Uplink Interference Cancellation in HSPA: Complexity Analysis

Sharad Sambhwani, Wei Zhang, Wei Zeng, Qualcomm Inc

Introduction

In [1], an interference cancellation (IC) scheme for the uplink of WCDMA/HSPA was presented. The motivation, theory, practical algorithms and the associated system performance benefits were presented. In this paper, we focus on the complexity analysis of the proposed IC scheme. Traditional hardware and software design philosophy usually considers the factors that influence processing power as deterministic variables. The design is typically dimensioned for the worst case situation. However, user behaviors such as packet arrival and successful decoding at the NodeB (Base Station) receiver are all random processes that cannot be described deterministically. The techniques of discrete event simulation are well suited to investigate the system requirement in the early stage of design. In this paper, we describe a methodology using discrete event simulation to study the complexity of the proposed IC scheme.

The rest of the paper is organized as follows. We first provide an overview of the interference cancellation schemes and assumed architecture. In particular, we are interested in the group IC scheme with scheduling [1]. Then we present the HSPA uplink system for a loaded VoIP application scenario, for which we perform a complexity analysis. Discrete event simulations are then developed to model the user traffic and the IC processing chain. Based on the user arrival statistics from simulation, we propose a baseline design. Finally, we perform a clock rate optimization for the different sub-systems in the IC architecture to further reduce the complexity.

Overview of Interference Cancellation

A discussion of various schemes to implement the post decoding IC is presented in [1]. Successive IC is an ideal scheme to achieve the sum-rate capacity of multiple access channels. However, a large processing delay is unavoidable due to serial decoding and cancellation processing. In Parallel IC, decoding for multiple users is carried out simultaneously to reduce the processing delay. Iterative processing can improve the performance of parallel IC by decoding the un-decoded users multiple times. On top of Parallel IC, smart scheduling can be added to further improve performance. The Group IC scheme is illustrated in Figure 1. Users are divided into groups according to certain criterion and then Parallel IC is performed on high priority groups to low priority groups successively. Iterative processing can be applied similarly. This scheme is a combination of Successive and Parallel IC in the case where users are grouped according to their decoding probabilities. The grouping approach can be also used to address the delay constraints, in which the low latency users are scheduled to a high priority group. IC scheduling plays a very important role in ensuring that the users are processed in the right order. The detailed explanation of such a scheduler is presented in [1]. This scheme is most attractive, not only because of the system performance benefits, but also the way it mimics the hardware implementation.

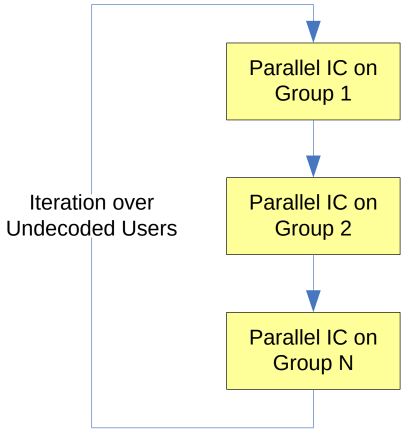

Figure 1: Iterative Group Interference Cancellation

In HSPA uplink, the data traffic is carried on the E-DCH Dedicated Physical Data Channel (E-DPDCH). A practical implementation of the traffic interference cancellation (TIC) given in [1] consists of the following processing blocks:

- Traffic Data Demodulation and Decode (TDD)
- Traffic Channel Decoder and Re-encoder
- Data Based Channel Estimation (DBCE)
- Traffic Interference Filtering (TIF)
- Traffic Interference Subtraction (TIS)

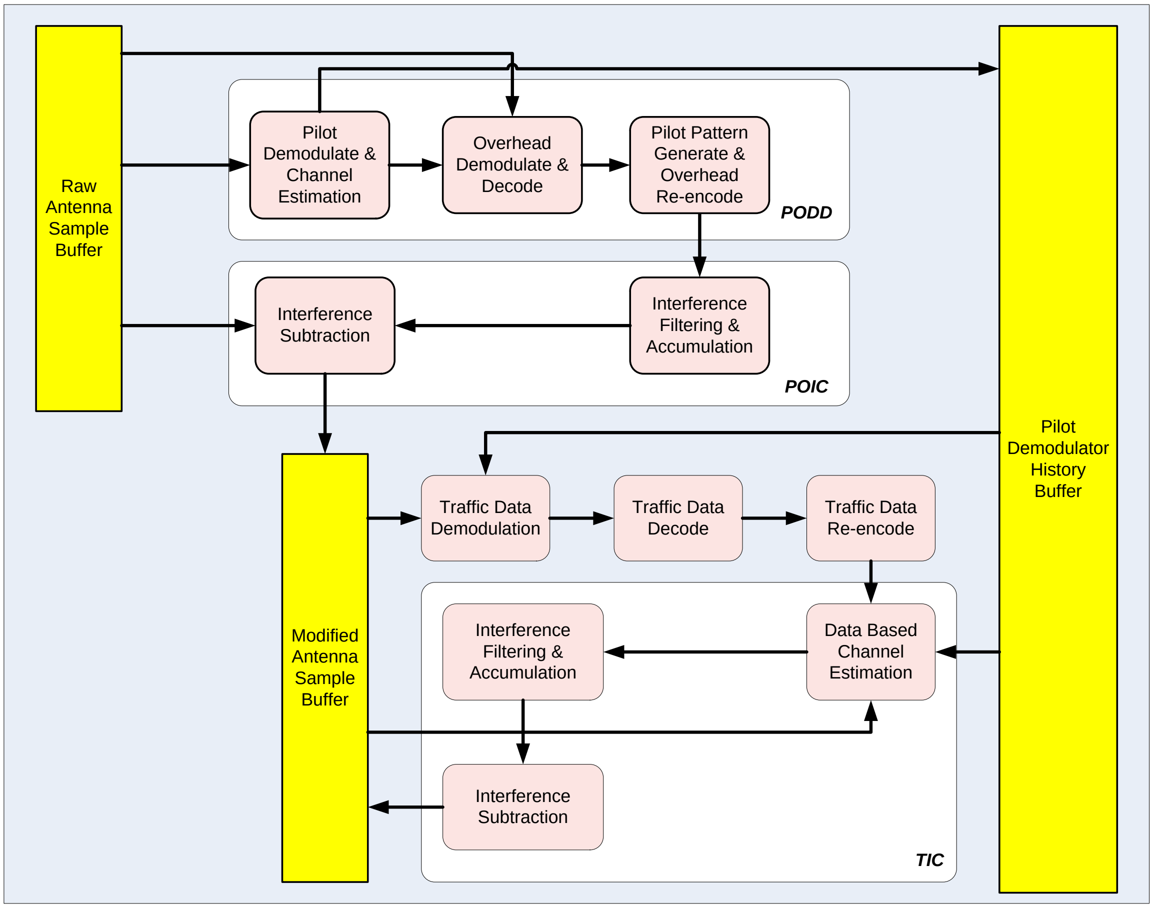
Raw Antenna Sample Buffer
Pilot Demodulate & Channel Estimation
Overhead Demodulate & Decode
Pilot Pattern Generate & Overhead Re-encode
PODD
Interference Subtraction
Interference Filtering & Accumulation
POIC
Pilot Demodulator History Buffer
Modified Antenna Sample Buffer
Traffic Data Demodulation
Traffic Data Decode
Traffic Data Re-encode
Interference Filtering & Accumulation
Data Based Channel Estimation
Interference Subtraction
TIC

Figure 2: Uplink Interference Cancellation Block Diagram

Figure 2 illustrates a high level block diagram of the uplink interference cancellation scheme as proposed in [1]. As seen in the figure, the TDD unit performs demodulation on the received signal in the waveform buffer. Once a data packet is successfully decoded by the channel decoder, the data is used as a pilot in the DBCE unit to re-estimate the channel. The interference waveform is then reconstructed using the new channel estimates and the re-encoded chip sequence. Finally, the interference waveform is subtracted from the waveform buffer. These subsystems are executed in a pipelined manner to accelerate the processing. The TIC processing can be illustrated as a timeline, as shown in Figure 3.

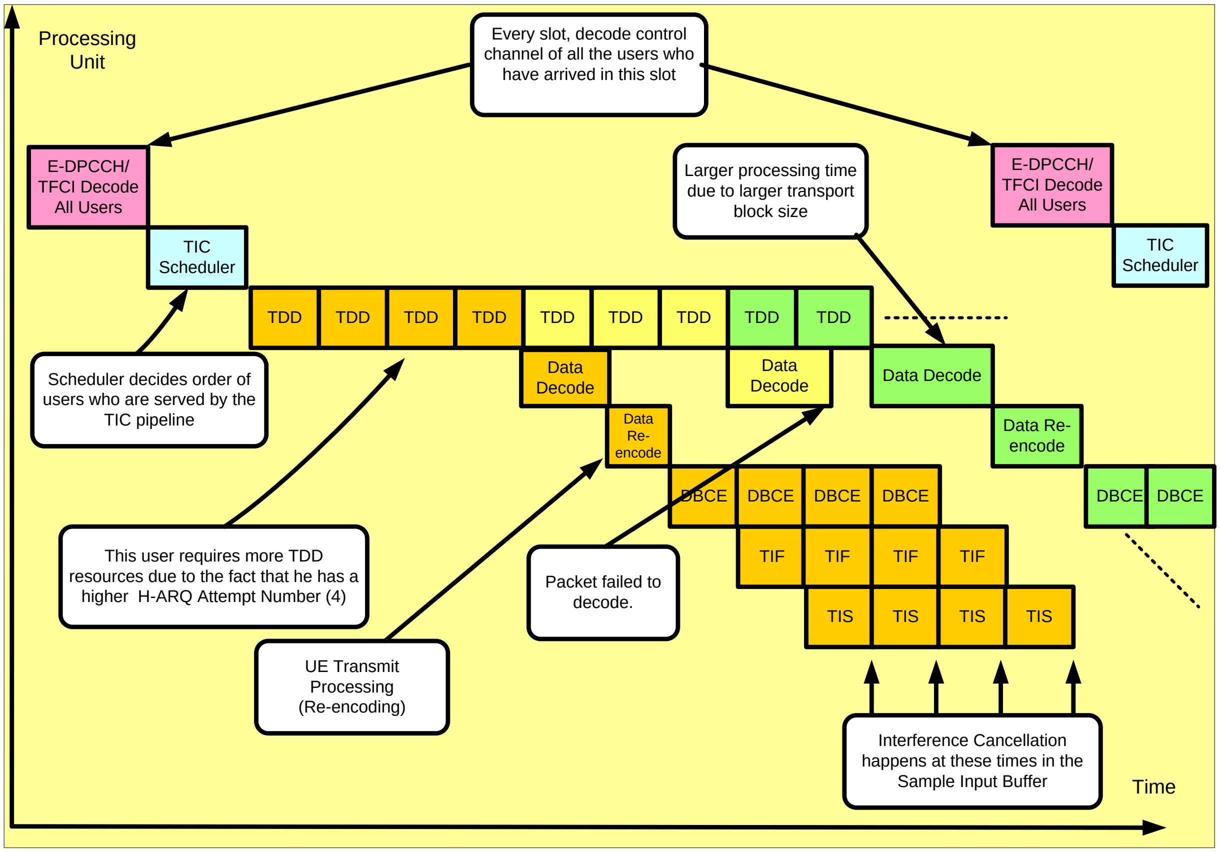
Processing Unit
Every slot, decode control channel of all the users who have arrived in this slot
E-DPCCH/ TFCI Decode All Users
TIC Scheduler
Larger processing time due to larger transport block size
E-DPCCH/ TFCI Decode All Users
TIC Scheduler
TDD
TDD
TDD
TDD
TDD
TDD
TDD
TDD
TDD
Scheduler decides order of users who are served by the TIC pipeline
Data Decode
Data Decode
Data Decode
Data Re-encode
Data Re-encode
DBCE
DBCE
DBCE
DBCE
DBCE
DBCE
TIF
TIF
TIF
TIF
TIS
TIS
TIS
TIS
This user requires more TDD resources due to the fact that he has a higher H-ARQ Attempt Number (4)
Packet failed to decode.
UE Transmit Processing (Re-encoding)
Interference Cancellation happens at these times in the Sample Input Buffer
Time

Figure 3: Uplink Interference Cancellation Processing Timeline

When a packet arrives at the antenna, it is processed by the different stages of the IC sub-system until the deadline to respond with an ACK/NACK to the user equipment (UE). In every slot, the TIC scheduler dispatches packets to the TIC processing pipeline. Packets are dispatched to the TDD, Data Decoder, DBCE, TIF and TIS in a sequential manner. Due to the delay of the pipeline, the packets dispatched before the time instant when a packet was cancelled will not benefit from the cancellation of this packet. These packets will still benefit from packets that were already cancelled from waveform buffer prior to their dispatch time. Note that if a packet fails to decode and there is still time to respond with an ACK/NACK back to the UE, the TIC scheduler may chose to re-dispatch this packet to the pipeline. This technique can be deemed as a form of iterative group IC where a group corresponds to a set of users to be processed in parallel.

NodeB Receiver Architecture Modeling

In this section we describe a model for the assumed architecture of the NodeB receiver capable of uplink IC. It should be stressed here that the assumed architecture is an example used to demonstrate the method of uplink IC complexity analysis and does not represent a unique design choice.

Figure 4 illustrates a high level block diagram of the NodeB receiver architecture. The NodeB receiver handles reception of users across three cells. A single processor architecture may not satisfy the processing requirements. Scalability for future system enhancement is also an important factor to consider. In order to meet these design challenges, as shown in the figure, the advanced NodeB receiver adopts a highly scalable symmetric multiprocessing (SMP) architecture.

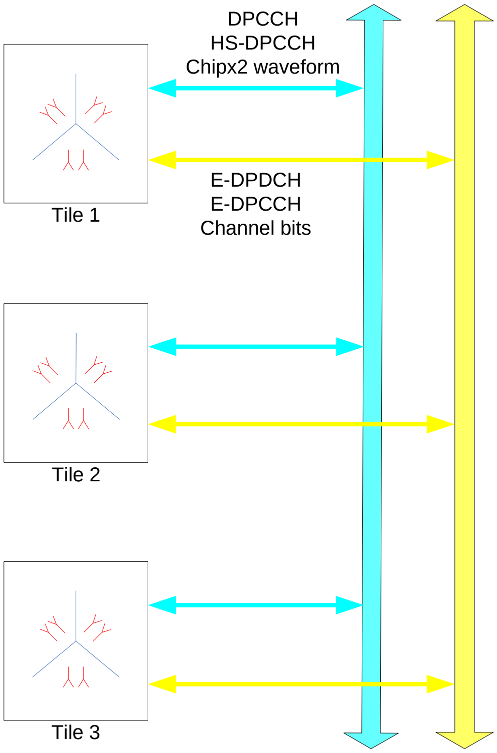

Figure 4: Three-tile Symmetric Multiprocessing Architecture of Advanced NodeB Receiver

Some key points of the architecture are summarized here. The receiver consists of three identical tiles of processors. The users in the three cells are distributed evenly to the three tiles to balance the processing. Each tile has its own processing engines. The three tiles operate on separate waveform buffers to facilitate parallel processing. To share interference cancellation results among the tiles, once a user is cancelled in one tile, it should be subtracted from other tiles. To save the bandwidth between tiles, E-DPDCH and E-DPCCH re-encoded channel bits are communicated among tiles. The waveform for each traffic channel is reconstructed locally on each tile.

The requirement for processing power within each tile and the communication bandwidth among the three tiles must be characterized for the purpose of the analysis. The assumptions used to establish the processing complexity metrics are listed in Table 1. We will define the processing complexity in the TDD and TIC sub-systems in later sections.

Table 1: Architecture Model Assumptions

Parameter	Value	Comments
N_{tile}	3	Number of Tiles
N_{ant}	2	Number of Antennas per cell
Bit-width of modified antenna sample buffer (bits)	16	8 bits for I, 8 bits for Q
Modified antenna sample buffer length (Mbytes)	~5.84	2ms TTI: 4 transmissions with 8 HARQ processes
N_{TDD}	1	Number of TDDs per tile
$N_{c,TDD}$	16	Chips per clock cycle of TDD
$N_{Decoder}$	2	Number of Decoders per tile
$N_{b,DEC}$	0.25	Information bits per clock cycle of Decoder
N_{Re}	1	Number of Re-encoders per tile
$N_{b,Re}$	1	bits per clock cycle of Re-encoder
N_{TIC}	1	Number of TICs per tile, Include DBCE, TIF, and TIS
$N_{c,TIC}$	16	Chips per clock cycle of DBCE, TIC, and TIS
a_{ov}	5%	Memory access overhead

The model for one tile of the NodeB receiver is illustrated in Figure 5, where each unit in the IC processing chain is modeled as a queue-server. The processing chain consists of: TIC Scheduler, TDD, Decoder, Re-encoder, and TIC. To enforce the delay constraint, several checkpoints are inserted in the processing chain at the TIC scheduler, and Decoder. At these checkpoints, the packet arrival time is checked to see if there is more time left for further processing before the NodeB must send the ACK/NACK to UE (referred to as ACK timeline). The IC iteration number is also checked to ensure at most two IC iterations are performed on each packet per ACK timeline.

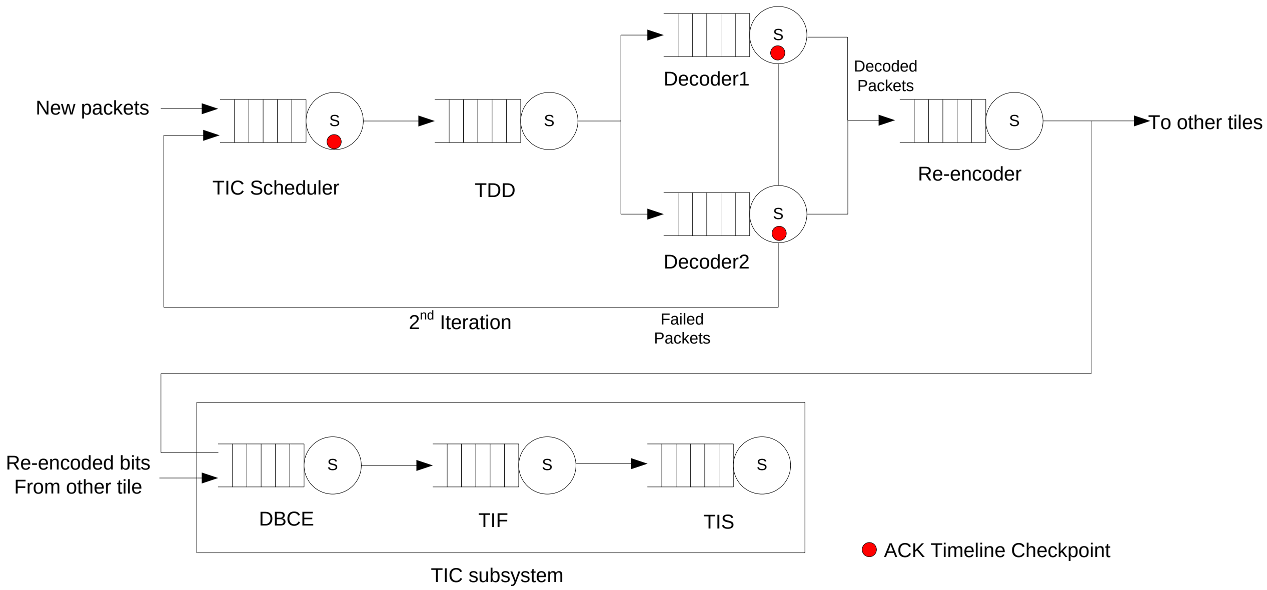

Figure 5: Discrete Event Model of IC Processing Chain (One Tile)

HSPA Uplink System Model

The system model parameters used to evaluate the complexity of uplink traffic IC are listed in Table 2. In particular, the complexity is estimated assuming a VoIP scenario in HSPA operating at a capacity of 192 users per cell.

Table 2: System Model Parameters

Parameter	Value	Comments
Application	VoIP	
Users per Cell	192	These users transmit on E-DCH channel based on voice source model
Cells per NodeB	3	
Receive Antennas per Cell	2	
TTI [ms]	2	
Max H-ARQ Transmissions	4	
Transport Block Sizes [bits]	307 (Full Rate), 120 (SID)	Voice source modeling: 2-state Markov model [1]; average talk spurt = 2seconds; voice activity factor = 0.5.
Source Time Interval [ms]	20 (active user), 160 (SID)	A Full Rate AMR packet is transmitted once every 20ms. A SID packet is transmitted once every 160ms.
Channel Types	3GPP channel mix	Number of fingers: Ped A3 = 1; Ped B3 = 5; Veh A30 = 4; Veh A120 = 4.
Tau_DPCH [n*66.67us]	0 to 149	Tau_DPCH is uniformly distributed in the range for all users.

VoIP Source Modeling

The VoIP source model can be described via a simple 2-state (1^{st} order Markov) voice activity model [4] as shown in **Error! Reference source not found.**. State 1 is referred to as the Active State and State 2 is referred to as the Inactive State.

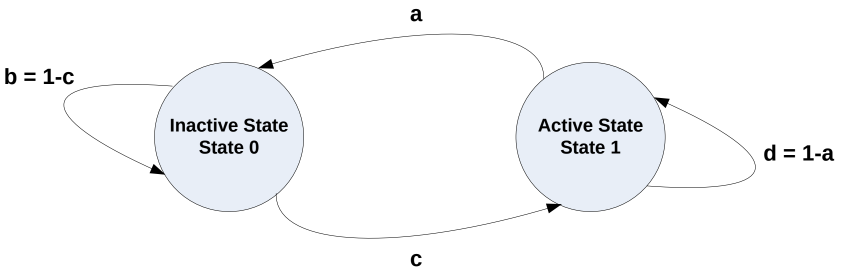

Figure 6: Two-state voice activity model

In the model:

- Pr {state 1 to state 0} = 0.01.

- Pr {state 0 to state 1} = 0.01.

The model is assumed to be updated at the voice frame rate $R = 1/T$, where T is the voice frame duration (20ms typically). Thus the Voice Activity Factor (VAF) is given by

$$\lambda = \frac{c}{a+c} = 0.5.$$

The mean talk spurt duration μ_{TS} and silence period μ_{SP} in voice frames are given by

$$\mu_{TS} = \frac{1}{a},\ \mu_{SP} = \frac{1}{c}.$$

The discrete event simulation model is illustrated in Figure 7. VoIP users are controlled by their state-machine to generate Full Rate or SID packets in each voice frame. The packet is associated by various attributes used in the simulation model, such as channel type, number of paths, decoding probabilities, TTI number, slot number and so on. After passing through a channel, whose behavior has been abstracted via the H-ARQ decoding probabilities, the packets are uniformly distributed to three tiles.

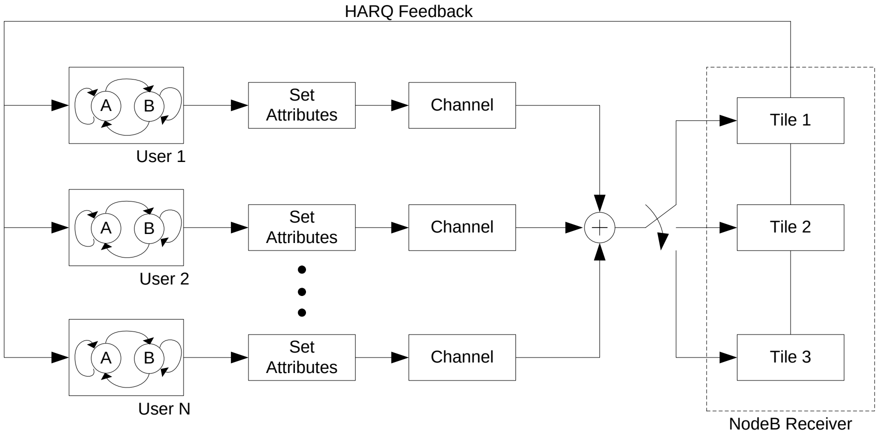

Figure 7: Discrete Event model of VoIP traffic

H-ARQ Fundamental Probabilities

In the following, the fundamental H-ARQ probabilities used in this study are listed. These probabilities are determined from link simulations. In particular, there are two sets of probabilities computed for the purpose of uplink IC complexity analysis:

- Pr{Transport Block error/user is on H-ARQ attempt n}

- Pr{User is on H-ARQ attempt n}

The first set of probabilities is listed in Table 3 and 4, and the second set of probabilities is listed in Table 5 and 6. The first set of probabilities is useful in assessing the complexity of the processing requirement of the re-encoder, DBCE, TIF and TIS sub-systems. This is because, these sub-systems are triggered only when a packet (or transport block) decodes correctly. On the other hand, the second set of probabilities influences the processing requirement of the TDD and channel decoder. This is because TDD consumes more resources and applies more delay for packet on the later H-ARQ attempt. For these packets, the channel decoder has to deal with more stringent delay requirement.

Table 3: BLER conditioned on H-ARQ attempt numbers, TBS = 120 bits

Probability	PA3	PB3	VA30	VA120	Description
BLER1	0.93	0.83	0.75	0.73	Pr [Block error / user is on H-ARQ attempt 1]
BLER2	0.52	0.41	0.42	0.38	Pr [Block error / user is on H-ARQ attempt 2]
BLER3	0.21	0.21	0.23	0.22	Pr [Block error / user is on H-ARQ attempt 3]
BLER4	0.10	0.14	0.14	0.17	Pr [Block error / user is on H-ARQ attempt 4]

Table 4: BLER conditioned on H-ARQ attempt numbers, TBS = 307 bits

Probability	PA3	PB3	VA30	VA120	Description
BLER1	0.98	0.96	0.87	0.84	Pr [Block error / user is on H-ARQ attempt 1]
BLER2	0.68	0.55	0.50	0.44	Pr [Block error / user is on H-ARQ attempt 2]
BLER3	0.22	0.21	0.22	0.22	Pr [Block error / user is on H-ARQ attempt 3]
BLER4	0.07	0.09	0.10	0.12	Pr [Block error / user is on H-ARQ attempt 4]

Table 5: H-ARQ Attempt Probabilities, TBS = 120 bits

Probability	PA3	PB3	VA30	VA120	Description
P_1	0.0125	0.0125	0.0125	0.0125	Pr [user is on H-ARQ attempt 1]
P_2	0.0116	0.0104	0.0094	0.0091	Pr [user is on H-ARQ attempt 2]
P_3	0.006	0.0043	0.0039	0.0034	Pr [user is on H-ARQ attempt 3]
P_4	0.0013	0.0009	0.0009	0.0008	Pr [user is on H-ARQ attempt 4]
P_5	0.9686	0.9719	0.9733	0.9743	Pr [user is inactive]

Table 6: H-ARQ Attempt Probabilities, TBS = 307 bits

Probability	PA3	PB3	VA30	VA120	Description
P_1	0.1	0.1	0.1	0.1	Pr [user is on H-ARQ attempt 1]
P_2	0.0987	0.0963	0.0869	0.0838	Pr [user is on H-ARQ attempt 2]
P_3	0.0674	0.0532	0.0439	0.0369	Pr [user is on H-ARQ attempt 3]
P_4	0.015	0.011	0.0098	0.0081	Pr [user is on H-ARQ attempt 4]
P_5	0.7189	0.7395	0.7593	0.7711	Pr [user is inactive]

Simulation Results

Baseline Design Requirements

In the following, some critical statistics about the user traffic through the discrete event simulation model are derived. In the simulation, we forcibly run the processing at an extremely high clock rate so that the processing delay and backlog are negligible. The required processing power is solely determined by the number of arrival packets in one TTI and the channel type associated with the packets. For convenience, we measure the volume of data to be processed by TDD in unit of TTIs. Samples of 7680 chips (one TTI) processed by a RAKE finger is one TTI. Thus we calculate the processing complexity as follows. The number of TTIs processed by TDD in one TTI is

$$N_{TTI,TDD} = \sum_{i=1}^{N_{pkt,TDD}} N_{harq}(i) \cdot N_{path}(i) \cdot N_{ant} \cdot I_{CH}(i) \cdot I_{SHO}(i) \,,$$

where $N_{pkt,TDD}$ is the number of packets arriving at TDD in the given TTI, $N_{harq}(i)$ is the H-ARQ attempt number for the i-th packet, $N_{path}(i)$ is number of paths of the channel, $I_{CH}(i)$ is the number of channelization codes, and the softer handover indicator $I_{SHO}(i)$ is given by

$$I_{SHO}(i) = \begin{cases} 2 & \text{if user i in SHO} \\ 1 & \text{otherwise} \end{cases}$$

The processing complexity of decoder is calculated as follows. The number of information bits decoded by the channel decoder in one TTI is

$$N_B = \sum_{i=1}^{N_{pkt,DEC}} TBS(i)$$

where $N_{pkt,DEC}$ is the number of packets arrived at Decoder in the TTI , $TBS(i)$ is the transport block size of the i-th packet.

The processing complexity of TIC is very similar to TDD. The number of TTIs processed by TIC in one TTI is

$$N_{TTI,TIC} = \sum_{i=1}^{N_{pkt,DEC}} N_{harq}(i) \cdot N_{path}(i) \cdot N_{ant} \,,$$

where $N_{pkt,TIC}$ is the number of packets arrived at TIC in the TTI. Since $N_{pkt,TDD}$, $N_{pkt,DEC}$, $N_{pkt,TIC}$, $N_{harq}(i)$, $N_{path}(i)$, and $TBS(i)$ are random variables, the number of TTIs to be processed in one TTI and the number of information bits decoded in one TTI are both random variables. In Figures 8-10, we plot the CDFs of number of TTIs or information bits processed by TDD, Decoder, and TIC units in one TTI.

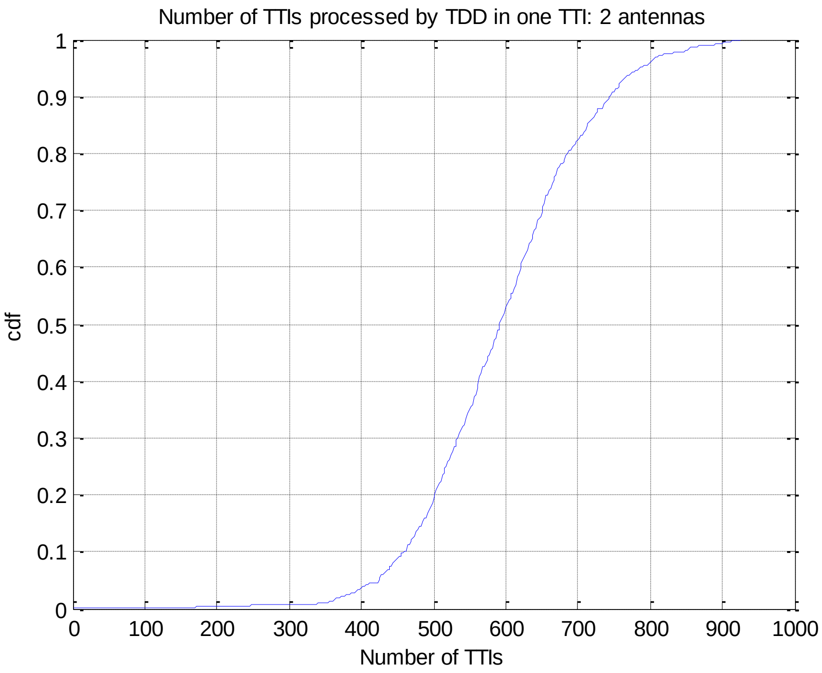

Figure 8: CDF of number of TTIs processed by TDD in one TTI

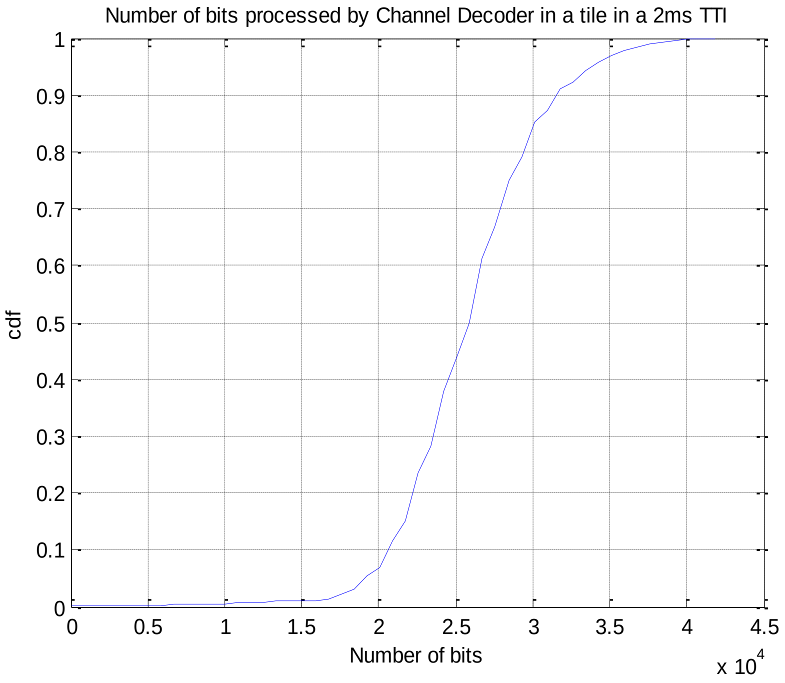

Figure 9: CDF of number of bits processed by Decoder in one TTI

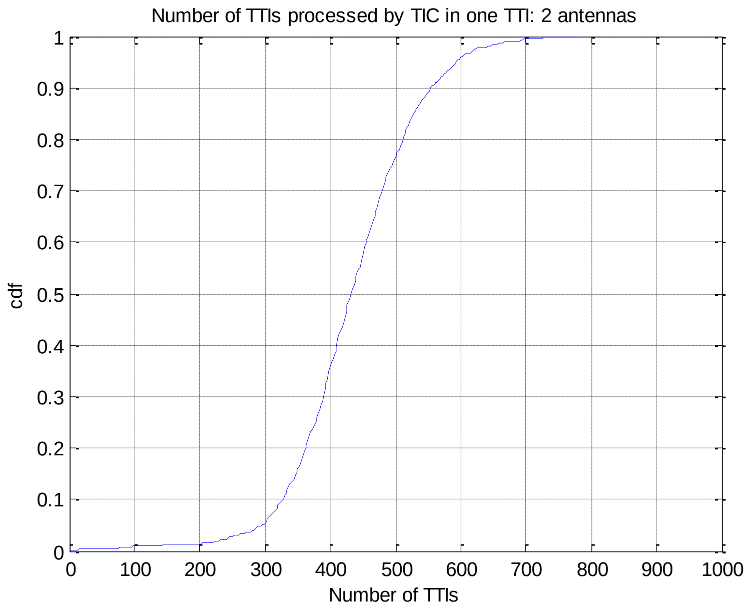

Figure 10: CDF of number of TTIs processed by TIC in one TTI

The baseline clock rate of TDD, Decoder, and TIC are derived to meet the processing requirements.

$$f_{TDD} = \frac{7680 \cdot N_{TTI,TDD} \cdot (1 + a_{ov})}{N_{C,TDD} \cdot TTI}$$

$$f_{DEC} = \frac{N_B \cdot (1 + a_{ov})}{N_{b,DEC} \cdot TTI}$$

$$f_{TIC} = \frac{7680 \cdot N_{TTI,TIC} \cdot (1 + a_{ov})}{N_{C,TIC} \cdot TTI}$$

Intuitively, if the processing unit can finish processing a packet in the TTI it is received, then there is very minimal impact on the total IC delay. Thus for the baseline design requirement, we use this criterion as a design measure. Based on the CDF, we can calculate the required clock rates to achieve this aim with probability 0.99. The statistics and clock rates are shown in Table 7. These clock rates are used as baseline design requirement.

Table 7: Number of TTIs processed in one TTI (99 percentile) and Clock Rate

Processing Unit	99% of CDF	Clock Rate
TDD	872 TTIs	220MHz
Decoder	3.76e4 bits	78.9MHz
TIC	680 TTIs	170MHz

Design Optimization

The baseline clock rates are designed for the worst case of arrival distribution. The delay constraint is stringent such that the receiver must finish processing of each packet in the TTI when it receives the packet. Actually, this is an over-design since we can distribute the packet processing to consecutive TTIs if allowed by the ACK and IC turnaround time. This motivates us to optimize the clock rates under certain constraints.

The TDD and Decoder are critical parts in the delay line. We define the ACK headroom time as

$$T_{headroom} = T_{ACK} - T_{DEC}$$

where T_{ACK} is the time when NodeB must send ACK/NACK information to UE, and T_{DEC} is time when decoding finishes. The headroom time is the margin left to the NodeB to send the ACK/NACK. When the headroom is negative, then the packet misses its ACK timeline. The optimization over clock rate can be written as

$$\min(\text{clock rates})\ \text{s.t.}\ \Pr(T_{headroom} > H) = P_{headroom}$$

where H is the headroom target and $P_{headroom}$ is the probability that headroom is greater than the design target. Given H, $P_{headroom}$ can be derived easily from CCDF of $T_{headroom}$.

The minimization involves a two dimensional search over the TDD and channel decoder's clock rates. To simplify the procedure, a sequential strategy is adopted. When optimizing the TDD, the decoder is running at extremely high clock so that no backlog happens at the decoder. Then we fix the TDD clock rate and optimize the decoder. The CCDF of $T_{headroom}$ for TDD clock rate optimization is plotted in Figure 11, where we vary the TDD clock rate from 130MHz to 180MHz. The utilization of TDD, defined as the proportion of the time the TDD sub-system is busy, is plotted in Figure 12. We can observe the following:

At TDD clock rate = 160MHz, Pr [$T_{headroom}$ > 0] = 0.99

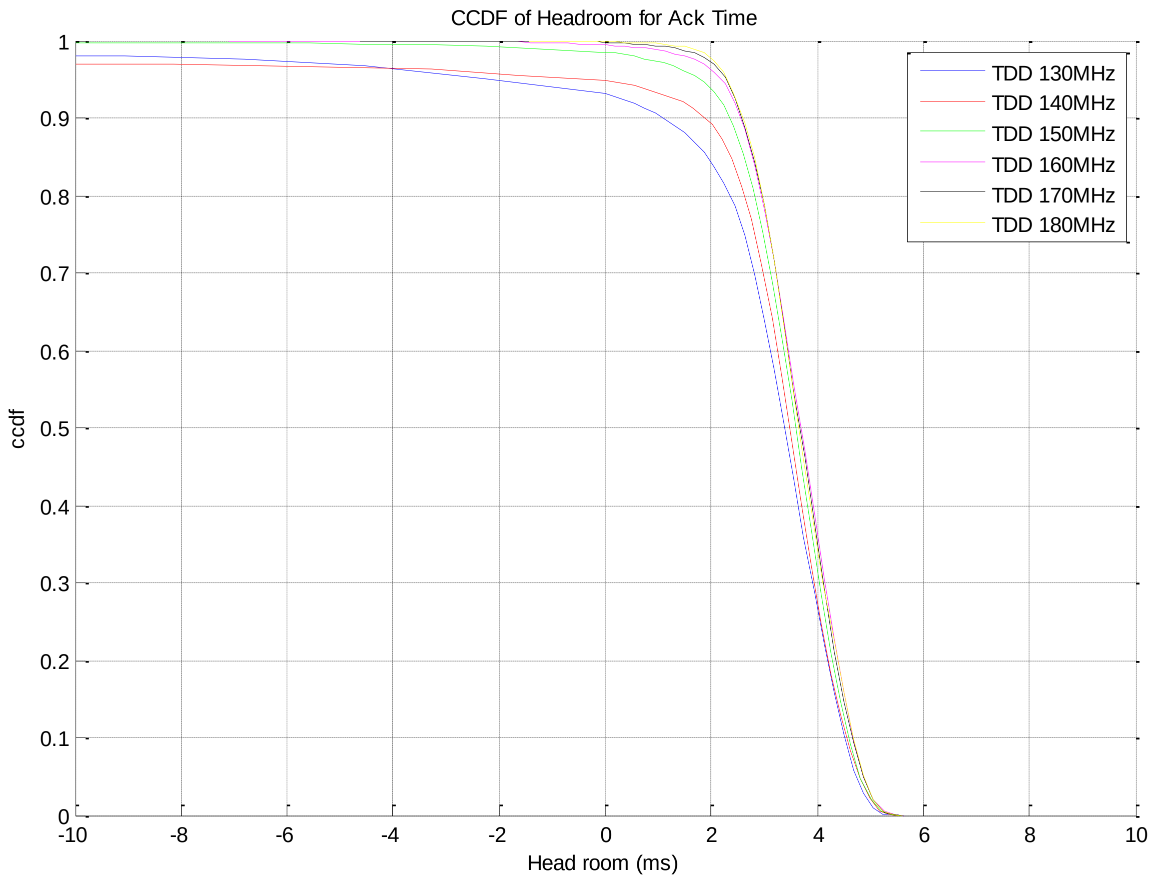

Figure 11: CCDF of $T_{headroom}$ for TDD clock rate optimization

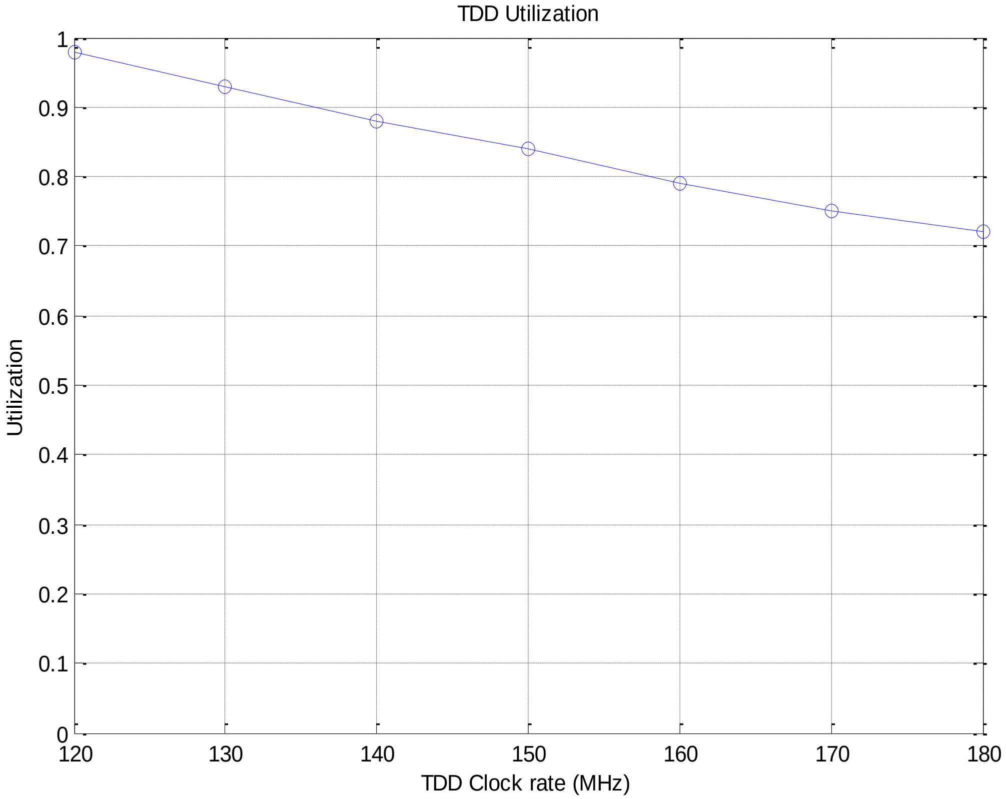

Figure 12: Utilization of TDD at different clock rates

The CCDF of $T_{headroom}$ for decoder clock rate optimization is plotted in Figure 13, where we vary the Decoder clock rate from 30MHz to 70MHz but fix the TDD clock rate to 160MHz. The utilization of the decoder is plotted in Figure 14. We can observe that:

At Decoder clock rate = 70MHz, Pr [$T_{headroom} > 0$] = 0.99

Compared with the baseline design in Table 7, optimization can significantly reduce the clock rate of TDD and Decoder.

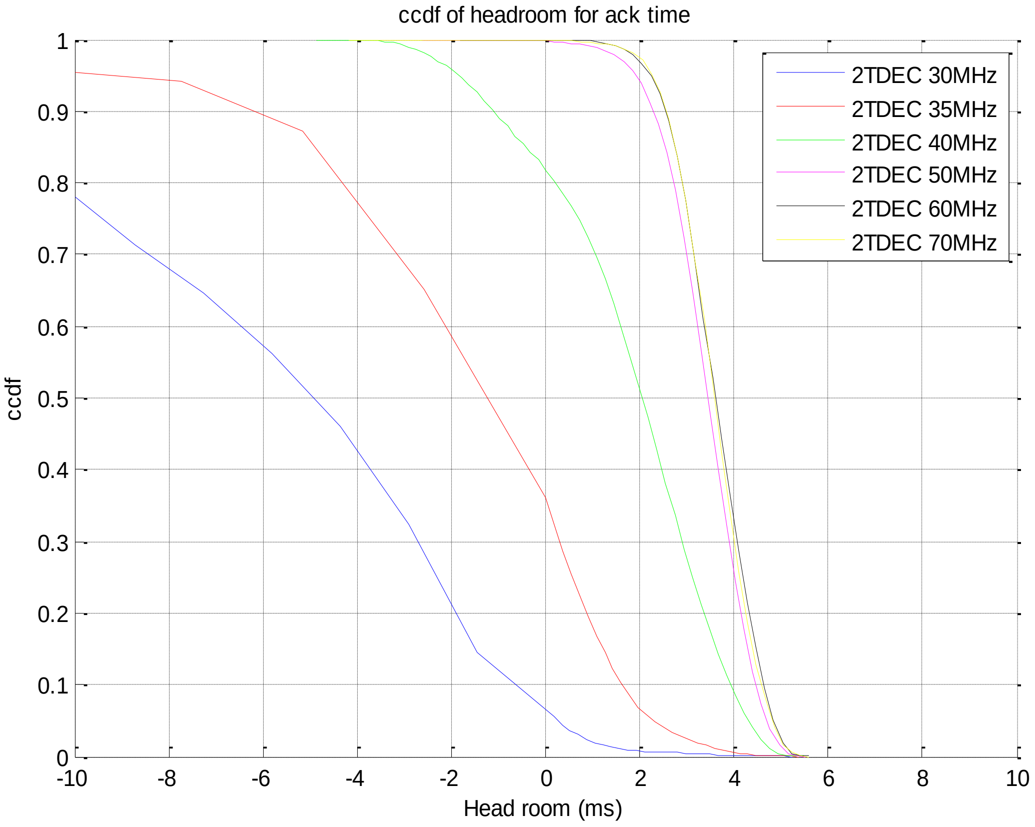

Figure 13: CCDF of $T_{headroom}$ for Decoder clock rate optimization

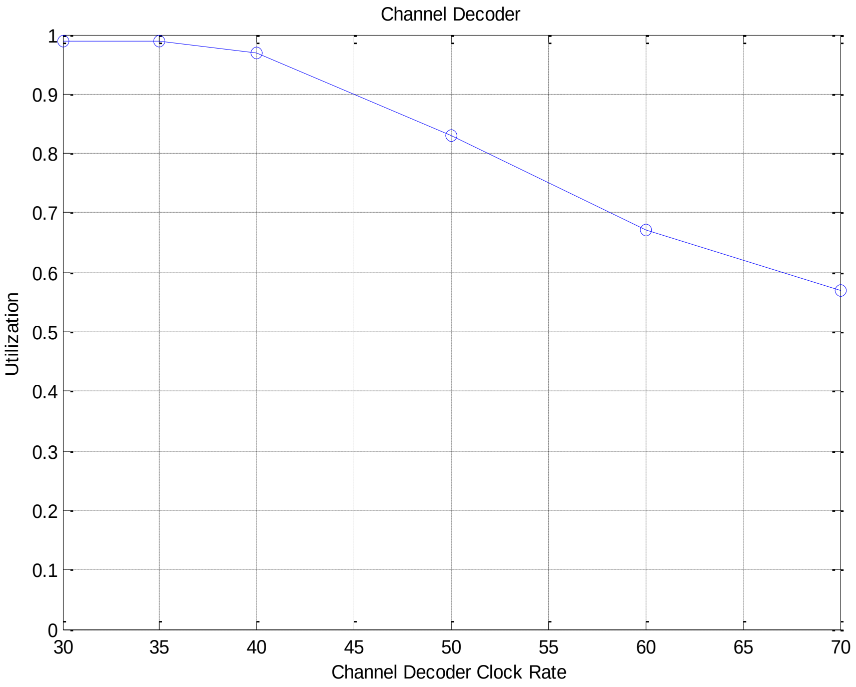

Figure 14: Utilization of Decoder at different clock rates

Conclusion

In this paper we have described a methodology to systematically evaluate the complexity of a NodeB receiver with interference cancellation for the uplink of WCDMA/HSPA. The procedure is illustrated by an example of a fully loaded VoIP application. A discrete event simulation model is developed to model the processing chain of interference cancellation system. Via this model, various statistics of the system can be studied by simulations. We can further optimize the system clock rate to reduce the implementation complexity and meet the processing headroom simultaneously. It is shown that the optimized clock rate is significantly lower than the baseline design for the worst case situation. Through the study, the interference cancellation is demonstrated to be fully feasible for the current VLSI technology.

Acknowledgements

References

[1] S. Sambhwani, W. Zhang, W. Zeng, "Uplink Interference Cancellation in HSPA: Principles and Practice", QUALCOMM Inc.
[2] H. Holma and A. Toskala, "WCDMA for UMTS: HSPA Evolution and LTE", 4th Edition, John Weily & Sons, 2007
[3] A. J. Viterbi, "Very Low Rate Convolutional Codes for Maximum Theoretical Performance of Spread-Spectrum Multiple-Access Channels", IEEE JSAC, vol. 8, no. 4, May 1990
[4] NGMN White Paper, "Next Generation Mobile Networks Radio Access Performance Evaluation Methodology", V1.3, June 2007.