12th Workshop on Electronics for LHC and Future Experiments
25-29 September 2006, Valencia

An Error-Correcting Line Code for a HEP Rad-Hard Multi-GigaBit Optical Link

Giulia Papotti

Universita' degli Studi di Parma

CERN (European Organization for Nuclear Research)

Outline

- motivation
  - TTC and GBT
- coding requirements for optical links
  - radiation environments
- line code architecture
  - main building blocks
  - two options
- properties of the presented line code
  - error correction capability
  - implementation complexity
  - DC-wander
- demonstrator ASIC implementation

TTC system

- distribution at the LHC of:
  - Timing: LHC clock
    - need clean clock
  - L1 Trigger
    - need low latency
    - only 1 bit information
  - “slow” Control
    - 1 bit per LHC clock, to build longer frames
  - total: 2 bit per bunch crossing = 80Mb/s

Gigabit Bidirectional Transceiver

- upgrade of the TTC link for SLHC
  - still need low latency and precise timing
  - bidirectional
  - broadcast and point-to-point
  - recent tech allows higher line speed
    - more bits per bunch crossing
      - more than only 1 bit for L1T
      - slow control can now be “fast”
      - protect trigger information with EC schemes

giulia.papotti@cern.ch

Line code requirements

- sufficient timing information for clock recovery (bit lock)
  - ~20ps jitter required
- dc-free line bit stream for AC coupling
  - statistically same number of 1s and 0s on the line
- frame synchronization (frame lock)
  - framed stream for bunch crossing information
- non periodic stream
  - low pattern dependent jitter

- high efficiency / little redundancy
  - to keep noise bandwidth and circuitry rate minimum

Optical links in rad environment

- ASIC hardening
  - Total Dose and Single Event Effects
  - layout (i.e. ELT) and circuit (i.e. majority voting) techniques
- Optical components
  - increased power budget
  - photodiodes detect high energy particles as light
    - radiation particle creates a current by ionization
    - if current high enough, detect a false signal

Line code requirements (2)

- sufficient timing information
- dc-free line bit stream
- frame synchronization (frame lock)
- non periodic stream
- high efficiency / little redundancy

- error-correction capability
  - target: photodiode SEU

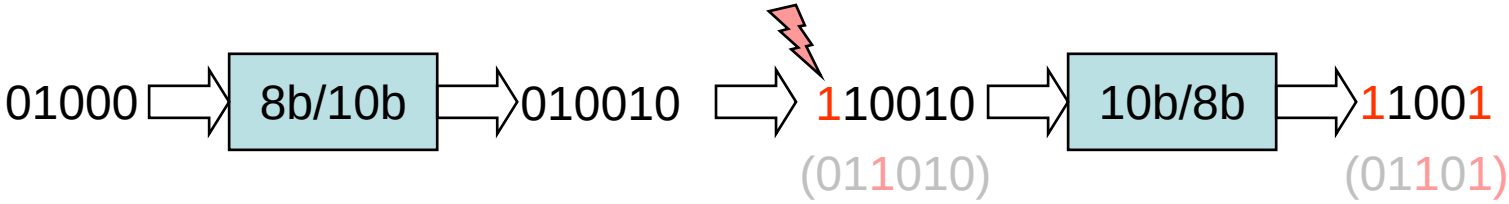
- low latency

Existing line codes

- scrambling
  - randomizes data with no bandwidth increase
- CIMT
  - Conditional Inversion + one Master Transition
- 8b/10b
  - mapping that guarantees minimum number of transitions and DC-balance
- 64b/66b
  - lower overhead needed for 10Gb Ethernet
  - scrambler + fixed transition for frame lock

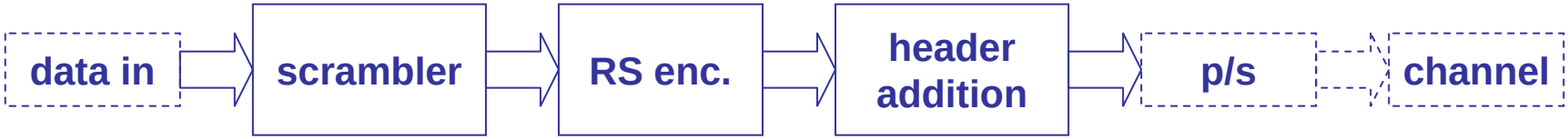
Error correction and line codes

- this leads to error multiplication
  - need more complicated EC algorithms
    - more time consuming
    - ex: error multiplication with 8b/10b



Proposed code block scheme

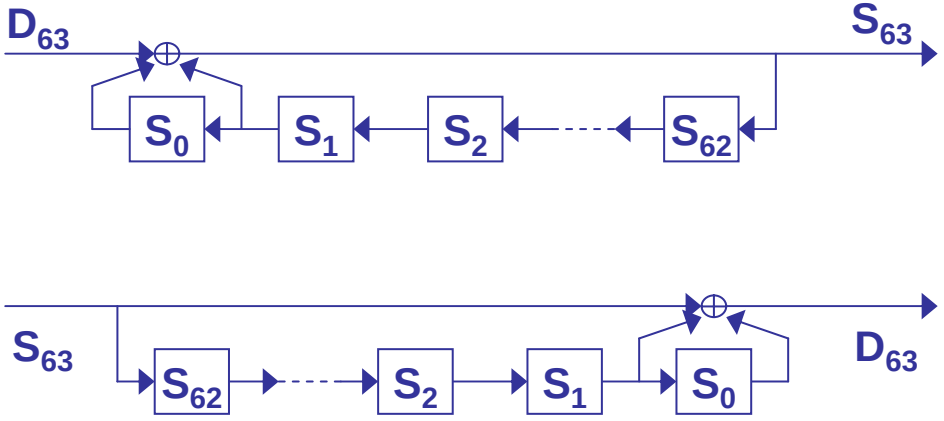
- line coding external to RS
  - to avoid error multiplication



- two options
  - 64-bit data word converted in 88-bit line word
    - code efficiency: ~73%, line speed: ~3.52 Gbps (@ 40 MHz)
  - 60-bit data word converted in 90-bit line word
    - code efficiency: ~67%, line speed: ~3.60 Gbps (@ 40 MHz)
    - lower efficiency but higher error correction capability

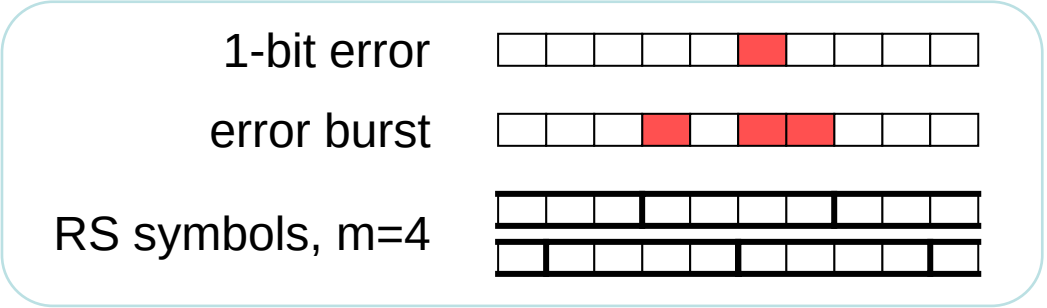
Scrambler

- randomizes the data with no bandwidth increase
- Linear Feedback Shift Register
  - long LFSR: better randomization
- self-synchronizing
  - broadcast link!



Reed-Solomon EC - 1

- family of linear cyclic block codes
  - very good efficiency
  - very good modularity/scalability
- feature: treat groups of m bits as single entities
  - correct burst errors
    - used in cds and space communication
  - m-parallel structure

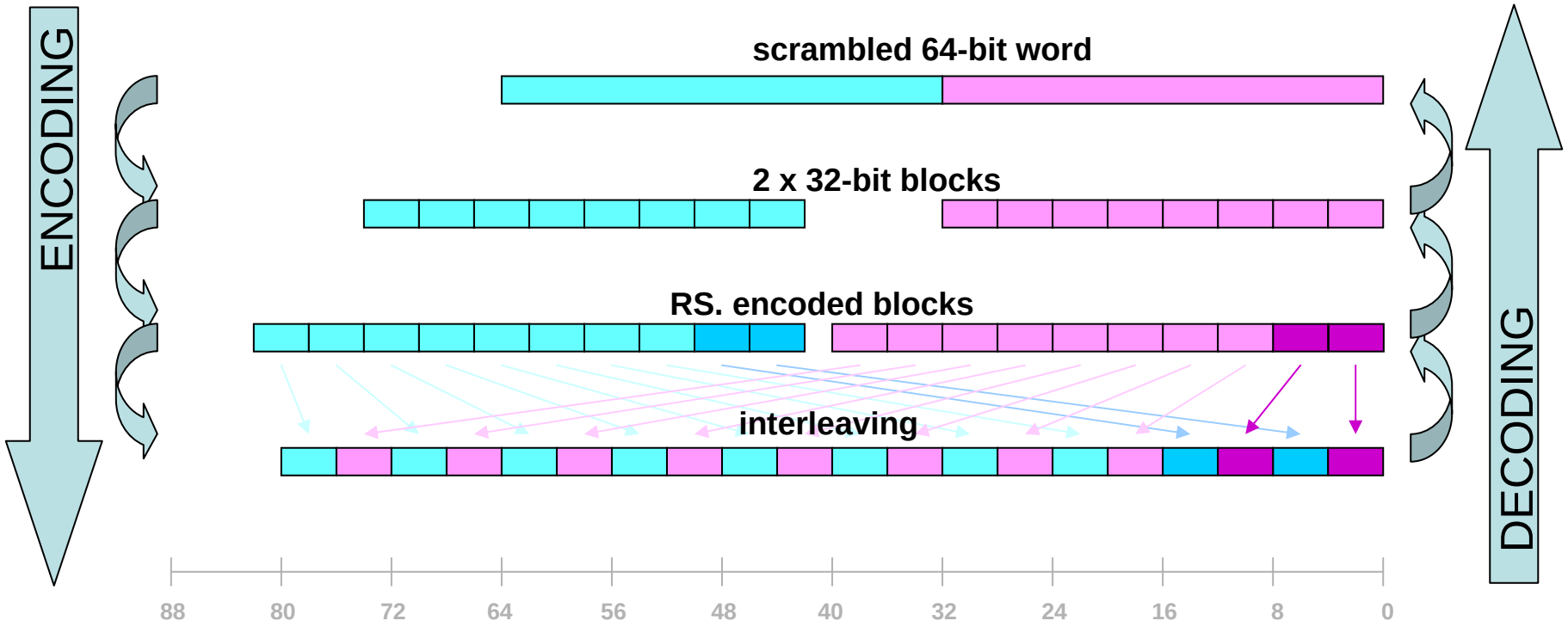


Reed-Solomon EC - 2

- encoding done via LFSR (cyclic code)
  - performed in 1 T_{FRAME}
- decoding: 4 “steps”
  - complexity growing with number of correctable errors
  - correct only one error
    - latency issues
  - with interleaving
    - for extended EC capability
  - performed in 2 T_{FRAME}

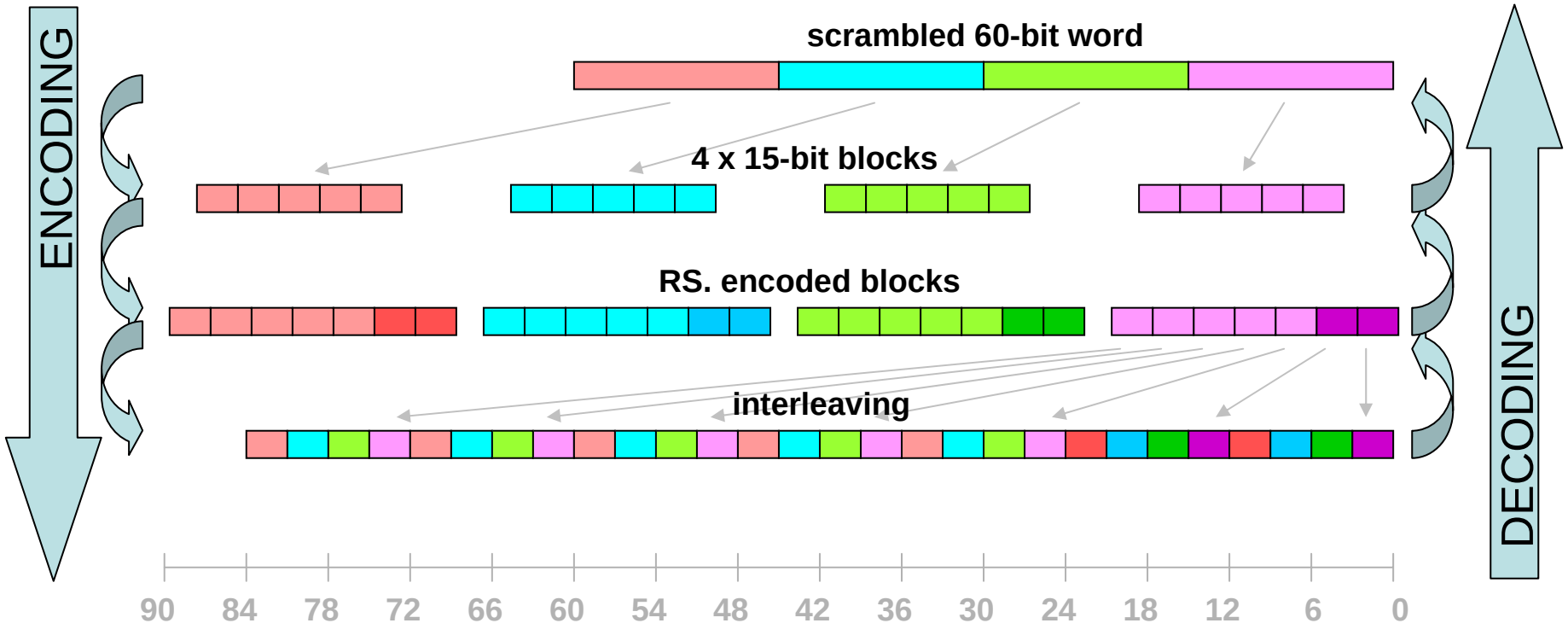
giulia.papotti@cern.ch

“m=4” option



giulia.papotti@cern.ch

“m=3” option



giulia.papotti@cern.ch

Summary of the two options

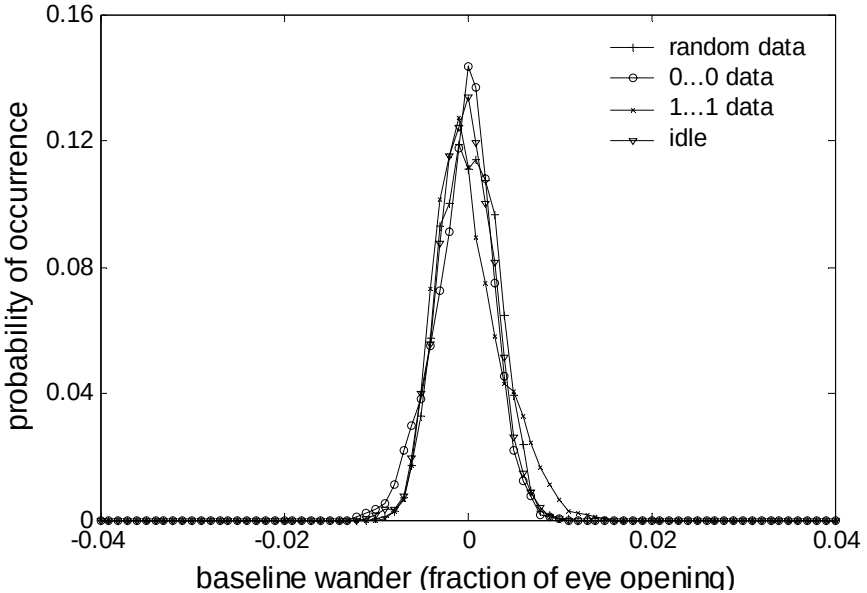
code char.	“m=4” option	“m=3” option
L interleaved bl.	**2**	**4**
N_s	10	7
K_s	8	5
$N_b = N_s L m$	80	84
$K_b = K_s L m$	**64**	**60**
scr. order n	63	60
H	8	6
$N_{tot} = N_b + H$	**88**	**90**
eff.= K_b / N_{tot}	**~73%**	**~67%**
Err.Corr.	**1+1/2**	**1+3/4**

- “m=4” option: higher efficiency
- “m=3” option: higher error correction capability

giulia.papotti@cern.ch

Simulation results

- average run-length about 3 bits
- verification of DC-balance
  - not compromised by the fact that scrambling is performed before RS encoding



Implementation complexity

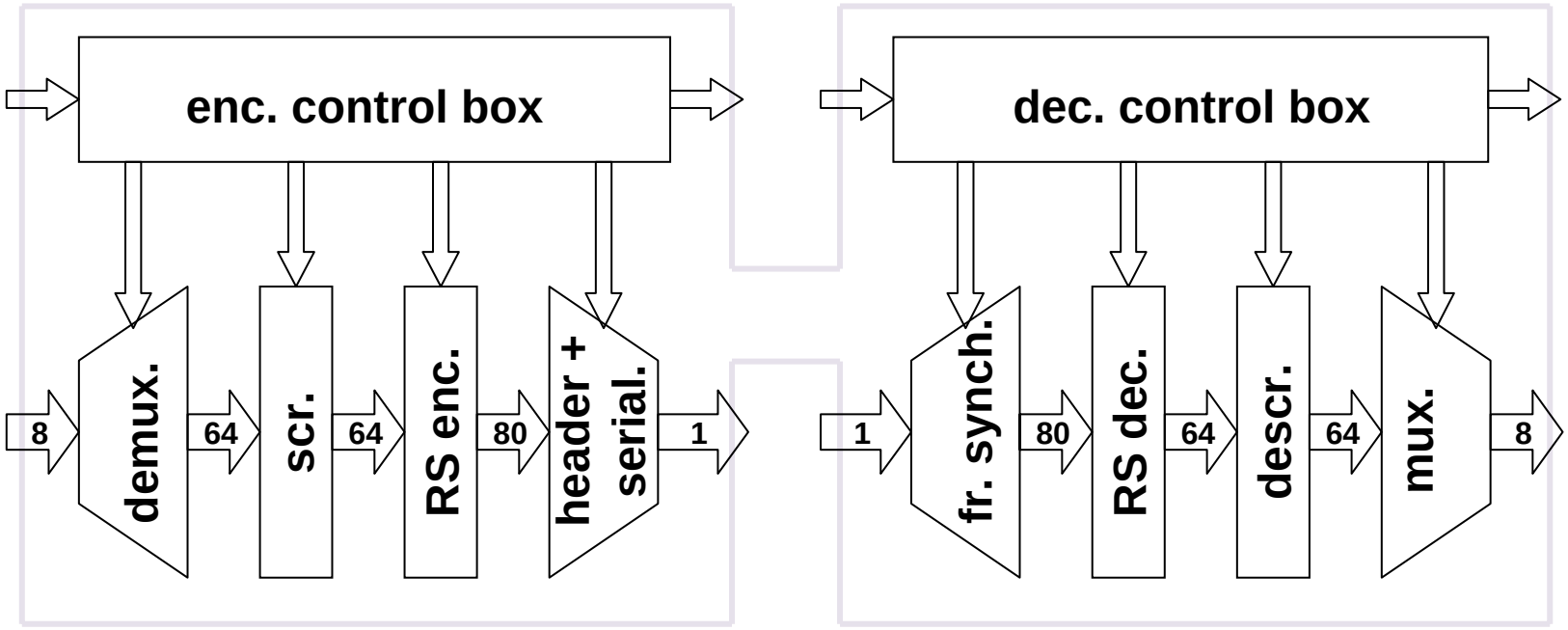
– for the .13um tech and Artisan digital library

Module	Cell count	Power cons. (dyn)
1st option enc.	**1066**	**5.2 mW**
2nd option enc.	**1098**	**6.4 mW**
1st option dec.	**2794**	**10.6 mW**
2nd option dec.	**2402**	**13.3 mW**

– second option slightly more “expensive”

– double error correction without interleaving
  - 3x cells and power
  - and worse >2x as much time for decoding

Line code demonstrator ASIC

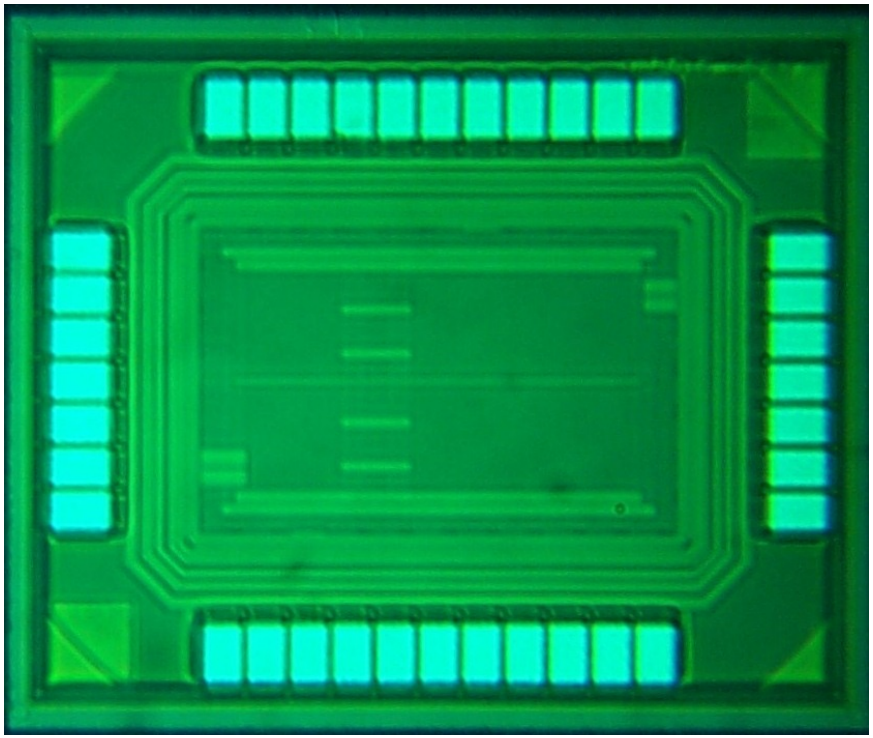


- “m=4” option implemented
- encoder and decoder can be tested separately or back-to-back
- limitation about number of pads required multiplexing and demultiplexing

giulia.papotti@cern.ch

ASIC implementation details

- 0.13μm technology
- ARTISAN standard cell library
- area: 1mm x 1.3mm
- encoder:
  - ~1700 cells
  - ~25mW/GHz
- decoder:
  - ~5000 cells
  - ~50mW/GHz



- successfully tested

giulia.papotti@cern.ch

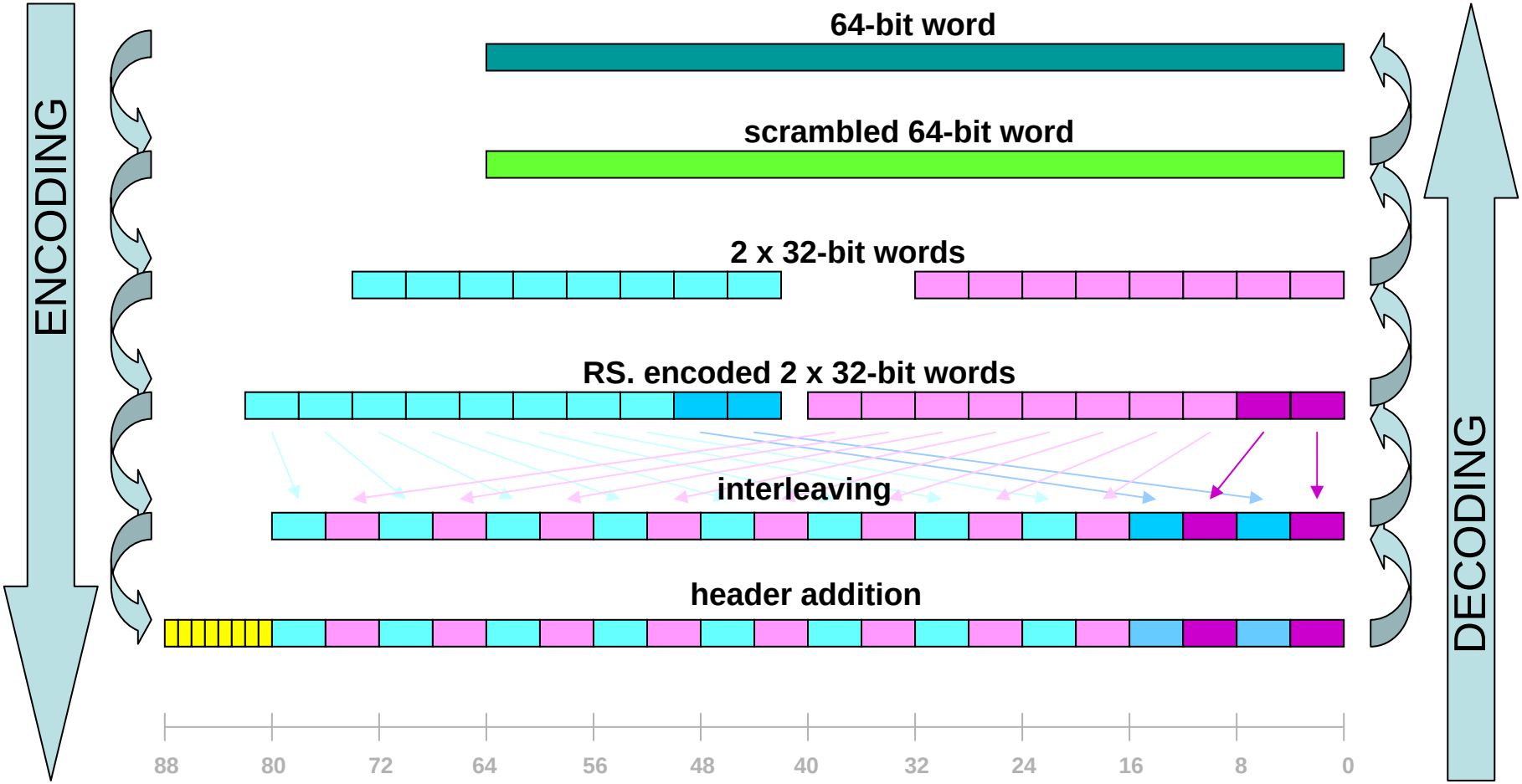
Summary

- line code requirements
- proposed line code blocks
  - scrambler + RS error correction + header addition
  - 2 options
    - 73% efficiency for (1+1/2) error correction capability
    - 67% efficiency for (1+3/4) error correction capability
- code simulation results
  - average run-length < 3 bits
  - DC-wander well within 1% of eye opening
- ASIC implementation
  - fully digital chip was produced and successfully tested

Header addition

- governs the frame locking mechanism
  - repeated header recognition allows locking
  - repeated wrong header causes out-of-lock
- need to distinguish three block types
  - “data”, “idle”, or “trigger”
  - done through different header patterns
  - SEU tolerant and DC-balanced
  - 5 bits minimum

visually (1st option)



giulia.papotti@cern.ch

visually (2nd option)