17

Fall 2001

A Method of Implementing the Interleaver in 3G Wireless Communication Systems for Randomization of Burst Errors

Eric Duquette
University of Arkansas, Fayetteville

A METHOD OF IMPLEMENTING THE INTERLEAVER IN 3G WIRELESS COMMUNICATION SYSTEMS FOR RANDOMIZATION OF BURST ERRORS

by Eric Duquette
Department of Electrical Engineering
College of Engineering

Faculty Mentor: Wookwon Lee
Department of Electrical Engineering

Abstract:

In less than a year or two, third generation (3G) wireless communication systems will be replacing the currently existing second generation (2G) wireless systems and be used worldwide. The 3G systems will have a higher data transfer capacity, which allows for simultaneous data, voice, and video transfer (e.g., multimedia applications) while most of 2G systems only allow for low-rate data and voice transfer. The wireless communications industry has set standards for 3G, which detail various operations of the technology. It should be noted that the 3G standards outline only the different operations and not the processes for the production of these operations. The research presented addresses a method of production of the interleaver operation for randomization of burst errors.

In this paper, the details of the interleaver are described and a novel method to implement the interleaving operation in a 3G system, called Universal Mobile Telecommunication Systems (UMTS) is presented. The implementation was aided by a sophisticated tool from Cadence, "Signal Processing Workstation" (SPW), which was used in both the design and testing of the interleaver. Of the two interleavers in UTMS standards, the first interleaver is the focus this research. However, this research would be applicable to the second interleaver.

Introduction:

The research presented discusses a method in the design and testing of the first interleaver operation in the Universal Mobile Telecommunication Systems (UMTS) third generation (3G) wireless technology standard. The research was done for the Wireless Communications Research Group (WCRG) of the University of Arkansas. The design and testing was aided by the Cadence software tool "Signal Processing Workstation" (SPW). SPW uses a graphical interface to allow users to build floating-point and fixed-point digital circuitry. Graphical representations of important digital blocks (such as memory, logic, and math blocks) in SPW expedite design of a digital system. To understand the design of the interleaver operation presented, it is first important to understand the basic reason for the need of the interleaver in wireless communications. The transmission of data signals through a transmission channel (air in this case) is subject to random disturbances. These disturbances can cause random errors in the data being transmitted. Through a process called channel encoding, the random errors can be compensated for before sending the signal by attaching additional information about the original data to the end of the original data. After the signal, along with its random errors, is received by the receiver, the additional information sent with the original data is decoded (the inversion of encoding) to correct the random errors.

Unfortunately, the transmission of a signal in air is also subject to burst errors. Burst errors occur in response to a fading transmission channel, where a fading channel develops from a rapid change in the distance between the base station and mobile handset (in most cases, this is a cell phone). The problem with burst errors is that they cannot be corrected with encoding techniques alone. However, in the interleaving process, the burst errors can be converted to random errors, known as randomization, by interleaving before transmission and then deinterleaving after reception of the signal. The burst errors can then be corrected by channel encoding/decoding.

The interleaver operation:

Interleaving is a process in which the bits that compose a digital signal are reordered before transmission of the signal to compensate for the effects of burst errors resulting during signal propagation of the transmission channel. The type of bit reordering is determined by the standard of the communication system. In a typical interleaver, the bits are written into a memory block in sequential order, and then read out of the memory block in a different order. One way of thinking of the typical interleaver is to think of a matrix representation of the memory block. The bits are first written into the memory row by row (starting at the top), and after filling up the memory, the columns of the matrix are then permuted. Permutation is a reordering of objects. One

permutation of the sequence of numbers {0, 1, 2, 3} might be {0, 2, 1, 3}. In this case, the set {0, 2, 1, 3} is said to be a permuted sequence of the original sequence. In general, for a sequence of n numbers there are n! = n(n-1)(n-2)...(2)(1) different permutations of that sequence. Thus, when the columns in the interleaver are permuted, the order in which they appear in the matrix changes. The output signal of the interleaver is then fabricated by reading the permuted matrix column by column (from the left) (see Figure 1). At this point, the signal is said to be interleaved.

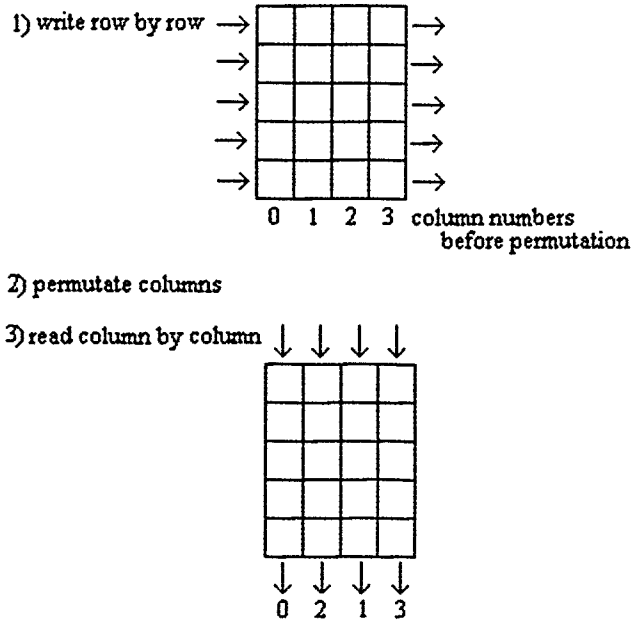

Figure 1. *Example of the three steps needed to generate the ideal permuted data sequence.*

Several parameters are needed to successfully interleave. The first of these is the number of bits in the data signal, commonly referred to as the length of the signal. The length of the signal must always be known because the amount of memory allocated should to be the same as the length of the signal. The number of columns and the type of column permutation must also be known. This research used parameters specified by the UMTS 3G standard.

Design:

The interleaver's basic algorithmic architecture was designed to meet the specifications outlined in the UTMS standards. The design of the interleaver can be broken down into five sections: memory allocation, writing to memory, column permutation, reading from memory, and changing read/write states. As explained above, the interleaver reorders the bits of a data signal. For practical reasons, the number of bits must be finite. Also, the number of bits must be small, often less than a thousand (for reasons beyond the scope of this paper). The number of bits needed to transmit a second's worth of voice signal is approximately seventy thousand bits. Therefore, the transmission of a data signal is processed in small intervals. The time it takes to transmit these small intervals is called the transmission time interval (TTI). The TTI is measured in milliseconds (ms) and is given as a parameter in the UTMS standard. The number of bits in one TTI is a small portion of the number of bits in the total data signal. Consequently, a certain number of TTI's must be processed in a communication system in order to deliver the whole data signal. The number of bits in a TTI will be denoted as L. In the UTMS standard, the TTI has possible values of 10ms, 20ms, 40ms, and 80ms. For this research, the different TTI's will correspond with the values of *TTI#*, which are 0, 1, 2, and 3 (for example, if the TTI is 40 ms, *TTI#* will be 2). The total number of columns, *col_total*, needed in the memory of the interleaver is dependent on the TTI of the system. The number of columns for a *TTI#* of 0, 1, 2, and 3 is 1, 2, 4, and 8 columns, respectively. The permutation column sequence for the different number of columns, as specified by the UTMS standard, is {0}, {0, 1}, {0, 2, 1, 3}, and {0, 4, 2, 6, 1, 5, 3, 7}, respectively. The permutation sequence for a particular value of *TTI#* will be denoted by $P_{TTI\#}$. Also, a value in a permutation sequence will be denoted by s_n, where n is the nth value in the sequence starting with zero (for example, the value of s_6 in P_3 is 3). The total number of rows, *row_total*, needed in the memory is just *L* divided by *col_total* .

A delay in data signal is usually unwanted in a communication system. For instance, a delay in hearing a voice on a cell phone would be irritating to the listener. A continuous data signal should be kept in order to avoid pauses, or delays, in the reception of the signal. Hence, when processing multiple TTI's, there should be no delays between the different TTI's at the output of the interleaver. These delays were avoided by using two different random access memory (RAM) blocks, MEM1 and MEM2 (see Figure 2).

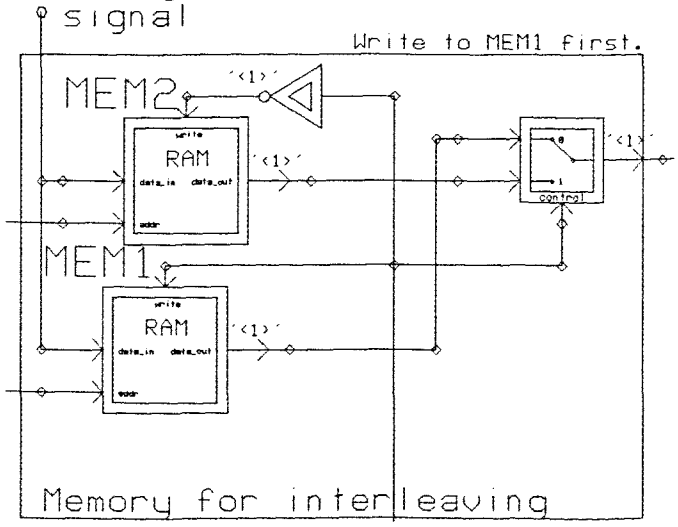

Figure 2. *(A snapshot from the final design). Two RAM blocks, MEM1 and MEM2, are used for the first interleaver memory allocation.*

During one TTI, L bits (the number of bits in a TTI) are being read from MEM1 and sent to the output on the interleaver. While this is going on, the next L bits from the input of the interleaver are being written into MEM2. On the next TTI, MEM1 is being written into, while MEM2 is being read from.

This pattern continues, where the read/write processes alternate among MEM1 and MEM2 on each subsequent TTI. Thus, a two-block memory allocation was implemented to maintain a continuous output data signal with no delay between adjacent input TTI's.

A memory block in SPW has an unspecified number of memory locations inside the block in which a single bit of information can be stored. For this research, the total number of locations for each of MEM1 and MEM2 is L. Each location in the memory has its address specified in decimal value, with the first location as value zero and the last location as value L-1. To write to a location in the memory, the appropriate address value must be inputted into the address input pin of the memory. In the design, the appropriate address value, when writing to either MEM1 or MEM2, was generated by one counter, denoted CT1 (see Figure 3). A counter is a device which, as indicative of its name, counts in an orderly fashion. The counter CT1 starts at zero and counts by one up to L-1. After the counter reaches L-1, it rolls back over to zero and starts over again. Every counter has a count output pin. The count output pin outputs the present value of the counter. Using the count output pin of CT1 as the address location for MEM1 and MEM2 during their respective writing cycles, each bit per TTI is loaded in sequential locations in the memory from zero to L-1 for a total of L bits. Thus, at the end of a TTI, the location of each bit will have the same order in the memory as the order inputted to the interleaver.

Ideally, after writing the bits per TTI to either MEM1 or MEM2, the columns in the memory must be column-wise permuted. In this design, permutation of the columns is not done directly before reading because that would increase the delay of the data signal to the output. Instead, the permutation step is bypassed and incorporated into the reading step. So, the process of writing to the memory row by row, permutating the columns, and then reading from the memory column by column is shortened to writing to the memory row by row and reading from the memory in a different sequence. This sequence of reading produces the same output as in the ideal case when the columns would be first permuted and then the memory would be read column by column. The problem of generating the correct interleaved output sequence now resides in how the memory should be read.

Reading from MEM1 and MEM2 during their respective reading cycles is accomplished with the use of two counters, CT2 and CT3, four read-only memories (ROM's), and a constant value block. Before going into detail about how the different SPW blocks were used to read from MEM1 and MEM2, the basic idea behind the method needs to be explained. In the ideal case, the interleaver design is centered about a memory block represented as a *row_total* by *col_total* matrix. Each position in the matrix has some address location associated with it. In this research, the memory location addresses are numbered in respect to the input sequence. For instance, memory address 1 contains the first bit of the input sequence and memory address 2 contains the second bit, continuing until the last bit of L bits. From this knowledge, the sequence of address locations that should be read from the memory for proper interleaving can be summarized in an equation. If *c_addr* represents the current address location that is being read from the memory, then $c\_addr = (\ c\_count\ )(\ col\_total\ ) + s_n$, where *c_count* is the current count value, *col_tot* is the total number of columns being used, and s_n is the current column in the column permutation sequence for a particular *TTI#*. The current count value, *c_count*, controls when the *c_add* changes to its next address location. The current count value is implemented using the counter, CT2 (see Figure 3). Since the read operation is done column by column, the maximum number of CT2 should match the total number of memory locations per column, which is the total number of rows. In this case, the maximum value of CT2 is simply *row_total*. For a *TTI#* of 2 and L of 20 bits, the sequence of *c_addr* is { 0, 4, 8, 12, 16, 2, 6, 10, 14, 18, 1, 5, 9, 13, 17, 3, 7, 11, 15}. Now, the hardware can be implemented by using the guidance of the *c_addr* equation. In the design, the *c_count* value is the same as the output count of CT2. Whenever the count of CT2 rolls back over to zero, the next column in the column permutation sequence is counted. Therefore, s_n should be incremented every time *c_count* becomes zero (excluding the first time *c_count* is zero) and is done so by using another counter, CT3 (see Figure 3).

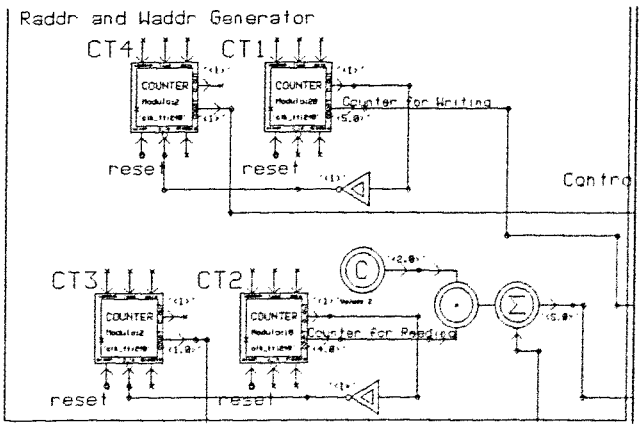

Figure 3. *A close up of the four counters in the first interleaver final design (TTI# = 1, L =20).*

The four ROM's contain the four different permutation sequences P_0, P_1, P_2, and P_3, respectively. The permutation sequence in each ROM is generated by reading the memory of

the ROM in sequential address locations (starting with zero). This is accomplished with the counter output of CT3. Thus, CT3 should count only whenever CT2 has reaches zero, as stated above. This type of configuration is common with the use of counters (especially when used in the design of clocks) and was easily arranged by using special purpose output and input pins named c_in and c_out on the SPW counter. Now, whenever value CT2 reaches zero, the value of CT3 increments by one and the next permutation value, s_n , in $P_{TTI\#}$ is outputted by $P_{TTI\#}$'s respective ROM. The *col_total* is a constant value in the interleaver, and thus, does not change in the operation of the interleaver. Hence, *col_total*'s value is controlled by an SPW constant block. All necessary values for the *c_addr* equation have now been developed in the hardware and are ready for the mathematical operations as described in the *c_addr* equation. SPW has a math block that has three inputs. Two of the three inputs are multiplied by each other and the third is added to the result of that multiplication. This SPW block is used in the design to multiply *c_count* by *col_total* and then add s_n to that result to obtain *c_addr*. The current address, *c_addr*, becomes the input to MEM1 and MEM2 during their respective read cycles.

The discussion of the design so far has just been for the processing of one TTI. Since an ideal data signal will be composed of many TTI's, the interleaver must accommodate for multiple TTI's. The design of the interleaver must be able to change states from reading MEM1 and writing to MEM2 to writing to MEM1 and reading from MEM2, and vice versa to process more then one TTI. The changing of read/write states in the interleaver is achieved by using an additional counter, CT4 (see Figure 3). CT4 has only two values, zero and one. CT4 is dependent on CT1 like CT3 is dependent on CT2. That is, CT4 is only incremented when CT1 reaches its maximum value of L-1. Thus, CT4 changes its value between 1 and 0 after every TTI length of bits. The value of CT4 is used to control several switches. One switch redirects the read/write address locations between MEM1 and MEM2 and changes the read/write state on MEM1 and MEM2 at the same time. Another switch changes the output of the interleaver between the output of MEM1 and the output of MEM2.

All of the necessary operations of the interleaver design are now finished. For the complete view of the design see Figure 4. Testing is the last stage in the completion of the interleaver operation.

Testing:

Testing of the interleaver was done with the Signal Calculator (SC) tool in the SPW software. When testing the interleaver, a data signal of a known number of bits was inputted into the interleaver. The operation of the interleaver was verified by confirming that the SC output of the interleaver matched the hand-calculated output for a sufficient number of bits at the beginning and the end of the output signal. SC is a graphical display, which proved to be beneficial when scrolling through a large number of bits.

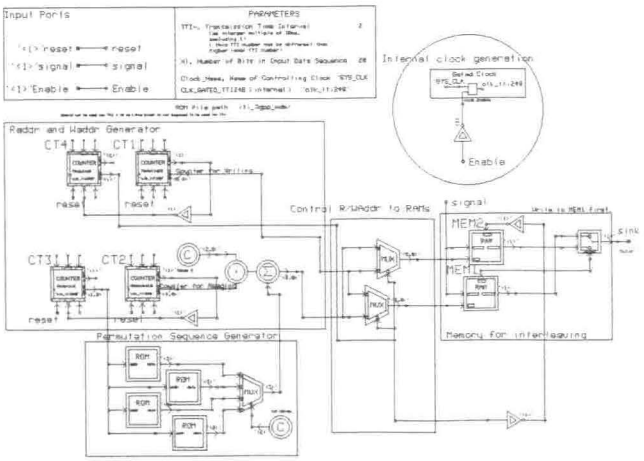

Figure 4. Final design of first interleaver (TTI# = 1, L = 20)

Conclusion:

The testing verified that the final design of the interleaver for randomization of burst errors was correct. The final design was completed after several other designing and testing phases and became part of a much bigger project headed by my mentor. The interleaver was the first design of many other designs for that project. For this reason, the inteleaver was the most challenging and problematic of all my designs. The pace and accuracy of my other designs improved from my initial work on the interleaver. Any future designs for the WRCG will without a doubt enjoy the same benefits.

In wireless communication systems, a data signal often has to be processed during various stages of the transmission and reception of the signal. A processed signal can be defined as a signal that has to be analyzed or changed. Whenever a signal has to be processed, it causes a delay in the amount of time it takes the signal to reach its final destination. It is important then that a communication system have sufficiently small delay. The outcome of the research was a functional implementation of the first interleaver under the UTMS standards that avoided delays in between adjacent TTI's and delays in the permutation of the data signal. For this reason, the final design is advantageous to any UTMS 3G technology. This is important since second generation (2G) wireless technology systems will soon be replaced by 3G systems in the next few years. 3G technology is better because it will offer a higher data rate and data capacity, allowing for simultaneous multimedia applications, such as data, voice, and video transfer.

Works Consulted:

3GPP 3G TS 25.201, "Physical layer – general description," Release 99, V3. 1.0 , June 2000.

3GPP 3G TS 25.212, "Multiplexing and channel coding," Release 99, V3. 1.0 , June 2000.

V.K. Bhargava and I.J. Fair, "Forward Error Correct Coding," *The Communications Handbook*, ed. Jerry D. Gibson. Boca Raton: CRC Press, 1997. 177-178.

A. Paulraj , "Diversity Techniques," *The Communications Handbook*, ed. Jerry D. Gibson. Boca Raton: CRC Press, 1997. 216-217.

Eric Duquette

Faculty comments

Mr. Duquette's faculty mentor, Wookwon Lee, made the following comments about the work:

> Eric Duquette's paper describes in a concise manner the fundamentals of the interleaver, which is an essential component in wireless communications, and its implementation using creative control circuitry along with use of memory components. As described in the paper, the core part of the interleaver design was the control circuitry to write and read information data to/from memory components in a timely fashion in order to transmit data in a noisy radio channel in a form that can be reliably received and decoded at the receiving end. The interleaver also has to accommodate various transmission-time intervals that might be associated with greater complexity for the entire interleaving operation. Eric successfully designed this "might-have-been-complicated" interleaver with an efficient control circuitry by using a few counters for generation of write/read addresses for the memory components.
>
> I found Eric to be an innovative, hard working undergraduate student. He has made excellent contributions to my own research project for the design of a sophisticated transmitter for a European-proposed, wideband Code-Division, Multiple-Access (W-CDMA) system for the next generation wireless terrestrial cellular system designed to provide multimedia services to ordinary cellular phone customers. I am confident that his work on the interleaver described in this paper is the result of creativity and innovation that can rarely be found in an undergraduate student, particularly in one who had little background in the areas of wireless communications when he began the project.

Neil Schmitt was also enthusiastic about Mr. Duquette's work. He said:

> It is my pleasure to provide a letter of support on behalf of Mr. Eric Duquette who is submitting some of his undergraduate research work for publication in *Inquiry*, the University of Arkansas undergraduate journal. Eric is a responsible individual who has achieved an impressive academic record in electrical engineering. His quiet unassuming approach to technical challenges belies his tenacious pursuit of solutions that are economically and technically feasible. Eric distinquished himself by becoming the top student in my Digital Signal Processing class - a class most students find very challenging.
>
> I am intimately familiar with the research topic that Eric elected to pursue. It would be very very appropriate for a Master's thesis topic. In spite of not having the academic coursework one would normally assume to be a pre-requisite to attacking this problem, Eric has achieved significant results. His efforts involve state-of-the-art research on a component of a new digital communication system that is destined to replace what is currently in use in the United States and that will be used world-wide as well. His design had to comply with international standards.