Architecture multi-ASIP pour turbo décodeur multi-standard

Amer Baghdadi
Amer.Baghdadi@telecom-bretagne.eu
Télécom Bretagne
Technopôle Brest-Iroise - CS 83818
29238 Brest Cedex 3 - FRANCE

GDR-ISIS 2010, Paris.
14 janvier, 2010

Outline

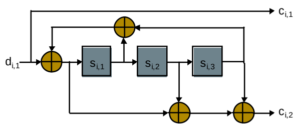
Flexibility requirements in standards for forward error correction codes

Standard	Codes	Rates	States	Block-size	Throughput
EDGE	CC	6/7, 1/3	64	39 - 870	5 - 62 kbps
UMTS	CC	1/2, 1/3	256	1 - 504	up to 32 kbps
	bTC	1/3	8	40 - 5114	up to 2 Mbps
LTE	CC	1/3	64	-	-
	bTC	1/3	8	40 - 6144	up to 200 Mbps
HSDPA	bTC	1/2 - 3/4	8	40 - 5114	up to 14.4 Mbps
CDMA2000	CC	1/2 - 1/6	256	1 - 744	up to 38 kbps
	bTC	1/2 - 1/5	8	378 - 20736	up to 2 Mbps
IEEE802.11 (WLAN)	CC	1/2 - 3/4	64	1 - 4095	6 - 54 Mbps
	CC	2/3	256	-	-
	LDPC	1/2 - 5/6	-	up to 1944	up to 450Mbps
IEEE802.16 (WiMax)	CC	1/2 - 7/8	64	up to 2040	-
	dbTC	1/3,1/2,2/3,3/4, 5/6	8	up to 4800	up to 75 Mbps
	LDPC	1/2 - 3/4	-	up to 2304	up to 75 Mbps
Inmarsat	bTC	1/2	16	up to 2608	up to 64 kbps
DVB-S2	LDPC	1/4 - 9/10	-	16000,64000	up to 255 Mbps

CC: Convolutional Code **bTC**: binary Turbo Code **dbTC**: duo-binary Turbo Code **LDPC**: Low-density parity-check code

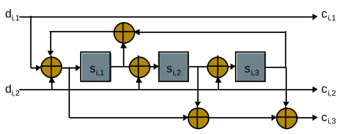
Convolutional turbo codes

- **Code parameters**
  - Number of States
  - Input bits (M-binary)
  - Code rate
  - Puncturing
  - Polynomials

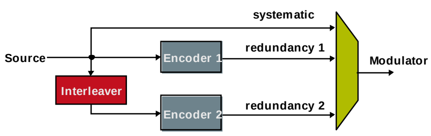


R=1/2, 8 state, binary

R=2/3, 8 states, duo-binary



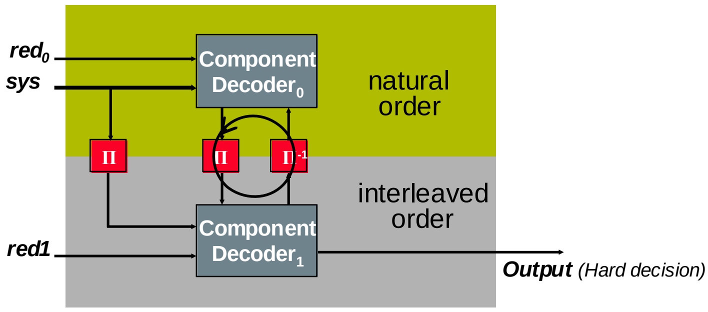
- **Parallel concatenation of recursive systematic convolutional codes**



- **Turbo principle in receiver**

Convolutional turbo decoding

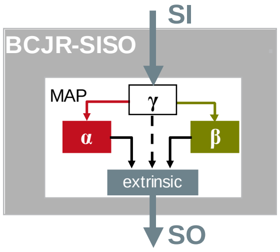
- **Iterative decoding of concatenated convolutional codes**
- **SISO module (Soft Input Soft Output)**
  - Exchange of probabilistic information between decoders
  - MAP algorithm: BCJR or Forward Backward



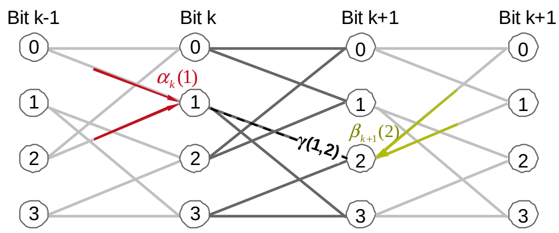
- **Improve convergence speed using parallelism**

BCJR SISO algorithm

- **max-log-MAP computations**



Trellis of a 4-states binary code



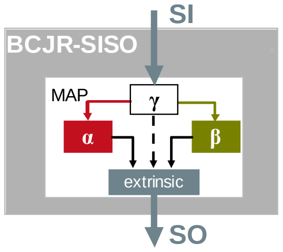
$$\alpha_k(s) = \max_{s'} \left(\alpha_{k-1}(s') + \gamma_k(s', s)\right)$$

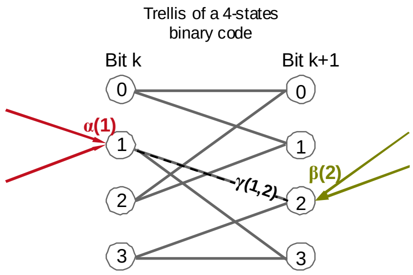
$$\beta_k(s) = \max_{s'} \left(\beta_{k+1}(s') + \gamma_{k+1}(s, s')\right)$$

$$L_k^{ex}(\mathbf{d}_k \equiv i) = \max_{(s',s)/\mathbf{d}(s',s)\equiv i} \left(\alpha_{k-1}(s') + \gamma_k^{ex}(s', s) + \beta_k(s)\right)$$

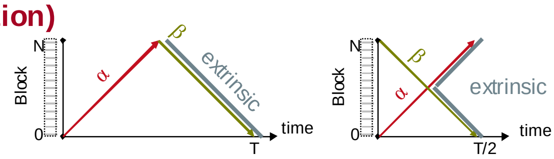
BCJR Metric Level Parallelism (level 1)

- **Parallelism of trellis transitions**



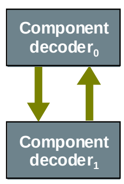


- **Parallelism of BCJR computations (recursions, extrinsic information)**

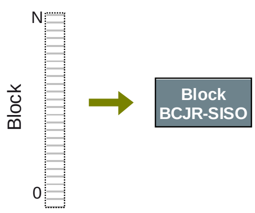


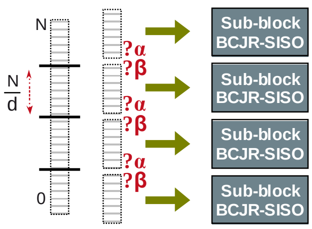
BCJR-SISO Decoder Level Parallelism (level 2)

- **Component-decoder parallelism**
  - Classical (serial) decoding
  - Shuffled decoding



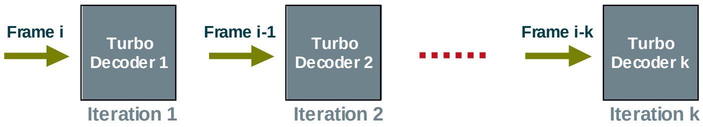
- **Sub-block parallelism**



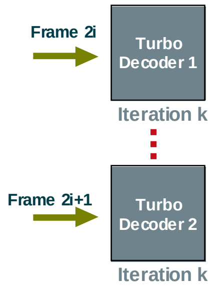


Turbo-decoder Level Parallelism (level 3)

- **Iteration parallelism**

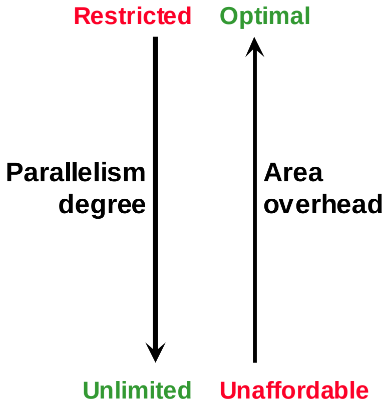


- **Frame parallelism**



Parallelism technique classification

Level	*Parallelism*
BCJR metric	Trellis transitions
	BCJR computations
BCJR-SISO decoder	Sub-blocks
	Component decoders
Turbo-decoder	Iterations
	Frames



Outline

- Introduction
  - Flexibility requirements in forward error correction
  - Convolution turbo codes
  - Turbo decoding iterative principle
  - BCJR SISO Algorithm
  - Parallelism levels classification
- **ASIP design for flexible turbo decoding**
  - Parameters of flexibility
  - Overall ASIP Architecture
  - Basic Computational Blocks
  - Instruction Set
  - Sample illustration program
- NoC-based multi-ASIP turbo decoding
  - Multi-ASIP platform
  - Adaptive NoC architecture
  - Prototyping and on-board validation
- Conclusion

Flexibility requirements in applications using convolutional turbo codes

Application	Rate	Throughput (Mbps)	Code	Block size (bits)
CCSDS (deep space)	1/6 … 1/2	1,6	binary, 16-state	1784 … 16384
UMTS (mobile 3G)	1/3	2	binary, 8-state	40 … 5114
CDMA2000 (mobile 3G)	1/5 … 1/2	2	binary, 8-state	378 - 20736
LTE (mobile)	1/3	100	binary, 8-state	40… 6144
DVB-RCS (TV)	1/3 … 6/7	2	duo-binary, 8-state	96 … 1728
DVB-RCT (TV)	1/2, 1/4	2	duo-binary, 8-state	144 … 648
Inmarsat (multimedia)	1/2	0,064	binary, 16-state	… 2608
WiMAX	1/2 … 5/6	75	duo-binary, 8-state	48 … 4800
Homeplug AV	1/2,16/21	200	duo-binary, 8-state	128, 1088, 4160

+ Trellis, puncturing, interleaver, concatenation, # of component decoders

Design decisions and ASIP flexibility

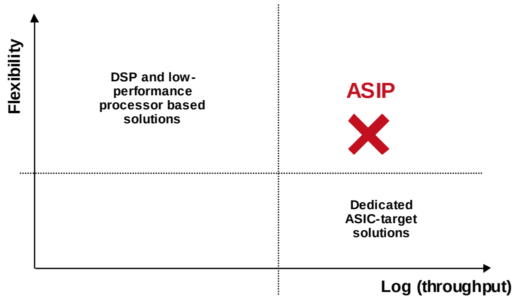
Flexibility

Variants	Decisions
Concatenation	Parallel
Polynomial	systematic recursive
Encoder	< 16 binary states or 8 duo-binary states
Interleaver	All
Rate	>= 1/7 for simple binary and 1/3 for duo-binary
Puncturing	Patterns of 16 bits
Block size	< 65536 symbol (1024 windows)
Algorithm	Max-log-MAP
Iterative process ctrl.	Fixed iterations # SF for extrinsic info.

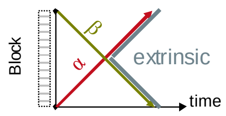
Target parallelism (level 1)

- 32 transitions
- 2 BCJR units

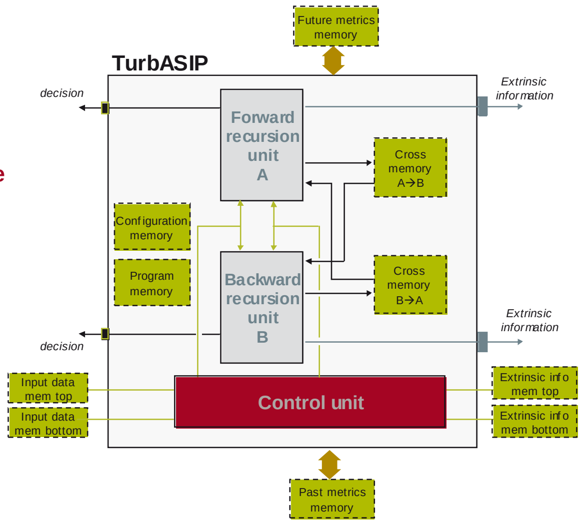
ASIP Architecture



Complete ASIP Architecture

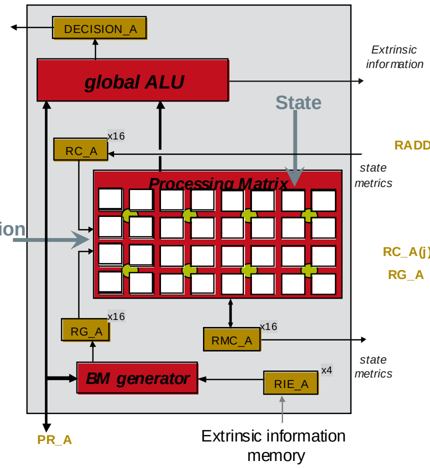


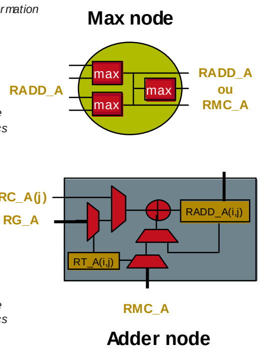
- SIMD architecture, 7-stage pipeline, dedicated and extensible instruction set (~ 31 insts)
- Maximum internal parallelism
- IO: packet interface to exchange information with other processors
- Supports all standardized turbo codes (single and double binary)



BCJR recursion unit

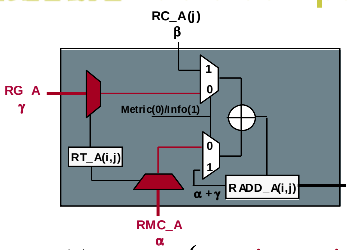
- **16 branch metrics, depending on puncturing**
- **Up to 32 transitions per processing matrix**
- **8 Max nodes**
  - Column wise: state metrics
  - Row wise: extrinsic information





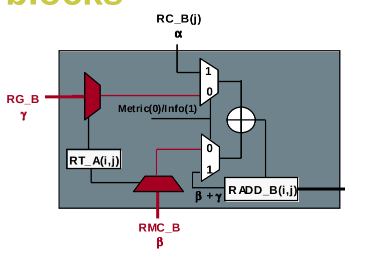
Basic computation blocks

Adder Node Forward



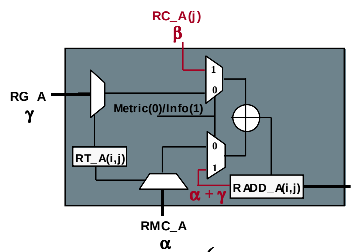
$$\alpha_k(s) = \max_{s'} \left(\alpha_{k-1}(s') + \gamma_k(s', s)\right)$$

Adder Node Backward



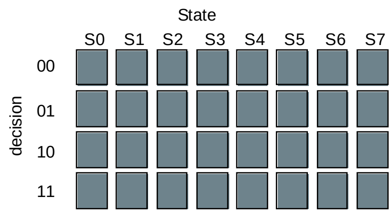
$$\beta_k(s) = \max_{s'} \left(\beta_{k+1}(s') + \gamma_{k+1}(s, s')\right)$$

Adder Node Fw/Bw



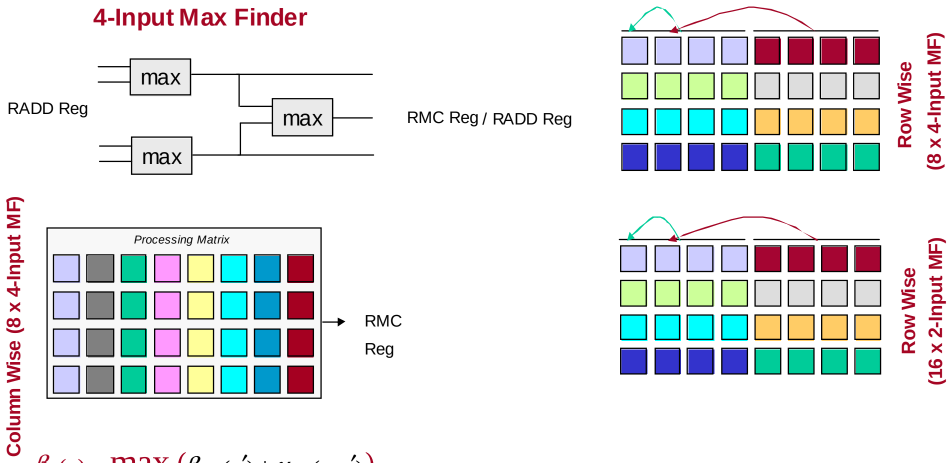
$$L_k^{ex}(\mathbf{d}_k \equiv i) = \max_{(s',s)/\mathbf{d}(s',s)\equiv i} \left(\alpha_{k-1}(s') + \gamma_k^{ex}(s', s) + \beta_k(s)\right)$$

Processing Matrix



Config for 8 states db

Basic computation blocks (Cont.)

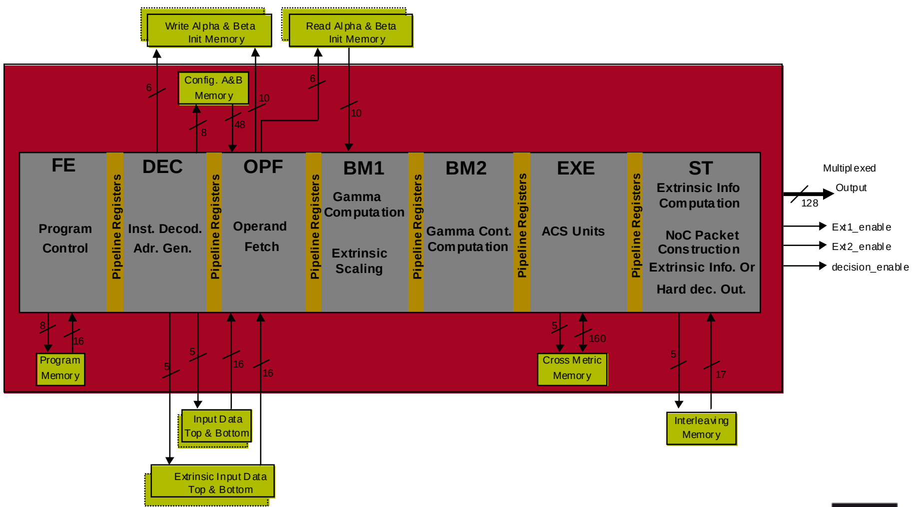


$$\beta_k(s) = \max_{s'} \left(\beta_{k+1}(s') + \gamma_{k+1}(s,s')\right)$$

$$\alpha_k(s) = \max_{s'} \left(\alpha_{k-1}(s') + \gamma_k(s',s)\right)$$

$$L_k^{ex}(\mathbf{d}_k \equiv i) = \max_{(s',s)/\mathbf{d}(s',s)\equiv i} \left(\alpha_{k-1}(s') + \gamma_k^{ex}(s',s) + \beta_k(s)\right)$$

Control, pipeline stages, and memories



ASIP instruction set

- **Registers/Memory Transfer**
  - Trellis configuration
  - State metric initialization
  - Windowing support
- **Control**
  - ZOL
  - Repeat
- **Operative Map**
  - Forward/Backward recursion
    - State metrics
    - Extrinsic info.
    - Hard decisions
- **Max tree**
  - Column
    - Max between 2 Inputs
    - Max between 4 Inputs
  - Row
    - Max between 2 Inputs
    - Max between 4 Inputs

ASIP: application example

double binary 8-states code

```
        LD_CONFIG 0
        LD_CONFIG 1
        LD_CONFIG 2
        LD_CONFIG 3
        SET_SIZE 48
_loop:  LD_REC
        ZOLB _LW,_LW,_RW
        DATA_LEFT add m      |
_LW:    max2 m               | x24
        DATA_RIGHT add m     |
        max2 m               |
        NO_LD add i          | x24
        max2 i               |
_RW:    max1 i ST_EXT        |
        ST_REC
        jmp   _loop
```

single binary 8-states code

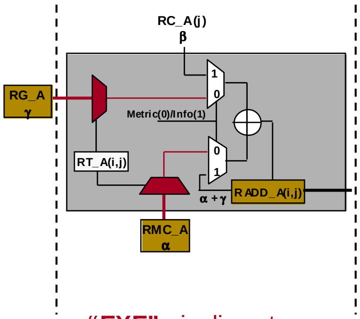
```
        LD_CONFIG 0
        LD_CONFIG 1
        SET_SIZE 40
_loop:  LD_REC
        ZOLB _LW,_LW,_RW
        DATA_LEFT add m      |
_LW:    max1 m               | x20
        DATA_RIGHT add m     |
        max1 m               |
        NO_LD add i          | x20
        max2 i               |
_RW:    max1 i ST_EXT        |
        ST_REC
        jmp   _loop
```

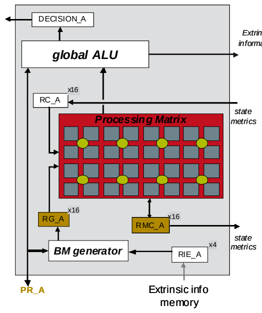
~ 3,5 cycles /symbol

ASIP: application example (cont.)

BM computations and **accumulation with old state metrics**

…
DATA_LEFT add m
max2 m
…
…
x24





- **All trellis transitions are processed in parallel**

(parallelism level 1: ***Parallelism of trellis transitions*****)**

"EXE" pipeline stage

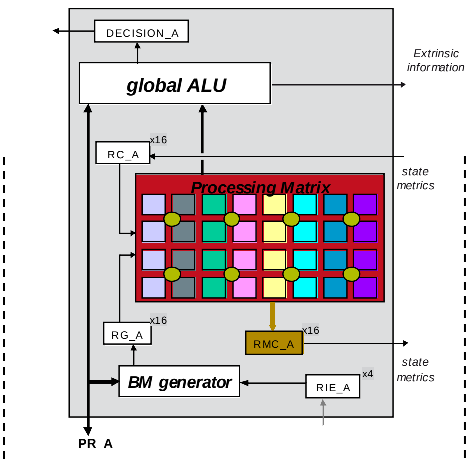
TELECOM Bretagne

ASIP: application example (cont.)

Select and update state metrics

…
DATA_LEFT add m
max2 m
…
…
x24

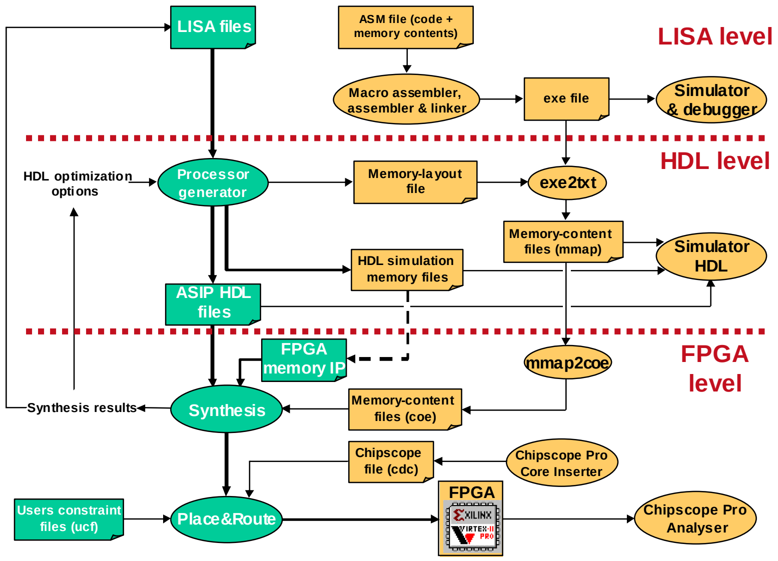
- **Use of 4-input Max function in column wise configuration for each state**
- **State metrics are saved in RMC registers**



"EXE" pipeline stage

TELECOM Bretagne

Design and verification flows

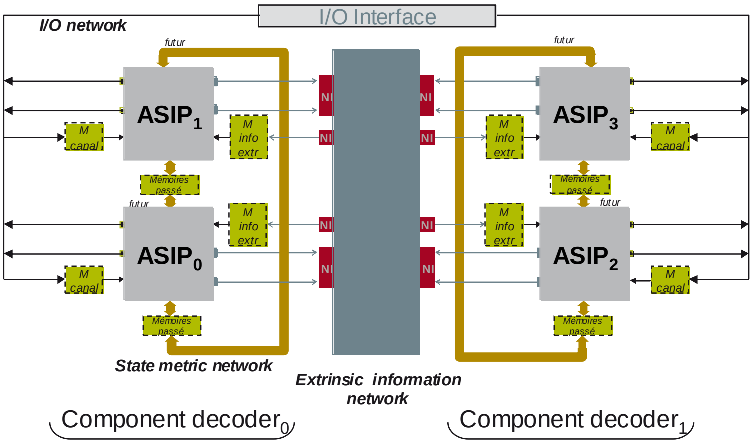


Outline

- Introduction
  - Flexibility requirements in forward error correction
  - Convolution turbo codes
  - Turbo decoding iterative principle
  - BCJR SISO Algorithm
  - Parallelism levels classification
- ASIP design for flexible turbo decoding
  - Parameters of flexibility
  - Overall ASIP Architecture
  - Basic Computational Blocks
  - Instruction Set
  - Sample illustration program
- **NoC-based multi-ASIP turbo decoding**
  - Multi-ASIP platform
  - Adaptive NoC architecture
  - Prototyping and on-board validation
- Conclusion

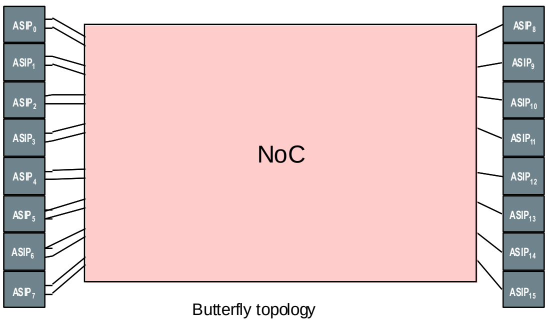
High-throughput multi-ASIP turbo decoder

- 3 dedicated networks
- Network interface: interleaver
- Low latency for the extrinsic information network
- Multi-stage interconnection networks

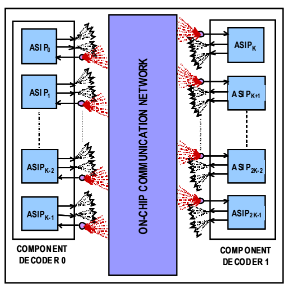


NoC for flexible turbo decoding

- Iterative extensive exchanges of interleaved data require a fully flexible on-chip interconnection network
- Support of all interleavers – avoid communication conflicts



Butterfly topology



- Appropriate NoC topologies explored (Butterfly, Benes, De Bruijn)

Results and more information…

- **AFANA Project (Application-Field-Aware Adaptive Network on chip Architecture)**
  (http://recherche.telecom-bretagne.eu/afana)

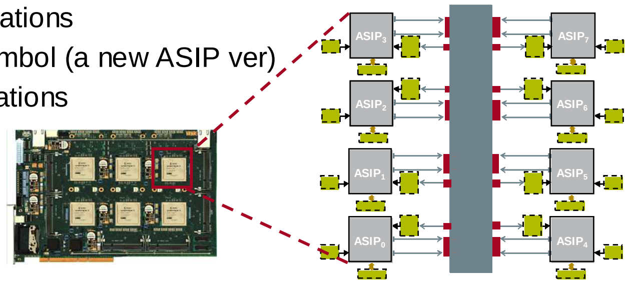
THOMSON images & beyond

InPixal

ANR – Architecture du futur
2007-2010

Prototyping and on-board validation

- **8-ASIP prototype using a logic emulation board**
  - F = 135 MHz
  - 53% of a single FPGA Xilinx Virtex 5 LX330
  - Turbo decoding:
    50 Mbps @ 6 iterations
  - With 1,5 cycle/symbol (a new ASIP ver)
    120 Mbps @ 6 itrations



  - Target ASIC: 1Gbps with 20-ASIP architecture

References

- **Research contributions from my past and current supervised PhD students:**
  - Olivier Muller
  - Atif Raza Jafri
  - Hazem Moussa

- **Selected reference publications:**

1. O. Muller, A. Baghdadi, M. Jezequel, "From Parallelism Levels to a Multi-ASIP Architecture for Turbo Decoding", IEEE TVLSI, Vol. 17, Issue: 1, Page(s): 92-102, Jan. 2009.
2. O. Muller, A. Baghdadi, and M. Jezequel, "From application to ASIP-based FPGA prototype : a case study on turbo decoding", 19th IEEE/IFIP international symposium on Rapid System Prototyping, June 2008.
3. H. Moussa, O. Muller, A. Baghdadi, M. Jézéquel, "Butterfly and Benes-Based on-Chip Communication Networks for Multiprocessor Turbo Decoding", DATE'07, April 2007.
4. O. Muller, A. Baghdadi, and M. Jezequel, "On the Parallelism of Convolutional Turbo Decoding and Interleaving Interference", IEEE Global Telecommunications Conference (GLOBECOM 06), November 2006.

Conclusions

- **Flexibility requirements in channel decoding and turbo decoding applications**
- **Turbo decoding and parallelism levels**
- **Detailed architecture of a typical ASIP for flexible turbo decoding**
  - Illustration of decoding process, pipeline, and resources utilization throughout a detailed application example
  - ASIP design and verification flow
- **NoC-based multi-ASIP turbo decoding**

Perspectives – on going work…

- **Towards an innovative universal channel decoder architecture**
  - Unifying flexibility-oriented and optimization-oriented approaches

- **UDEC Project (Universal channel DECoder)**

  (http://recherche.telecom-bretagne.eu/udec)

TELECOM Bretagne

cea leti

THALES

ANR – Architecture du futur
2008-2011

Agence Nationale de la Recherche ANR