

# Chapel Hierarchical Locales:

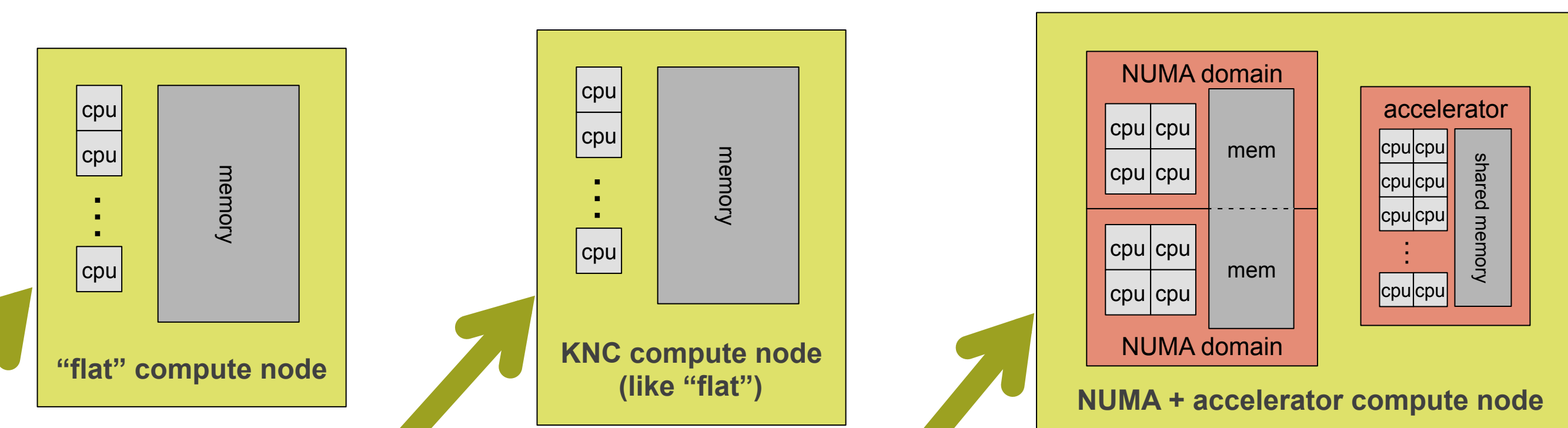
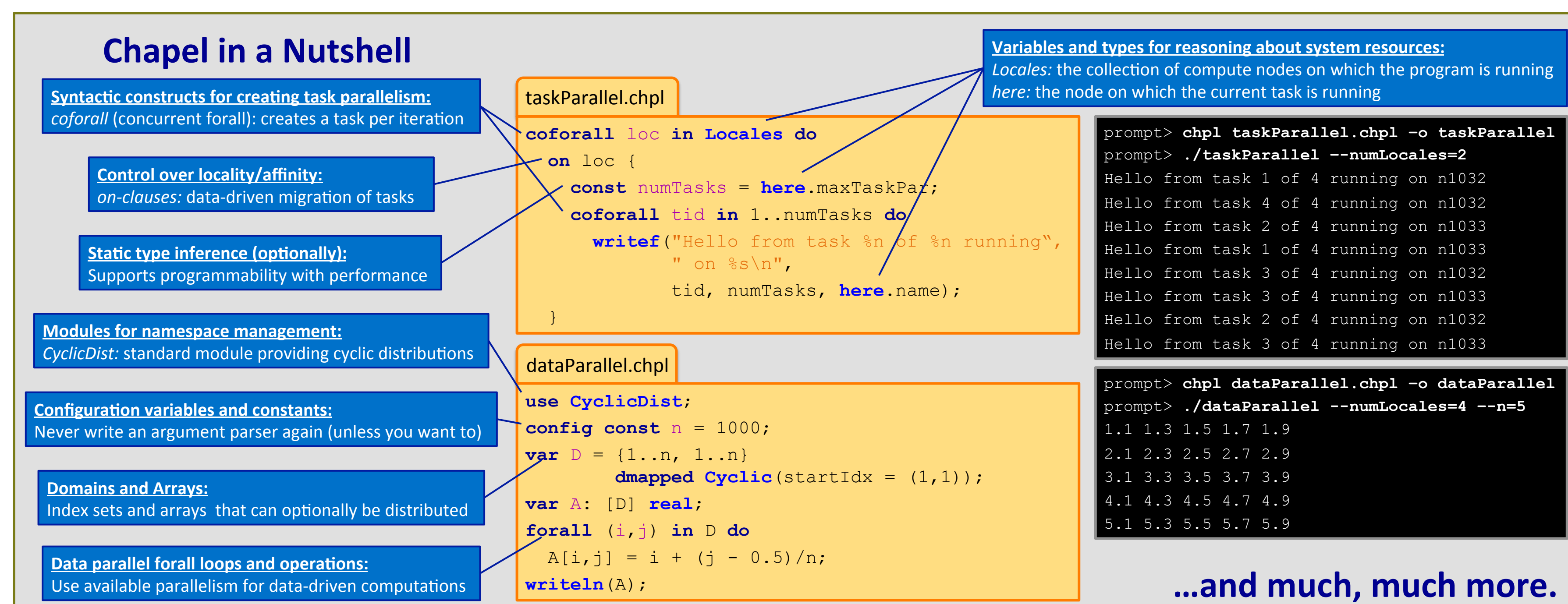
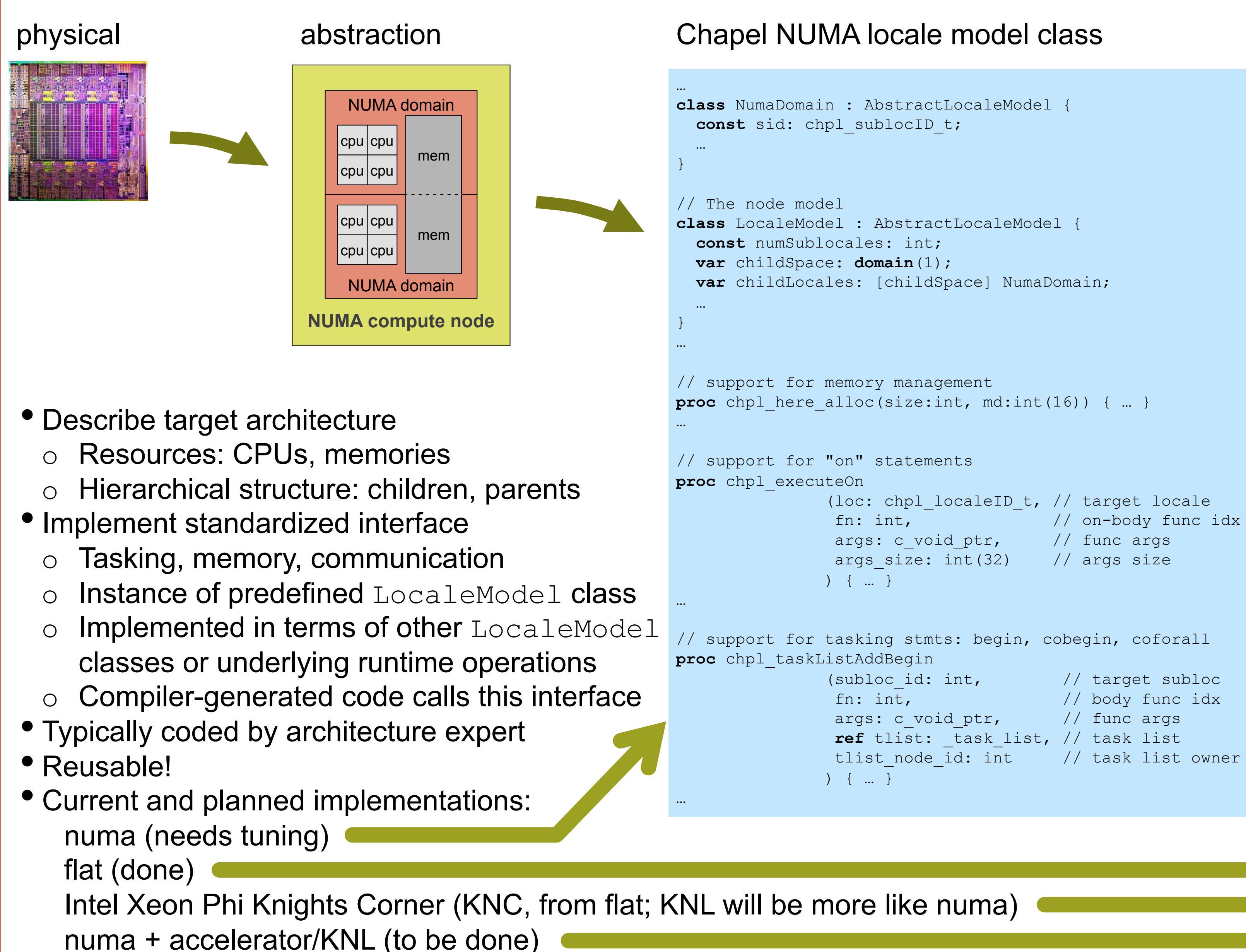
## Adaptable Portability for Exascale Node Architectures

### Complicated compute node architectures got you down?

Wishing for a programming model that adapts to new architectures so your applications don't have to?

### Our Solution:

#### Chapel's Hierarchical Locale Models



architecture & resources

#### Chapel's Domain Maps

- Describe distribution of indices (block, cyclic, etc.)
- Tie together locality, parallelism, affinity
- Interrogate locale model regarding resources
- Translate abstract parallelism into actual
- Typically coded by affinity/locality expert
- Reusable!

```

// Block domain map
class Block: BaseDist {
  param rank: int;
  type idxType = int;
  var boundingBox: domain(rank, idxType);
  var targetLocDom: domain(rank);
  var targetLocales: [targetLocDom] locale;
  var locDist: [targetLocDom] LocBlock(rank, idxType);
  var dataParTasksPerLocale: int;
  var dataParIgnoreRunningTasks: bool;
  var dataParMinGranularity: int;
}
  
```

locality & parallelism

#### Chapel Applications

- Specify domain maps in application code
- Express parallelism abstractly
  - no reference to physical architecture
- Specify locale model as part of Chapel configuration when compiling application

```

// Stream Triad
config const m = 1000,
              alpha = 3.0;
const ProblemSpace = {1..m} dmapped Block(...);
var A, B, C: [ProblemSpace] real;
B = 2.0;
C = 3.0;
A = B + alpha * C;
  
```

Chapel compiler

Conceptual translation  
Actual translation

#### Parallel Implementation

conceptually equivalent Chapel code

```

coforall loc in targetLocales do on loc {
  coforall subloc in loc.getChildren() do on subloc {
    coforall tid in here.NumCores {
      for (a,b,c) in zip(A,B,C) {
        a = b + alpha * c;
      }
    }
  }
}
  
```

emitted C code with calls to Chapel runtime etc.

```

void main(...) {
  chpl_taskListAddCoStmt(fn_for_outer_coforall_stmt);
}

void fn_for_outer_coforall_stmt(...) {
  chpl_executeOn(loc, fn_for_on_stmt);
}

void fn_for_on_stmt(...) {
  chpl_taskListAddCoStmt(fn_for_inner_coforall_stmt);
}

void fn_for_inner_coforall_stmt(...) {
  for (...) { a[i] = b[i] + alpha * c[i]; }
}
  
```

#### Next Steps

- Enhance and tune NUMA locale model
- Tune Intel KNC locale model
- Start on Intel KNL and accelerator

