Karlsruhe Institute of Technology

Institut für Technik der Informationsverarbeitung

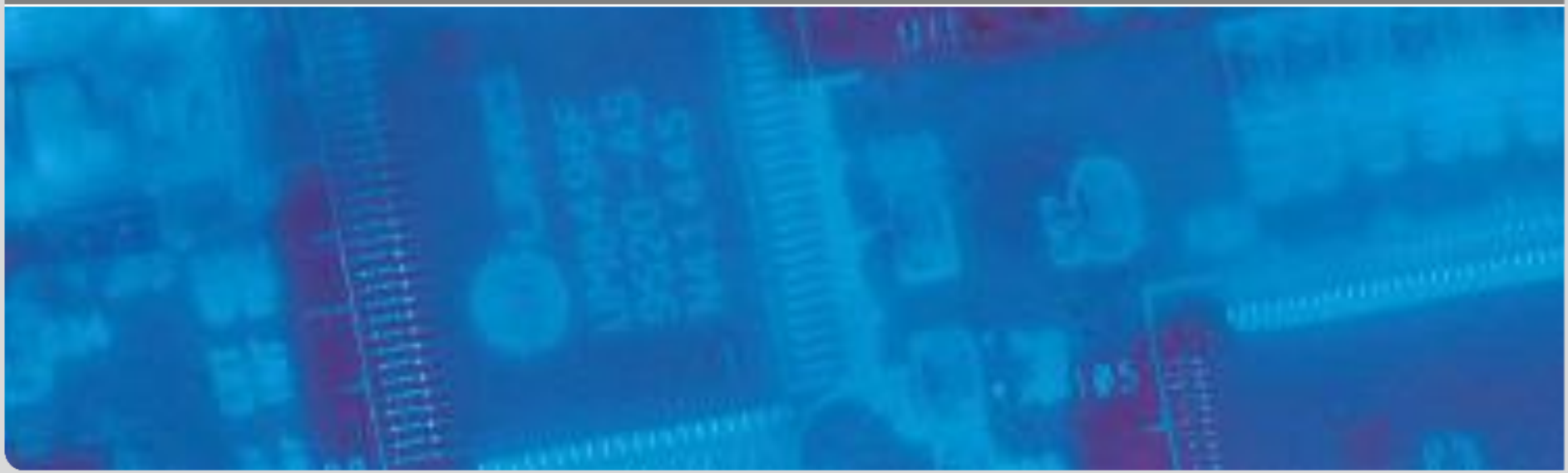
Institut für Prozessdatenverarbeitung und Elektronik - IPE

FPGA-Based Architecture for Pattern Recognition

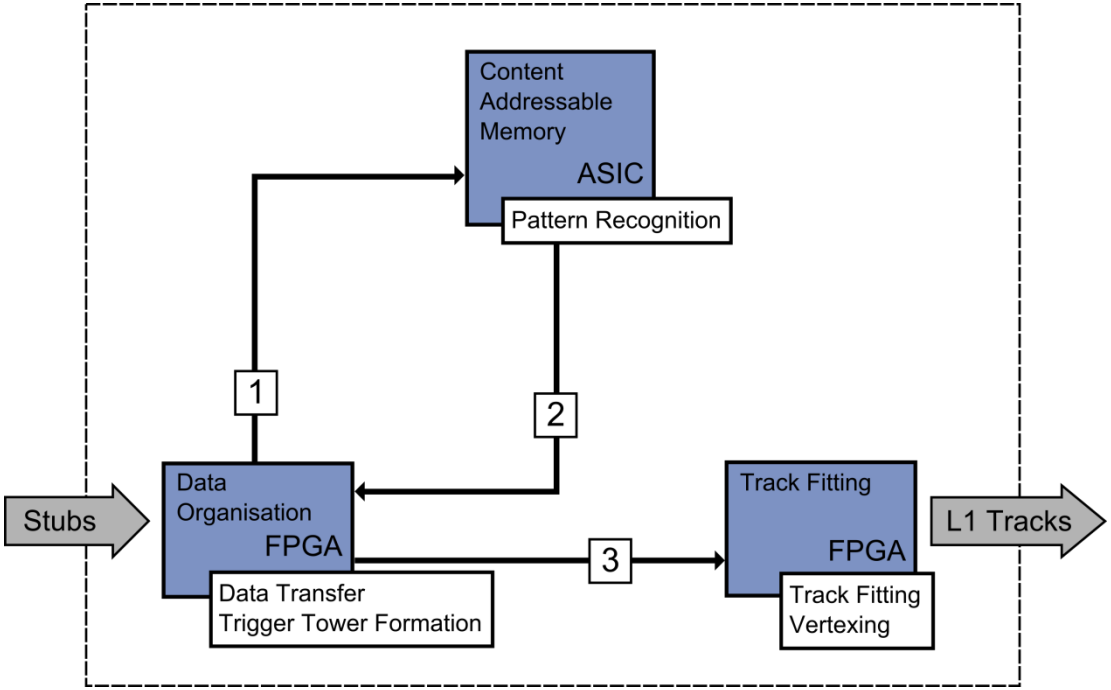
Tanja Harbaum, Thomas Schuh

Institut für Technik der Informationsverarbeitung (ITIV)

Institut für Prozessdatenverarbeitung und Elektronik (IPE)



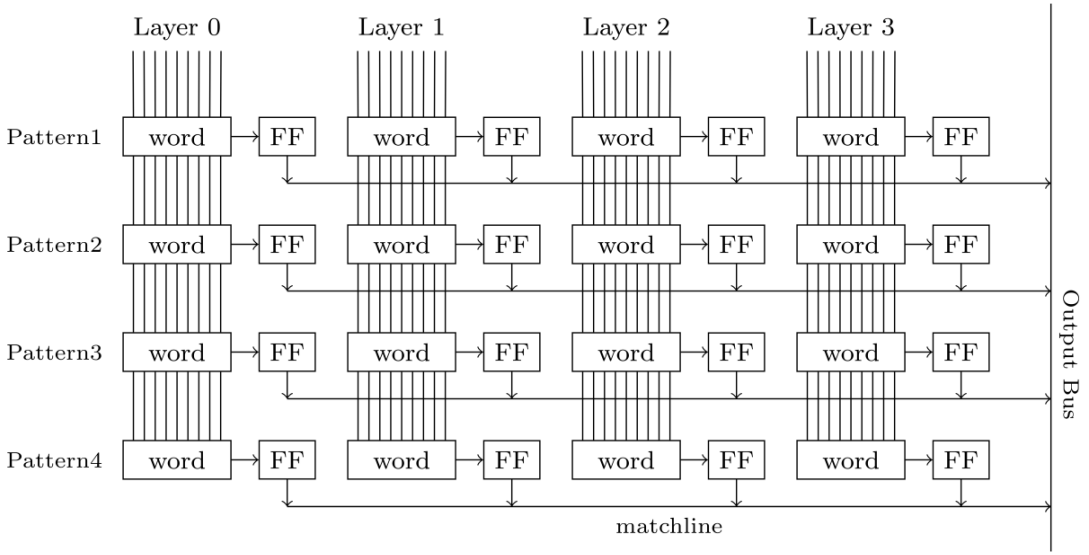
FPGA Approach – Motivation

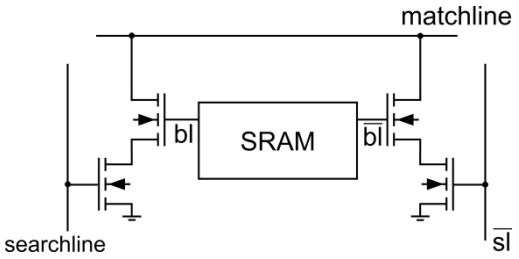


- Merge FPGAs
- Flexible design

FPGA Approach – Starting Point

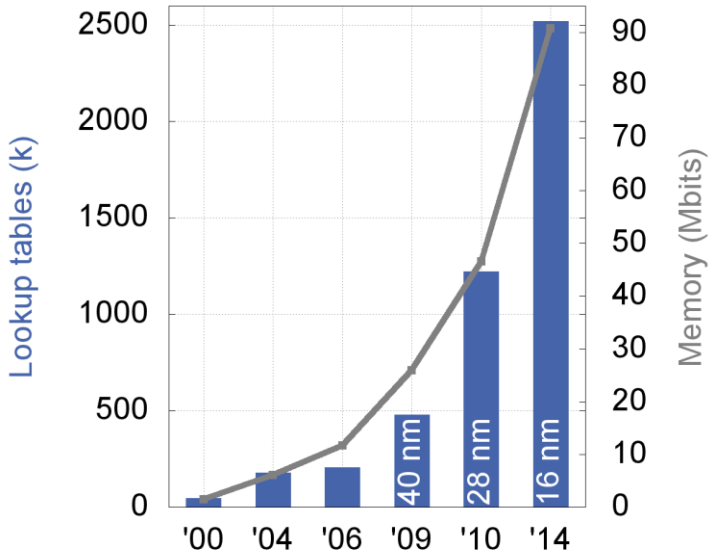
- Pattern Recognition - AM Chip
- Content-addressable memory





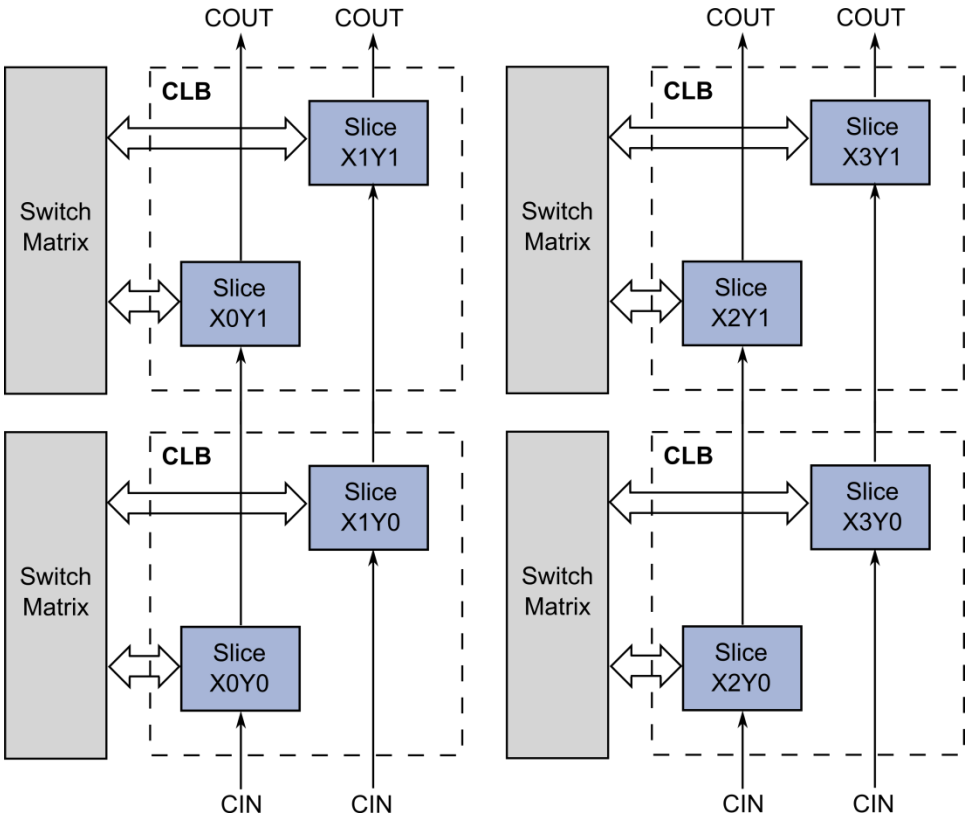
Structure of a FPGA - Resources

- Configurable Logic Blocks (CLBs)
  - Major resource
  - Includes Lookup tables (LUTs)
  - Implements logic
- Block RAM (B-RAM)
  - Scarce resource
  - Implements memory

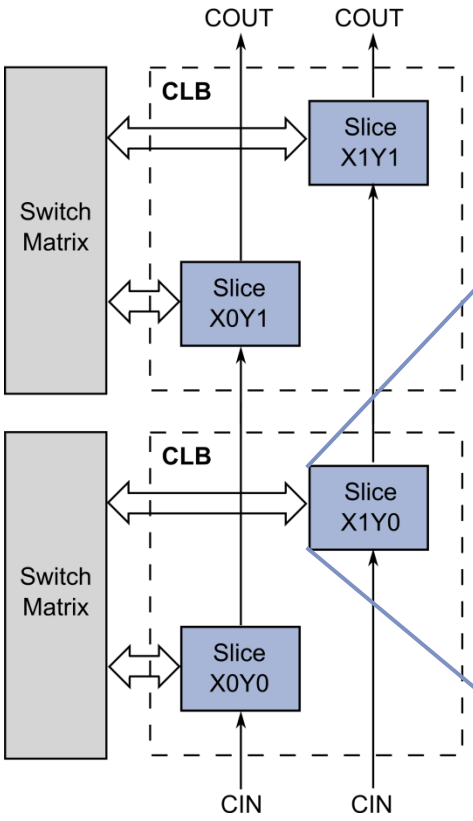


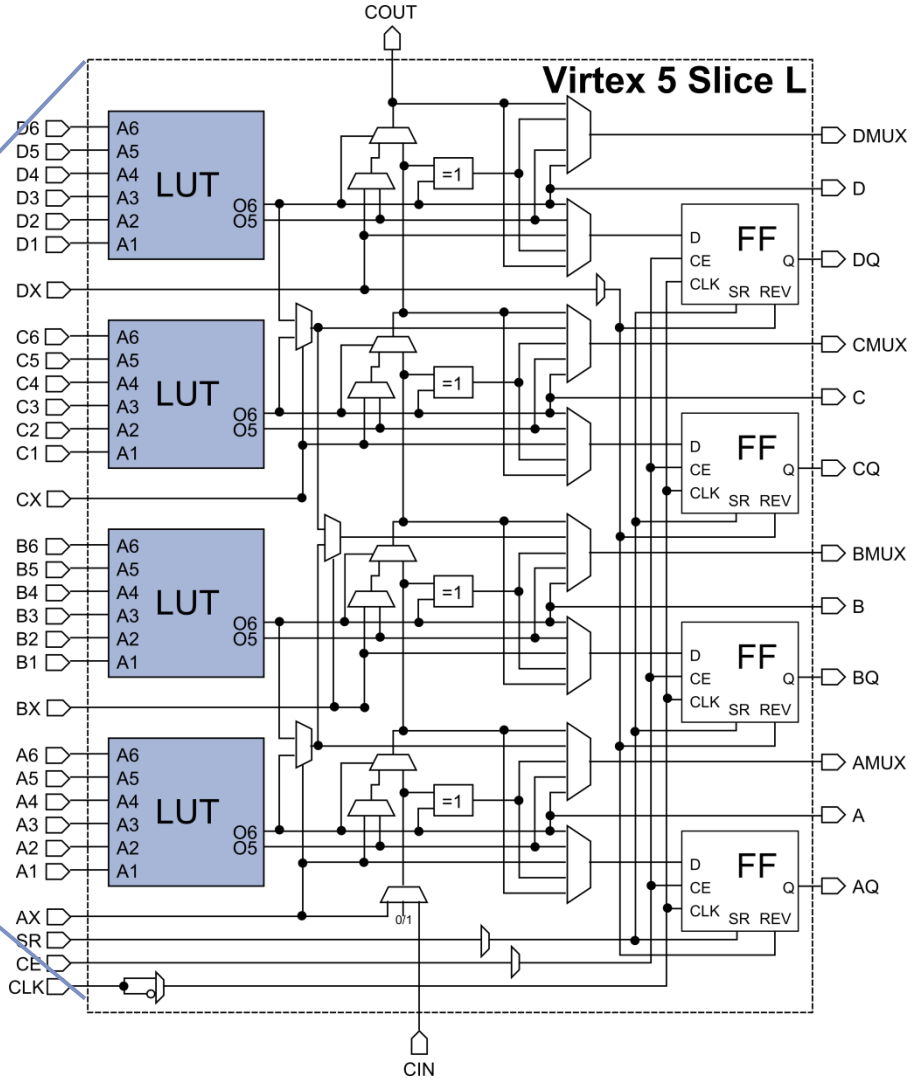
➡ Implement logic and not memory!

Structure of an FPGA – Configurable Logic Block



Structure of an FPGA - Slice

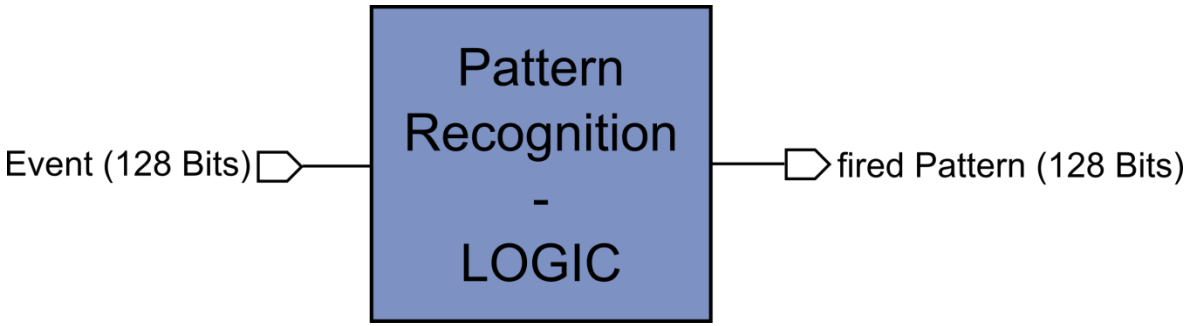




- 6 entries per LUT

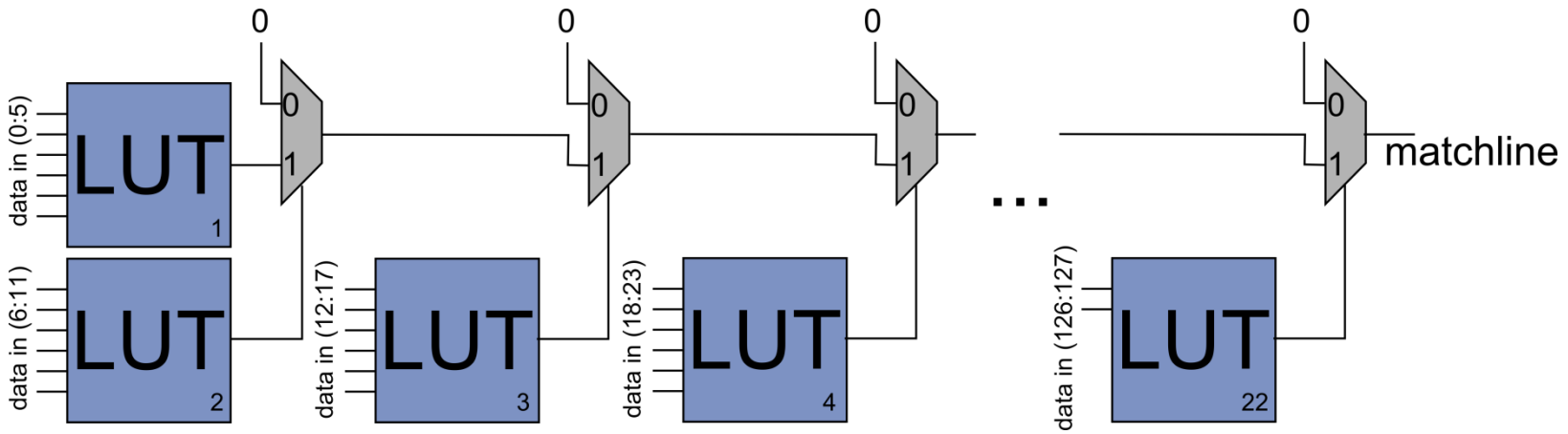
The Ideas behind an FPGA Approach

- Treat the Pattern Recognition as “logic unit”



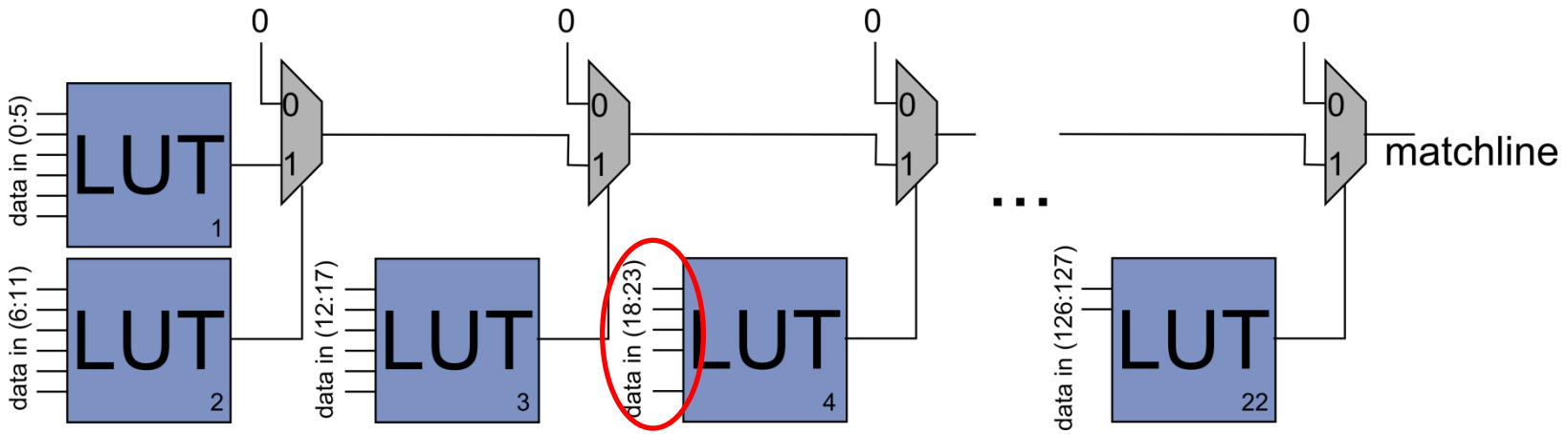
- 128 Bits incoming
  - ➡ 22 LUTs minimum (6 entries per LUT)
  - ➡ Multiplexors

FPGA Approach – Minimization by Logic



- LUT structure for one Pattern
  - 22 LUTs
  - Pure Combinatorial – no clock cycle
  - Plus one 128 bit register to store the input

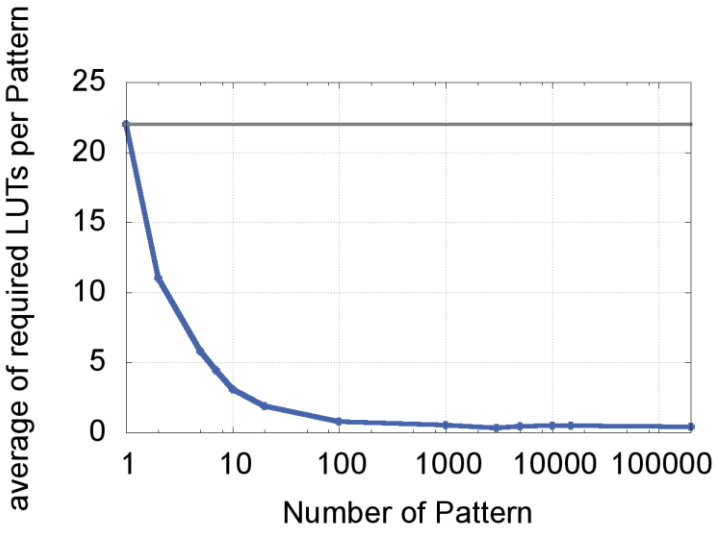
FPGA Approach – Minimization by Logic

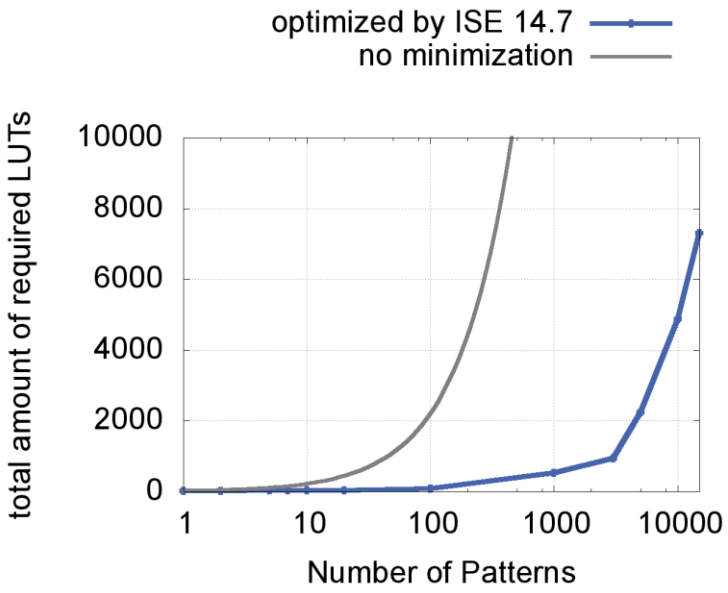


- LUT structure for two Patterns
  - One bit differences
  - 22 LUTs
  - Pure Combinatorial – no clock cycle
  - Plus one 128 bit register to store the input

➡ decreased by a factor of 2

FPGA Approach – First Results

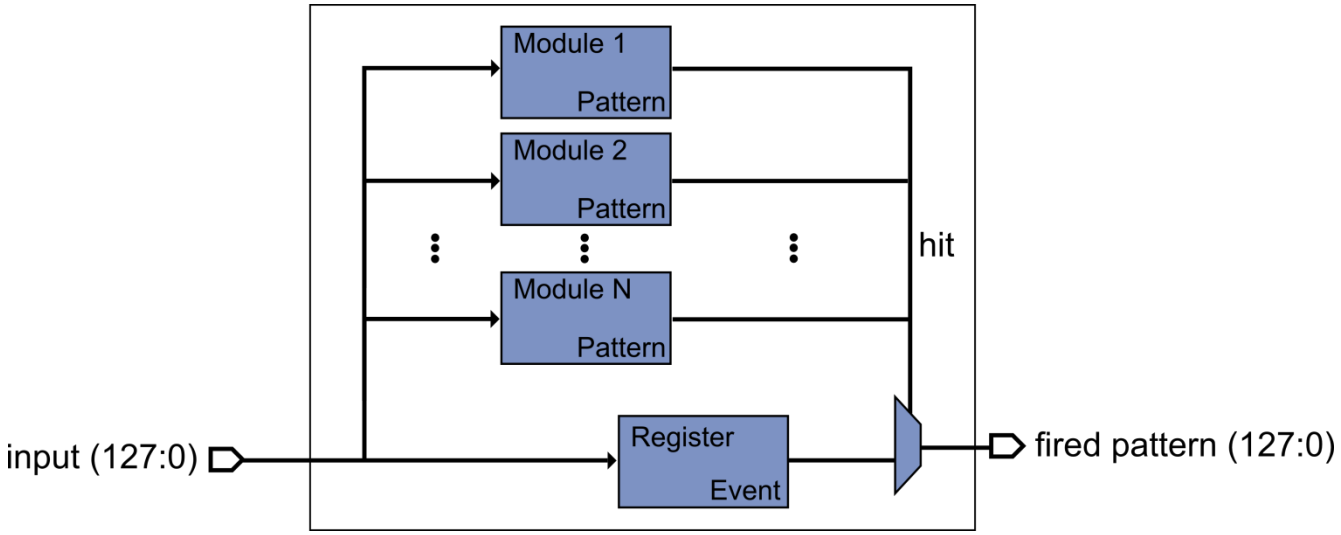




- Gain saturates – 0.5 LUT per Pattern in average
- Modularization without significant overhead
- Depends on the composition of the Pattern Bank

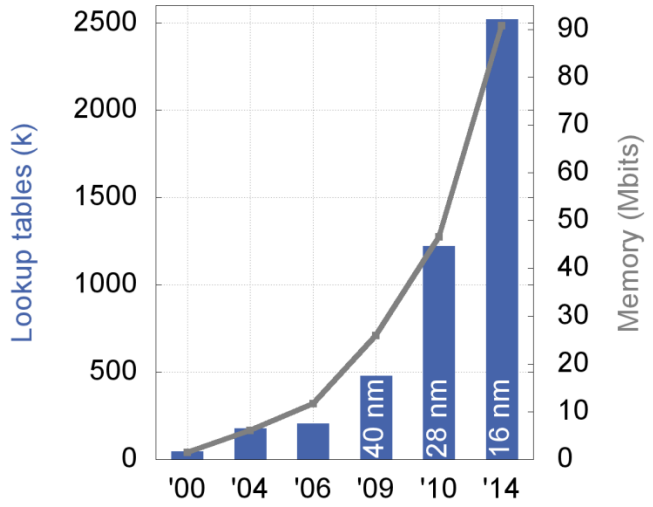
➡ Extensive minimization by logic is possible

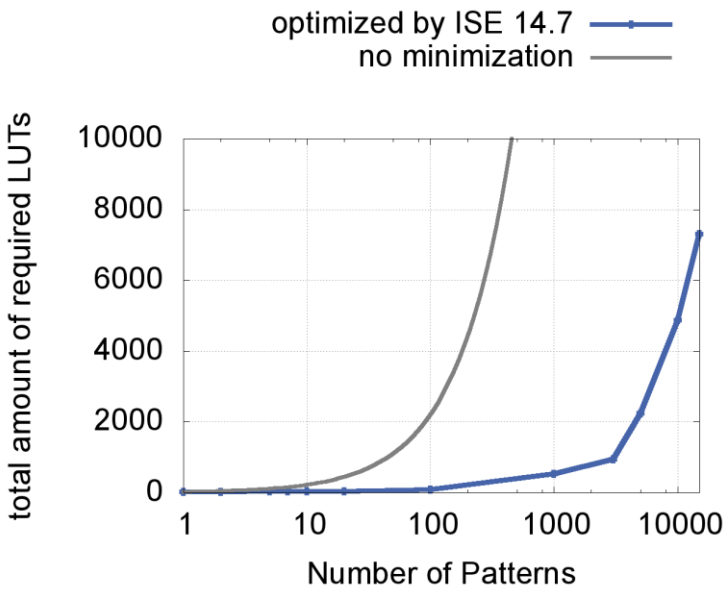
FPGA Approach – Parallel Structure



- N Modules
  - Parallel comparison
- First Results
  - 10.000 Pattern per module
  - 20 modules → 200.000 Patterns
  - Latency 8ns (Xilinx Virtex 7)

FPGA Approach – Outlook



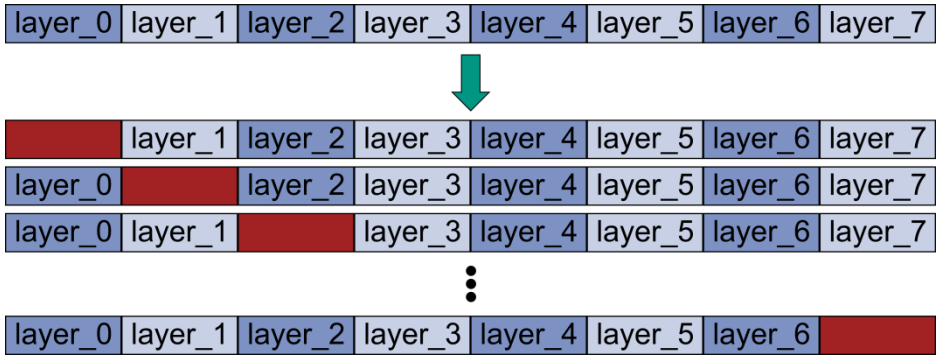


- Number of LUT per FPGA are increased
  - Around factor five in the last seven years
  - Extra devices added (e.g. multiplexors)
  - 5M Patterns per FPGA imaginable

Comparison to AM Chips

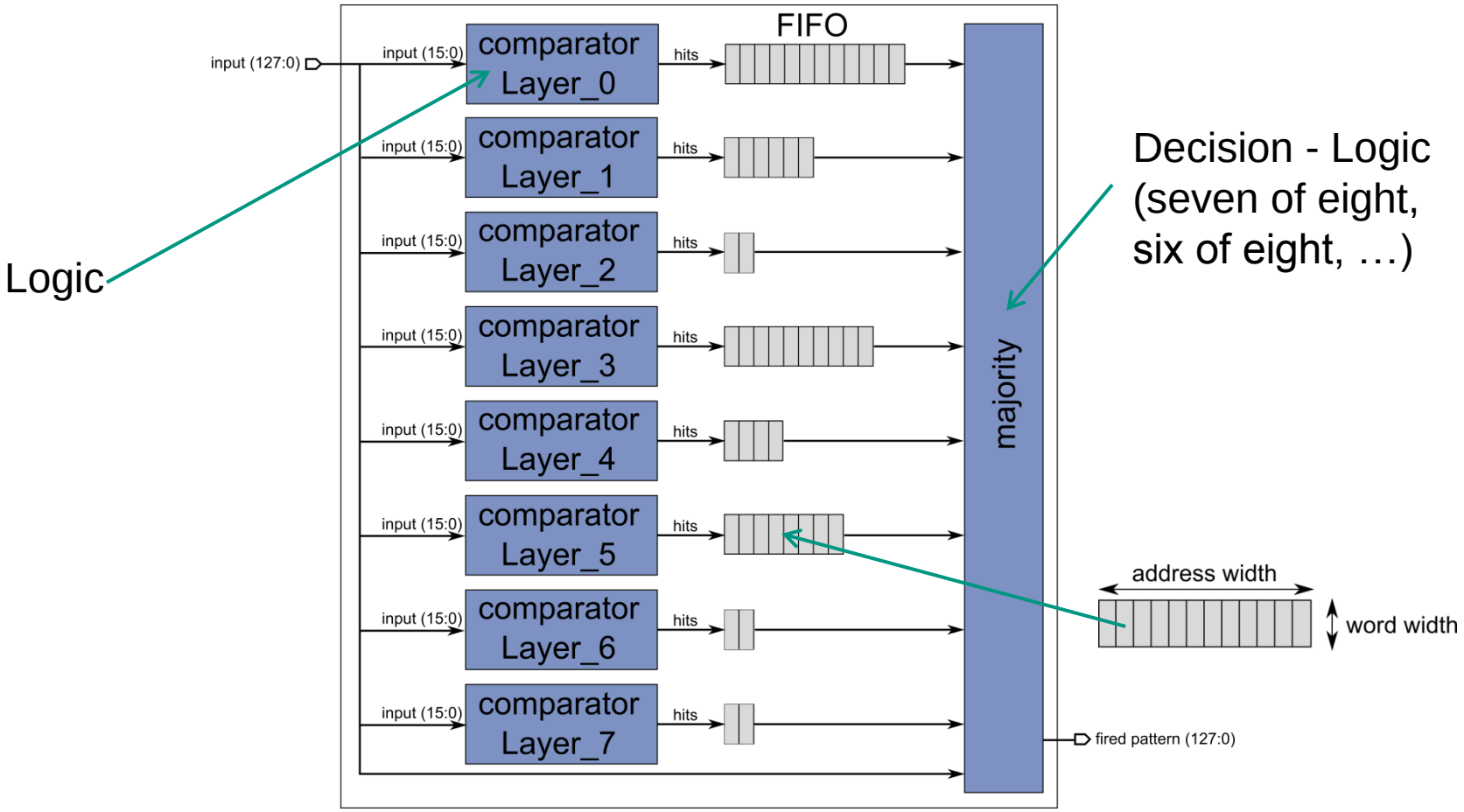
- AM-chip offers more possibilities
- Writable memory
  - ➡ Synthesize the FPGA
- Handle failure layers
  - ➡ Split the patterns into layers (128 Bits → 8*16 Bits)

1 Multiply Patterns



2 Layer Based Approach

Layer Based Approach

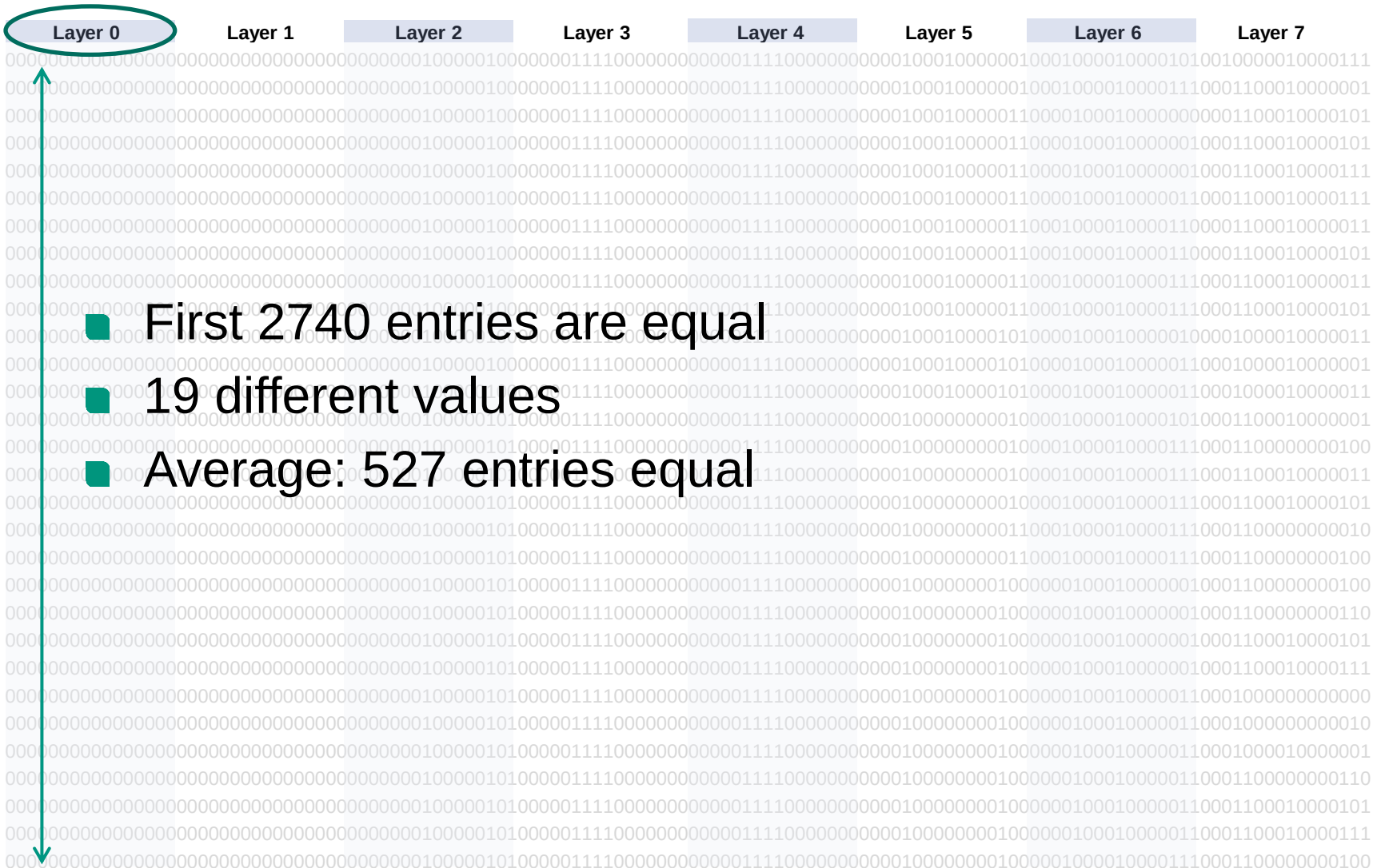


Analyze the Pattern Bank

Pattern Bank

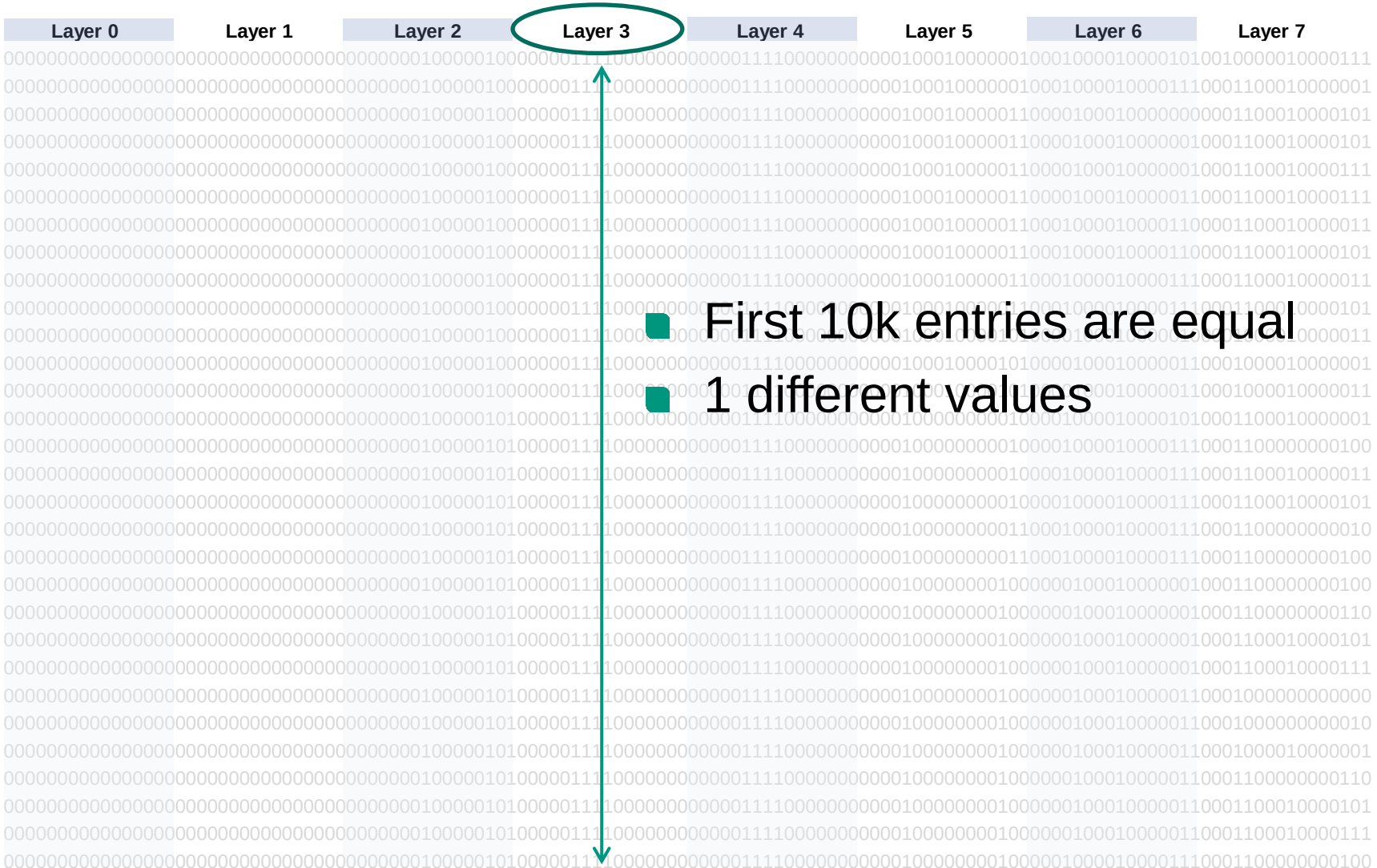
Layer 0	Layer 1	Layer 2	Layer 3	Layer 4	Layer 5	Layer 6	Layer 7
0000000000000000	0000000000000000	0000000100000100	0000001111000000	0000011110000000	0001000100000100	0100010000101001	0000010000010111
0000000000000000	0000000000000000	0000000100000100	0000001111000000	0000011110000000	0001000100000100	0100011100011000	0100000100000001
0000000000000000	0000000000000000	0000000100000100	0000001111000000	0000011110000000	0001000100000110	0000100010000000	0001100010000101
0000000000000000	0000000000000000	0000000100000100	0000001111000000	0000011110000000	0001000100000110	0000100010000001	0001100010000101
0000000000000000	0000000000000000	0000000100000100	0000001111000000	0000011110000000	0001000100000110	0000100010000001	0001100010000111
0000000000000000	0000000000000000	0000000100000100	0000001111000000	0000011110000000	0001000100000110	0000100010000011	0001100010000111
0000000000000000	0000000000000000	0000000100000100	0000001111000000	0000011110000000	0001000100000110	0000100010000011	0000100010000011
0000000000000000	0000000000000000	0000000100000100	0000001111000000	0000011110000000	0001000100000110	0000100010000011	0001100010000101
0000000000000000	0000000000000000	0000000100000100	0000001111000000	0000011110000000	0001000100000110	0000100010000011	0001100010000011
0000000000000000	0000000000000000	0000000100000100	0000001111000000	0000011110000000	0001000100000110	0000100010000011	0001100010000101
0000000000000000	0000000000000000	0000000100000100	0000001111000000	0000011110000000	0001000100001010	0000100010000010	0000100010000011
0000000000000000	0000000000000000	0000000100000100	0000001111000000	0000011110000000	0001000100001010	0000100010000011	0000100010000001
0000000000000000	0000000000000000	0000000100000100	0000001111000000	0000011110000000	0001000100001010	0000100010000011	0000100010000011
0000000000000000	0000000000000000	0000000101000001	0000001111000000	0000011110000000	0100000000010000	1000010000101000	0001100010000001
0000000000000000	0000000000000000	0000000101000001	0000001111000000	0000011110000000	0100000000010000	1000010000111000	0001100000000100
0000000000000000	0000000000000000	0000000101000001	0000001111000000	0000011110000000	0100000000010000	1000010000111000	0001100010000011
0000000000000000	0000000000000000	0000000101000001	0000001111000000	0000011110000000	0100000000010000	1000010000111000	0001100010000101
0000000000000000	0000000000000000	0000000101000001	0000001111000000	0000011110000000	0100000000011000	1000010000111000	0001100000000010
0000000000000000	0000000000000000	0000000101000001	0000001111000000	0000011110000000	0100000000011000	1000010000111000	0001100000000100
0000000000000000	0000000000000000	0000000101000001	0000001111000000	0000011110000000	0100000001000000	1000010000010000	0001100000000100
0000000000000000	0000000000000000	0000000101000001	0000001111000000	0000011110000000	0100000001000000	1000010000010000	0001100000000110
0000000000000000	0000000000000000	0000000101000001	0000001111000000	0000011110000000	0100000001000000	1000010000010000	0001100010000101
0000000000000000	0000000000000000	0000000101000001	0000001111000000	0000011110000000	0100000001000000	1000010000010000	0001100010000111
0000000000000000	0000000000000000	0000000101000001	0000001111000000	0000011110000000	0100000001000000	1000010000011000	0001000000000000
0000000000000000	0000000000000000	0000000101000001	0000001111000000	0000011110000000	0100000001000000	1000010000011000	0001000000000010
0000000000000000	0000000000000000	0000000101000001	0000001111000000	0000011110000000	0100000001000000	1000010000011000	0001000010000001
0000000000000000	0000000000000000	0000000101000001	0000001111000000	0000011110000000	0100000001000000	1000010000011000	0001100000000110
0000000000000000	0000000000000000	0000000101000001	0000001111000000	0000011110000000	0100000001000000	1000010000011000	0001100010000101
0000000000000000	0000000000000000	0000000101000001	0000001111000000	0000011110000000	0100000001000000	1000010000011000	0001100010000111
0000000000000000	0000000000000000	0000000101000001	0000001111000000	0000011110000000	0100000001000000	1000010000011000	0001100000000100

Pattern Bank – Analysis 10k Pattern



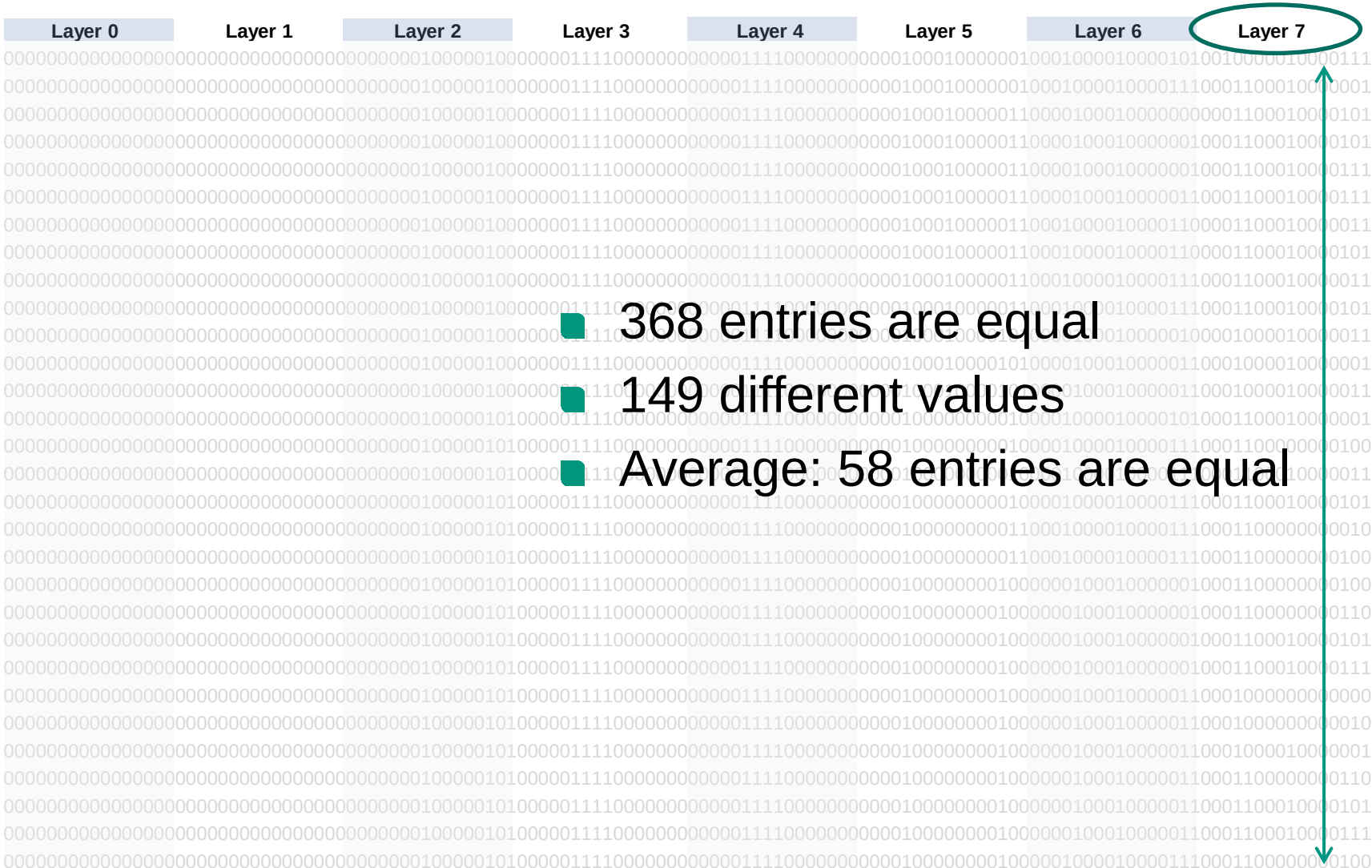
- First 2740 entries are equal
- 19 different values
- Average: 527 entries equal

Pattern Bank – Analysis 10k Pattern



- First 10k entries are equal
- 1 different values

Pattern Bank – Analysis 10k Pattern



- 368 entries are equal
- 149 different values
- Average: 58 entries are equal

Conclusion

- First results looks promising
- Analyses the whole Pattern Bank
  - Maybe re-order the Pattern
- Carry out more test runs
  - Synthesize the Layer Based Approach

- Do not hesitate to contact us!
  - harbaum@kit.edu or thomas.schuh@kit.edu