# A 150Mbit/s 3GPP LTE Turbo Code Decoder

Matthias May, Thomas Ilnseher, Norbert Wehn
Microelectronic Systems Design Research Group, University of Kaiserslautern
67663 Kaiserslautern, Germany
{may, ilnseher, wehn}@eit.uni-kl.de

Wolfgang Raab
Infineon Technologies AG
81726 München, Germany
wolfgang.raab@infineon.com

***Abstract*—3GPP long term evolution (LTE) enhances the wireless communication standards UMTS and HSDPA towards higher throughput. A throughput of 150 Mbit/s is specified for LTE using 2×2 MIMO. For this, highly punctured Turbo codes with rates up to 0.95 are used for channel coding, which is a big challenge for decoder design. This paper investigates efficient decoder architectures for highly punctured LTE Turbo codes. We present a 150 Mbit/s 3GPP LTE Turbo code decoder, which is part of an industrial SDR multi-standard baseband processor chip.**

## I. Introduction

The outstanding forward error correction capabilities of Turbo codes [1] made them part of many today's communications standards. These include 3GPP UMTS with its increased throughput extensions HSDPA and HSUPA and its long term evolution (LTE) [2]. While these standards' initial application was video telephony with limited throughput requirements, they are now tailored for high throughput mobile internet access. Downlink data rates of 75 Mbit/s per spatial MIMO stream with up to 4 MIMO layers are specified for LTE [3]. To provide this high throughput LTE Turbo codes are highly punctured. While for UMTS a code rate of $1/3$ was used, LTE Turbo codes are punctured to a code rate of 0.95 for maximum throughput of 150 Mbit/s for 2 layers and of 300 Mbit/s for 4 layers, respectively.

High throughput Turbo Code (TC) decoder architectures have to split a received data block into so called windows for parallel decoding. An acquisition process estimates the initial values at the window borders. The accuracy of this acquisition depends on the number of available channel values. Decoding highly punctured TC requires a very long acquisition to incorporate an adequate number of channel values for a communications performance near to that of the non-windowed, continuous decoding. This acquisition results in a considerable amount of additional area, latency, and power. Thus, for highly puntured TC, it is a challenge to reach a communications performance near to continuous decoding at very high throughput, low power and area consumption. This paper presents a 150 Mbit/s 3GPP LTE compliant TC decoder and to the best of our knowledge for the first time a thorough investigation of the decoding problems for highly punctured TC.

The paper is structured as follows. After a brief introduction in TC decoding in Section II, previously published decoder architectures are presented in Section III. Section IV investigates parallel TC decoder architectures. In Section V, the acquisition length for the decoding of highly punctured LTE TC is discussed. Further, we investigate the scalability of different TC decoder architectures to a throughput of 300 Mbit/s and higher for upcoming releases of LTE. The implementation of a 150 Mbit/s 3GPP LTE compliant TC decoder is presented in Section VI.

## II. Turbo Code Decoding

For TC decoding a complete data block is iteratively processed in a loop which comprises two component decoders using the Log-MAP algorithm [1]. These two decoders exchange so called *a posteriori probability* information ($\Lambda^e$). The iterative exchange continues until a stopping criterion is fulfilled. A general TC decoder structure is shown in Figure 1. The decoder receives three different sequences of soft values which are denoted as systematic ($\lambda^s$) and parity information ($\lambda^{p_0}$, $\lambda^{p_1}$), respectively. The systematic information corresponds to the original transmitted data sequence. The parity information is generated by two recursive convolutional encoders. One encoder works on the original data sequence and the other one on an interleaved sequence to destroy correlations.

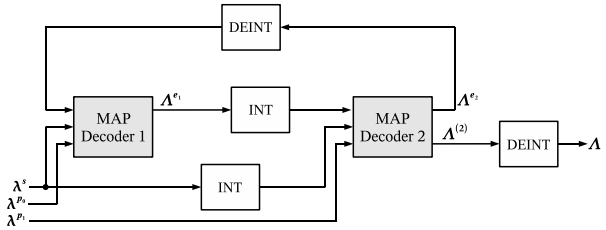


Fig. 1. General Turbo code decoder

The Log-MAP algorithm computes the probability for each bit to have been sent as $d_k = 0$ or $d_k = 1$. Given the received channel values $\lambda$ and potential a priori information, the *logarithmic likelihood ratio* (LLR) of these probabilities is denoted as:

$$\Lambda_{d,k} = \ln \frac{\Pr\{d_k = 0|\lambda\}}{\Pr\{d_k = 1|\lambda\}} \tag{1}$$

It can be calculated using the *state metrics* $\alpha_k^m$ and $\beta_{k+1}^{m'}$ referring to the encoder states and the *branch metrics* $\gamma_{k,k+1}^{m,m'}$ referring to the state transitions. The $\alpha$- and $\beta$-metrics are

978-3-9810801-6-2/DATE10 © 2010 EDAA

gathered in a *forward* and *backward recursion*, respectively:

$$\alpha_{k+1}^{(m')} = \min_{\forall m}^{*} \left( \alpha_k^{(m)} + \gamma_{k,k+1}^{m,m'} \right) \tag{2}$$

$$\beta_k^{(m)} = \min_{\forall m'}^{*} \left( \beta_{k+1}^{(m')} + \gamma_{k,k+1}^{m,m'} \right) \tag{3}$$

where $\min^*$ is a minimum selection with an additional correction term resulting from the transformation of the addition into the logarithmic domain.

Equation 1 shows that a $\Lambda_{d,k}$ of 0 yields a propability of $\Pr\{d_k = 0|\lambda\} = \Pr\{d_k = 1|\lambda\} = \frac{1}{2}$. This can be exploited to perform so called *puncturing*. If a code rate higher than the base rate of $1/3$ is desired, redundant information is removed (punctured) from the original rate $\frac{1}{3}$ stream created by the encoder. These removed bits are replaced in the decoder by an LLR value $\Lambda_{d,k} = 0$.

The Max-Log-MAP algorithm uses the min operation instead of $\min^*$. This leads to a slight loss in communications performance which is reduced to less than 0.1 dB if the exchanged soft values between the component decoders are scaled by a so called *extrinsic scaling factor* (ESF) [4].

The data dependency of the state metrics calculation (see Equations 2, 3) throughout a whole block can be loosened by starting the recursion on arbitrary positions in the block with approximated initialization values. This technique is called *windowing*. It allows memory and latency reduction at the expense of additional computations. The resulting sub-blocks can be mapped to individual decoders for parallel processing and thus for high throughput. In this case, a recursion on a certain number *AL* of preceeding bits must be performed to obtain sufficiently accurate estimates. This is called *acquisition*.

The length of the acquisition at the window borders is critical for highly punctured codes. A method to avoid acquisition at the cost of additional decoding iterations is *next iteration initialization* (NII) [5]. Using NII the final state metrics are exchanged and used for initialization in the next iteration.

## III. Related Work

Previously published high throughput architectures for decoding 3GPP TC are summarized in Table I. Most of the published decoders applying acquisition [6]–[9], [11], [12] perform not more than 64 acquisition steps. We will see later that these acquisition lengths are not suitable for decoding high-rate LTE TC. Using next iteration initialization instead of acquisition [13], [14] also does not achieve the required communications performance which has to be near to non-windowed, continuous decoding in the limit of 6 to 7 decoding iterations for the specified thoughput of 150 Mbit/s. Note that for 100 Mbit/s LTE [10], [13] a code rate of about $2/3$ (2 MIMO layers) is used. In this case, a smaller acquisition length or NII is sufficient.

Recently, Kim and Park published a decoder prototype for LTE and WiMAX [15]. The proposed architecture consisting of 8 parallel SMAP units with radix-4 recursion units provides a throughput of 186 Mbit/s at 8 iterations and 250 MHz. This points to the fact that the decoder performs 128 acquisition steps. This complies with our results in Section V.

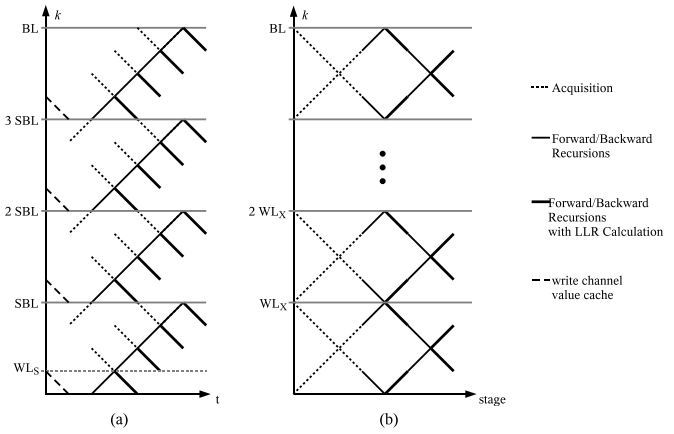


Fig. 2. Windowing schemes: (a) parallel SMAPs, *N*=4; (b) XMAP

## IV. Parallel TC Decoder Architectures

Parallel decoder architectures are key for high throughput. Parallelism can be applied on all levels of the TC decoder design [8]. On the system level, several TC decoders can be implemented in parallel. The throughput nearly linearly scales with the number of TC decoders while the decoding latency is still equal to that of a single decoder. Such an architecture has a low architectural efficiency and high latency which makes it infeasible under strong latency constraints.

High throughput *and* low latency TC decoders are using the windowing scheme to split the block into sub-blocks. Decoding is distributed on parallel working MAP units or the MAP decoder itself is parallelized. Other techniques can be applied on lower levels, e.g., replacing the radix-2 unit for state metrics calculation by a radix-$2^n$ unit to process several bits simultaneously [6], [16].

The most widely used parallel decoder architecture is based on parallel working MAP decoders [6]–[8], [10]–[12], [17]. In this architecture, a data block of length *BL* is divided into *N* sub-blocks of length *SBL*. Each sub-block is again divided into several windows of length $WL_S$. Each sub-block is processed independently by a dedicated MAP decoder. The corresponding sub-block trellis in such a MAP decoder is *serially* processed on window level. Thus, it is called SMAP in the following. Each SMAP decoder produces at maximum one LLR per clock cycle and uses two recursion units to perform one forward- and backward-recursion step in each clock cycle and one additional recursion unit for acquisition in parallel. The resulting processing order for the three recursion units is shown in Figure 2 (a). The recursion unit for acquisition can be omitted if NII is used. Dotted lines indicate the acquisition steps. Solid lines symbolize forward and backward recursion steps, respectively. Thick lines indicate that the LLR output calculations is performed in parallel to the corresponding backward recursion.

TABLE I
PREVIOUSLY PUBLISHED HIGH THROUGHPUT 3GPP TC DECODER ARCHITECTURES

| Work | Standard(s) | Architecture | $WL$ | $AL$ | Throughput [Mbit/s] | @ iter. | @ $f_C$ [MHz] | Technology | Area [mm$^2$] | Power [mW] |
|---|---|---|---|---|---|---|---|---|---|---|
| [6] | HSDPA | Single SMAP unit with 3 radix-4 recursion units | 40 | 40 | 21.6 | 6 | 129.6 | 180 nm | 14.5 | 956 |
| [7] | HSDPA | Single SMAP unit with 3 recursion units | 40 | 40 | 20.2 | 5.5 | 246 | 130 nm | 1.2 | 61.5 |
| [8] | UMTS | Multiple parallel SMAP units with 3 recursion units | 20 | 20 | 60 (8 SMAP) | 6 | 166 | 180 nm | 16.0 | 650 |
| [9] | 3GPP | Fully pipelined XMAP | 4–16 | 20 | 308 ($WL$=8) | 6 | 500 | 130 nm | 3.8 | – |
| [10] | LTE, WiMAX | 4 parallel SMAP units | – | – | 100 | – | – | – | – | – |
| [11] | Multiple | Multiple parallel SMAP units with 3 radix-4 recursion units | 64 | 64 | 758 (64 SMAP) | 6 | 256 | 130 nm | 13.1 | 573 |
| [12] | Multiple | 32 parallel SMAP units with 3 radix-4 recursion units | $\leq 64$ | $\leq 64$ | 711 | 6 | 200 | 65 nm | – | – |
| [13] | Multiple | Unified ASIP architecture for Turbo and LDPC decoding | 1 | Only NII | 100 (LTE TC) | – | 333 | 45 nm | 0.23 (logic only) | 230 |
| [14] | Multiple | Multiple ASIPs, 2 recursion units per ASIP | 64 | Only NII | 128 (WiMAX TC, 8 ASIP) | 5 | 400 | 90 nm | 3.96 | – |
| [15] | LTE, WiMAX | 8 parallel SMAP units with 3 radix-4 recursion units | 128 | 128 | 186 (LTE TC) | 8 | 250 | 130 nm | 10.7 | 900 |
| **This work** | LTE (punctured up to rate 0.95) | XMAP, 8 LLRs per cycle, 4 recursion operations folded to one recursion unit | 32 | 96 | 150 | 6.5 | 300 | 65 nm | 2.1 | 300 |

In contrast to the SMAP architecture, which processes *several* sub-blocks in parallel on individual MAP decoders, the XMAP architecture [9], [18] processes a *single* window at a time. The corresponding window calculation is performed by a monolithic fully pipelined MAP engine. Again, a data-block is partitioned into $BL/N$ windows of length $WL_X = N$. The spatial processing of the XMAP pipeline is shown in Figure 2 (b). The recursions in each window are fully unrolled and pipelined. Every cycle a complete new window is fed into the decoder yielding $WL_X$ decoded LLRs per clock cycle. For lower throughput requirements, $H$ recursion operations can be *folded* onto a single recursion unit, i.e., the recursions are only partially unrolled. In this case the XMAP decoder calculates $\frac{WL_X}{H}$ LLRs per clock cycle.

The XMAP architecture takes advantage of the limited lifetime of the state metrics. This can be exploited to reduce the storage requirements yielding smaller area and energy. This architecture was presented for the first time in [18]. Later, Mansour and Shanbhag [19] have theoretically proven the optimality of the XMAP architecture for a pipelined windowing scheme with respect to the state metrics memory.

A comparison of the parallel SMAP and XMAP architecture is shown in Table II. For moderate acquisition lengths the XMAP needs less recursion units than a parallel SMAP decoder. However, this turns around for very long acquisitions. As will be shown in Section V, LTE requires very long acquisitions for high code rates. Though, the disadvantage of long acquisitions can be absorbed by folding as described above and by applying next iteration initialization (see Section V) in combination with acquisition.

The window length $WL_X = N \cdot H$ in an XMAP decoder is usually smaller than the window length $WL_S$ in an SMAP decoder, which results in a *smaller latency and a considerably smaller amount of state metrics storage*. Note that the quadratic scaling of the state metrics memory with $N$ for the XMAP can be broken by processing more than one window in the decoder pipeline [19] for large $N$.

TABLE II
PARALLEL TURBO DECODER ARCHITECTURES

| | $N$ parallel SMAP | XMAP |
|---|---|---|
| # recursion units | $2 \cdot N$ for $AL = 0$<br>$3 \cdot N$ for $0 < AL \leq WL_S$<br>$4 \cdot N$ for $WL_S < AL \leq 2WL_S$<br>... | $2 \cdot \left(\frac{AL}{H} + N\right)$ |
| MAP latency | $AL + 2 \cdot WL_S$ | $AL + N \cdot H$ |
| state metrics memory | $N \times WL_S \times S \cdot bw_{\alpha\beta}$<br>dual-port RAMs | $\frac{N^2 \cdot H}{4} \cdot S \cdot bw_{\alpha\beta}$ bits<br>register pipeline |
| NII memory | $2 \cdot \frac{BL}{WL_S} \times S \cdot bw_{\alpha\beta}$<br>single-port RAM | $4 \cdot \frac{BL}{N \cdot H} \times S \cdot bw_{\alpha\beta}$<br>single-port RAM |

$bw_{\alpha\beta}$: bit width of state metrics
$S$: number of encoder states (8 for LTE TC)

Due to the both-sided acquisition at every window, the XMAP decoder requires double the memory for NII. However, this memory is relatively small compared to the state metrics memory and channel value cache.

A higher degree of parallelism $N$ becomes mandatory for further throughput increase to 300 Mbit/s or higher in upcoming LTE releases. For a higher $N$, like 16, 32, 64,

- the XMAP shows an improved communications performance due to less acquisitions,
- the MAP latency becomes the determining factor for decoding throughput. Due to the pipelined processing, the XMAP latency can be *completely* hidden by mul-

tiplexing the half-iterations of two Turbo code blocks. For an SMAP architecture, this is only possible with a disproportionally large overhead.

*Interleaving*

In parallel architectures multiple extrinsic values are produced per clock cycle. These values have to be interleaved, i.e. they must be fetched from and stored to the memory at the same time. For that, the extrinsic memory is partitioned into smaller memories to allow for concurrent, independent read and write operations. Nevertheless this scrambling can still result in memory access conflicts which occur in those cases where multiple values are addressed in the same physical memory in the same clock cycle. Various techniques exist to solve these conflicts at design-time or run-time [11], [20]–[23].

LTE adopts quadratic permutation polynomial (QPP) interleavers for Turbo coding [2] to avoid conflicts at run-time. The interleaver is defined as:

$$\Pi(i) = (f_1\, i + f_2\, i^2) \mod BL \tag{4}$$

It has been shown in [24] that QPP interleavers are free of memory access conflicts if the number of parallel processed sub-blocks, i.e. the number of SMAPs, is a factor of the interleaver length. Note that this is only valid if all component decoders access the memory in the same order. Therefore, the latency of the SMAP cannot be reduced by applying a *double windowing scheme* as proposed in [8].

For conflict-free interleaving in an XMAP architecture each memory $m(i) = \Pi(i) \bmod N$ with $i \in [wN, (w+1)N-1]$, $w$ is an integer, $0 \le w < \frac{BL}{N}$, has to be accessed exactly once. Assuming that $N$ is a factor of $BL$, it is obvious that:

$$\Pi(i) \equiv \Pi(i + w \cdot N) \mod N \tag{5}$$

Using (5) and the fact that an interleaver generates each address $\Pi(i)$ for $i \in [0, BL-1]$ exactly once, we can prove that for XMAP decoders interleaving is also conflict-free if $N$ is a factor of the interleaver length.

In summary, although the XMAP decoder has some disadvantages to the SMAP approach in some aspects, the better scalability and advantages for future very high throughput requirements dominate.

## V. Acquisition Length in LTE TC Decoder Design

The maximum data rate of UMTS Rel. 99 was 384 kbit/s at a code rate of $1/3$. In the meantime, UMTS has been advanced to HSDPA and further to LTE which apply different techniques to improve the data rate. Among these techniques a higher code rate is used, generated from the rate $1/3$ mother code by puncturing. For LTE, the throughput of coded bits is limited to 80 Mbit/s per spatial MIMO stream of 20MHz channel bandwidth with 64QAM. To achieve a payload throughput of 150 Mbit/s with 2×2 MIMO a highly punctured TC of rate 0.95 is used. For such a highly punctured code, many bits of the rate-$1/3$ stream are removed, e.g., only 6452 out of 18444 encoded bits are transmitted for a block of 6144 information bits. As already mentioned the acquisition length

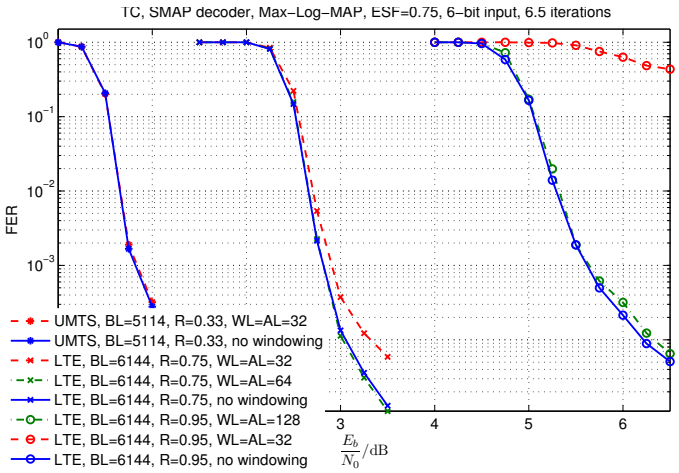


Fig. 3. Frame error rate (FER) for different code rates $R$ and acquisition lengths $AL$

for such highly punctured codes is a big challenge in decoder design. Here we investigate for the first time this problem for high throughput LTE TC decoding.

In Figure 3, the communications performance with different acquisition lengths is compared to the (optimal) performance of continuous, non-windowed decoding. For the rate $1/3$ UMTS TC an $AL$ of 32 yields optimal performance. Decoding a rate $3/4$ LTE TC with $AL$=32 results in a higher error floor, whereas decoding with $AL$=64 obtains the same performace as continuous decoding. The decoder practically refuses to work with $AL$=32 for a rate 0.95 LTE TC. In this case, an acquisition of 128 steps is necessary. Here the simulation results for the widely used parallel SMAP architecture are presented. Simulations with an XMAP TC decoder showed similar results.

Reducing the acquisition length is beneficial for area, latency, and energy. This can be achieved using NII. NII so far was only used to avoid any acquisition. The communications performance for a rate 0.95 LTE TC applying NII is shown in Figure 4. Even with a window length of 128, NII instead of acquisition is not sufficient for this code rate at 6.5 iterations.

Here we propose for the first time to combine NII and acquisition. Only in the first iteration the acquistion is initialized with states of equal probability. In each iteration, state metrics for initializing the acquistion in the next iteration are stored. NII in combination with 32 acquisition steps yields still a performance loss of about 0.3 dB. Applying NII in combination with an acquisition of 64 steps shows nearly the same convergence as the decoder with $AL$=128 at the same number of decoding iterations. Moreover, the decoder with $AL$=128 has double the latency as the decoder with $AL$=64 and NII. Hence, it can run only 6 instead of 6.5 iterations for the same throughput which yields a degradation in the communications performance. Furthermore, the decoder with $AL$=64 requires only 50% of the memory for state metrics and input value caches, whereas the additional memory for NII is relatively small (see Table II). So, in total, the memory is reduced.

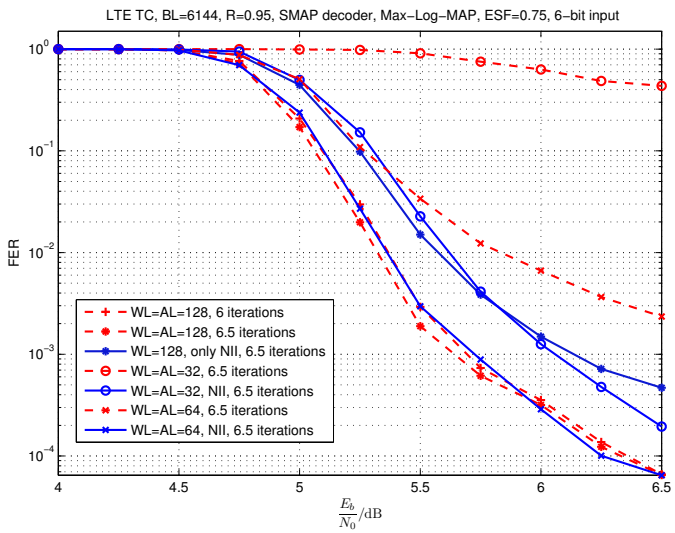


Fig. 4. LTE TC with code rate 0.95: Impact of NII

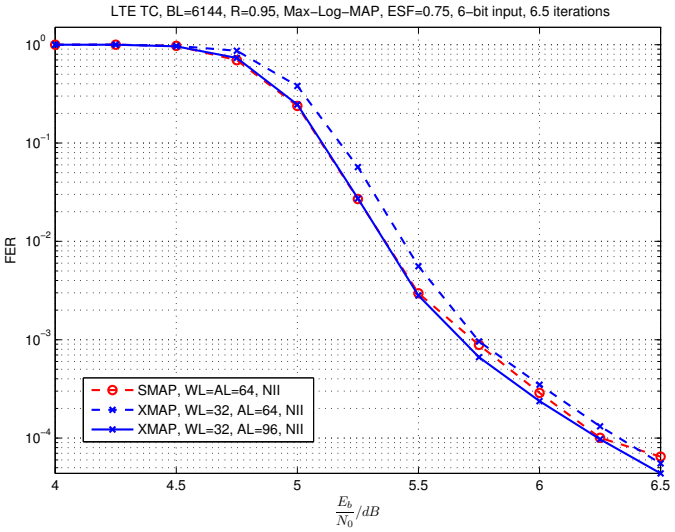


Fig. 5. LTE TC with code rate 0.95, XMAP and SMAP architectures.

An acquisition of 96 steps is necessary for the XMAP architecture, which is shown in Figure 5. It is longer compared to the acquisition in the SMAP decoder due to the both-sided acquisition at every window.

## VI. LTE TC Decoder Implementation

Based on the investigations presented in the previous sections we implemented an LTE compliant TC decoder based on the XMAP architecture. The peak throughput for large frame sizes slightly exceeds the required 150 Mbit/s at 6.5 iterations and 300 MHz. The 300 MHz frequency constraint was imposed by the baseband chip in which the LTE TC decoder is embedded. The corresponding parameters are listed in Table III. The decoder architecture is shown in Figure 6. I/O memories are doubled to hide the latency of input and output data streaming.

TABLE III
Turbo Decoder Parameters

| | |
|---|---|
| Algorithm | Max-Log-MAP with ESF of 0.75 |
| Number of half-iterations | run-time configurable |
| Throughput | 153 Mbit/s ($BL$=6144, 13 half-iterations, 300 MHz) |
| Channel value quantization | 6 bit |
| Extrinsic value quantization | 7 bit |
| State metrics quantization | 11 bit |
| Frame sizes | All LTE frame sizes |
| Code rate $R$ | Up to 0.95 (by external puncturing, interface for $R$=1/3) |
| Architecture | XMAP |
| Window length $WL_X$ | 32 |
| Acquisition length $AL$ | 96 |
| Recursion unit folding factor $H$ | 4 |

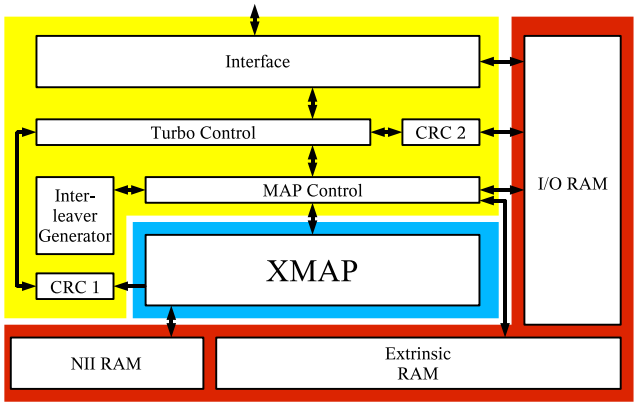


Fig. 6. Decoder architecture

The LTE standard mandates that a CRC is added to every turbo code block. This CRC is used to stop the iterative decoding process as soon as possible. Due to the high puncturing, the decoding process can oscillate, i.e., although the decoder converges to a valid code word in the $n$th half-iteration, the code word after the $n+1$th half iteration is invalid again. To avoid this, we implemented a scheme that allows to check the CRC after every half-iteration. Due to interleaving, this scheme requires two CRC units, see Figure 6.

Interleaver addresses are generated on-the-fly. Existing interleaver generator architectures [25], [26] generate one address per cycle. The interleaver addresses (see Equation 4) can be calculated recursively:

$$\Pi(i+1) = (\Pi(i) + f_1 + f_2 + 2\,f_2\,i) \mod BL \qquad (6)$$

with $\Pi(0) = 0$ and an interleaver length of $BL$. For $f_1$ and $f_2$ see [2]. A parallel architecture like the XMAP with $N$=8 requires the calculation of 8 addresses per clock cycle in parallel with steps of 8, respectively. We therefore implemented generators which calculate the addresses:

$$a(\Pi(i+8)) = (a(\Pi(i)) + f_1 + 8\,f_2 + 2\,f_2\,i) \mod (BL/8) \qquad (7)$$

The decoder has been synthesized using Synopsys design compiler under the conditions listed in Table IV, placed and routed, and integrated in an industrial SDR chip. The chip

TABLE IV
PROCESS PARAMETERS

| | |
|---|---|
| Process technology | 65 nm Low Power CMOS Process |
| Operational voltage | 1.1 V |
| Clock frequency | 300 MHz |

TABLE V
POST-LAYOUT AREA RESULTS

| Component | Area | |
|---|---|---|
| Memory (I/O, extrinsic, NII) | 1.13 $mm^2$ | (54 %) |
| XMAP Core | 0.84 $mm^2$ | (40 %) |
| Controllers, Interleaver generators, CRC | 0.13 $mm^2$ | ( 6 %) |
| $\Sigma$ | 2.10 $mm^2$ | |

has left the tape-out phase and is currently in the sample production. In the case that the Decoder is 100% loaded, power estimation shows a power consumption of about 300 mW with a voltage of 1.1 V and a temperature of 100 °C. An operational voltage of 1.1 V is optimal with respect to power at the target frequency of 300 MHz. Area results after place and route are shown in Table V. The colors correspond to the colored partitions in Figure 6. The XMAP core including the state metrics storage consumes only 40% of the overall area.

## VII. CONCLUSION

In this paper, we have presented a 3GPP LTE compliant Turbo code decoder which provides a throughput of 150 Mbit/s at 6.5 decoding iterations and 300 MHz clock frequency with a power consumption of about 300 mW. The decoder has been integrated in an industrial SDR chip in 65 nm low power CMOS process. The architecture has a very good scalability for further throughput demands. Special emphasis was put on the problem of acquisition in highly punctured LTE Turbo codes with code rates up to 0.95. We considerably reduced the high acquisition length needed for this code rate by implementing NII in addition to the acquisition.

## REFERENCES


[1] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon Limit Error-Correcting Coding and Decoding: Turbo-Codes," in *Proc. 1993 International Conference on Communications (ICC '93)*, Geneva, Switzerland, May 1993, pp. 1064–1070.

[2] Third Generation Partnership Project, "3GPP home page," www.3gpp.org.

[3] J. Berkmann, C. Carbonelli, F. Dietrich, C. Drewes, and W. Xu, "On 3G LTE Terminal Implementation - Standard, Algorithms, Complexities and Challenges," in *Proc. International Wireless Communications and Mobile Computing Conference IWCMC '08*, 6–8 Aug. 2008, pp. 970–975.

[4] J. Vogt and A. Finger, "Improving the Max-Log-MAP Turbo Decoder," *IEEE Electronic Letters*, vol. 36, 2000.

[5] J. Dielissen and J. Huiskens, "State Vector Reduction for Initialization of Sliding Windows MAP," in *Proc. 2nd International Symposium on Turbo Codes & Related Topics*, Brest, France, Sep. 2000, pp. 387–390.

[6] M. Bickerstaff, L. Davis, C. Thomas, D. Garrett, and C. Nicol, "A 24Mb/s Radix-4 LogMAP Turbo Decoder for 3GPP-HSDPA Mobile Wireless," in *Proc. 2003 IEEE International Solid-State Circuits Conference (ISSCC '03)*, San Francisco, CA, USA, Feb. 2003, pp. 150 – 151,484.

[7] C. Benkeser, A. Burg, T. Cupaiuolo, and Q. Huang, "Design and Optimization of an HSDPA Turbo Decoder ASIC," *IEEE JOURNAL OF SOLID-STATE CIRCUITS*, vol. 44, no. 1, pp. 98–106, Jan. 2009.

[8] M. J. Thul, F. Gilbert, T. Vogt, G. Kreiselmaier, and N. Wehn, "A Scalable System Architecture for High-Throughput Turbo-Decoders," *Journal of VLSI Signal Processing Systems (Special Issue on Signal Processing for Broadband Communications)*, vol. 39, no. 1/2, pp. 63–77, 2005, springer Science and Business Media, Netherlands.

[9] M. May, C. Neeb, and N. Wehn, "Evaluation of High Throughput Turbo-Decoder Architectures," in *Proc. IEEE International Symposium on Circuits and Systems (ISCAS 2007)*, New Orleans, USA, May 2007.

[10] S.-J. Lee, M. Goel, Y. Zhu, J.-F. Ren, and Y. Sun, "Forward error correction decoding for WiMAX and 3GPP LTE modems," in *Proc. 42nd Asilomar Conference on Signals, Systems and Computers*, 26–29 Oct. 2008, pp. 1143–1147.

[11] G. Prescher, T. Gemmeke, and T. Noll, "A Parameterizable Low-Power High-Throughput Turbo-Decoder," in *Proc. 2005 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2005)*, Philadelphia, Pennsylvania, USA, Mar. 2005, pp. V–25–28.

[12] Y. Sun, Y. Zhu, M. Goel, and J. R. Cavallaro, "Configurable and scalable high throughput turbo decoder architecture for multiple 4G wireless standards," in *Proc. International Conference on Application-Specific Systems, Architectures and Processors ASAP 2008*, Jul. 2–4, 2008, pp. 209–214.

[13] F. Naessens, B. Bougard, S. Bressinck, L. Hollevoet, P. Raghavan, L. Van der Perre, and F. Catthoor, "A unified instruction set programmable architecture for multi-standard advanced forward error correction," in *Proc. IEEE Workshop on Signal Processing Systems SiPS 2008*, 8–10 Oct. 2008, pp. 31–36.

[14] O. Muller, A. Baghdadi, and M. Jezequel, "From Parallelism Levels to a Multi-ASIP Architecture for Turbo Decoding," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 17, no. 1, pp. 92–102, Jan. 2009.

[15] J.-H. Kim and I.-C. Park, "A unified parallel radix-4 turbo decoder for mobile WiMAX and 3GPP-LTE," in *Proc. IEEE Custom Integrated Circuits Conference CICC '09*, Sep. 13–16 2009, pp. 487–490.

[16] Y. Zhang and K. K. Parhi, "High-Throughput Radix-4 logMAP Turbo Decoder Architecture," in *Proc. Fortieth Asilomar Conference on Signals, Systems and Computers ACSSC '06*, Oct. 2006, pp. 1711–1715.

[17] D. Gnaedig, E. Boutillon, J. Tousch, and M. Jzquel, "Towards an optimal parallel decoding of turbo codes," in *Proc. 4nd International Symposium on Turbo Codes & Related Topics*, Apr. 2006.

[18] A. Worm, H. Lamm, and N. Wehn, "A High-Speed MAP Architecture with Optimized Memory Size and Power Consumption," in *Proc. 2000 Workshop on Signal Processing Systems (SiPS '00)*, Lafayette, Louisiana, USA, Oct. 2000, pp. 265–274.

[19] M. M. Mansour and N. R. Shanbhag, "VLSI architectures for SISO-APP decoders," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 11, no. 4, pp. 627–650, Aug. 2003.

[20] A. Tarable, S. Benedetto, and G. Montorsi, "Mapping interleaving laws to parallel turbo and LDPC decoder architectures," *IEEE TRANSACTIONS ON INFORMATION THEORY*, vol. 50, no. 9, pp. 2002–2009, Sept. 2004.

[21] T. Richter and G. Fettweis, "Parallel Interleaving on Parallel DSP Architectures," *Proc. 2002 Workshop on Signal Processing Systems (SiPS '02)*, pp. 195–200, Oct. 2002.

[22] M. J. Thul, F. Gilbert, and N. Wehn, "Concurrent Interleaving Architectures for High-Throughput Channel Coding," in *Proc. 2003 Conference on Acoustics, Speech, and Signal Processing (ICASSP '03)*, Hong Kong, P.R.China, Apr. 2003, pp. 613–616.

[23] M. J. Thul, , C. Neeb, , and N. Wehn, "Network-on-Chip-Centric Approach to Interleaving in High Throughput Channel Decoders," in *Proc. 2005 IEEE International Symposium on Circuits and Systems (ISCAS '05)*, Kobe, Japan, May 2005, pp. 1766–1769.

[24] O. Y. Takeshita, "On maximum contention-free interleavers and permutation polynomials over integer rings," *IEEE TRANSACTIONS ON INFORMATION THEORY*, vol. 52, no. 3, pp. 1249–1253, March 2006.

[25] R. Asghar and D. Liu, "Dual standard re-configurable hardware interleaver for turbo decoding," in *Proc. 3rd International Symposium on Wireless Pervasive Computing ISWPC 2008*, 7–9 May 2008, pp. 768–772.

[26] P. Salmela, H. Sorokin, and J. Takala, "Low-complexity polynomials modulo integer with linearly incremented variable," in *Proc. IEEE Workshop on Signal Processing Systems SiPS 2008*, 8–10 Oct. 2008, pp. 251–256.