ISTC 2025 Implementation Competition

Competition Committee: Henry Pfister, Chair (Duke), Gianluigi Liva (DLR) , Sebastian Cammerer (NVIDIA)

There are a variety of encoding/decoding solutions for short blocklength (e.g. K <= 512 message bits) including (among others) polar codes with CRC-aided successive cancellation list decoding, BCH codes with ordered statistics decoding, polarization adjusted convolutional codes with a variety of decoding approaches, and tail-biting convolutional codes with expurgating linear functions (CRCs) decoded with serial or parallel list decoding. ISTC 2025 will feature a competition to explore the frame error rate, average latency, and maximum latency performance of actual C++ implementations of short blocklength encoders and decoders running on a single CPU.

The goal of this contest is to provide a hands-on event that encourages interest in coding and provides a starting point for technical discussions. Designing codes and decoders gives rise to a complicated multi-objective optimization problem. To make fair comparisons, we need to fix some design decisions that inevitably favor some approaches over others. For example, the implementations in this contest are run on a CPU and may not use multiple cores or multiple threads. We understand that this constraint is somewhat limiting but this is the first competition being hosted by ISTC and we are trying to keep it simple and have the focus on the conceptual differences between different coding schemes. It is likely that future competitions will address other architectures.

To enter the implementation competition, contestants must register their intention to compete on the conference website before the paper submission deadline. Submissions may have an accompanying paper submission but this is not required. The deadline for initial code submission is April 30th, 2025. Code should be submitted by emailing a zip file (specifications discussed below) to istc2025comp@gmail.com. As per standard review policies, submitted papers and source code are treated as confidential information and not released publicly. If a paper is submitted and accepted, it will be published like any other conference paper. If the code executes properly, its performance results will be reported publicly. The source code itself will not be released by conference but we encourage authors to release their source code publicly so that others can replicate their amazing results.

There are a variety of categories for the competition. To submit an entry to the competition you should plan to submit C++ source code for an encoder/decoder pair for at least one column (all three rates for a given K) or one row (all four values of K for a given rate) of the configurations listed below. We encourage submission of encoder/decoder pairs that operate for all twelve configurations as would be required in a standard. For each rate, we will simulate your code to verify FER performance and latency with particular attention to performance near the FER operating points of 10^-3 and 10^-5. The calling structure also allows submissions to be tuned independently for average run-time versus worst-case run-time. During encoder/decoder setup,

a flag will be passed that specifies whether average latency or maximum latency will be tested for that run.

Rate	$K = 64$ message bits	$K = 128$	$K = 256$	$K = 512$
$R = 1/4$	$(N = 256, K = 64)$	(512,128)	(1024,256)	(2048,512)
$R = 1/2$	(128,64)	(256,128)	(512,256)	(1024,512)
$R = 4/5$	(80,64)	(160,128)	(320,256)	(640,512)

The FER and latency performance of all submissions will be presented in a talk at the conference and a team of judges will determine winning encoder/decoder pairs for each row and each column as well as an overall winner.

Detailed submission guidelines:

(1) The GitHub repository: https://github.com/pfistergroup/istc25_contest.git demonstrates how your code will be compiled and executed.
(2) We plan to run the code on bare metal servers using the Intel E-2456 processor but this choice is not
(3) You need to add your code to the source directory (replacing enc_dec.cpp) and update the Makefile to compile your code. Note: the compiler flags should not be changed.
(4) Codes will be simulated on the binary-input AWGN channel. The testing framework allows one to pass integer representations of the received log-likelihood ratios (LLRs) to the decoder by remapping with an arbitrary function each floating point LLR. Moreover, this conversion is not included in the timing.
(5) You will need to update the file “test_params.txt” that indicates what EsN0 is required for your submission to achieve 10^{-3} and 10^{-5} frame error rate for each rate / length / latency triple.
(6) Verification will be done by running 10^5 and 10^7 blocks respectively and checking that not more than 130 errors (i.e., 3 standard deviations above the mean) are observed. All decoders will be tested using the same noise vectors. Execution will be on a Intel 64-bit processor such as the E-2456.
(7) Data memory usage shall be limited to 1MB (exact method to be determined) and very slow decoders will be disqualified if we cannot execute the above test within a reasonable amount of time.
(8) For submission, create a zip file containing your Makefile, your enc_dec.cpp, your test_params.txt, and any other course files required for your compilation.
(9) Each submission will be assigned a random identifier and performance statistics of all submissions will be released anonymously using this identifier.

Additional questions regarding submission should be directed to istc2025comp@gmail.com