PROGRAMMING MODELS FOR NEXT GENERATION HPC SYSTEMS

Jens Breitbart

Research Group Programming Languages / Methodologies
University of Kassel
Germany

UNCONVENTIONAL RESEARCH IN THE PROGRAMMING MODEL FIELD?

SHOULD WE EVEN TRY?

- No?
  - PMs evolve slowly for good reasons!
- Yes!
  - Steer existing PM model evolution
  - ... and maybe even find something worthwhile on its own

OVERVIEW

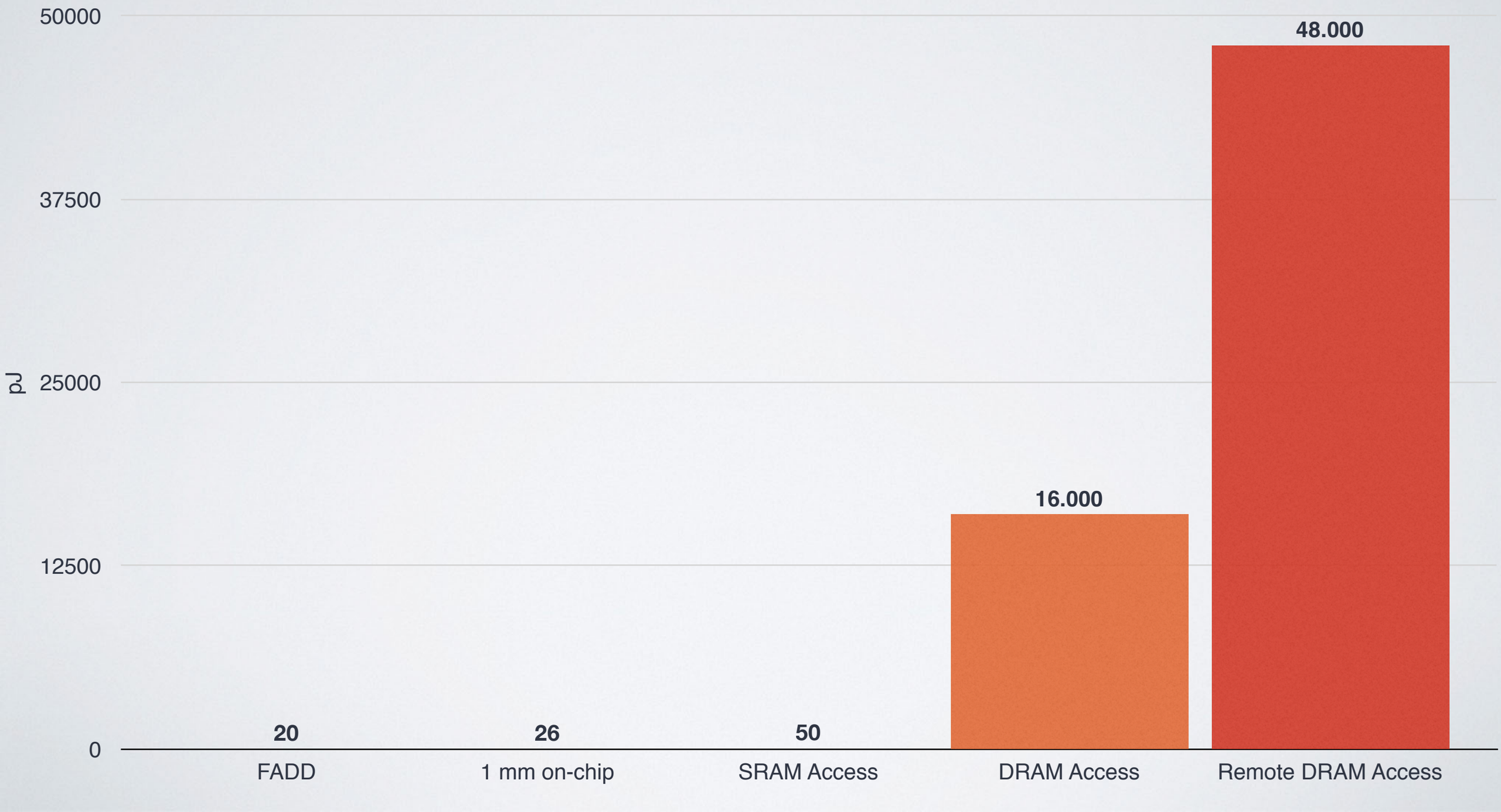
- Future hardware - A prediction
- Challenges for future programming models
- Current programming model - What is our there?
- Something unconventional: PSAM
- Short summary

FUTURE HARDWARE - WHAT CHALLENGES ARE AHEAD?

Performance = Performance / Watt

1 source for energy consumption = memory accesses

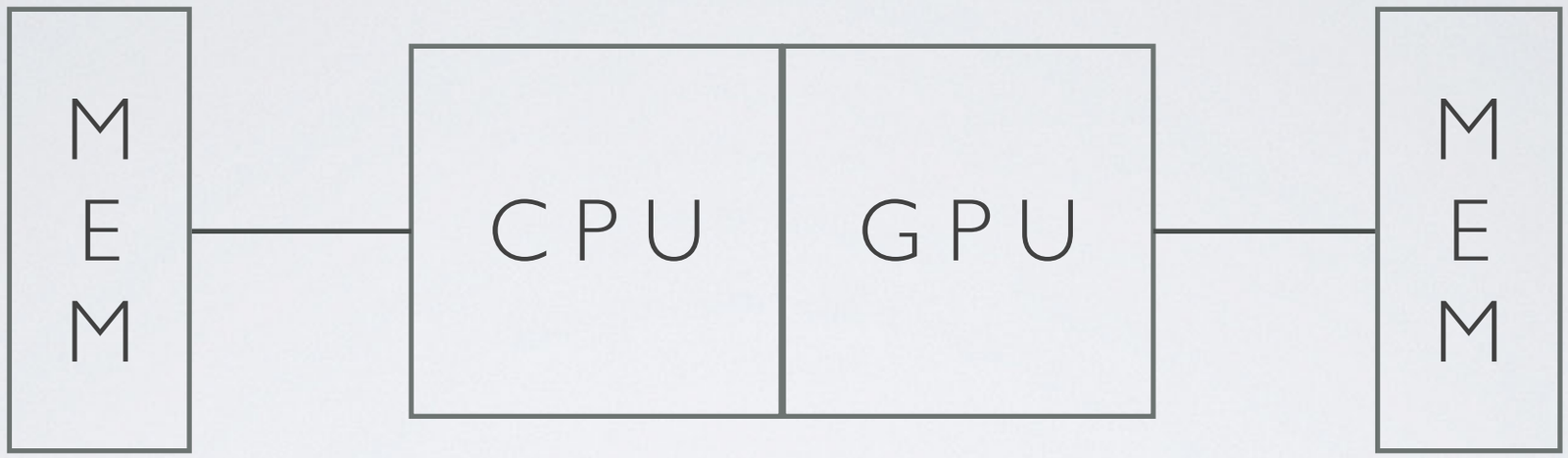
FUTURE HARDWARE - WHAT CHALLENGES ARE AHEAD?



CURRENT HARDWARE

- Cache coherent (prevents scalability)
- Not feasible for all problems
- Slow communication link

FUTURE HARDWARE?



- Multiple core types
  => and a lot of cores

- Still cache coherent?
  => Communication through memory

- NUMA effects on chip?
  => Know where your data is!

OVERVIEW

- ~~Future Hardware – A prediction~~
- Challenges for future programming models
- Current programming model - What is our there?
- Something unconventional: PSAM
- Short summary

CHALLENGES FOR FUTURE PROGRAMMING MODELS

- We may loose cache coherency
  - but users need an easy and efficient memory consistency model
- We need efficient synchronization
  - to get strong scaling
- We may get NUMA effects in every system
  - and first touch may not what user want to use

REQUIREMENTS

A NEW PROGRAMMING MODEL SHOULD:

- Efficient and easy to understand memory consistency model
- Allow efficient fine grained pair wise synchronization
- Provide control over NUMA effects
- Easy to use

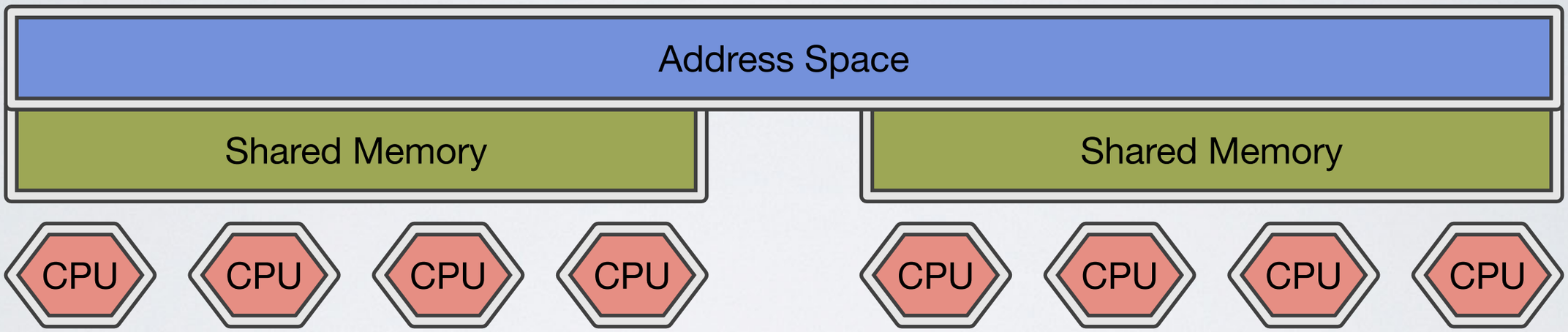
OVERVIEW

- ~~Future Hardware – A prediction~~
- ~~Challenges for future programming models~~
- Current programming model - What is our there?
- Something unconventional: PSAM
- Short summary

CURRENT PROGRAMMING MODELS - WHAT IS OUT THERE?

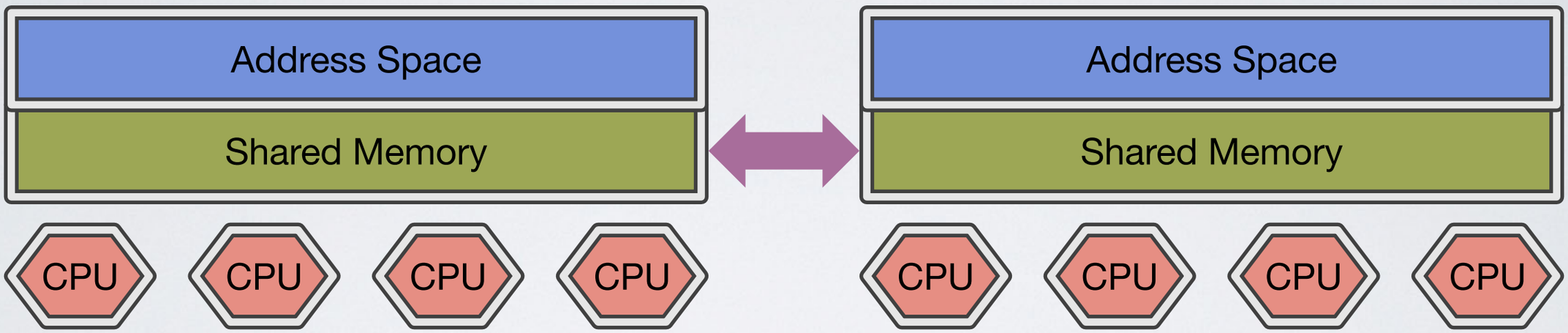
- OpenMP: Not good non cache coherent systems
- MPI: Not unconventional (and we'll concentrate on share memory)
- CUDA/OpenCL: Only barrier synchronization between tasks
- PGAS: Unconventional + already faces various issues expected for future hardware

CHAPEL



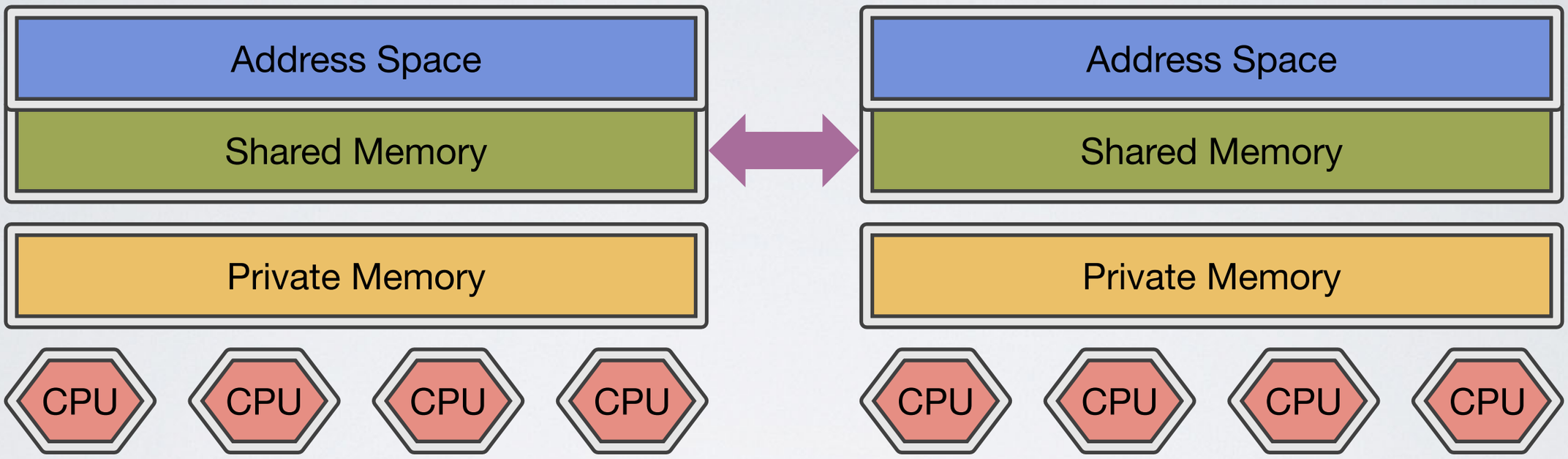
- APGAS language
- Relaxed memory consistency following shared memory systems
- Synchronization variables

X10



- APGAS language
- OpenMP like node local synchronization
- Active message semantic for remote memory reads/writes

XCALABLEMP

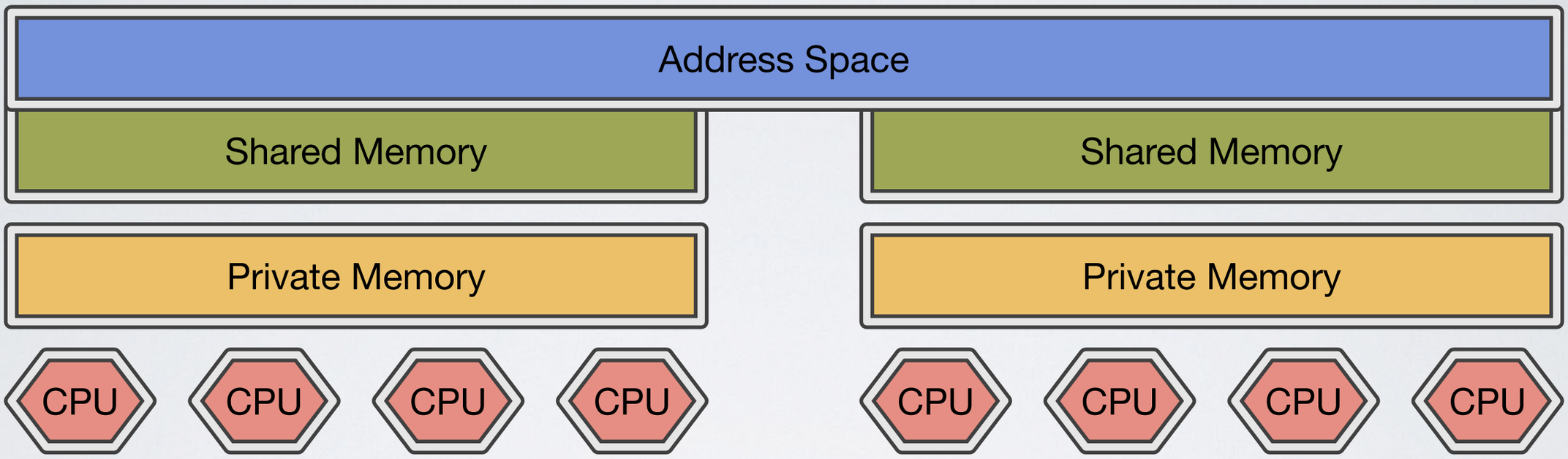


- Version 1.0 was released Nov. 2011
- Barrier synchronization between nodes
- Only explicit remote accesses

OVERVIEW

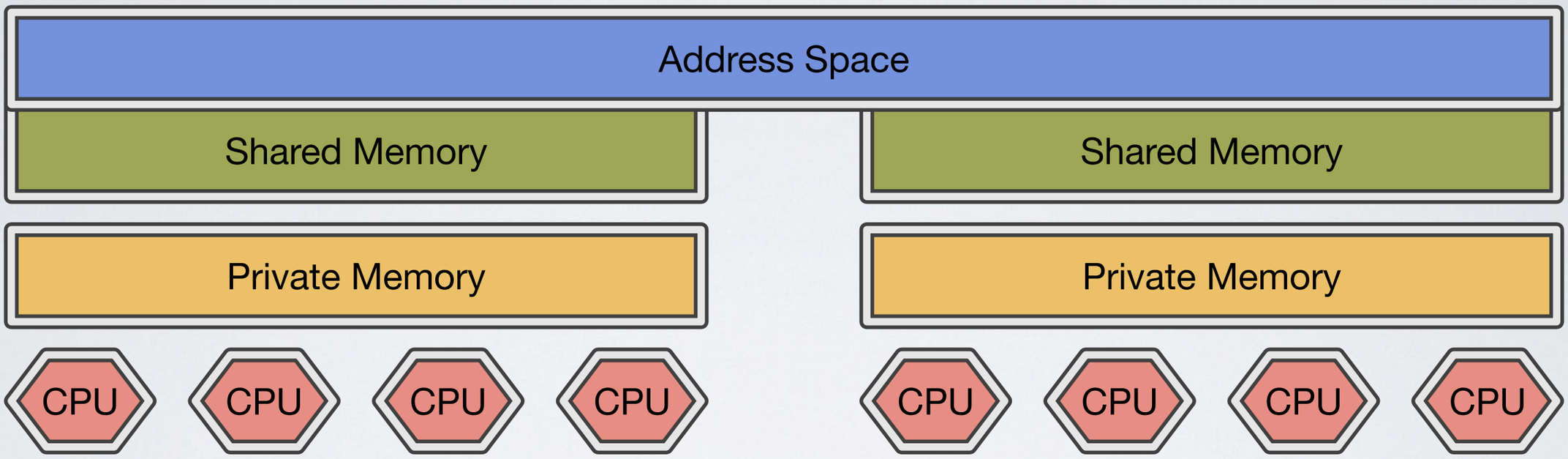
- ~~Future Hardware – A prediction~~
- ~~Challenges for future programming models~~
- ~~Current programming model – What is our there?~~
- Something unconventional: PSAM
- Short summary

PSAM



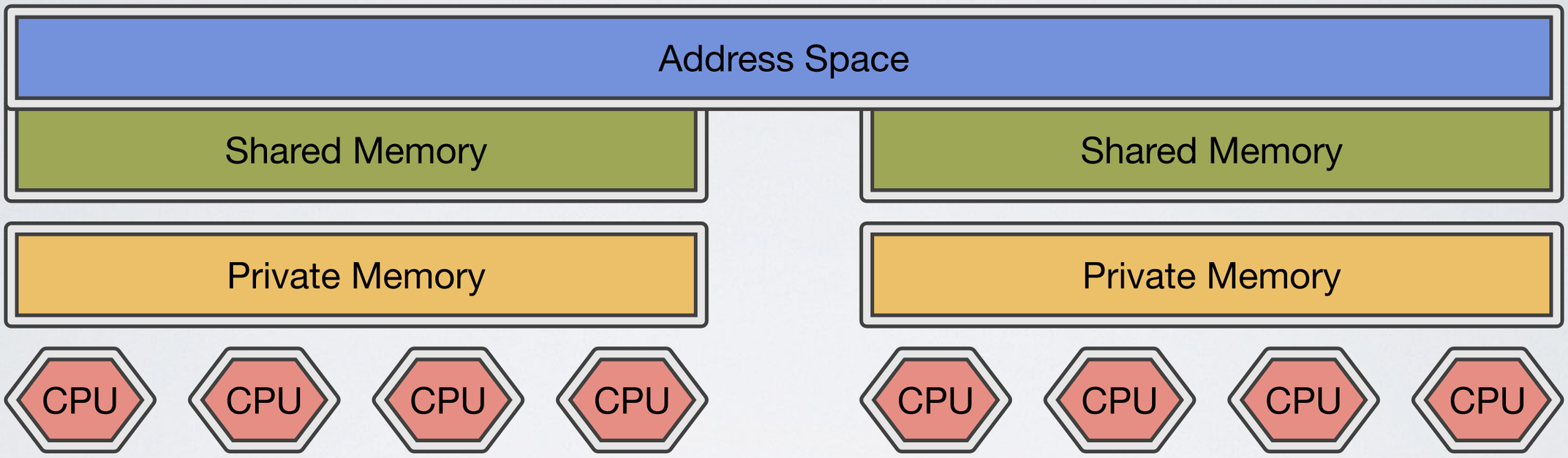
- Shared memory is single assignment
  - Empty: can be written, reading blocks
  - Full: can be read, cached everywhere

PSAM



- SA memory is divided in synchronization units
  - Think: matrix tiles, graph partitions, array stripes, ...
  - Reduces synchronization overhead, follows cache blocking

PSAM



- Dataflow-like synchronization without race conditions
- Use e.g. OpenMP for node local parallelisation
- PRAM like global reduction/scan

OUR SOLUTIONS

A PGAS PROGRAMMING MODEL SHOULD:

- Allow efficient fine grained pair wise synchronization
  - Synchronize on single assignment data batches
- Provide control over NUMA effects
  - Explicit memory placement
- Easy to use
  - No race conditions

THERE IS A PSAM LIBRARY

IT...

- is unstable and highly experimental
- mostly used to experience model trade offs
- includes a GPU/CPU implementation of the model
- is on Github

A LOT OF OPEN PROBLEMS

BUT IT LOOKS PROMISING

- And a lot of unconventional ways to solve them
- PGAS for NUMA effects?
- Use SA memory to deal with missing cache coherency?
  - and allows direct support for RDMA
- Dataflow-like synchronization?

Thank you for listening!

:-)