Executive Summary

T@MPO

Turbo Codec @ Minimum Power

Doc. no.: P50313-IM-DL-017

ESTEC Contract: CCN1 to 13781/99/NL/FM

B. Bougard, V.Derudder, L. Hollevoet, L. Van der Perre, J.-W. Weijers

-Imec, Kapeldreef 75, B-3001 Leuven, Belgium-

TABLE OF CONTENTS

1. Introduction and objectives

The explosive demand for wireless communication puts high requirements on technological developments. The spectrum scarceness and high market value require exploiting it as efficiently as possible. On the other hand, the trend to have any wireless system from cellular to WLAN and satellite receivers integrated in small, high autonomy portable devices, makes energy consumption another strategic stake. The optimum trade-off between spectrum and energy, the so-called Shannon limit, has been characterized in the late ’40s [1]. However, decades of innovation in communication theory, signal processing and VLSI were needed to approach this limit, using advanced Forward Error Correction (FEC) techniques.
In this context, the still recent introduction of turbo-codes appears as a breakthrough [2]: Turbo-codes outperform any previously known FEC scheme by at least 3 dB, consequently doubling the battery lifetime or saving 20% of the required spectrum. Those properties make turbo-codes attractive for most of current wireless applications.
The spreading out of Turbo coding has been spectacular in publications and theoretical developments. By contrast, their hardware implementation is evolving far more slowly. Speed, latency, and most of all energy consumption break the show when bringing the turbo coding principles into practice.
This project aimed to bridge this gap, *providing solutions to implement high-performance turbo-codes with throughput, latency and energy consumption that make them strong competitors with traditional FEC techniques.*

2. Performance analysis and algorithmic selection

Prior to any architectural and implementation-oriented analysis, it is crucial to compare and assess the different turbo-codes and decoding algorithms available, in term of coding performance and complexity.

The so-called turbo-codes always correspond to the concatenation of two simple codes (linear convolution or block codes) separated by an interleaving process. Two main classes are distinguished depending on the parallel or serial nature of the concatenation, respectively the Parallel Concatenated Codes and the Serially Concatenated Codes. Parallel-concatenated codes are most often made of Convolutional constituent codes, leading to the Parallel Concatenated Convolutional Codes (PCCC) class as introduce by Berrou in [Berrou93], and schematically depicted in Figure 1.

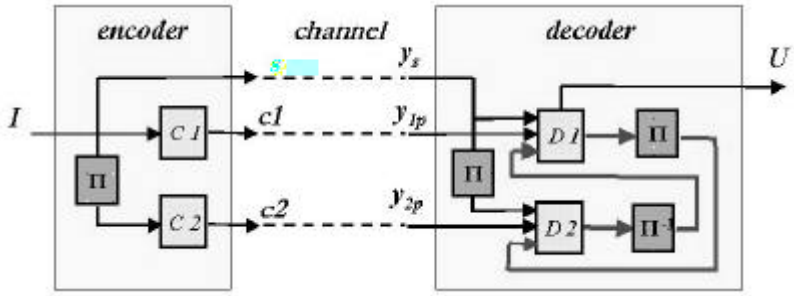
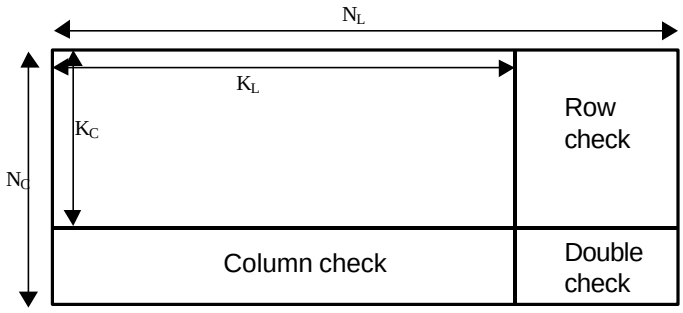


Figure 1: Parallel Concatenated Convolutional Code (PCCC)

They are often simply called *turbo-codes*. Serially concatenated codes can be made of Convolution constituent codes (SCCC) or block constituent codes (SCBC or *product code*). The latter scheme, as shown in Figure 2, organizes the data in a matrix and performs block coding (and decoding) on both rows and columns of this matrix.

Figure 2: Serially Concatenated Block Codes (SCBC)



PCCC's have traditionally better performance than SCCC's and SCBC's at low SNR but sometime present an error floor at higher SNR, where they are outperformed by SCCC's and SCBC's. However, recent progress in the construction of the interleaver, which is at the origin of the error floor, has lead to reduce this impairment significantly. Good PCCC's do not show any error floor down to BER 10^{-9}. SCCC's and SCBC's do not present significant difference regarding the performance, however, SCBC can be decoded more easily (due to an higher regularity). Therefore, we focused our attention in this project exclusively to PCCC and SCBC codes. Next to the coding scheme itself, the decoding algorithms have a decisive impact on the performances. All turbo-codes are decoded in an iterative way, by successive *soft-input soft-output* (*SISO)* decoding and (de-)interleaving operations. During the iterations, a so-called *extrinsic information* is refined, hopefully converging to the maximum likelihood symbols probabilities. Choosing the SISO decoding algorithm is a key issue since they affect as well the performance and the complexity. For the PCCC's, two SISO algorithms can be used: the

Soft-output Viterbi Algorithm (SOVA) or the *Maximum A Posteriori Algorihm* (MAP, also referred as BCJR algorithm). The latter outperforms significantly the former, this with an acceptable complexity penalty when suitably optimized. We have considered the MAP algorithm to decode PCCC codes, more specifically its *max-log-max* and *max*-log-max* variants, which have been traded-off. For the SCBC's, focus was on the *Augmented List Decoding-Fang-Battail-Buda* (ALD-FBBA) SISO decoding algorithm, which outperforms drastically the other available solutions, as well in terms of asymptotic performance as in terms of convergence properties (and consequently, in energy consumption).

The criteria considered to select the coding scheme to be implemented are the following:

- The coding gain in the useful region regarding the targeted application
- The flexibility (in term of achievable block sizes, code rates and scalability)
- The potentiality to be implemented with very low decoding latency, as required by reactive multimedia applications, and above all, with low energy consumption.

Considering those criteria, the performances in term of bit error rate (BER) and packet error rate (PER) have been assessed. An extensive comparative analysis of the performance of PCCC and SCBC has been carried out, as reported in [4]. To reduce the latency, which includes the time to receive one block at the considered data rate, small block sizes have been considered. We have shown that 3 dB extra gain is still achievable with turbo-codes with block sizes from 200 to 400. Smaller block sizes (down to 32) still present performance comparable with conventional convolutional codes. Different modulations (BPSK, QPSK, 16QAM and 64QAM) and code rate (1/3, ½, 2/3, 3/4) have been studied. A specific slightly sub-optimal soft-demapper has been derived for 16QAM and 64QAM. An AWGN channel has been assumed.

The results of the study show that, when using (*code-) optimized* interleavers, PCCC's outperform SCBC's by 0.2 to 1.5 dB depending on the block sizes and the modulation and provided that the code rate stays below 2/3. For code rate 2/3 and ¾, SCBC's are slightly better. For code rate 7/8, SCBC's show a superior performance. This can be explained by the fact that in the case of SCBC's, higher code rates are achieved by code selection instead of puncturing.
Besides the performance aspects, it was important to have an early estimate on the impact of selecting one scheme or the other on the implementation objectives. It was not the aim to have a fully detailed complexity assessment for both schemes, but rather to have sufficient information to make a sound scheme selection. Since our main implementation-related objective was energy efficiency, the complexity of the algorithms under study has been evaluated in terms of energy consumption. Both turbo-coding schemes under consideration (PCCC and SCBC) are notably data intensive. This is due to the fact that both of them are manipulating scalar *metrics* where bits are sufficient in conventional coding. Due to this data-intensive character, it can be assumed that most of the decoding

energy is dissipated by the data accesses in the memory architecture. Therefore, a good assessment of the energy consumption is reached by profiling the memory accesses. This can be done using profiling tools [9] handling high-level description of the decoder (C-code). Furthermore, an important part of the energy reduction measures such as the parallelisation and the introduction of memory hierarchy (see section 3) can be modeled at C-level and their impact can already be considered here.
We have profiled a C-level memory accesses-optimized versions of log-max(*)-MAP based PCCC and ALD-FBBA-based SCBC's decoders using the Atomium toolset [9]. Profiling data associated with memory-access energy models lead to decoding energy estimates that suffice to proceed to the scheme selection.

Considering performance and energy-consumption assessments, we have noticed the superiority of PCCC's in term of performance, energy consumption and flexibility. The highest flexibility of PCCC's is due to the not-required code-adaptation to achieve different block sizes and code rates. The block size depends only on the interleaver while virtually any code rate can be achieved by puncturing. Regarding latency, the studied schemes do not differ significantly. Moreover both can be parallelized.

In conclusion, we have decided to consider PCCC's in the remainder of the project. A state-of-the-art PCCC code has been selected [3GPP] and a corresponding turbo-CODEC has been functionally specified in WP130. Table 1 summarizes the main characteristics of the selected coding schemes, and its flexibility.

Code characteristics	
Inner Codes	Non-terminated RSC $g_0 = 1 + D^2 + D^3$ $g_1 = 1 + D + D^3$
Interleaver sizes	32, 48, 64, 72, 96, 128, 144, 192, 256, 288, 384, 432
Code rates	1/3, 1/2, 2/3, 3/4, 7/8
Fully programmable puncturing pattern	

Table 1: Characteristics and parameter options of selected coding scheme

3. Architectural optimization

Enabling high throughput, low latency and low power turbo-decoding requires a thorough architectural optimization. The turbo decoding algorithms suffer from major bottlenecks that hamper seriously the throughput and power consumption of their current implementations. In the case of PCCC, the major bottlenecks regarding latency and throughput as well as power consumption are:

the iterative aspect inherent to the turbo-decoding method;
the double-recursion of the MAP SISO decoding algorithm introducing a significant latency and requiring extensive storage of the state metrics;
the interleaving process that requires the knowledge of the fully decoded frame at a given iteration before starting the following and requiring extensive storage of the extrinsic values.

In order to cope with our objectives of high throughput, low latency and low energy, the standard MAP algorithm has been systematically optimized applying IMEC's Data Transfer and Storage Exploration (DTSE) methodology [5], leading to a parallel, energy conscious hardware architecture.

After selecting the coding scheme, a fixed-point model of the corresponding MAP decoder has been developed. Word-width for the critical data (soft-input, state- and branch-metrics) have been chosen in order to reduce the loss between the floating- and the fixed-point model to 0.2 dB. After this refinement, it has been noted that the max* correction leads to a marginal performance advantage while increasing significantly the critical path, introducing an extra look-up table access in the critical state-metrics calculation loop of the MAP. Therefore, the max-log-MAP algorithm has finally been selected for implementation.
The next step into the way to an optimized implementation is the parallelization of the SISO decoder. Parallelization leads intuitively to latency reduction and throughput improvement,. However, it also effects an energy reduction since more data can be processed into the same clock cycle, requiring a lower clock rate to achieve a given throughput and hence, a lower supply voltage and power.

The traditional MAP algorithm consist mainly of two recursions (forward- and backward-recursions) applied on the block to be decoded and followed by a soft-input calculation step requiring the knowledge of both forward- and backward-recursions results. Those dependencies lead generally to excessive latency and storage requirements. Therefore, a so-called Overlapping Sliding-Window (OSW) variant of the MAP is usually considered. The forward- and backward-recursions are applied to a sub-block (windows) and neighboring information is provided by *training sequences* on the neighboring windows. The OSW-MAP has a reduced intrinsic latency and storage requirement. However, when considered into a turbo-scheme, this benefit is lost since the complete SISO decoding and the complete extrinsic information have to be stored before starting a new iteration. To tackle those drawbacks, we have considered operating the windows in a parallel way. In the final MAP architecture, two adjacent windows are combined in a structure called *worker* that operate the MAP decoding on sub-block in parallel. To maintain the workers as independent as possible, the OSW training sequences have been replaced by a neighboring condition initialization between the iterations (next iteration initialization, NII). Hence, the data exchange between the workers can be done at low rate (i.e. at the iteration rate). This property has been exploited to enable energy scalability by clocking

independently the workers. Depending on the considered block size, more or less workers can be activated. Non-activated workers are not clocked, thus they do not toggle nor consume energy (except leakage).
In order to benefit from the parallelism benefits in terms of latency and throughput, it is not sufficient to parallelize the SISO-decoding; the interleaving has also to be parallelized. However, since interleaving corresponds to applying a permutation, its parallelization is challenging. A parallel operation requires to access during the same clock cycle data from the same nature, causing memory access conflicts. In the SISO-decoder, those conflicts can be resolved by splitting the considering memories. However, for the interleaving, such a splitting is not sufficient: it is possible that the permutation is such that two extrinsic values have to be written in the same memory at the same time, which is only possible using energy-hungry multi-port memories. To enable energy-efficient parallel interleaving, two solutions are possible: on one hand, a collision-resolution mechanism, which does not restrict the interleaving patterns but requires a significant amountt of extra-hardware; on the other hand, a collision-avoidance mechanism where collisions are avoided by adding restrictions in the interleaver design. We have applied the second strategy, designing collision-free interleavers [] which perform as good as the UMTS standard interleaver [3].The parallel SISO and interleaver lead to a completely parallel PCCC decoding solution, the first reported so far, featuring a record decoding latency performance (5Us/block @ 200MHz).
Optimizing the memory architecture and data mapping has further reduced energy consumption:
Efficient metrics normalization reduces the amount of data to be stored.
Parallel data accesses are economically enabled by packing concurrently accessed data in single long memory words.
An appropriate customized memory hierarchy reduces the access frequency in large memories and enables in-place calculation introducing small local buffers.

Finally, a last energy-reduction has been achieved by applying an early stop criterion that adapts constantly the number of iterations to the decoding convergence, monitored by examining the extrinsic information magnitude.
Table 2 lists main characteristics of the decoder that was optimized to minimize power and maximize flexibility, and is implemented in the T@MPO IP.

Decoder characteristics	
SISO algorithm	Log-max-MAP, 7 parallel workers
Number of iterations	Up to 7
Optional early stop criterion with settable QoS threshold	

Table 2: Low power optimized decoder characteristics

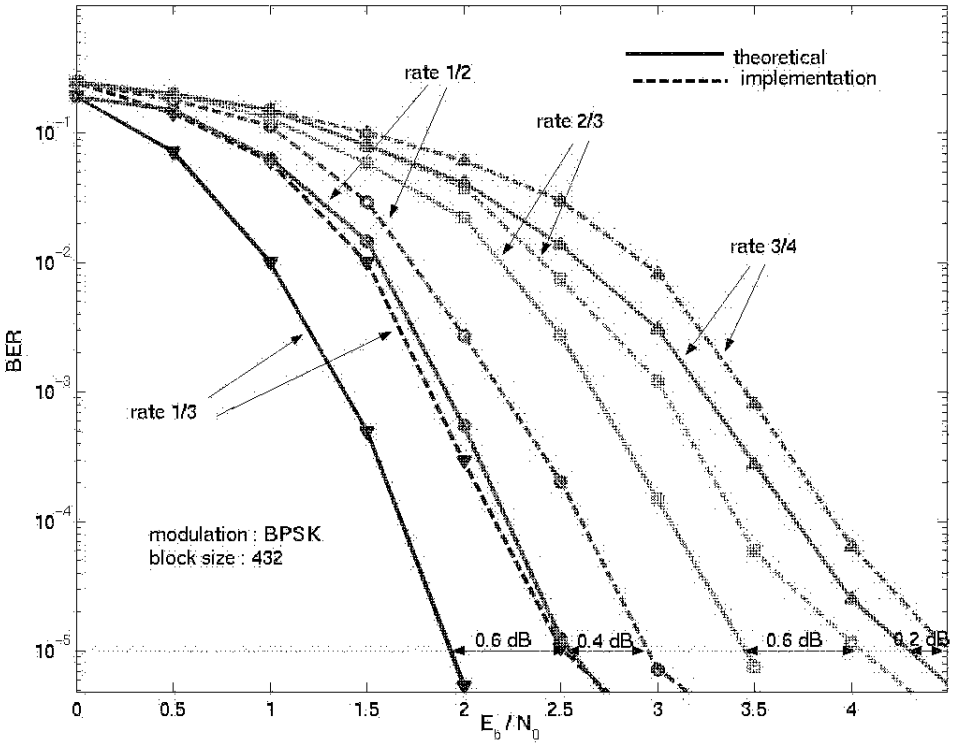


Figure 3: Theoretical and T@MPO IP performances demonstrating limited implementation loss

The optimized architecture has been modeled in C++ using IMEC's OCAPI library [8]. The model has been validated by simulating its BER performance with various modulations, on an AWGN channel and by comparing the results with the original C-code simulations. Figure 3 shows both the theoretical performances of the selected decoder, and the actual results showing limited implementation loss.

The C++ model has also been used as design reference for the VHDL development and to generate automatically stimuli and reference responses for the RT simulations; this, at various levels of the architectural hierarchy (top-level, MAP/interleaver level, worker level).

4. Hardware platform selection and design flow

For the hardware platform selection of the T@MPO, a feasibility assessment for state-of-the-art DSP, FPGA and ASIC technologies has been performed. In order to demonstrate a low power, low latency and high throughput turbo coder, an ASIC implementation has shown to be the only valid solution. The selected ASIC technology was the UMC 0.18u CMOS technology [6].
During the implementation phase of the turbo coder, a consistent design flow has been applied, ensuring a correct design at all stages of the implementation. A C++ Ocapi [8] dataflow model of the T@MPO has been developed which served as the reference for implementation.
In addition, the same system testbenches have been used for the T@MPO dataflow model, VHDL RT model as well as gate-level netlist.
Finally, a strategic test-plan has been defined which covered a wide range of operation scenarios. In this way, the design methodology guaranteed a maximum reliability without the need for extensive time-consuming testing. For logic synthesis, the bottom-up (or divide-and-conquer) compile strategy has been applied.

5. ASIC implementation

The T@MPO ASIC is implemented in a 0.18 µm CMOS process of UMC. This technology was chosen for the low power consumption at the required internal 160 MHz operation speed.
The ASIC design was based on a VHDL description. All technology specific component instantiations are concentrated in a few files. This should allow easy porting of the design to different technologies.
All RAMs are 2-port RAMs, because of the low power optimization. Indeed, the RAM generator available for the process technology consistently generated more power efficient RAMs for 2-port RAMs than for single port RAMs. Most RAMs can be replaced by single port RAMs if this would be more power efficient in another technology. The interleaving patterns are stored in ROM tables. The ROM tables are synthesized as gates due to the lack of a ROM generator.

The implementation process was split in 2 phases. In a first phase, the toplevels of the entities of the ASIC were written in VHDL and the control logic was designed. The data path and memory models of each block were reduced to very simple operations. With this so called “black box” model, system targets like throughput and latency could be checked. The end result of this first phase was a VHDL entity and an I/O behaviour definition of each top-level block. Figure 4 shows the scheme of the top-level model.

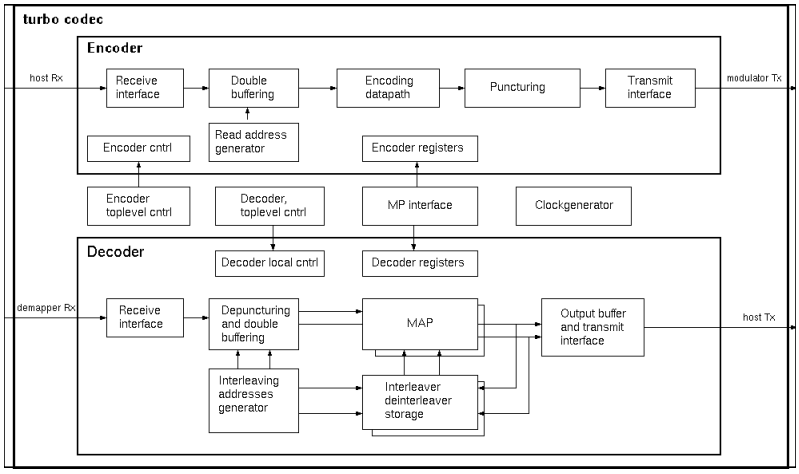


Figure 4: CODEC top-level block diagram

In the second phase, top-level locks were implemented in parallel. The implementation is driven by the need for low power. Low power operation is achieved by disabling the clock of all logic that is not used. The clock generator block is partly designed at the gate level. Together with a careful layout and routing of the clock nets, a maximum skew of 400 ps between any 2 gated clock nets is guaranteed. All signals traveling from one gated clock domain to another gated clock domain are delayed by a delay element of 600 ps. As a result, all communication between gated clock domains can be treated as synchronous.

After synthesis, placement and routing, the critical path is in the “worker” block of the decoder. The critical path has a maximum clock frequency of 170.9 MHz. The requirement of 160 MHz is thus met.
The most important clock domains that can be switched on and off are those for the encoder data reception, the encoder data path, the decoder data reception and the 7 clocks of the decoder workers. The encoder data path and decoder data path always run at 160 MHz, independent from the actual block length. Data is dumped in an internal output FIFO and the respective clock is switched off. After hardware reset all gated clocks, except those for the microprocessor interface, are switched off.

Figure 5 shows the external interfacing to the T@MPO ASIC. For debugging purposes 16 output pins of the ASIC are used to monitor one of several sets of internal signals of the ASIC. During VHDL design, Design For Testability (DFT) was taken into account. During full scan, the whole ASIC operates on a single clock and some nodes are fixed to

a known value. The RAMs have Build In Self Test (BIST) logic, which is turned off during normal mode to reduce power consumption.

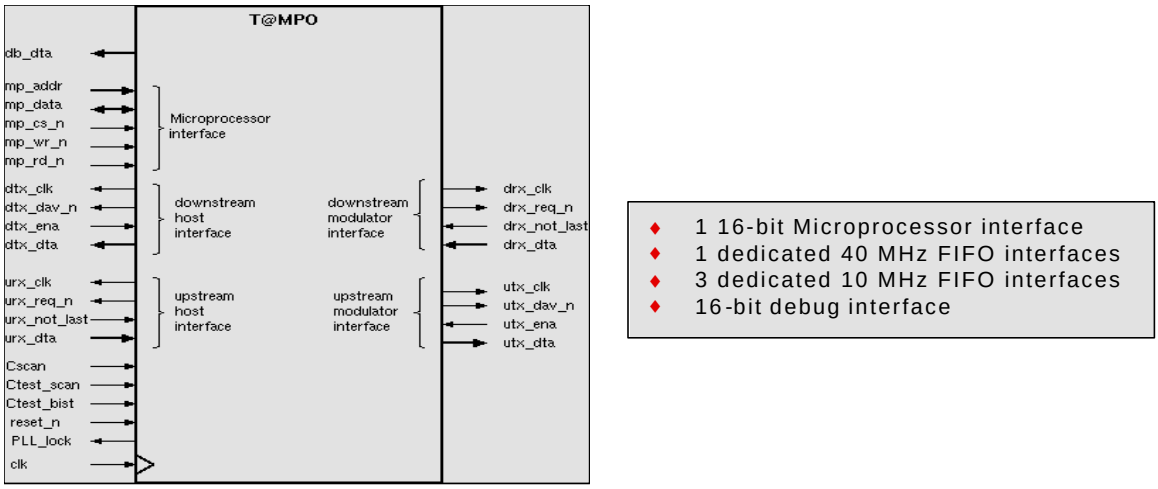


Figure 5 : External interfaces

The 4 data interfaces are designed to communicate with external FIFO components. The 2 host data interfaces are byte wide and run at 10 MHz. The encoded data interface is 12 bit wide and runs at 10 MHz. The decoder LLR data input interface is 12 bit wide (=3 LLR values in parallel) and runs at 40 MHz. The output clock signals at the data interfaces are switched off if no data can be received or needs to be transmitted. The data output interfaces each have a “transmit enable” pin which can postpone the start of the encoding or decoding process if the receiving FIFO is not yet ready to receive the next block.

Input data for the encoder and decoder is received at 80 and 120 Mbit/s respectively. This data is stored in a buffer until a complete block of data is received. In the decoder the data is depunctured (insertion of zero values) before it is stored. All data is stored only once and in the same order as in which it is received. Interleaving or de-interleaving is performed at the read side of the buffer. If a complete block is received, the encoding or decoding process can start. At the same time, new data of the next block is received and stored in a separate input buffer. Once a complete block is received by the encoder double buffering, the encoder data path and puncturing are started. The resulting data is copied to a small output buffer, which is dumped to the output whenever 12 bits are available. Once started, the encoding of a block cannot be stopped by reprogramming or external control signals.

Once a complete block is received by the decoder double buffering, the required number of workers is started. Only the longest block length uses all workers. The double buffering is actually built with 2*7 (2 windows in each of the 7 workers) times 3 (systematic, coded1 and coded2) RAMs. The lower and upper half of the RAMs represent the first and second buffer of the double buffer. Only for the largest block length, all RAMs are activated. The workers all run in parallel and dump their data in separate output buffers. The output buffers are read by a separate block, sliced into '0' or '1' output bits, grouped by 8 and sent to the host data interface. Once started, the decoding of a block cannot be stopped by reprogramming or external control signals.
Table 3 summarizes the main figures of the ASIC implementation.

IP main characteristics (UMC .18µ CMOS technology)	
Critical path	5.85 ns
Internal clock	Up to 170.9 MHz
External clock	Up to 42.7 MHz
19 internal clock regions	
embedded PLL	4x or bypass
Die size	14.7 mm^2
Complexity	373 Kgates
RAM	66 two-port SRAM (36 Kbit)

Table 3: Main T@MPO **ASIC characteristics**

After processing, samples were packaged, and verified in the test-house. The yield was 89%. Figure 6 shows a picture of a packaged T@MPO sample.

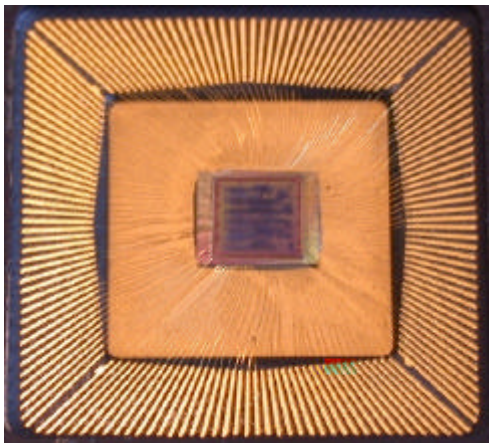


Figure 6: Packaged T@MPO sample

6. Prototype demonstrator and measurements

Prototype demonstrator

In order to test and demonstrate the T@MPO, 2 separate demonstrators have been designed. Both systems consist of a hardware part and a software part. The hardware part is identical for both systems and is based on the PICARD platform [7]. This in-house developed test platform is a modular, PC-based test system with dedicated extensions that allow for testing high-throughput IP cores. A PICARD add-in board that houses a

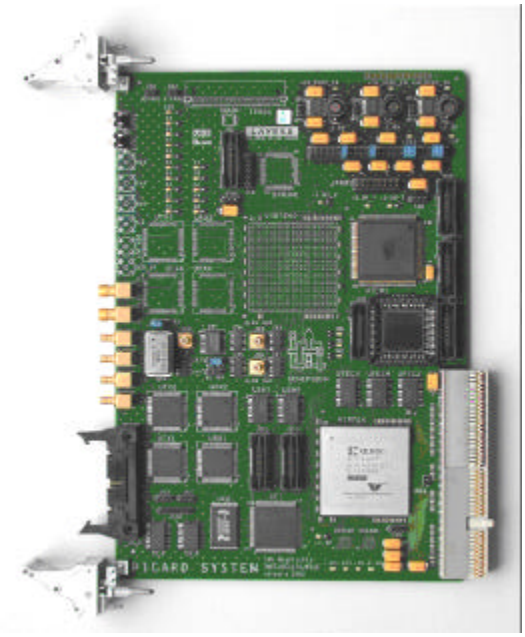


Figure7: PCB for T@MPO testing

T@MPO sample and a FPGA has been designed (see Figure 7). The FPGA contains the interface to PICARD and an AWGN channel model.

The software part of the demonstrators is different for both systems. It runs on the host PC that controls the complete PICARD system. Figure 8 shows the complete test set-up.

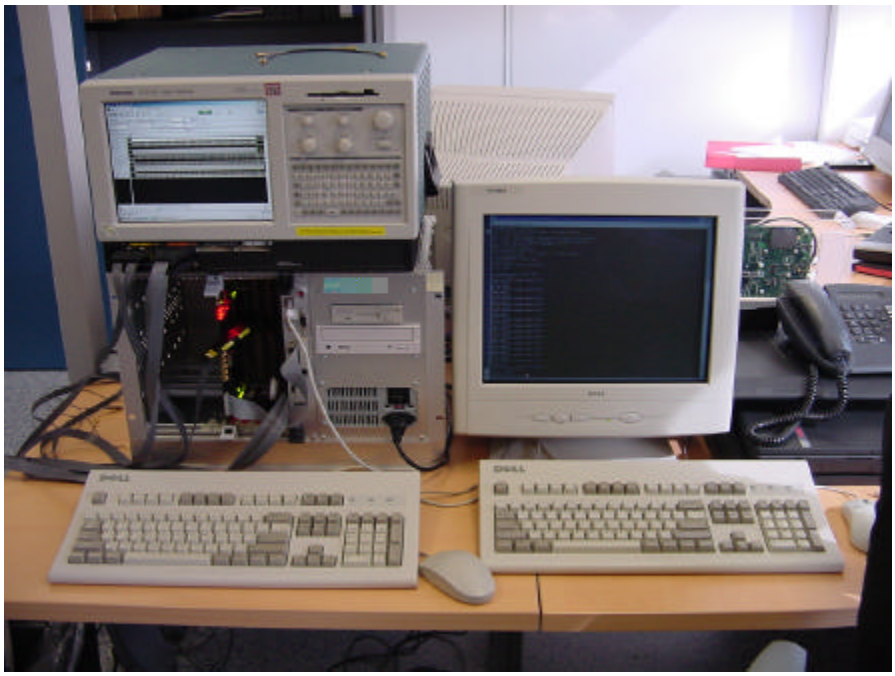


Figure 8: Complete test set-up

The first demonstrator is the so-called scientific demonstrator. This demonstrator has been developed to test the error correcting performance of the T@MPO and has a command-line based interface. This eases the use of the program in large series of tests e.g. to test the performance of all block sizes for a specific code rate and modulation. The output of this demonstrator is a log file that contains the bit error rate for all tested working points of a certain working mode. This file can then easily be imported in a spreadsheet for further processing.

The second demonstrator, denoted graphical demonstrator, is a visually more attractive program that allows the user to see, in real time, the effects of changing the most important working parameters of the T@MPO. It has a graphical user interface that shows the transmission of a video over the implemented channel model. The user can enable and disable the T@MPO, change the most important settings of the ASIC and can select the modulation scheme and Eb/N0 of the channel model.

Measurements

Four parameters have to be evaluated in order to be able to assess the performance of the T@MPO: the throughput, the latency, the power consumption and the error correcting capabilities. All parameters have been measured using the scientific demonstrator.

The throughput and latency have been measured using a logic analyzer connected to the PICARD add-in board. The throughput matches the expected figure of maximum 76 Mbit/s when the T@MPO is clocked at 160 MHz (internal clock). The latency is independent of the selected block size and code rate due to the parallel architecture of the T@MPO. When small block sizes are applied, only a part of the decoder is clocked in order to keep the power consumption low. Larger block sizes cause more parallel units to be activated, thereby keeping the latency constant. The obtained latency meets the required timing of less then 10 us processing time per block.

The energy consumption has been measured for all block sizes and for a varying number of iterations. The power consumption does not depend on the code rate, so the measurements have been limited to code rate 1/3. The power measurements show that the consumed energy is between 12 nJ and 17 nJ per decoded bit, depending on the configuration, for 6 iterations and without early stop criterion. With early stop criterion, it would be kept almost constant around 9 nJ/bit, for all configurations. This is below the targeted 10 nJ/bit. The variation of the decoding energy with the Eb/N0 proves the effectiveness of the early stop criterion, however the ESC design error hampers its demonstration on the T@MPO ASIC.

The error correcting performance has been evaluated for all combinations of code rates and block lengths and this for the four supported modulation schemes. The bit error rate is measured down to 10^{-8} using the Monte Carlo method. The complete set of results is documented in [11].

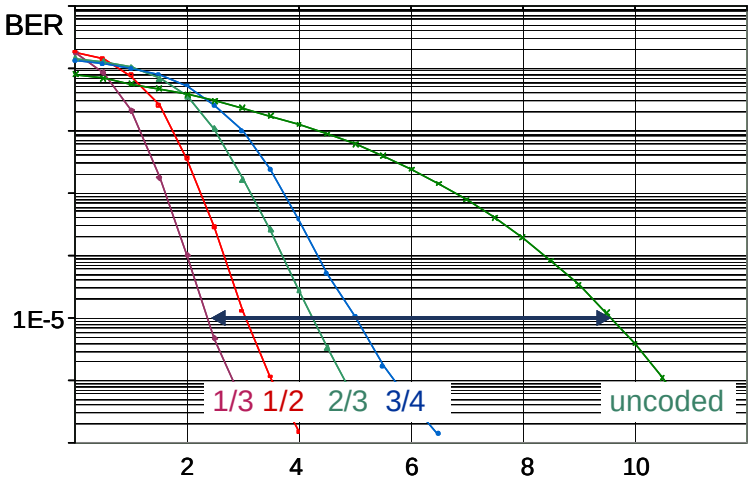


Figure 9: Measured performance of T@MPO IP with block size 288 and BPSK modulation

When evaluating all operation modes of the T@MPO, a problem is revealed for 6 out of the 12 supported block sizes. Examination of the bit error rate curves show that an error floor is detected at an unacceptable high bit error rate. This means that the bit error rate is not improving as expected below a certain threshold.

The cause of this type of errors is twofold. First, the four modes that have a block length that is not a multiple of 32 bits are not working properly. This is due to a problem in the design database that was used to implement the T@MPO. This problem was known before the tests were started and has been solved in the design database.

The second cause of performance loss was not known before the hardware evaluation was started and is due to the interleaver pattern that is used to scramble and de-scramble the data in the encoder and decoder. The patterns are different for every block size and have meticulously been selected at design time from a set of possibilities by running simulations on the dataflow model for every pattern. Since simulation time was restricted, each interleaver has been evaluated up to a bit error rate of 10^{-5}. When running tests on the T@MPO ASIC, more elaborate BER tests can be run and problems with the interleaver show up at the low bit error rates that were impossible to evaluate at design time. The operation modes with block length 256 and 384 bits are affected by this problem. By the nature of this problem, it is clear that this design error was not known before the start of the evaluation tests.

For the modes with block sizes that are working correctly, one can observe that the T@MPO is behaving as expected and that there are curves that show no error floor for bit error rates higher than 10^{-8}.

A further extension of the performance evaluations has been conducted in combination with the FlexCop, IMEC's OFDM core. When comparing the performance of OFDM-based transmission to the results of the single carrier tests over an ideal modem, a small performance loss is detected. This is due to inaccuracies in the functional block that translates the output of the FlexCop demodulator to input for the T@MPO decoder.

7. Conclusions on main results

The T@MPO IP exploits an innovative architecture, allowing to increase transmission range and lower power consumption in embedded wireless systems. The full-duplex turbo encoder/decoder breaks through performance bottlenecks by reducing latency and power.

In a first phase of the project a system and algorithmic level exploration was carried on, trading-off different flavors of turbo-codes (e.g. convolutional turbo-codes versus block product-codes) and leading to the specification of an attractive turbo-coding scheme to be prototyped (see section 2).

In parallel, thorough architectural explorations have been carried on, aiming at removing the bottlenecks hampering speedy and energy-efficient basic-blocks for turbo-decoders, highly parallel architectures were explored as well for the constitutive code soft-input soft-output decoder and for the interleaver processes (section 3).

On basis of the experience built in those first phases, a parallel-concatenated turbo-code (PCCC) has been selected, which has a slightly better performance but also presents a higher flexibility and higher potential energy and throughput gains when applying the innovative architectural solutions. A state-of-the-art PCCC scheme and its corresponding parallel turbo-decoder have been specified.

In order to demonstrate the effectiveness of the solutions in a real-time environment, a turbo-CODEC prototype has been designed. Due to the high requirements e.g. in term of clocking and to the necessity of accurate power measurements, an ASIC platform has been chosen (section 4). The implementation has been done according to an integrated design methodology (section 5) easing the tests and the validation of the design.

Finally, the CODEC prototype has been mapped on a specific testing board integrated in a real-time testing environment that allowed accurate performance and energy consumption measurements (section 6). State-of-art coding gain is measured (up to 8.5 dB) while demonstrated throughput equals 80Mbps and decoding latency and energy 5Us and maximum 10nJ/bit respectively.

References

[Lit. 1] C. Shannon, “A mathematical theory of communications”, Bell Sys. Tech. Journal, vol 27, October 1948.

[Lit. 2] C. Berrou, A. Glavieux, P. Thitimajshima, “Near Shannon limit error-correcting coding and decoding: turbo-codes”, Proc. IEEE ICC, pp.1064-1070, May 1993.

[Lit. 3] 3rd Generation Partnership Project (3GPP), Technical Specification Group (TSG), Radio Access Network (RAN), Working Group1, “Multiplexing and channel coding”, TS 25.222 V1.0.0 Technical Specification, 1999-04.

[Lit. 4] Deliverable P50313-IM-DL-013

[Lit. 5] F. Catthoor, S. Wuytack, E. de Greef, F. Balasa, L. Nachtergaele, A. Vandecapelle, “Custom memory management methodology, exploration of memory organization for embedded multimedia system design”, Kluwer Academic Publisher, 1998.

[Lit. 6] www.umc.com

[Lit. 7] http://www.imec.be/design/M4/Picard.shtml

[Lit. 8] www.imec.be/ocapi

[Lit. 9] www.imec.be/atomium

[Lit. 10] A. Giulietti, L. Van der Perre, M. Strum, "Parallel turbo code interleavers : avoiding collisions in accesses to storage elements ", *Electronics Letters*, vol.38, no.5, February 2002

[Lit. 11] Deliverable P50313-IM-DL-016