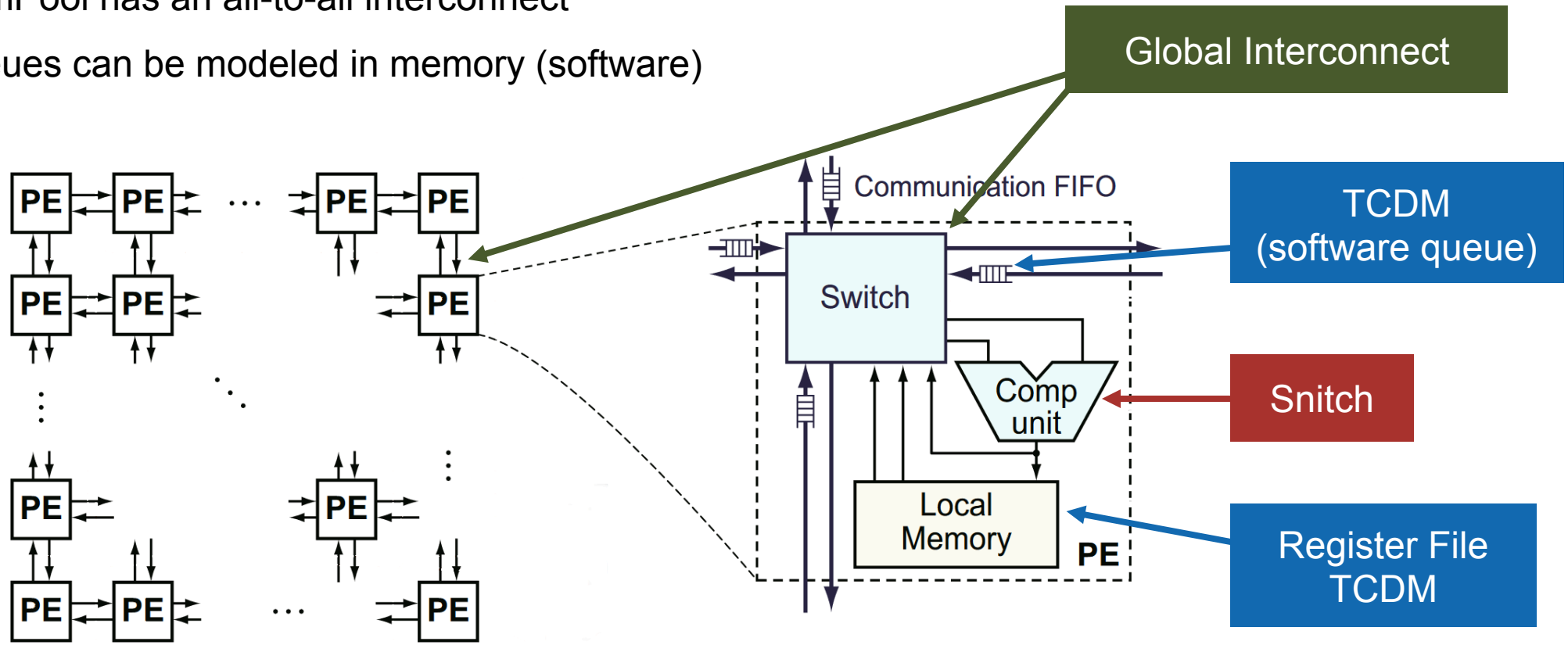
ETH zürich

MemPool meets Systolic

Samuel Riedel
Matheus Cavalcante
Prof. Luca Benini

Emulate Systolic Workloads on MemPool

- 256 cores can be thought of as a 16×16 grid
- MemPool has an all-to-all interconnect
- Queues can be modeled in memory (software)



Source: Domain-Specific Language and Compiler for Stencil Computation on FPGA-Based Systolic Computational-Memory Array - https://link.springer.com/chapter/10.1007/978-3-642-28365-9_3

Evolution of MemPool meets Systolic

```
// software queues
c = 0;
for (i=0; i<N; i++) {
  a = queue_pop(qa_in);
  b = queue_pop(qb_in);
  c += a * b;
  queue_push(a, qa_out);
  queue_push(b, qb_out);
}
```

```
// +queue pop/push extension
c = 0;
for (i=0; i<N; i++) {
  a = __builtin_pop(qa_in);
  b = __builtin_pop(qb_in);
  c += a * b;
  __builtin_push(a, qa_out);
  __builtin_push(b, qb_out);
}
```

```
// +SSR like extension
c = 0;
setup_ssr(a, qa);
setup_ssr(b, qb);
for (i=0; i<N; i++) {
  c += a * b;
}
```

Software Queues

- Emulate all communication queues in software
- Allows for:
  - Arbitrary number of queues
  - Arbitrary interconnect topology

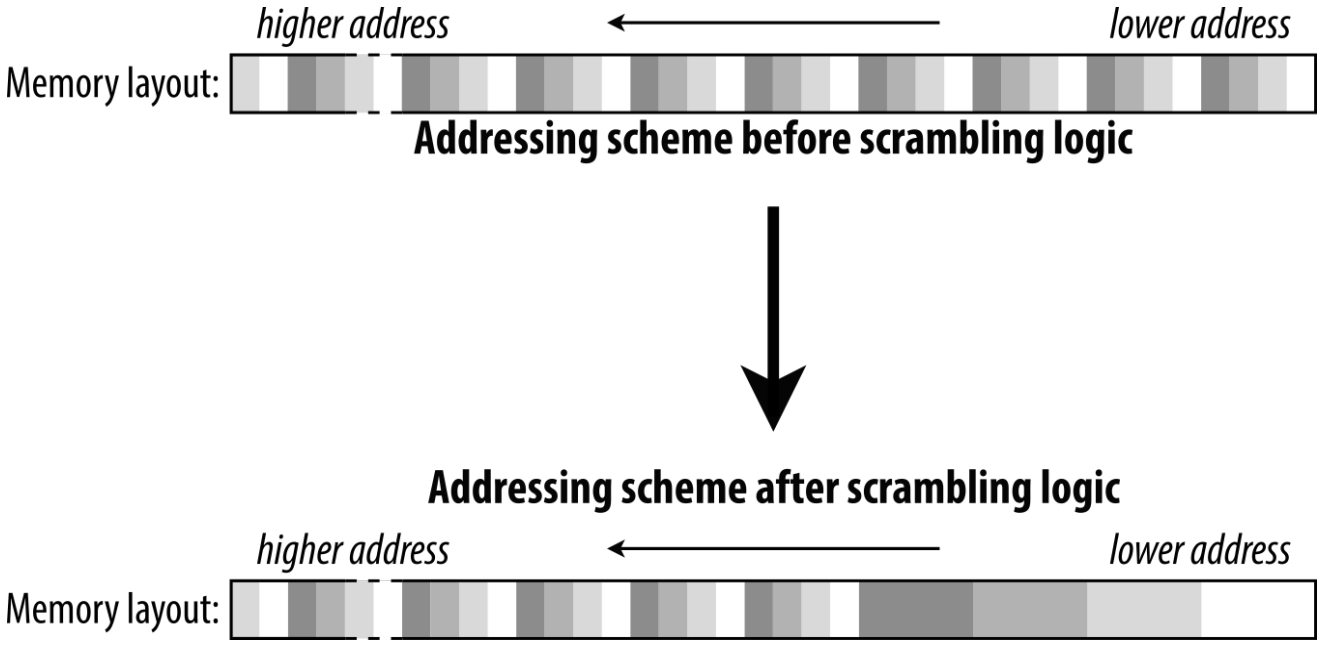
- Get a feel for systolic workloads
  - Boundary conditions
  - Behavior in MemPool

```
// Baseline
c = 0;
for (i=0; i<N; i++) {
  a = queue_pop(qa_in);
  b = queue_pop(qb_in);
  c += a * b;
  queue_push(a, qa_out);
  queue_push(b, qb_out);
}
```

→ Function calls

Keeping the queues local

- Represent fixed connection between producer/consumer
- Queues can be kept local to push/pop core
- Use/extend hybrid addressing scheme
  - Allows us to have continuous regions that stay local to a core



Memory-Allocated Queues

- Queues dynamically allocated in the sequential region
  - *malloc* (with different memory regions)

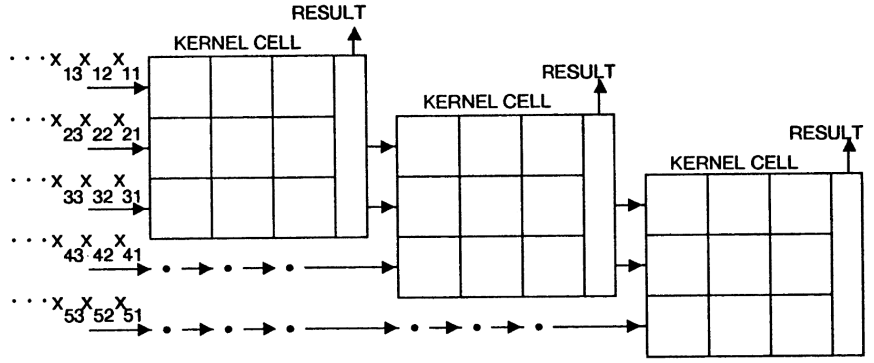
```
// Queue data type

typedef struct {
  int32_t *array;
  int32_t head;
  int32_t tail;
  int32_t size;
} queue_t;
```

→ Dynamically allocated struct

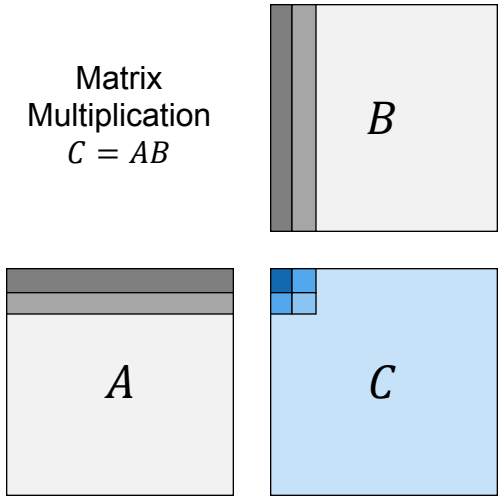
- Dynamic or static queues
- Circular buffer queue implementation

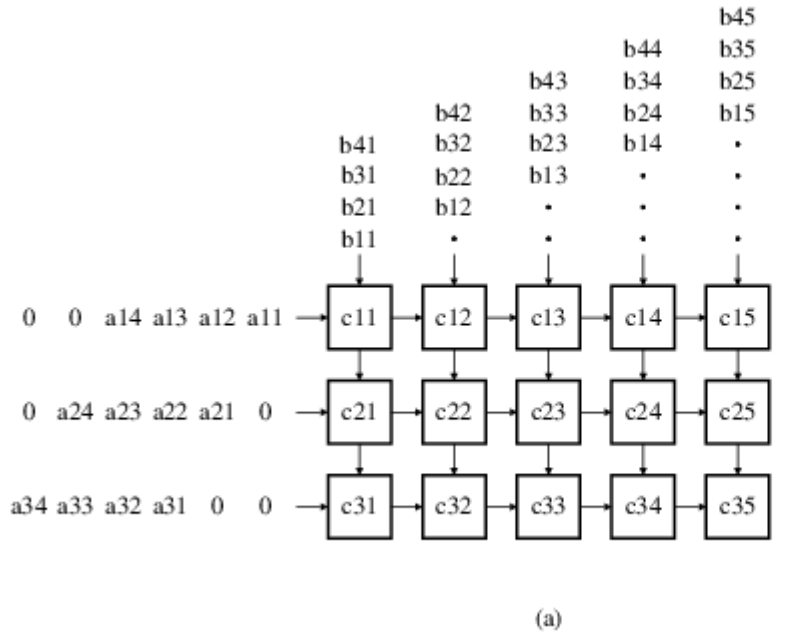
Benchmarking

- Kernels to use as benchmarks:
  - Matrix Multiplication
  - 2D Convolution
  - *What else?*
- Compare to default data-parallel implementation



Source: H. T. Kung and S. W. Song "A Systolic 2-D Convolution Chip"





Source: https://gyires.inf.unideb.hu/KMITT/a52/ch04.html