Granularity Exploration for Logic in Memory

I. O'Connor, M. Cantan, L. Mozzone
C. Marchand, A. Bosio, D. Deleruyelle
Lyon Institute of Nanotechnology – University of Lyon
ian.oconnor@ec-lyon.fr

3ε FERRO

Phase Change Workshop
Villars sur Ollon (CH)
14th January 2020

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 780302.

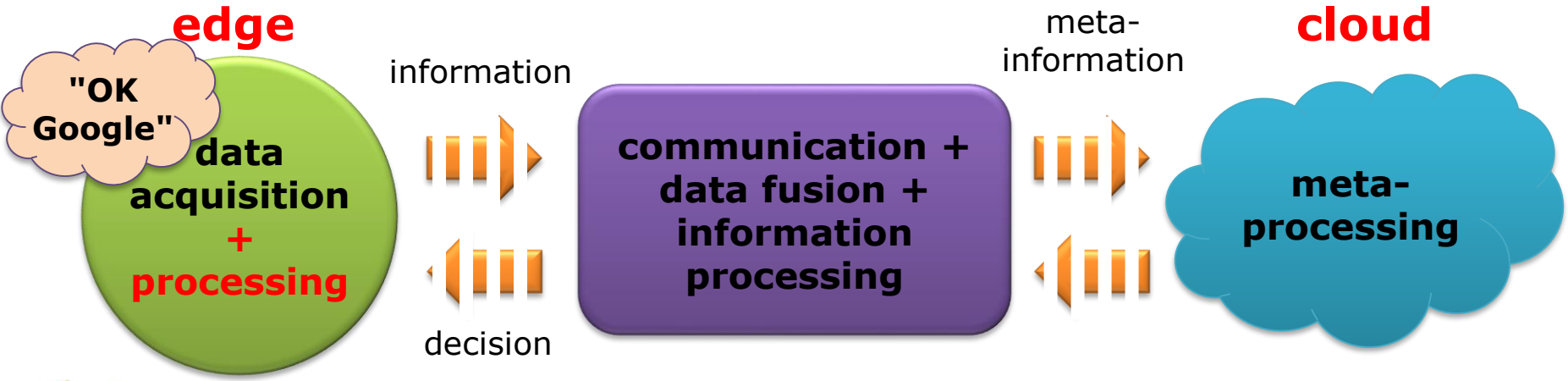
Tomorrow's IoT

- Ever-smarter "things" capable of specific (limited) processing on data to
  - extract and transmit meaningful information to computing resources in the cloud
  - make fast, location-aware and secure decisions

Pros
Low latency, low delay jitter
Location awareness
Low vulnerability
Lower communication data-rate

Cons:
Constrained energy

Need highly energy-efficient embedded computing platform



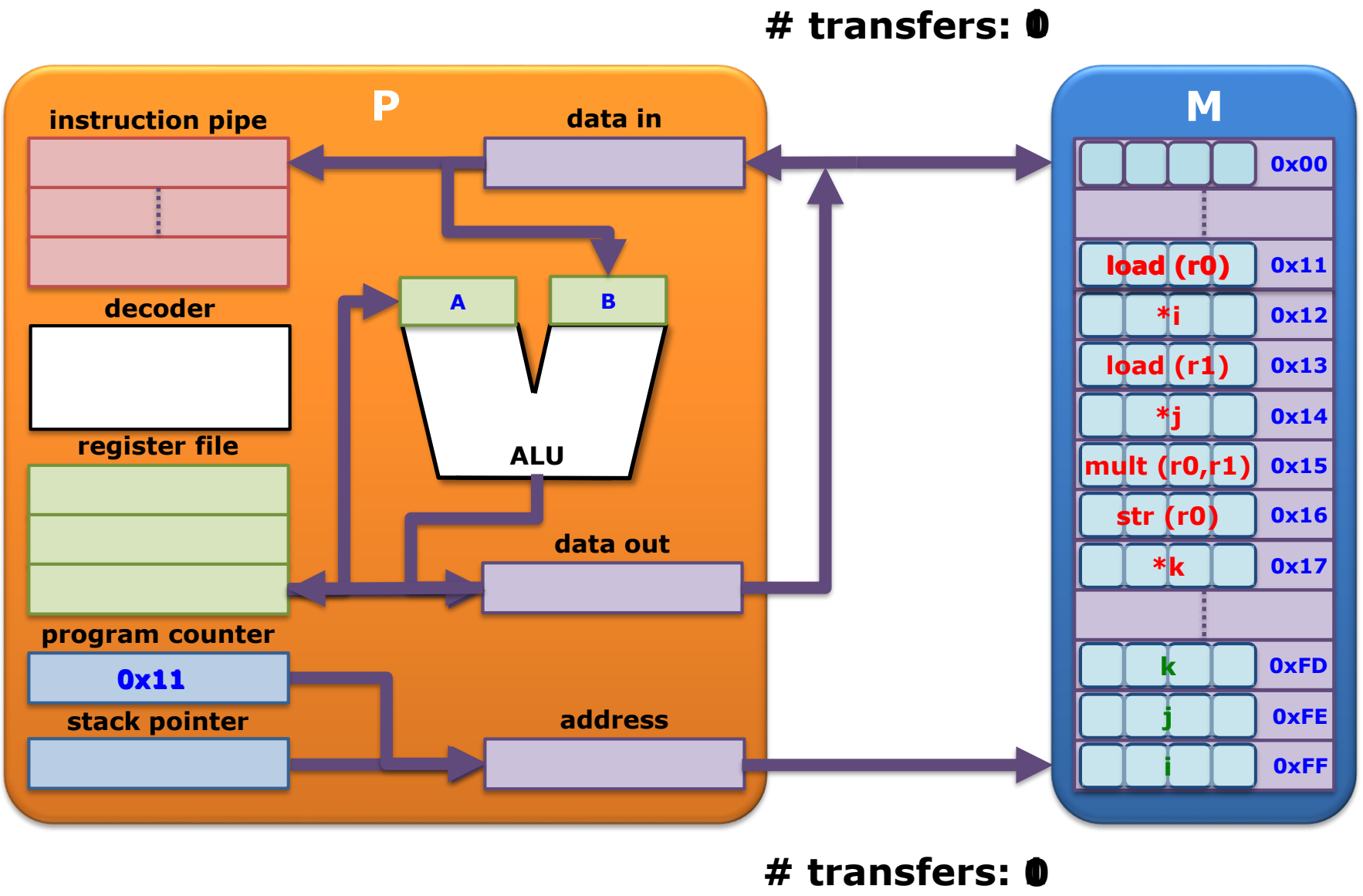
Energy-efficient edge computing

- Critical issues when energy is constrained:
  - Leakage (static) current
  - Processor-memory communication energy
- Non-volatile memory (NVM) is key to
  - Shut down computing resources as much as possible (normally-off computing)
  - Enable limited computation at the data source (logic in memory / in-memory computing)
- Conventional solution: eFlash
  - \+ high density, manufacturable, low-cost
  - – low read/write speed, high power requirements, extra masks, vulnerability to radiation
- Many NVM candidates emerging

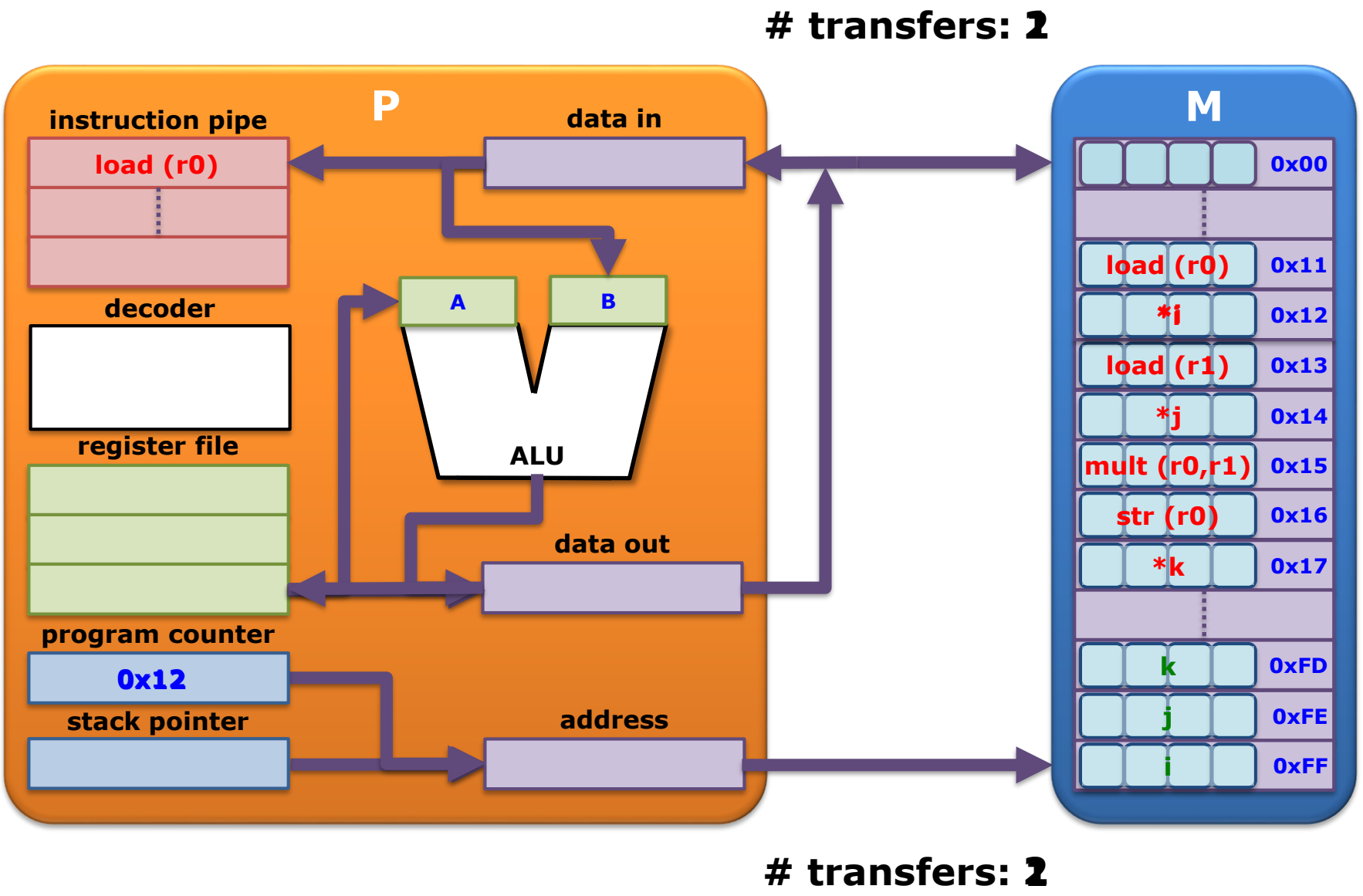
Agenda

- Von Neumann limits
  - Processor-memory communication costs
- New memory-logic paradigms
  - In-Memory-Computing
  - Function configuration (NV-LUT cells, NV-FPGA)
  - Coefficient programming (NV-logic cells, FGLiM)
  - Arithmetic table functions (memory cells, CGLiM)
- Benchmarking platform (work in progress)
  - Circuit level optimization
  - Architectural Design Space Exploration

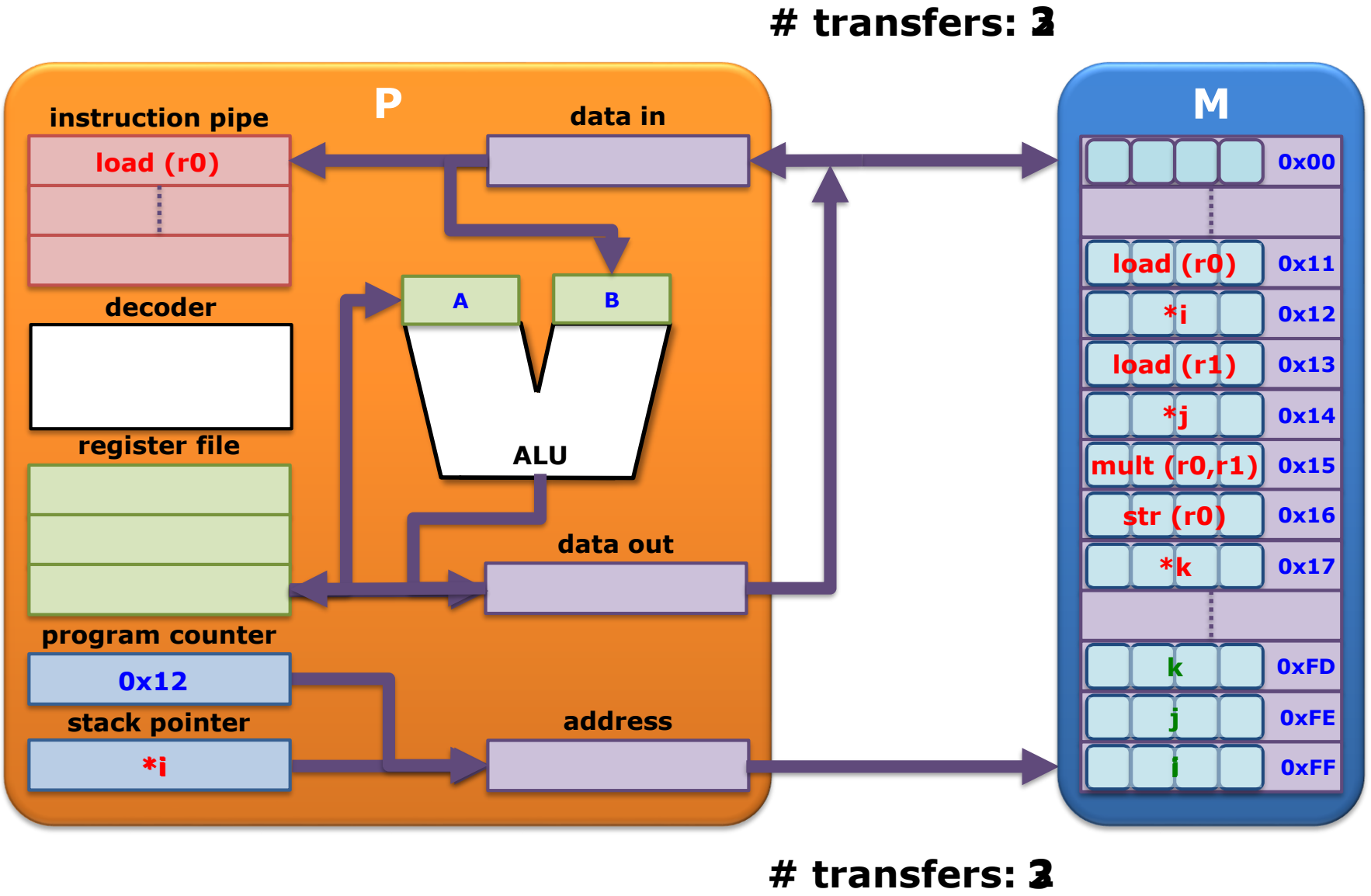
Von Neumann limits



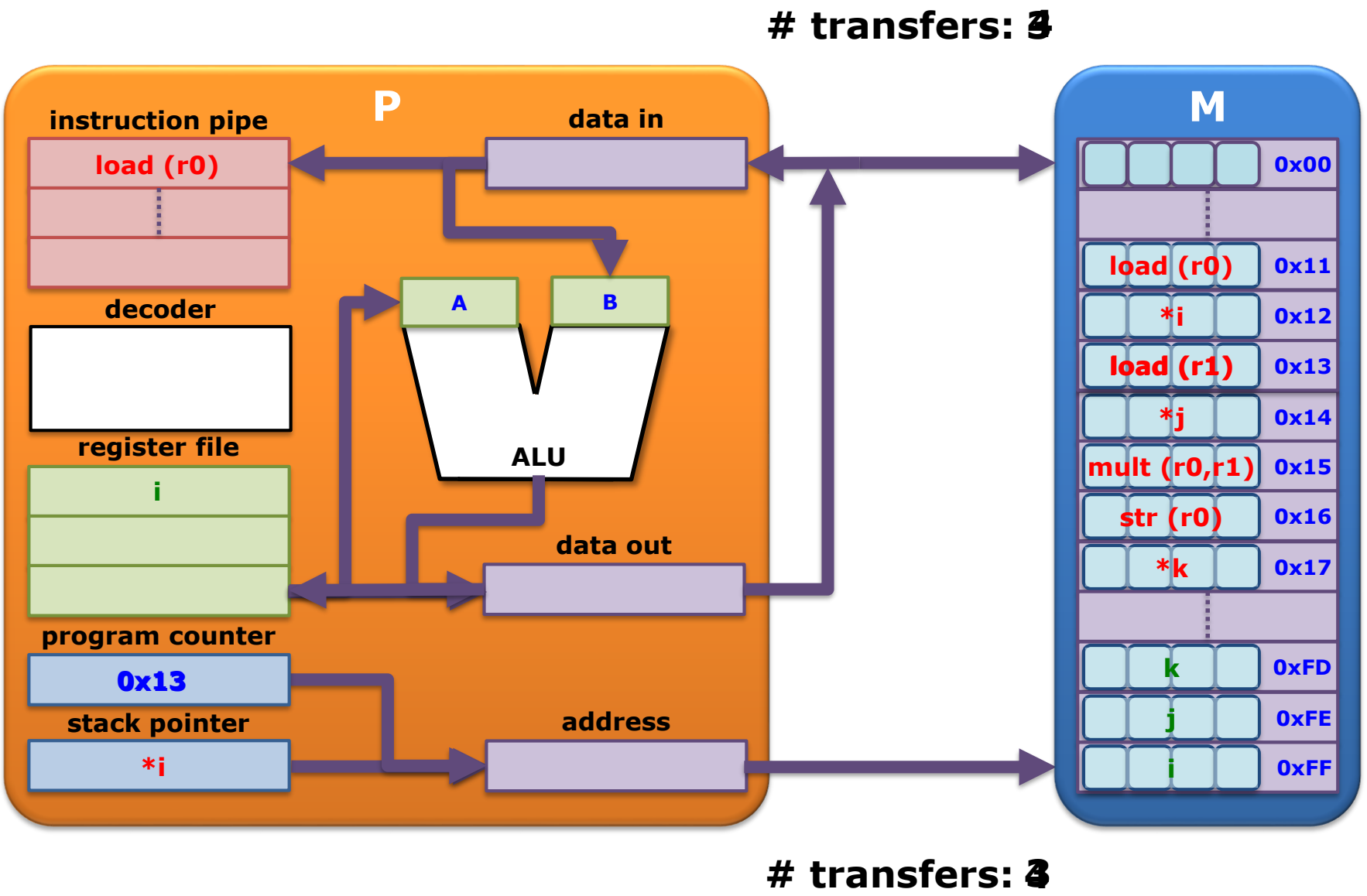
Von Neumann limits



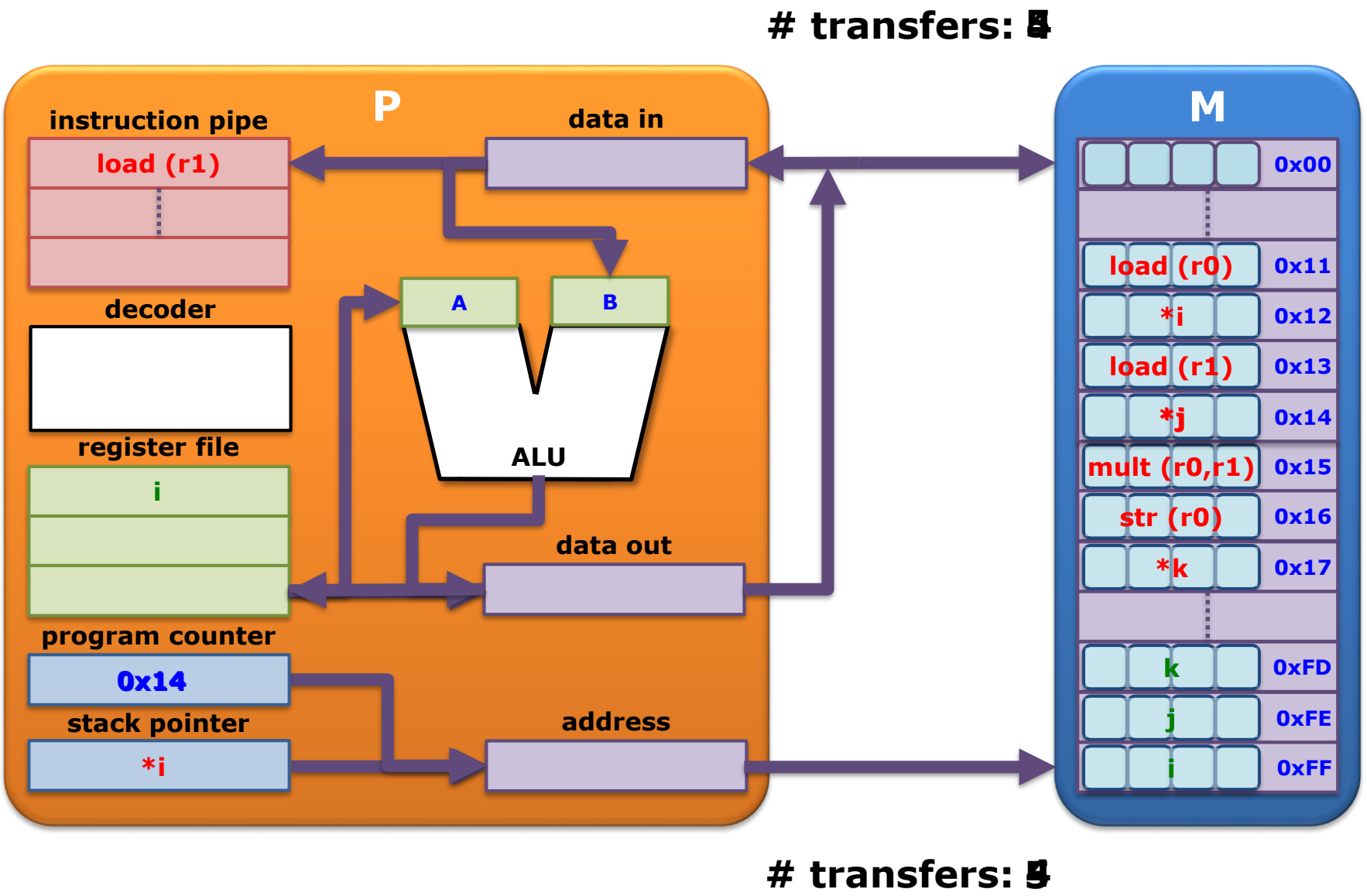
Von Neumann limits



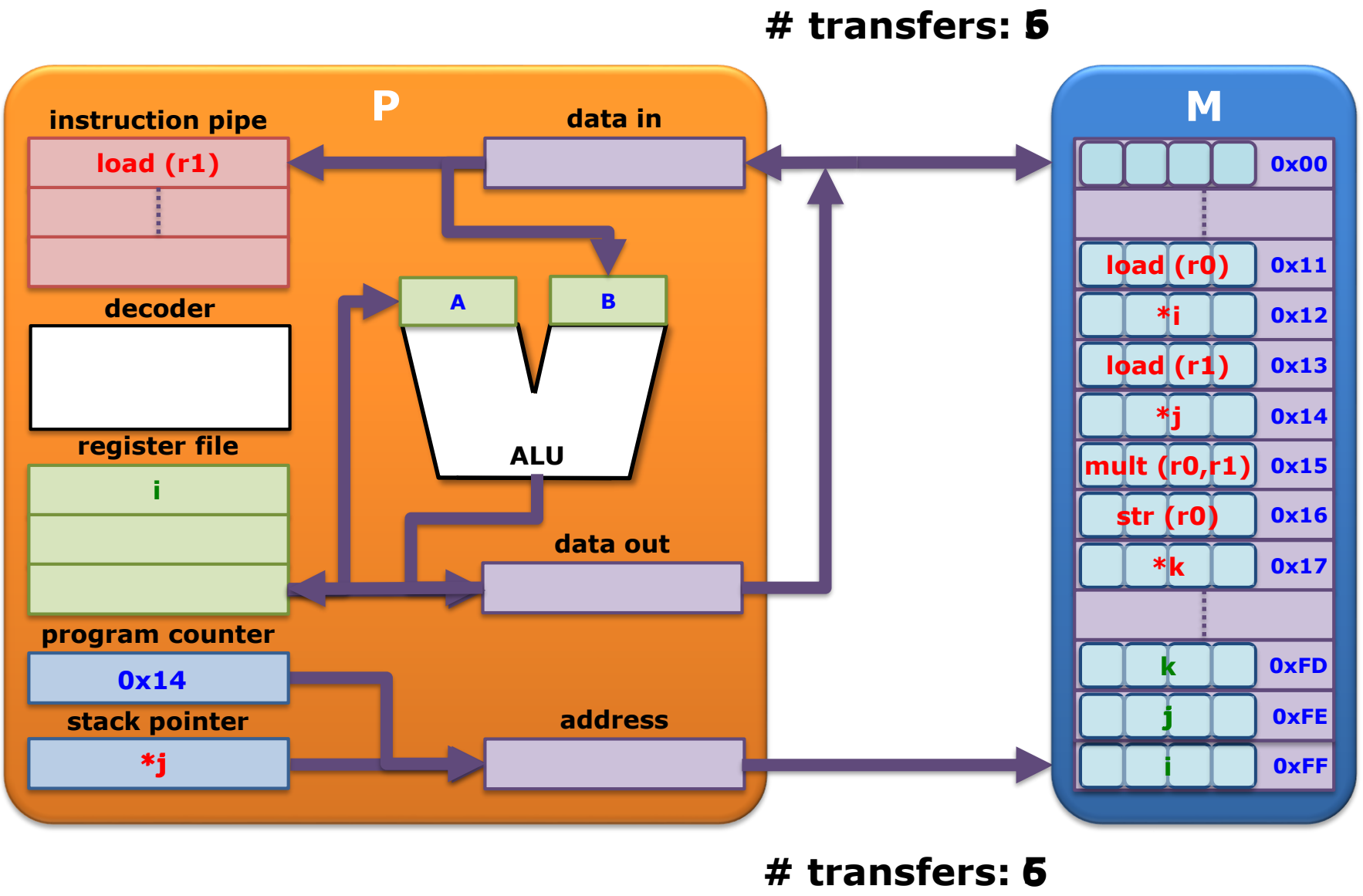
Von Neumann limits



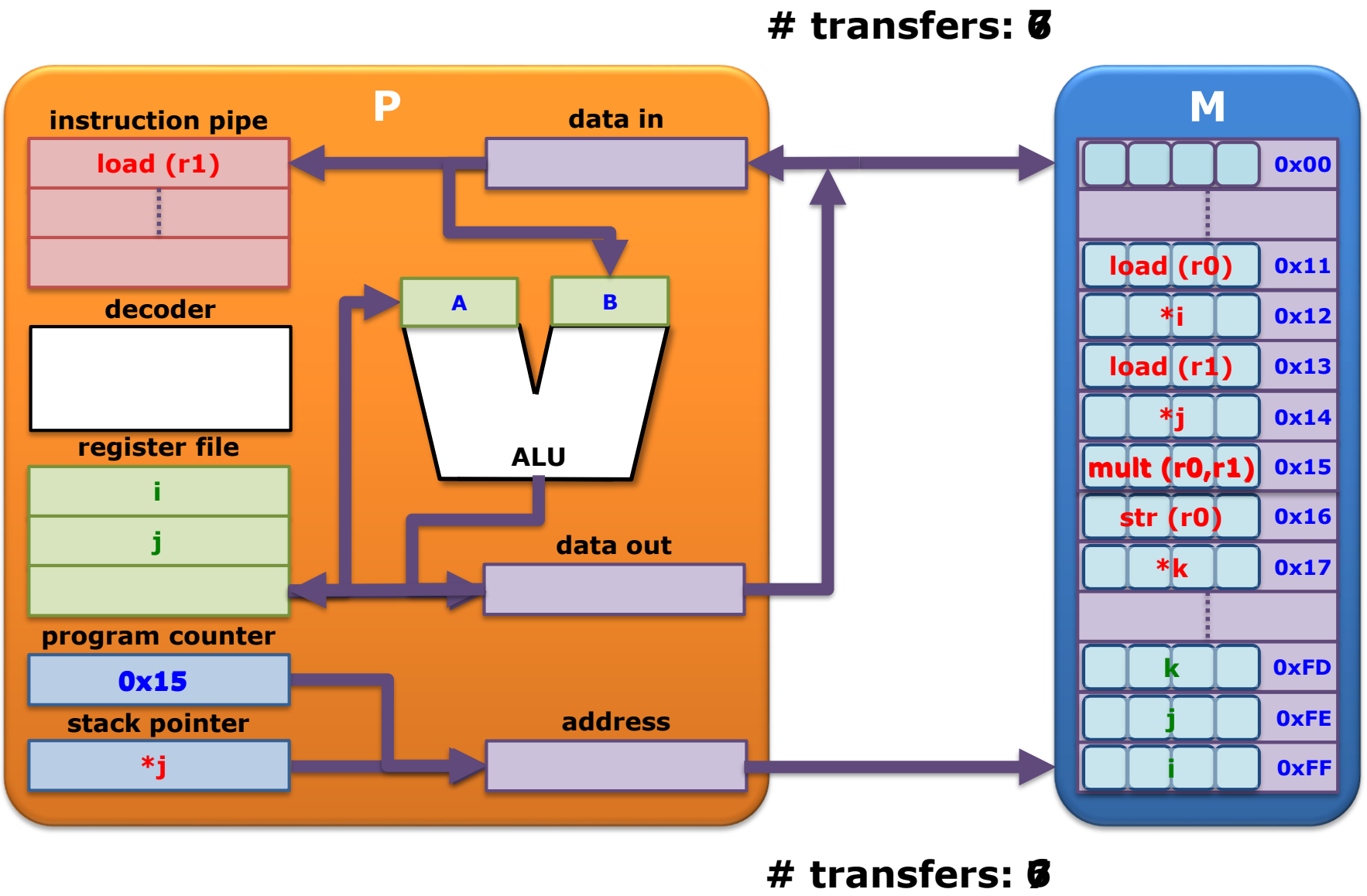
Von Neumann limits



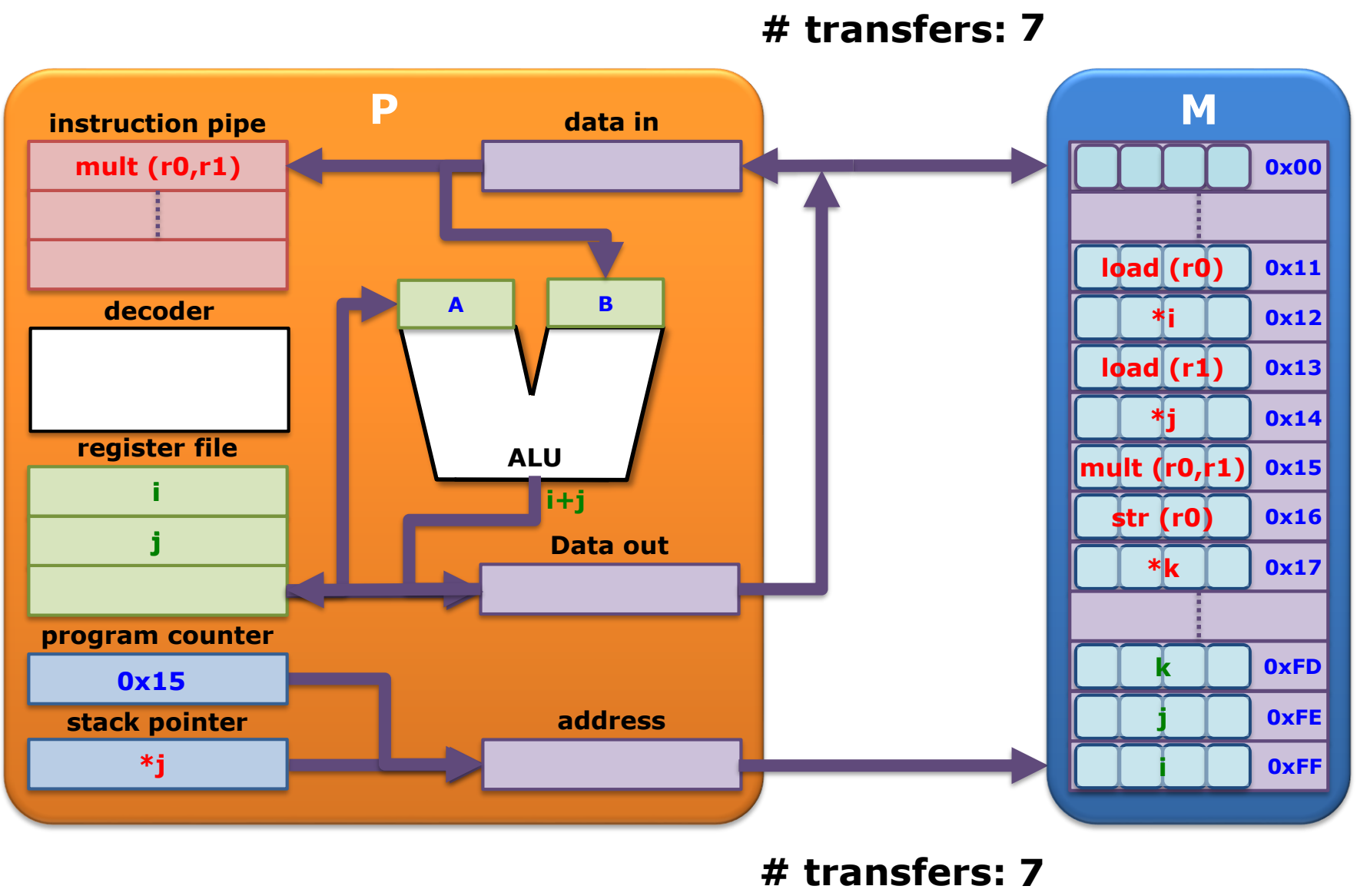
Von Neumann limits



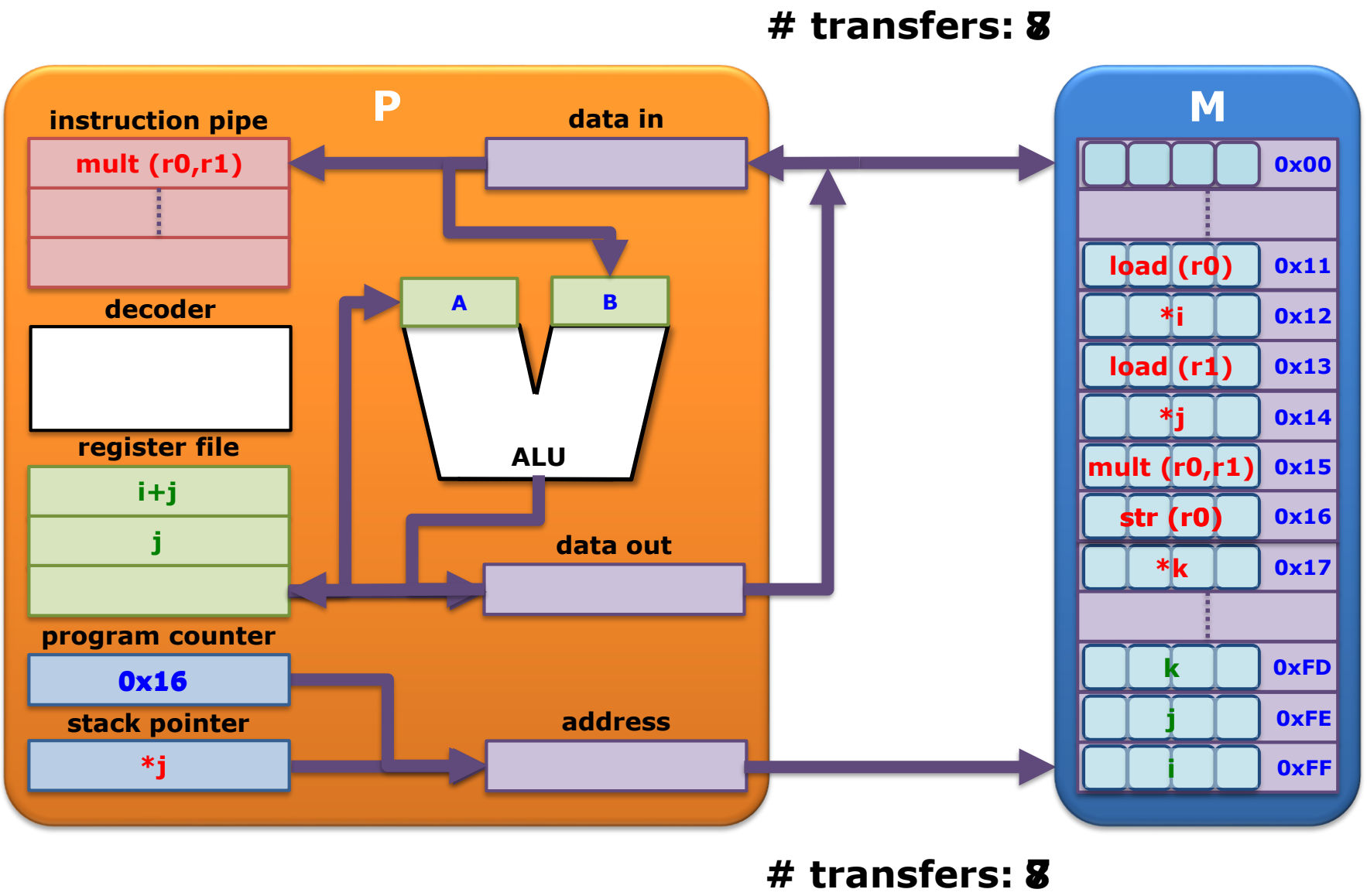
Von Neumann limits



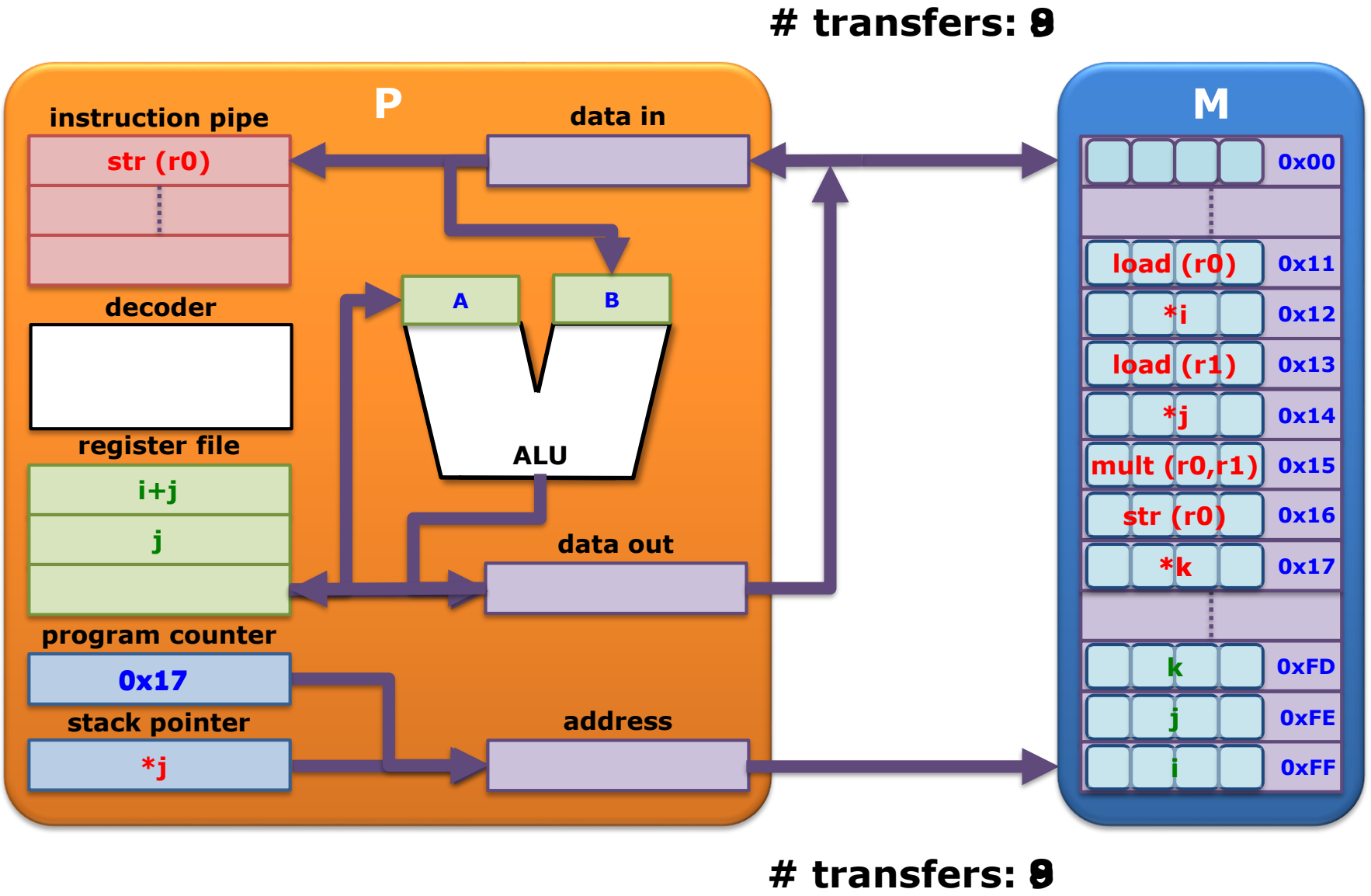
Von Neumann limits



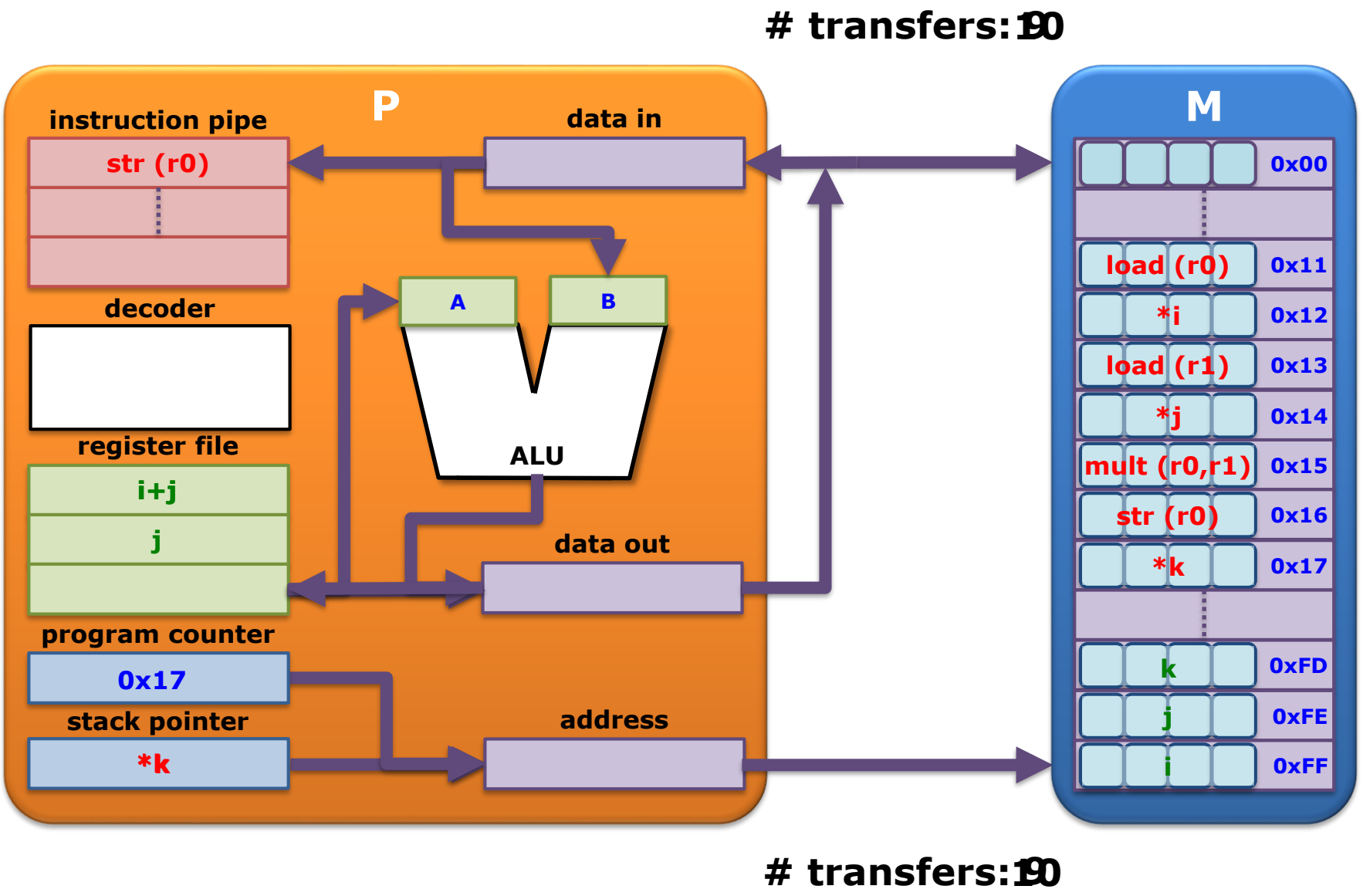
Von Neumann limits



Von Neumann limits



Von Neumann limits



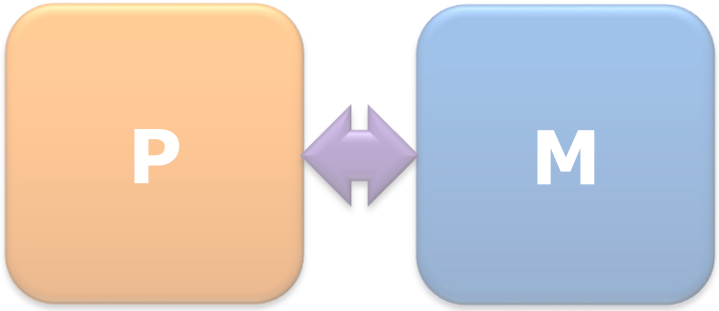
Von Neumann limits

- Separates processing (compute operations) from memory (data and instructions) and relies heavily on intensive communication
- Simple multiplication operation requires 10 address transfers + 10 data transfers
  - Latency (10 cycles) can be improved by increasing processor-memory bandwidth
  - Energy is the main issue – typical figures:
    - 1pJ/bit/mm for communication
    - 1-10aJ/bit for computing
    - 640pJ/mm per 32-bit multiplication **just for communication**
- **95% total energy consumption for communication typical to IoT nodes!**

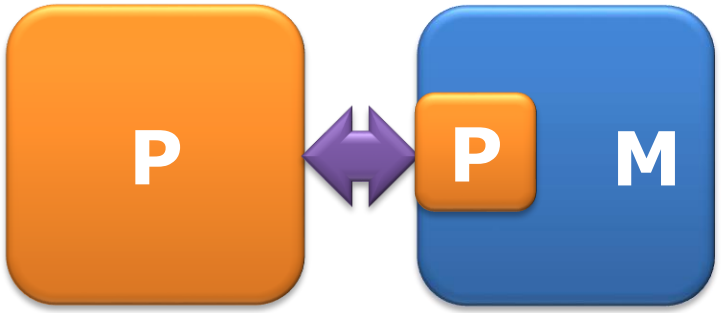
Agenda

- Von Neumann limits
  - Processor – memory transfer costs
- New memory-logic paradigms
  - In-Memory-Computing
  - Function configuration (NV-LUT cells, NV-FPGA)
  - Coefficient programming (NV-logic cells, FGLiM)
  - Arithmetic table functions (memory cells, CGLiM)
- Benchmarking platform (work in progress)
  - Circuit level optimization
  - Architectural Design Space Exploration

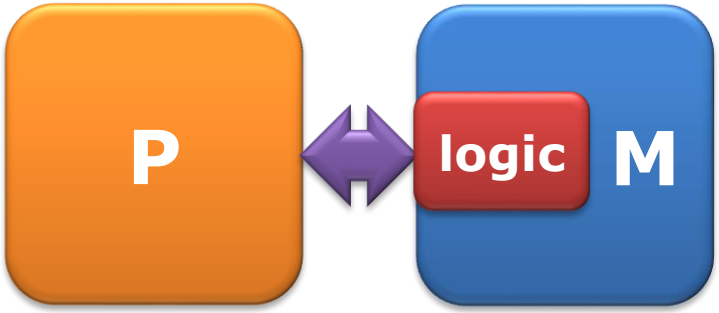
New memory-logic paradigms



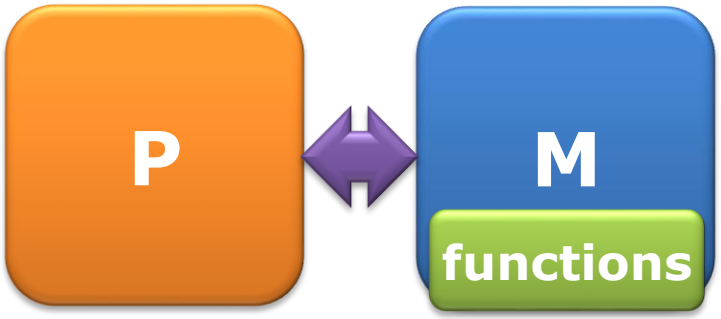
Conventional computing
Von Neumann to manycore



PiM – Processing In Memory
PROCESSING functionality
INside modified MEMORY



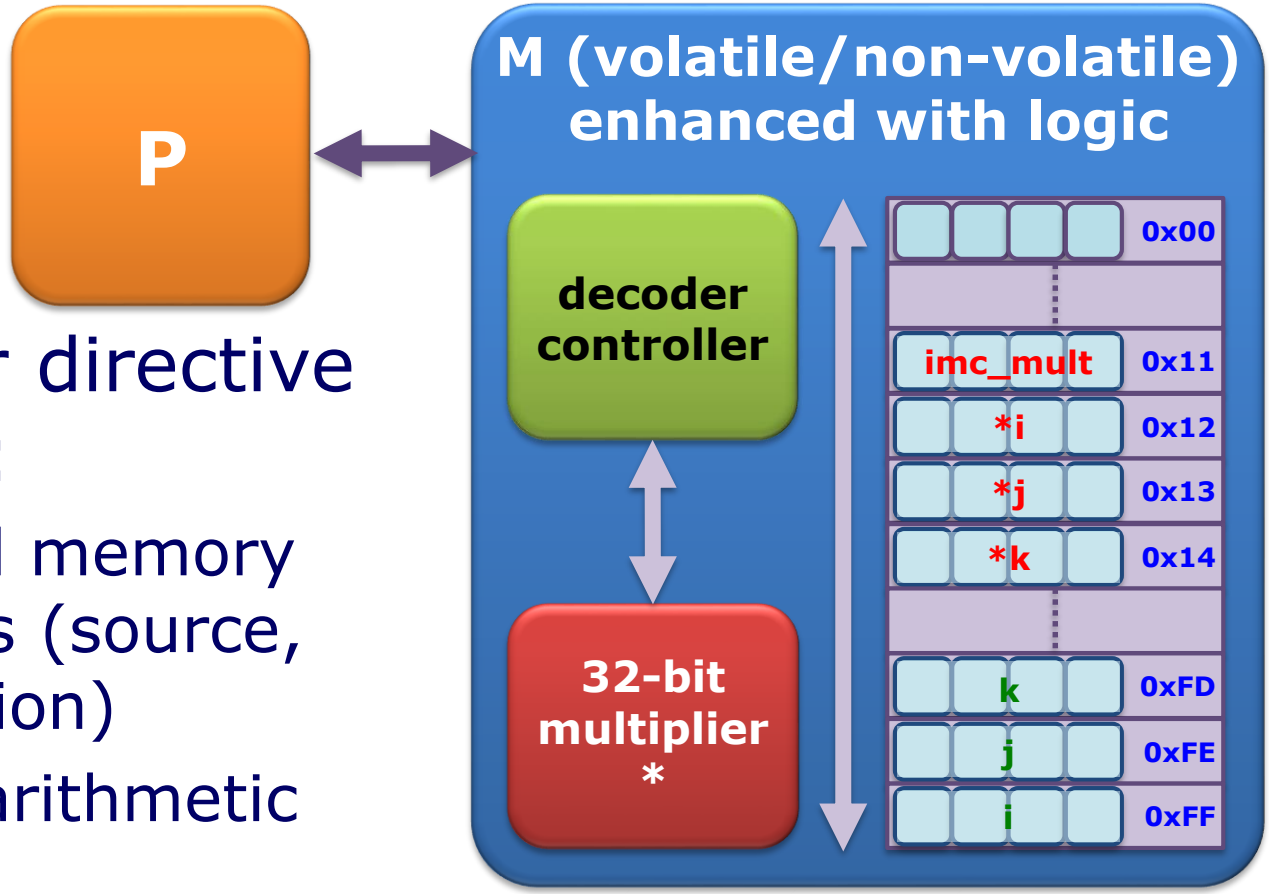
IMC – In Memory Computing
INside modified MEMORY add
elementary COMPUTING functionality



LiM – Logic in Memory
read LOGIC operation results
from existing MEMORY resources

IMC – In Memory Computing

- INside modified MEMORY add elementary COMPUTING functionality

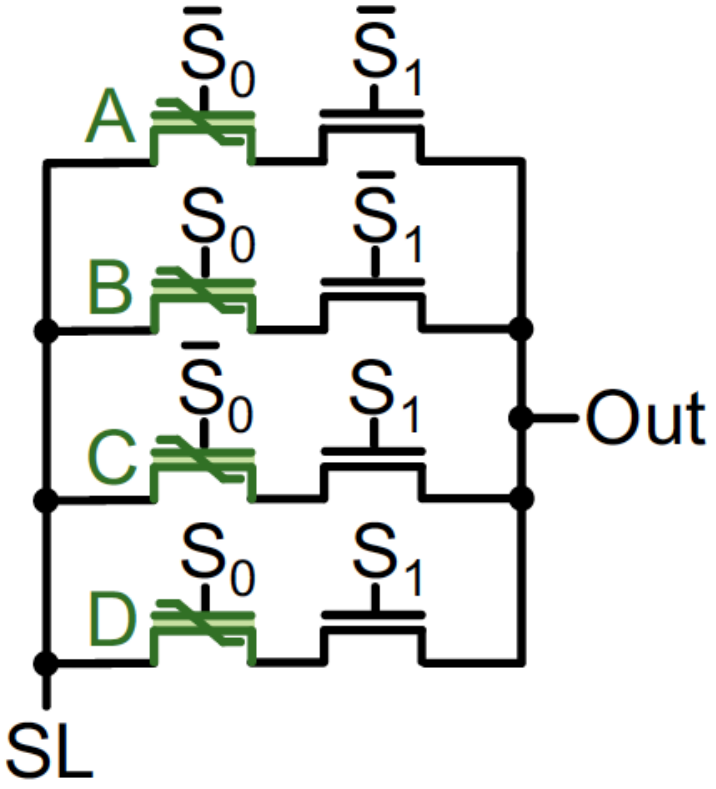


- Processor directive indicates:
  - Operand memory locations (source, destination)
  - Logic / arithmetic function
  - Writeback / no-writeback

Maha Kooli et al., DATE 2018

Reconfigurable In Memory Computing

- The previous example is only capable of executing 32-bit multiplication on memory data (hardwired logic)
- By using reconfigurable logic, resources can implement any function
- NVM is desirable to enable power-down schemes (and avoid superfluous reprogramming)

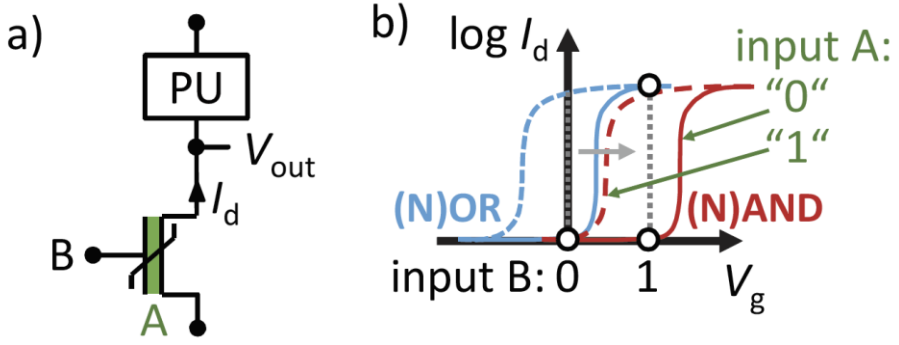


E. Breyer et al., ESSDERC 2019

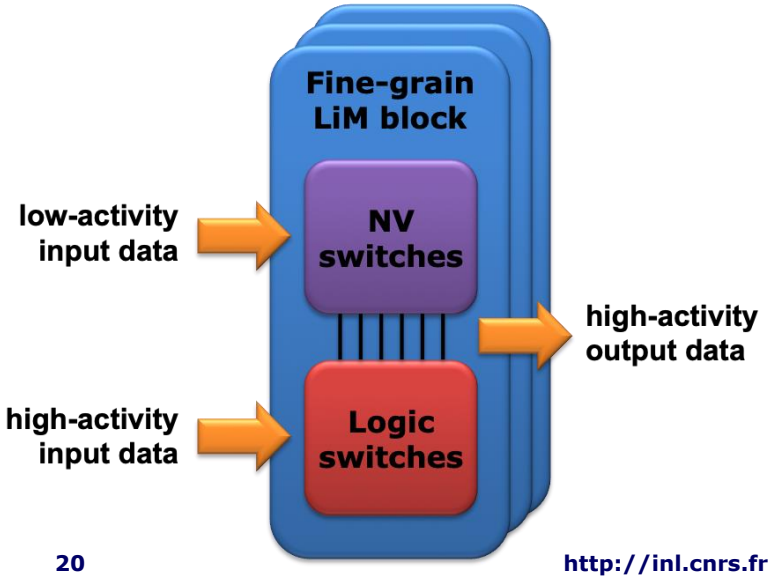
$$Out = A.\overline{S_1}.\overline{S_0} + B.\overline{S_1}.S_0 + C.S_1.\overline{S_0} + D.S_1.S_0$$

Data-programmable logic

- NV-programming of coefficients in data-intensive applications e.g. convolutional filters / neural networks
  - Non-volatile (programmable) logic input A stored in polarization state of the FeFET by applying write pulse (to gate)
  - Volatile (data) logic input B applied as readout voltage (also to gate)

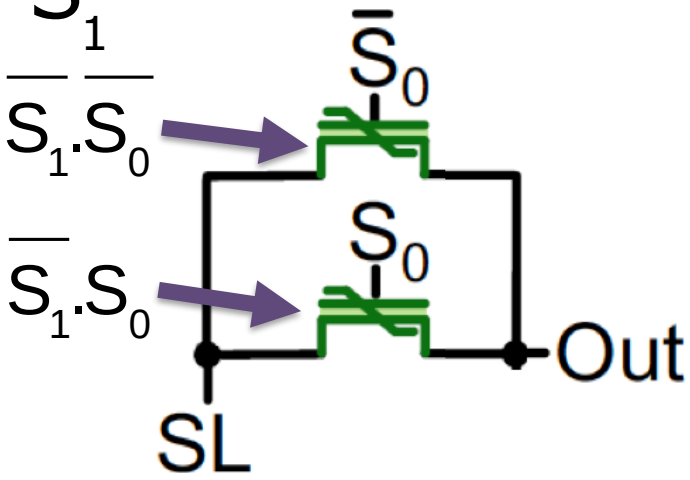


E. Breyer et al., IEDM 2017

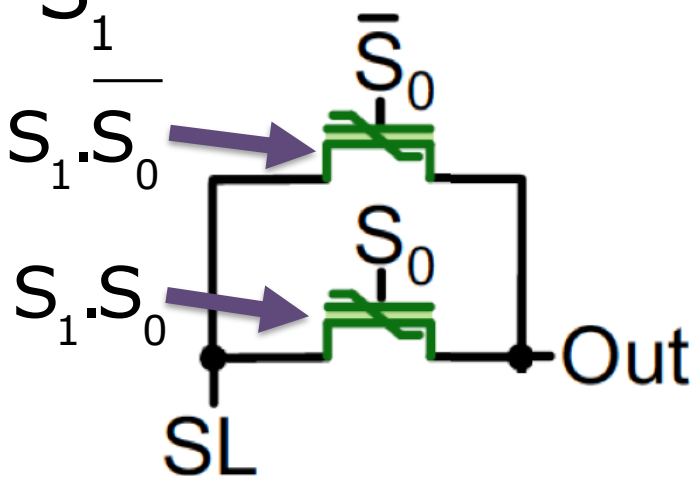


Reconfigurable and data-programmable

- In the previous slides:
  - Volatile data, reconfigurable function, OR
  - Volatile and non-volatile data, fixed function
- Calculating the stored state as a function of the non-volatile data enables combination of reconfigurability and data-programmability
- If $\overline{S_1}$

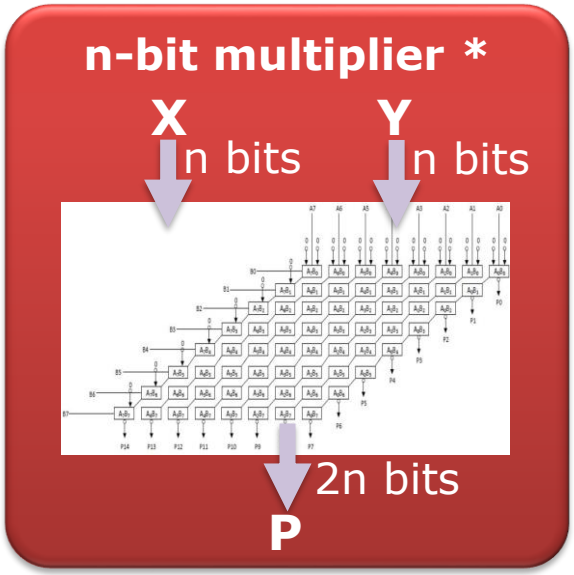


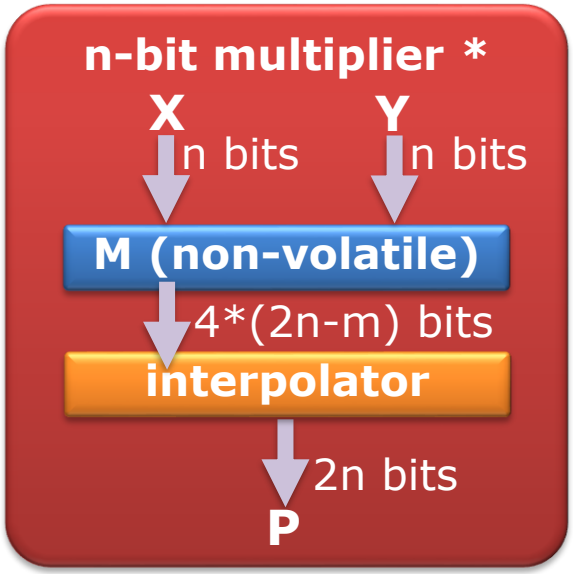
- If S_1



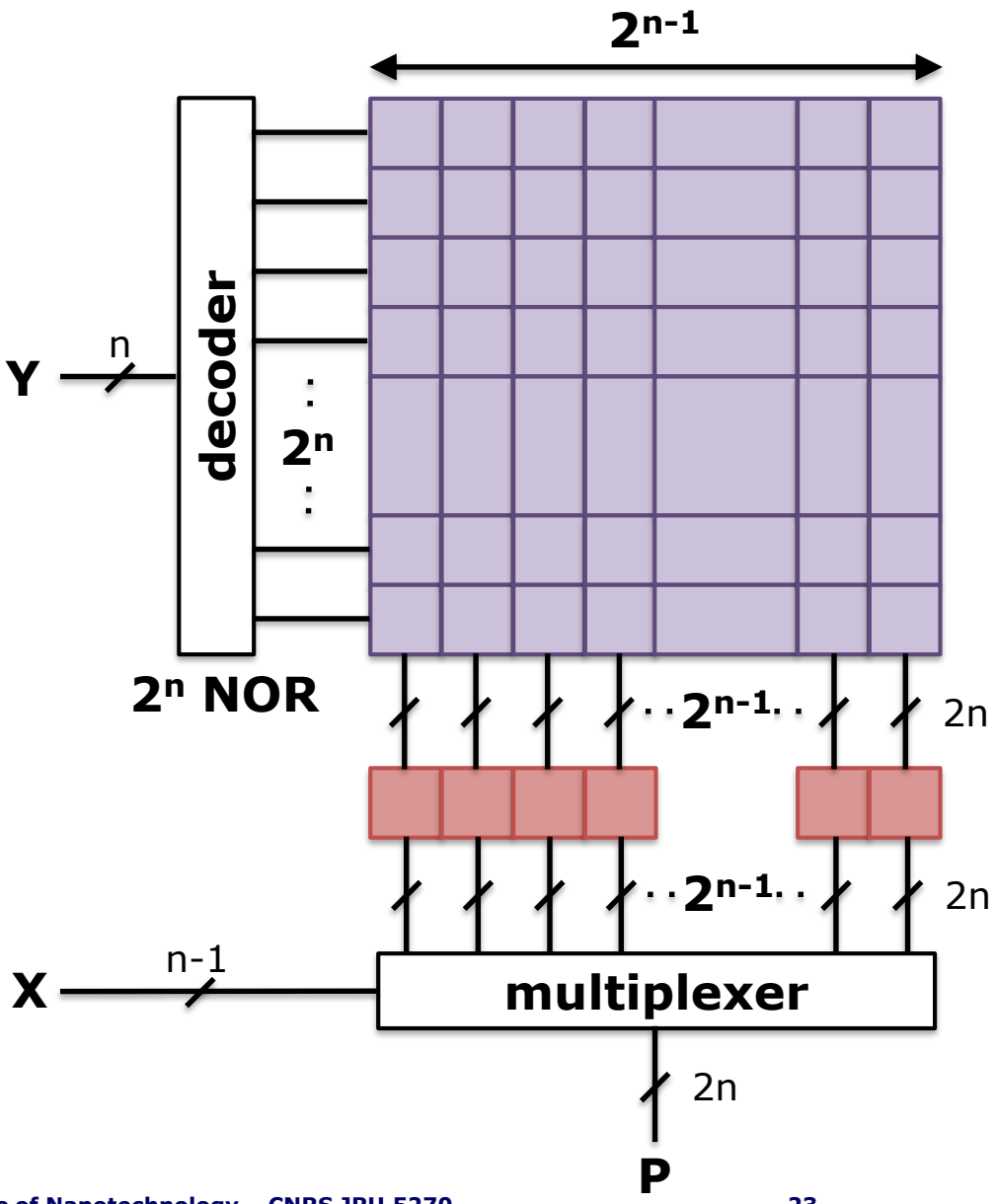
LiM: Arithmetic table functions

- read LOGIC operation results from existing MEMORY resources
  - Example: n-bit multiplication (unsigned integers)
  - **Computation** requires n^2 AND gates, $(n-1)^2$ 1-bit full adders
  - **Lookup table (LUT)** requires n^2 2n-bit memory cells, or fewer with bilinear interpolator

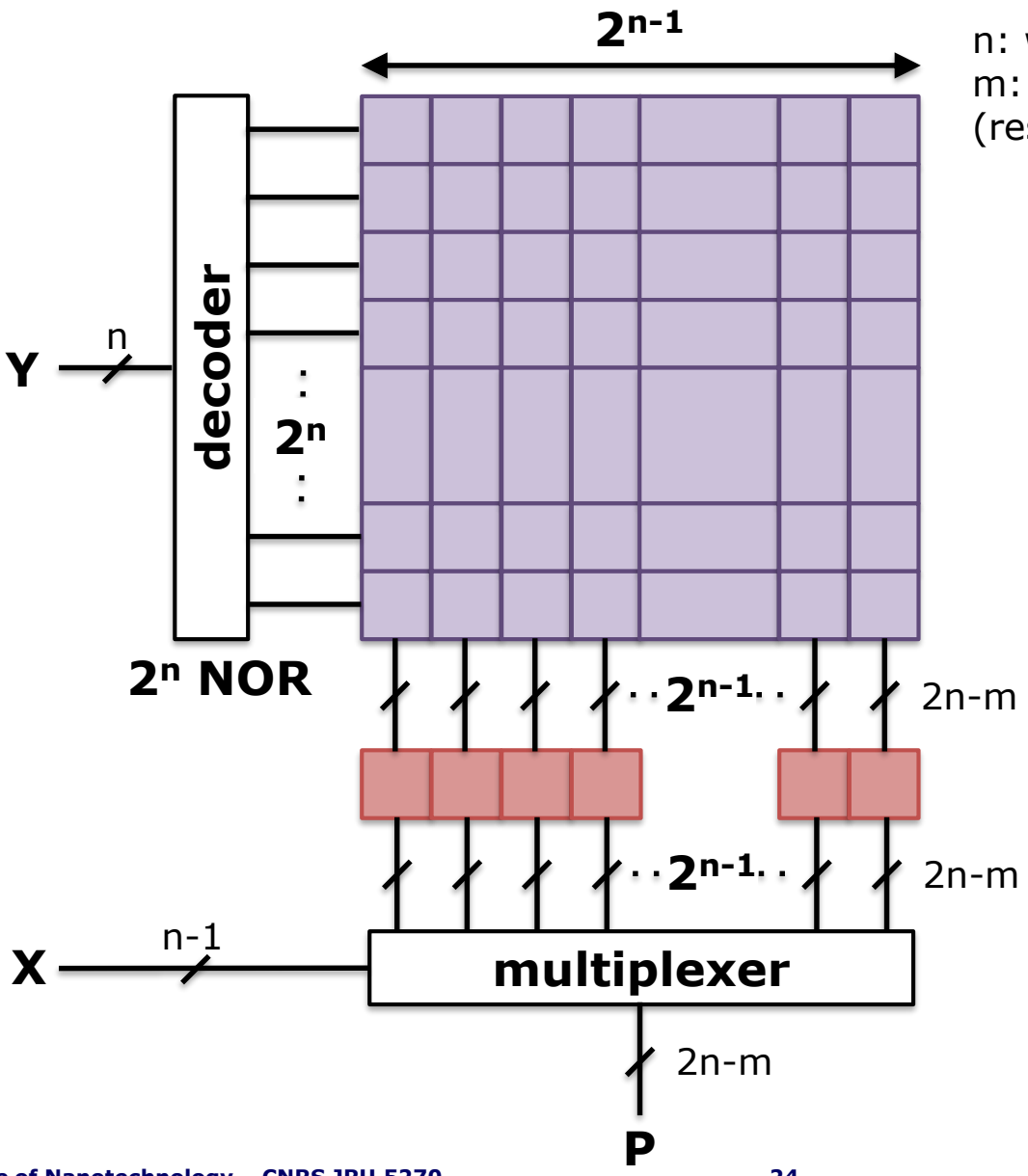




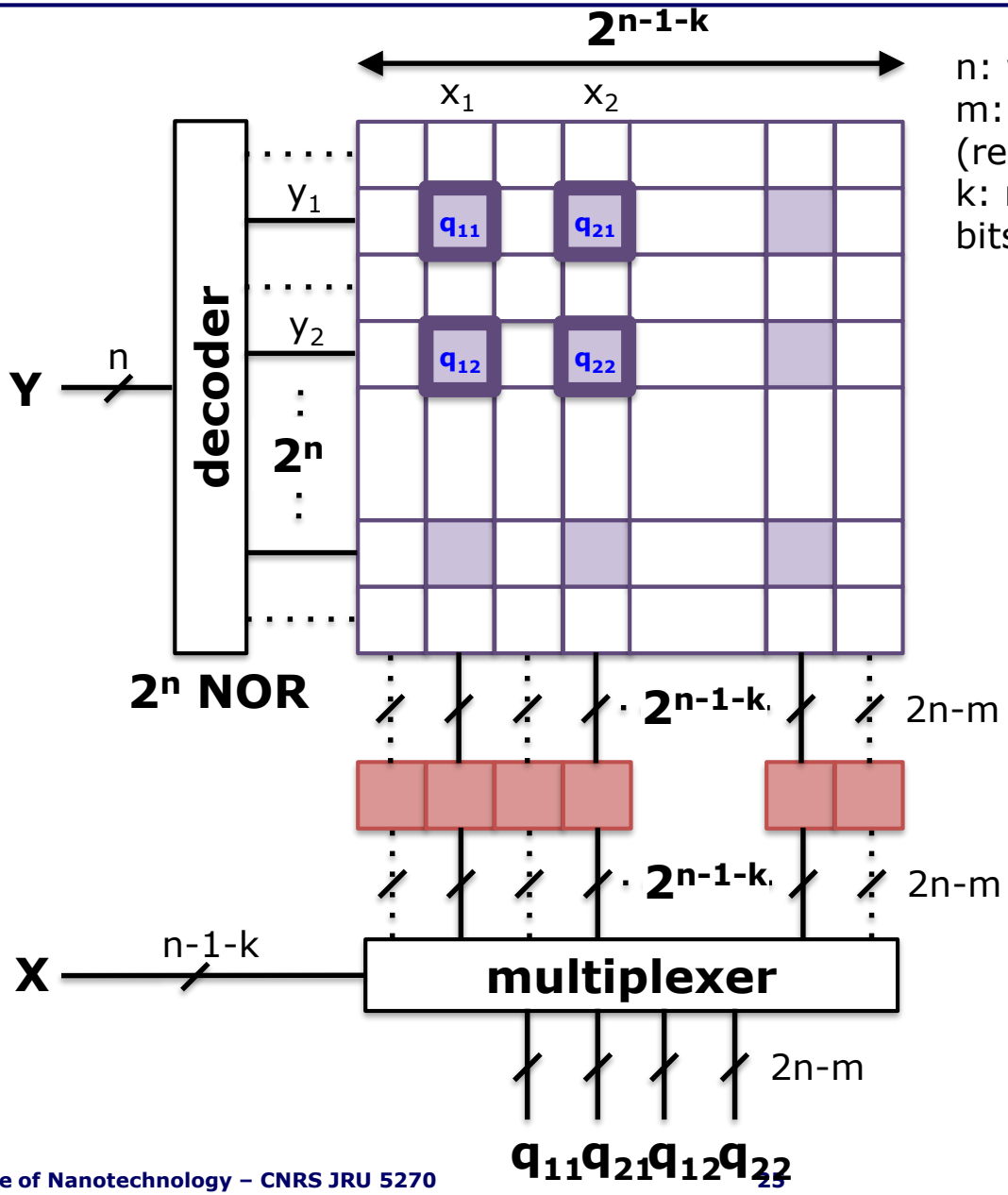
Exact LUT


2n-1
n: width of input operands
Y
n
decoder
2n
2n NOR
..2n-1..
2n
..2n-1..
2n
X
n-1
multiplexer
2n
P

Approximate LUT



Approximate and sampled LUT



n: width of input operands
m: number of truncation bits (results) [m<2n]
k: number of X-operand bits removed [k<n-1]

Bilinear interpolation

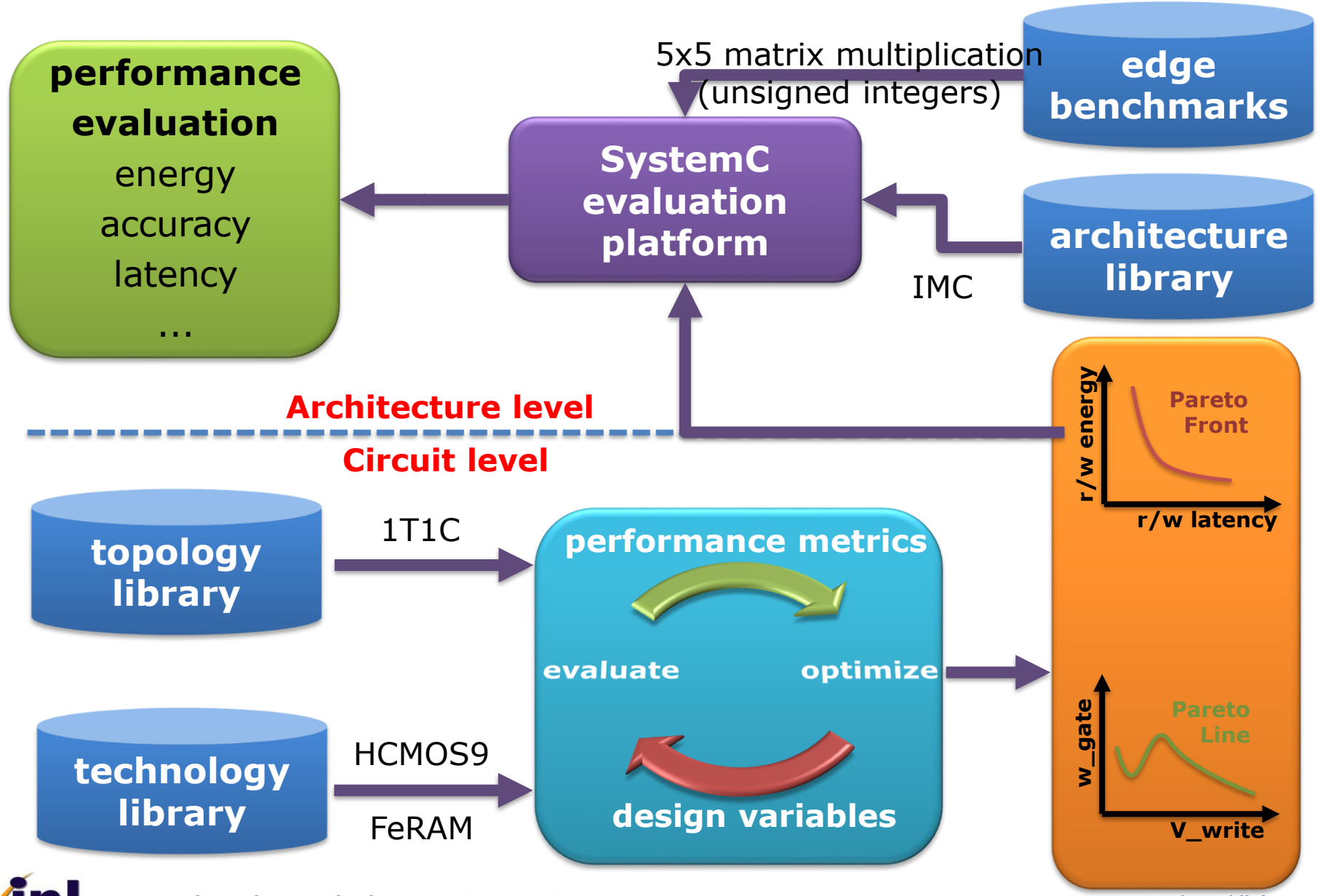
$$f(x,y) = \frac{y_2 - y}{y_2 - y_1}\left(\frac{x_2 - x}{x_2 - x_1} f(Q_{11}) + \frac{x - x_1}{x_2 - x_1} f(Q_{21})\right) + \frac{y - y_1}{y_2 - y_1}\left(\frac{x_2 - x}{x_2 - x_1} f(Q_{12}) + \frac{x - x_1}{x_2 - x_1} f(Q_{22})\right)$$

- 4 (n-m)-bit subtractions
  - (y_2-y), (y-y_1)
  - (x_2-x), (x-x_1)
- 4 2^k bit right-shifts
  - divide (y_2-y), (y-y_1) by (y_2-y_1)=2^k
  - divide (x_2-x), (x-x_1) by (x_2-x_1)=2^k
- 6 ((n-m)*(n-k))-bit multiplications
- Tradeoff to be found between LUT sampling and interpolation hardware in terms of hardware cost, energy consumption and accuracy

Agenda

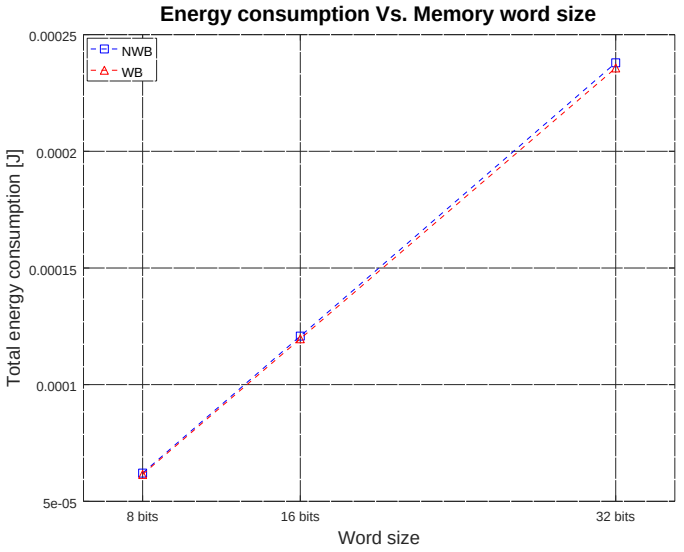
- Von Neumann limits
  - Processor – memory transfer costs
- New memory-logic paradigms
  - In-Memory-Computing
  - Function configuration (NV-LUT cells, NV-FPGA)
  - Coefficient programming (NV-logic cells, FGLiM)
  - Arithmetic table functions (memory cells, CGLiM)
- Benchmarking platform (work in progress)
  - Circuit level optimization
  - Architectural Design Space Exploration

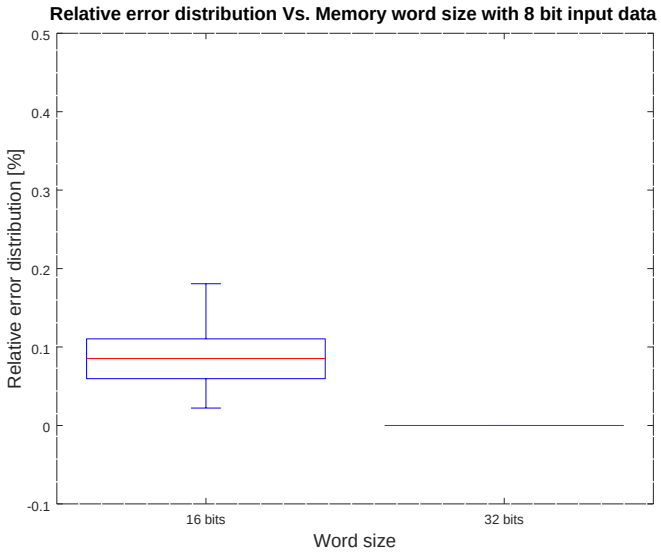
Benchmarking overview



Preliminary results

Parameter	Value
0 read energy	1nJ
1 read energy	2.5nJ
0->0 write energy	0.5nJ
1->0 write energy	0.5nJ
0->1 write energy	2nJ
1->1 write energy	0.5nJ
0 storage power	0.1mW
1 storage power	0.1mW
Restart energy	5nJ
Read latency	20ns
Write latency	20ns
Shutdown latency	200ns
Retention time	inf





Challenges

- Large-scale technology proof-of-concept
  - Memory matrix performance and capacity
  - Integration strategy
  - Overall energy efficiency
- Programming model and compilers
  - Application oriented hardware configuration
  - Agile instruction sets
  - Clear metrics and figures of merit
- Additional paradigms
  - Security
  - Approximate computing
  - Deep learning