TSG-RAN Working Group 1 meeting #3
Eskilstuna, Sweden
March 22-26, 1999

3GPP/TSGR.1#3(99)-299

TITLE:

Complexity analysis of the Motorola Turbo Code Interleaver

SOURCE:

Motorola

ABSTRACT:

This contribution provides an analysis of the hardware complexities of the proposed turbo code interleavers from Motorola, Hughes/Nortel, and the NTT DoCoMo merged proposal.

1.0 Introduction

This contribution compares the hardware implementation complexity of the proposed turbo code interleavers. In order to make a fair comparison among the interleavers, it is assumed that each interleaver supports all possible turbo code block lengths from 320 up to 8,192 bits in length.

2.0 Motorola Interleaver

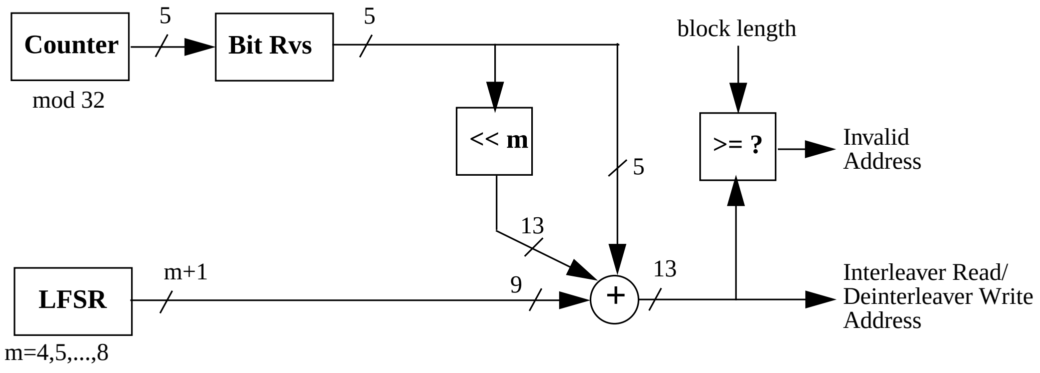
Figure 1 shows the implementation of the Motorola turbo code interleaver. The circuit has a maximum latency of two clocks (no consecutive puncturing). The gate count estimate is shown below in Table 1.

Module	Gate Equivalents	Comment
Counter	60	5-bit counter
Bit Reversal	0	
LFSR	194	(10g FF + 3g XOR) per bit * 8 bits + 5x8 LUT (40g) + 50g augmentation circuitry
Shift Left m Bits	60	Muxes required to shift a 5 bit value from 4-12. (3g/mux)*20 = 60g
Adder	155	15g per bit * 5 bits + 10g per bit * 8 bits
Comparator	130	10g per bit * 13 bits
Output Address Latch	130	10g FF per bit * 13 bits
TOTAL	729	

Table 1

Note that this implementation can support any block size from 320 bits up to 8,224 bits using a look up table (LUT) which contains only 5, 8-bit values (this can be a very small ROM but more likely would be synthesized into logic gates). **No other parameters are required**. If larger block sizes are required, this interleaver could easily be modified. For example, to handle a block size of 81920 bits (1.024 Mbps, 80 ms block), this interleaver would only grow to 1050 gates.

The 729 gates are all that is required to implement the interleaver. No additional parameters need to be stored and no additional memory is required. This interleaver is very well suited for hardware implementation. Interleaver addresses can be generated real-time. They do not need to be generated prior to use and stored. The interleaver is very flexible and easily adapts to changing block size.



Motorola Interleaver

Fig. 1

3.0 Hughes-Nortel Prunable interleaver

Based on Motorola's understanding, Table 2 shows estimated gate counts for the elements of the Prunable interleaver. Row permutation and the final address comparator (for puncturing) and the output latch are similar to the Motorola interleaver. In addition a column permutation must be done which requires a 7 x 13 bit multiplication and a modulo computation. Furthermore the prunable interleaver parameters must be stored. This requires a minimum of 1224 bits of ROM.

Module	Gate Equivalents	Comment
Counter	72	6-bit counter
Bit Reversal	0	
Shift Left m Bits	105	Muxes required to shift a 6 bit value from 3-7. (3g/mux)*35 = 105g
Adder	160	15g per bit * 6 bits + 10g per bit * 7 bits
Comparator	130	10g per bit * 13 bits
Output Address Latch	130	10g FF per bit * 13 bits
Sub-total	597	
multiplier	1000	7 x 13 bits, ~1k gates
ROM	1224	1224 bits, 1g per bit
TOTAL	2881	

Table 2

If larger block sizes were desired, the ROM table would grow significantly. The other logic would grow similarly to the Motorola interleaver.

4.0 NTT DoCoMo merged proposal

Based on the information given in Tdoc TSGR1#3(99)217, we were not able to do a detailed complexity analysis of the NTT DoCoMo merged proposal. Some observations are listed below based on the range of block sizes of 320 to 8192.

- Larger LUT (345) bits for root of prime number table for prime numbers from 31 to 409. This would get significantly larger if larger block sizes were desired.
- A multiplier and modulo operation are required. Optionally another LUT could be used but this would be large: 9 bits * sum of primes from 31 to 409.
- Need to select I_i's: computational complexity or 10984 bit LUT.
- Need to apply algebraic interleaving by factor I_i on each column, where sequence interleaved is index table used for 1st column permutation. This requires complexity of algebraic interleaver in and of itself.
- Need to determine if the number of columns is prime, prime+1, or prime-1 and have augmentation circuitry for each case as applicable.
- LUT for predetermined interleaver pattern (pip) of lengths 10 and 20: (140 bits).
- Still requires puncturing like Motorola and Hughes/Nortel.

To do a detailed complexity analysis, a more detailed description is needed.

- Given a block size, how is the number of columns calculated?
- How many candidate column sizes will need to be implemented?
- What parameters are best suited for LUT and what should be calculated real-time?
- If the I_i's are calculated real-time, how is this done?
- How many consecutive addresses need to be punctured worst case?
- Would interleaver addresses be calculated real-time or would they be stored in a RAM table prior to use?

5.0 Performance

Simulation results in AWGN are shown in figures 2 and 3. All interleavers perform similarly. Results presented in a previous submission have also shown that the performance in fading channels is also identical.

6.0 Implementation Analysis Conclusion

The preceding analysis has shown that the hardware complexity of the turbo code interleaver proposed by Motorola is approximately 1/4 the complexity than the Hughes/Nortel approach and appears to be of much lower complexity compared to the NTT merged proposal. The Motorola interleaver does not require any ROM or RAM thereby simplifying the implementation as well as any self-test circuitry. Furthermore, the latency of the Motorola interleaver can be reduced to a single clock cycle with only a 20% increase in complexity over the two-clock version. At the mobile, in order to reduce power consumption the turbo decoder would most likely be implemented such that each decoder time step would require only a single clock. The ability to operate the turbo interleaver at the same clock frequency would simplify the implementation and further reduce power consumption.

Also note that since the Motorola turbo code interleaver has only one parameter (the LFSR polynomial) for each non-pruned interleaver length, it is much simpler to specify and verify. Given that all the proposed interleaver methods are very similar in performance, the turbo code interleaver should be chosen based on simplicity.

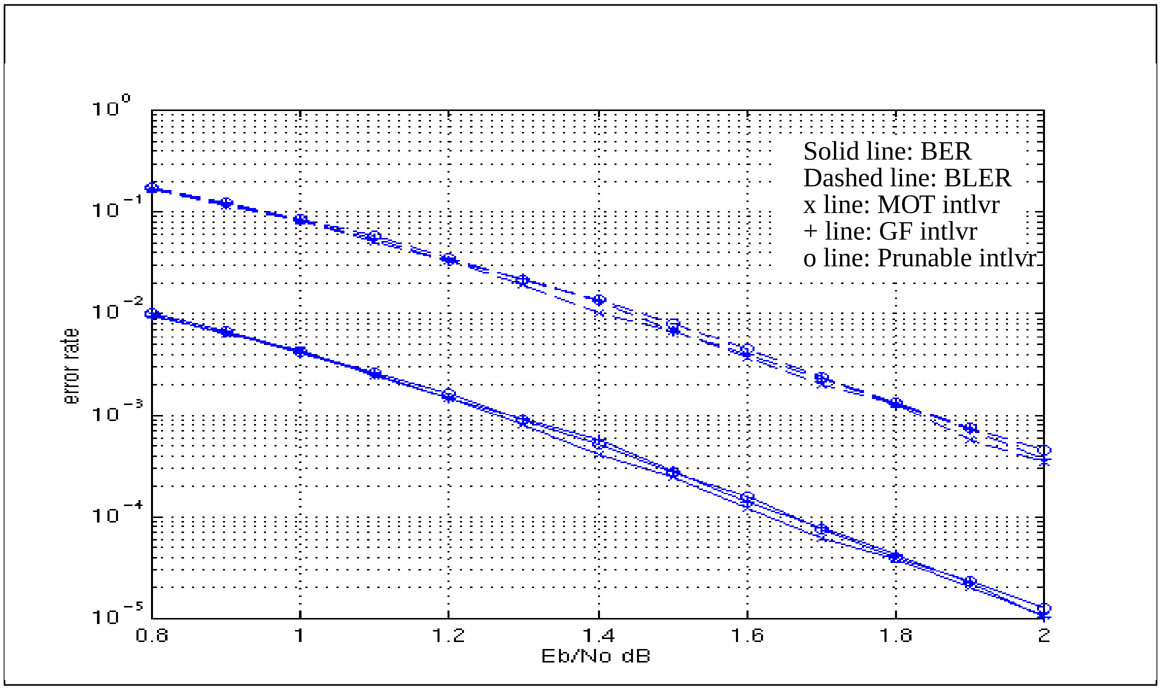


Figure 2 Turbo interleaver comparison using 8 states 4 iterations turbo code, data rate 32 kbps, 10 ms interleaving interval, static channel

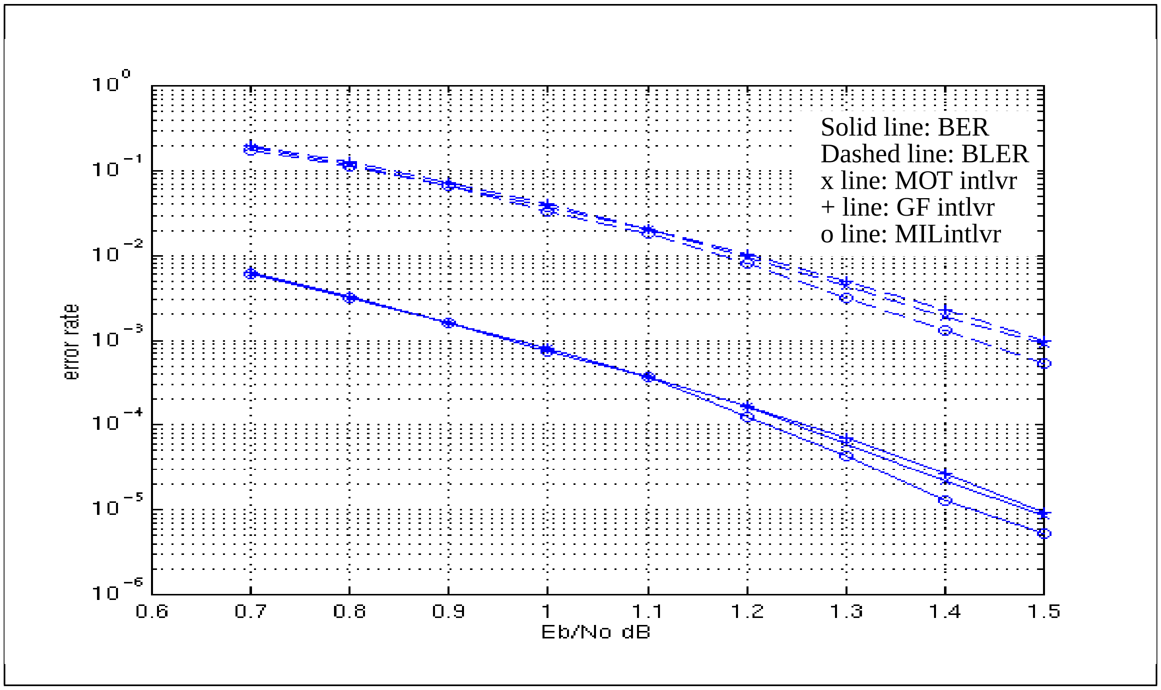


Figure 3 Turbo interleaver comparison using 8 states 4 iterations turbo code, data rate 64 kbps, 10 ms interleaving interval, static channel