European Processor Initiative

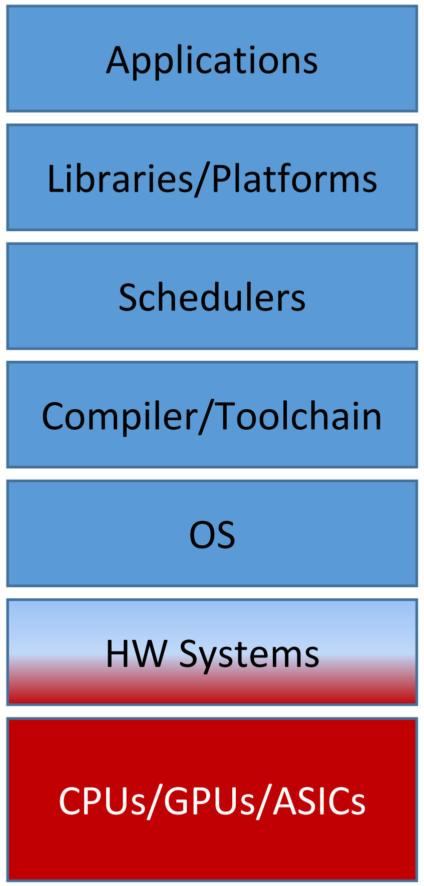
THE RISC-V VECTOR PROCESSOR IN EPI

JESUS LABARTA (@BSC.ES)

DISCLAIMER

- Personal opinions
- I grew up in HPCland

TOWARDS PROCESSOR DESIGN IN EUROPE



European Processor Initiative

TOWARDS PROCESSOR DESIGN IN EUROPE

Applications

Libraries/Platforms

Schedulers

Compiler/Toolchain

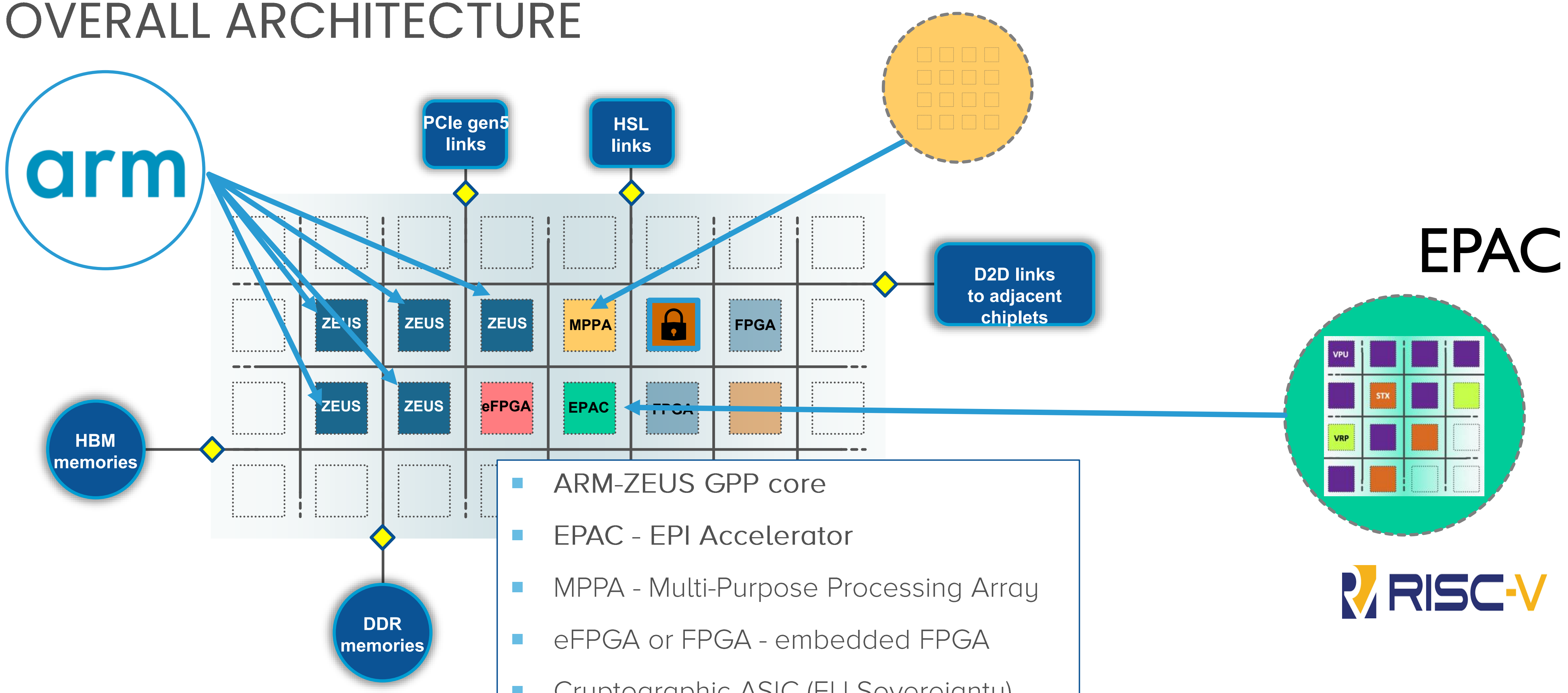
OS

HW Systems

CPUs/GPUs/ASICs

European Processor Initiative

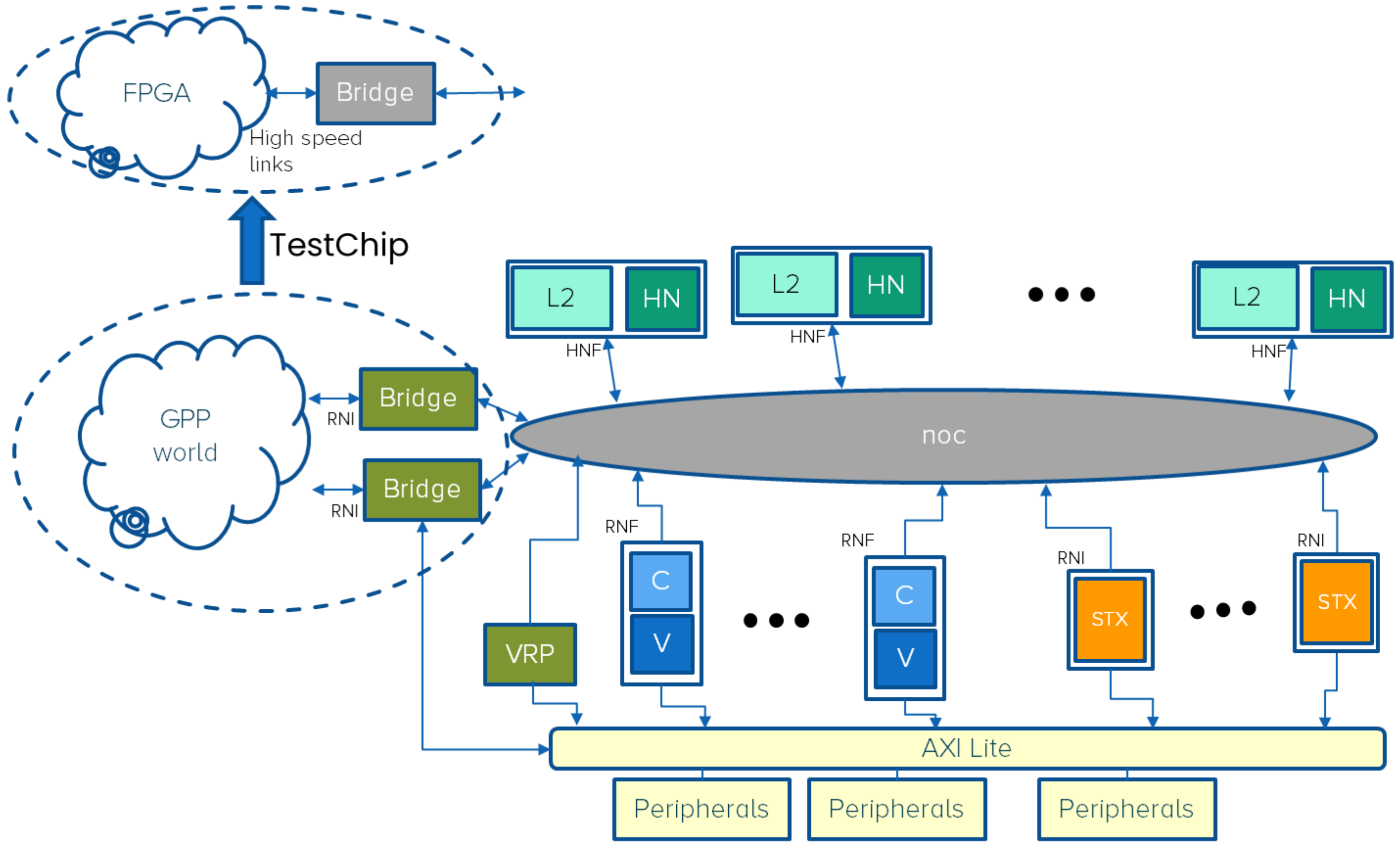
OVERALL ARCHITECTURE



- ARM-ZEUS GPP core
- EPAC - EPI Accelerator
- MPPA - Multi-Purpose Processing Array
- eFPGA or FPGA - embedded FPGA
- Cryptographic ASIC (EU Sovereignty)

EPAC

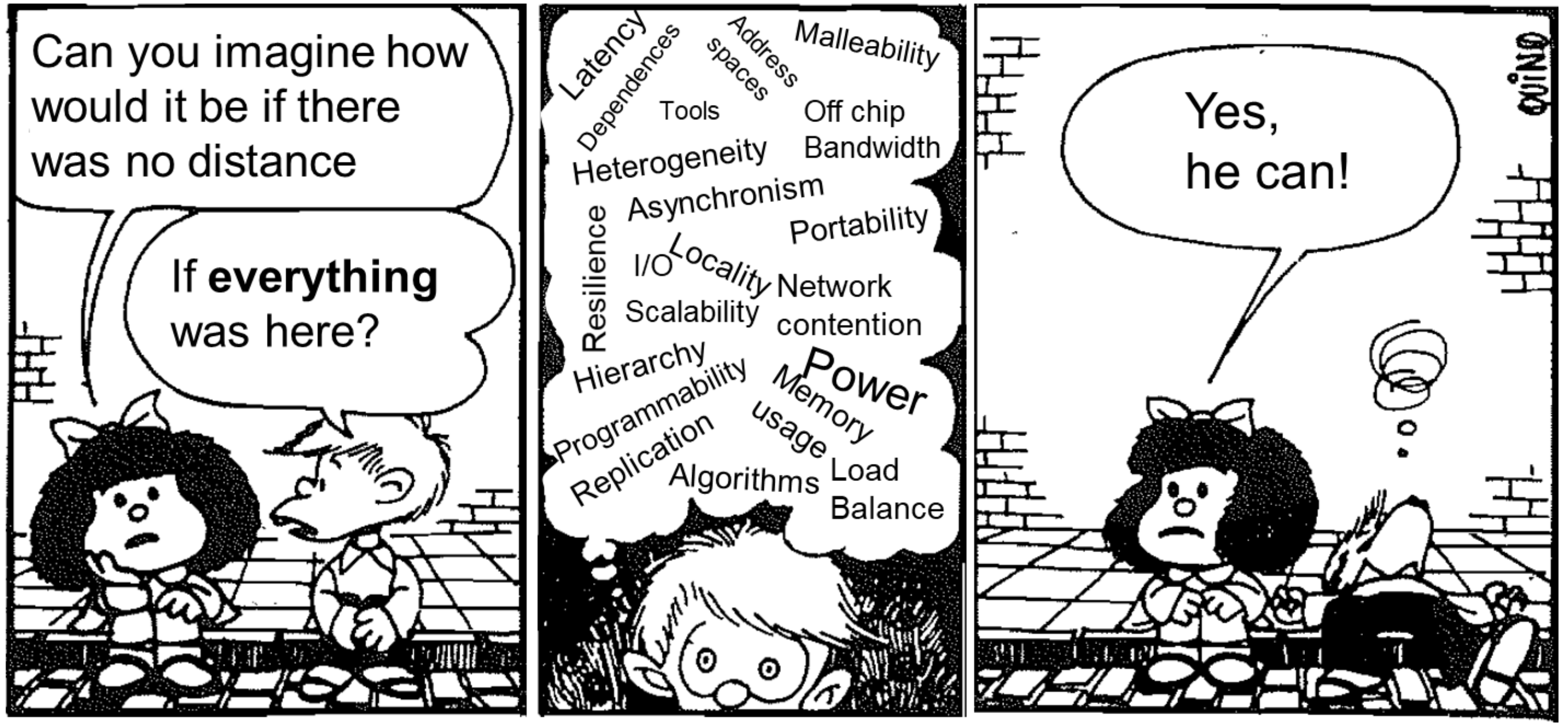
- RV64GCV (→ 8x)
  - 2 way in order core
  - Decoupled OoO VPU
    - 8 lanes
    - Long vectors (256 DP elements)
  - L1 - MESI coherency
- CHI interface NoC
  - 1 line / cycle (high B/F)
- L$2: 256KB/module
  - Allocation control mechanisms
- Linux



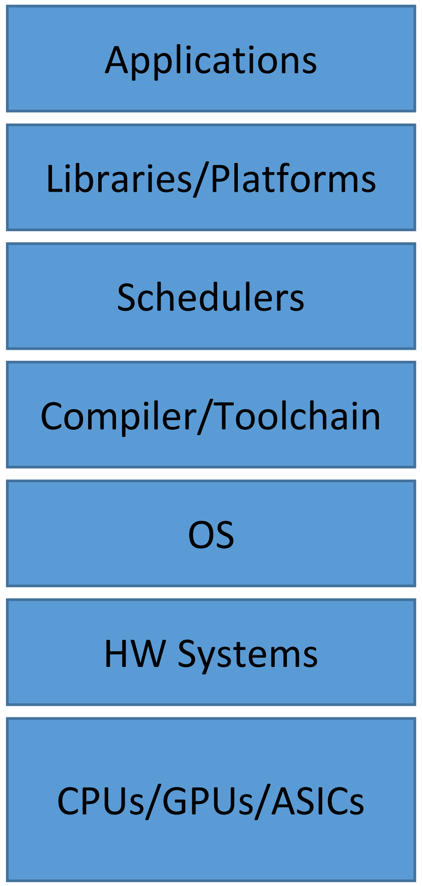
- STX: DL and stencil specific accelerators
- VRP: variable precision processors

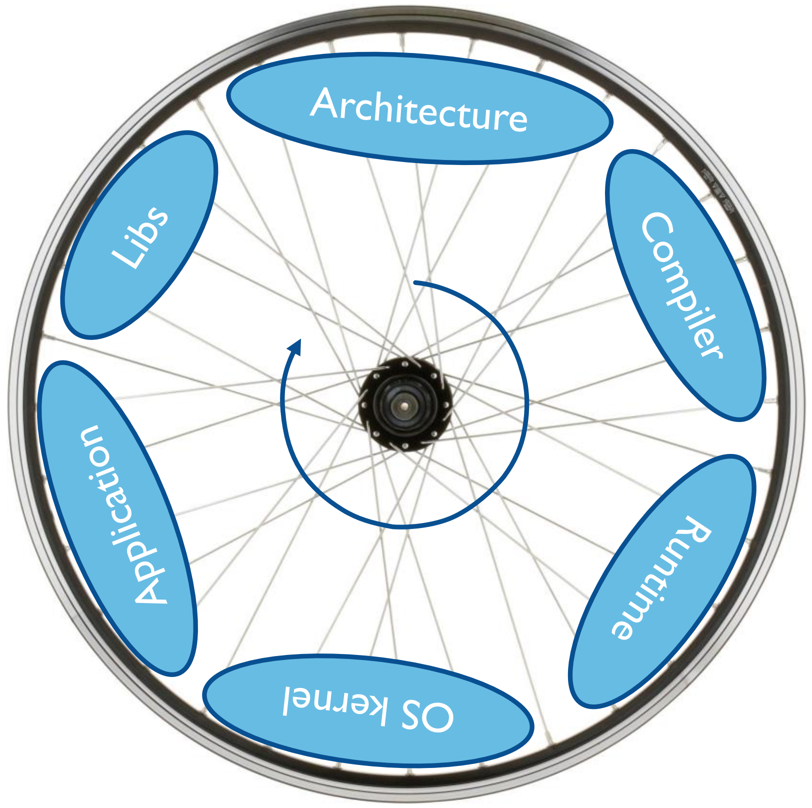
TOWARDS HOLISTIC CO-DESIGN

- Can we develop a unified model? Nicely integrate all levels?
- How do we ensure coordination/cooperation between levels at run time?



HOLISTIC CO-DESIGN





Best place to address an issue

Fundamentals

Balance

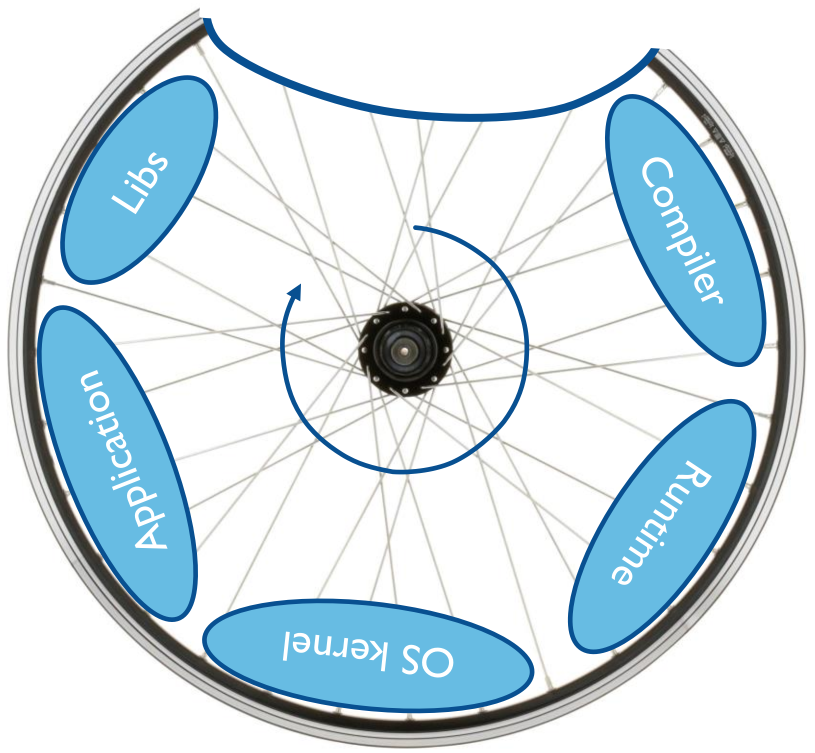
Mindset

Productivity

Efficiency

HOLISTIC CO-**DESIGN**

Applications
Libraries/Platforms
Schedulers
Compiler/Toolchain
OS
HW Systems
CPUs/GPUs/ASICs



Best place to address an issue

Fundamentals

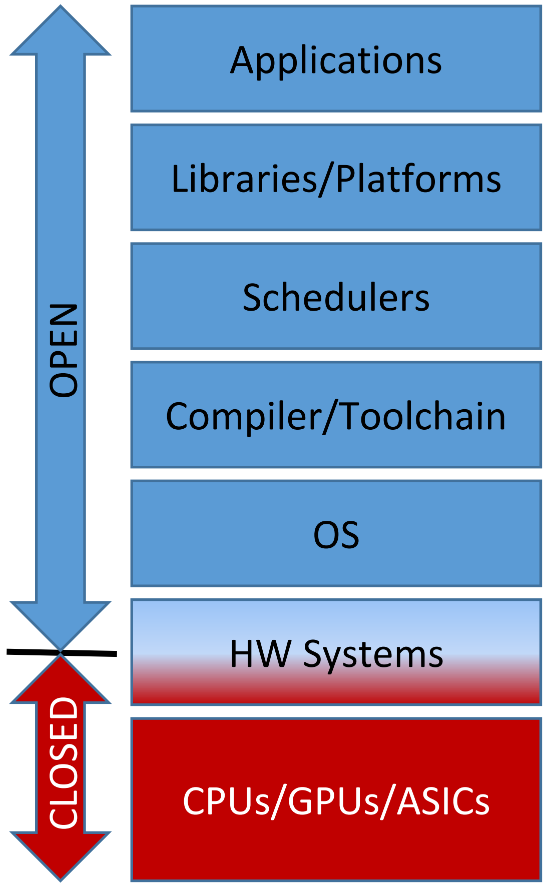
Balance

Mindset

Productivity

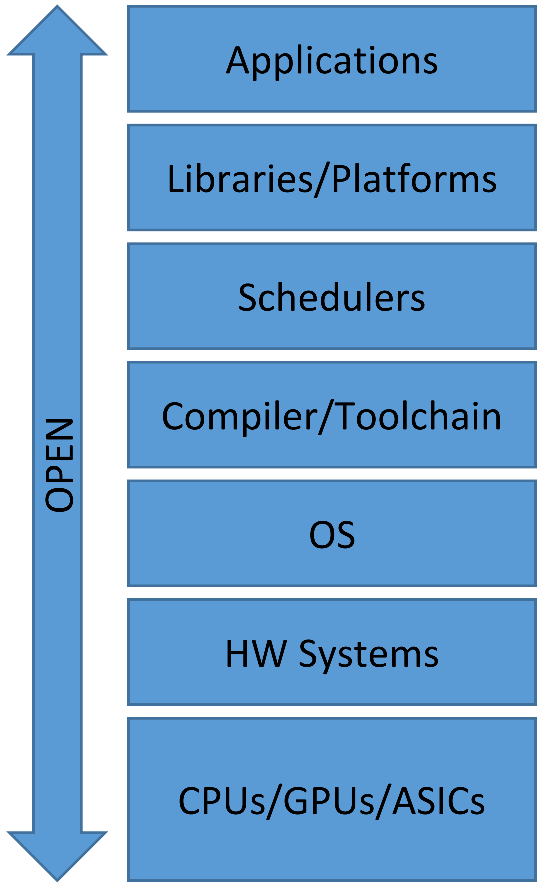
Efficiency

LEVERAGE INTERFACES AND IMPLEMENTATIONS



Leverage "standards"
Opportunity to innovate and contribute

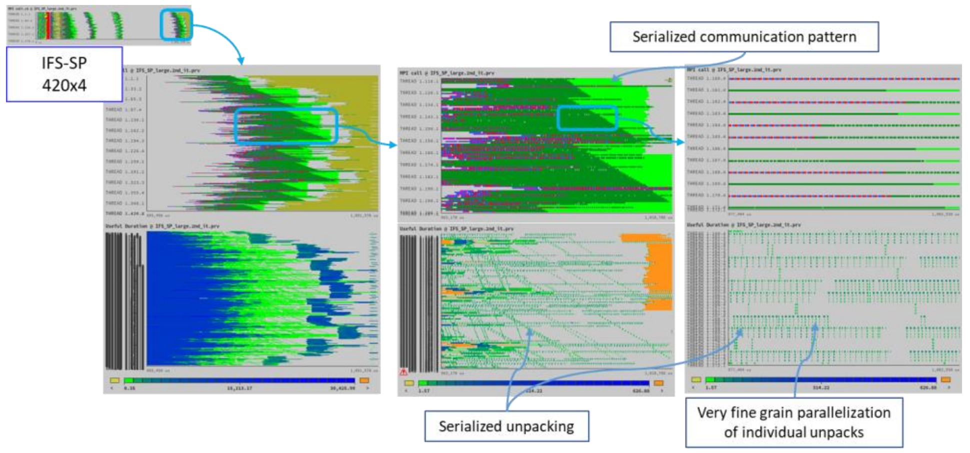
LEVERAGE INTERFACES AND IMPLEMENTATIONS

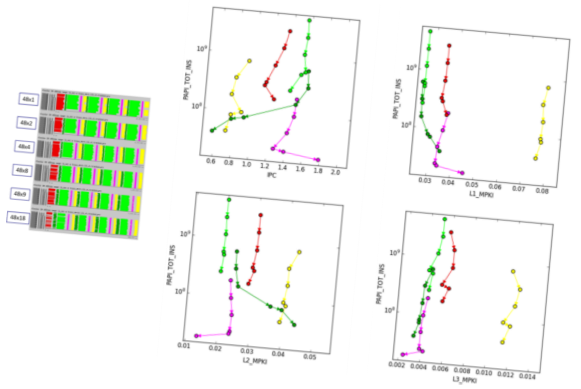


Leverage "standards"
Opportunity to innovate and contribute

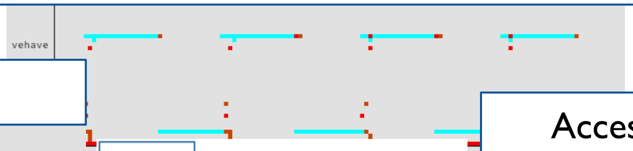
DETAILED ANALYSIS AND INSIGHT ON BEHAVIOR

- Applications
- Libraries/Platforms
- Schedulers
- Compiler/Toolchain
- OS
- HW Systems
- CPUs/GPUs/ASICs

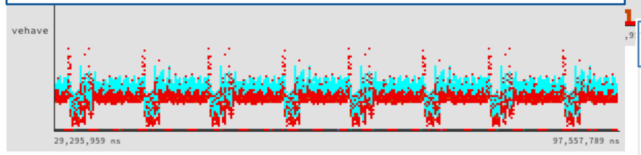

IFS-SP
420x4
Serialized communication pattern
Serialized unpacking
Very fine grain parallelization of individual unpacks



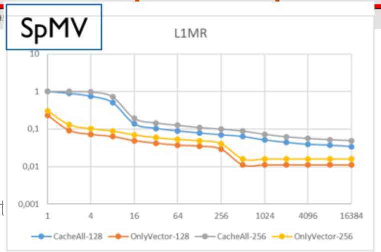
LRU Stack distance (colored by instr)



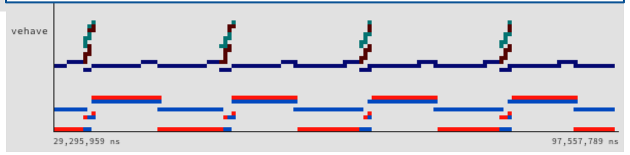
Instr timing (colored by instr)



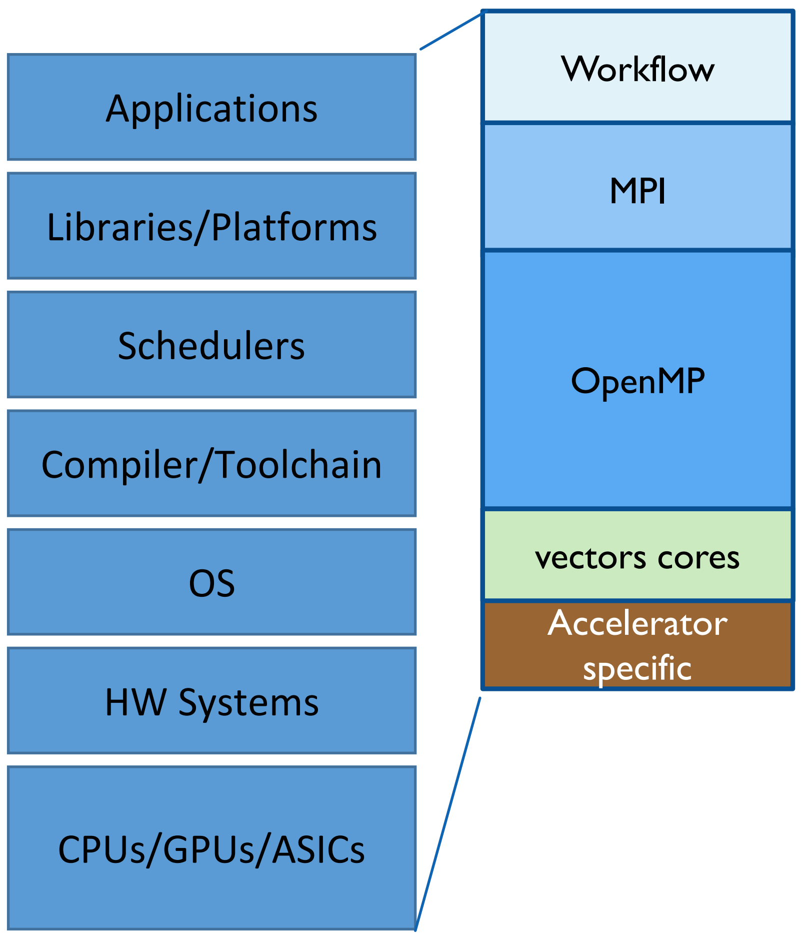
SpMV

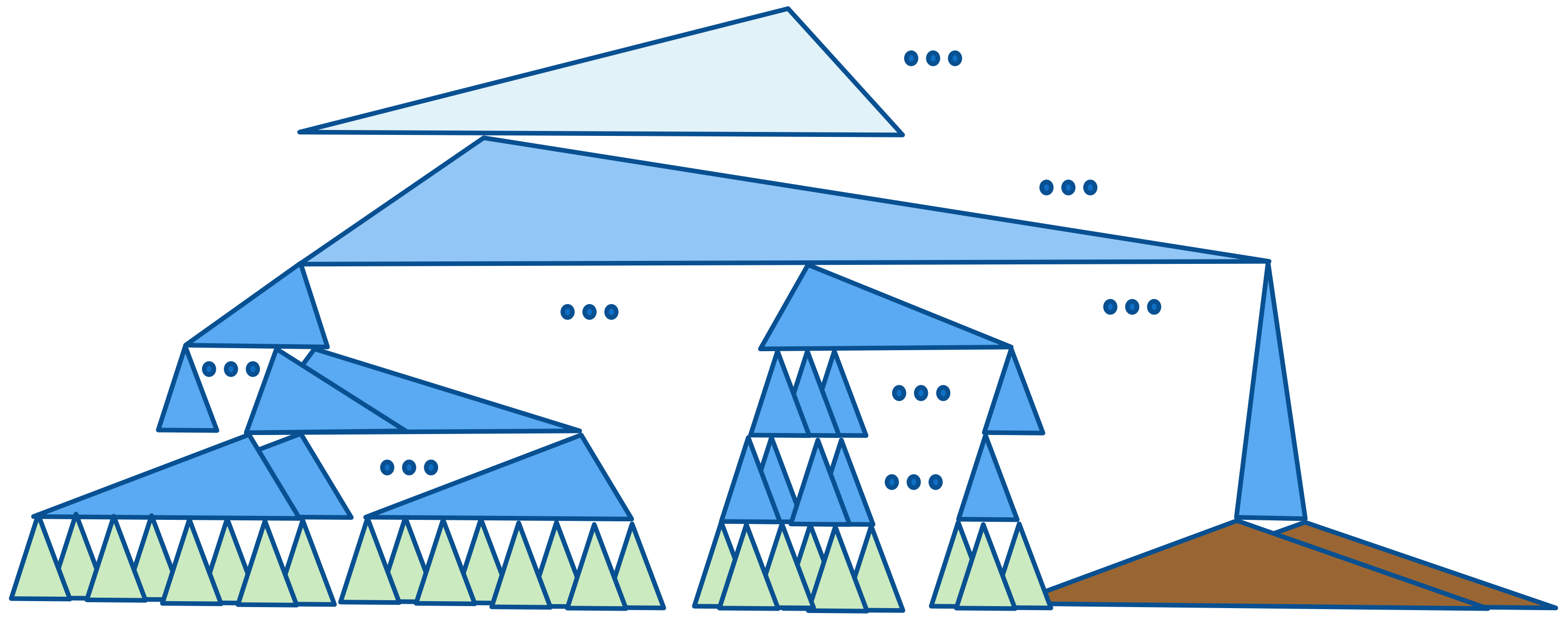


Access pattern (colored by PC)



BALANCED HIERARCHY





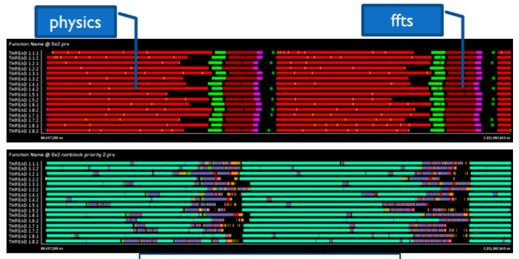
“Limited” number of “general purpose” control flows within tile
Long vectors. 8 lanes per core

LATENCY → THROUGHPUT: ASYNCHRONY AND OVERLAP

- Applications
- Libraries/Platforms
- Schedulers
- Compiler/Toolchain
- OS
- HW Systems
- CPUs/GPUs/ASICs

Interoperability MPI + OpenMP

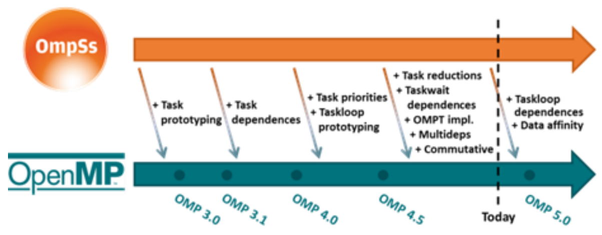
- Taskify MPI calls



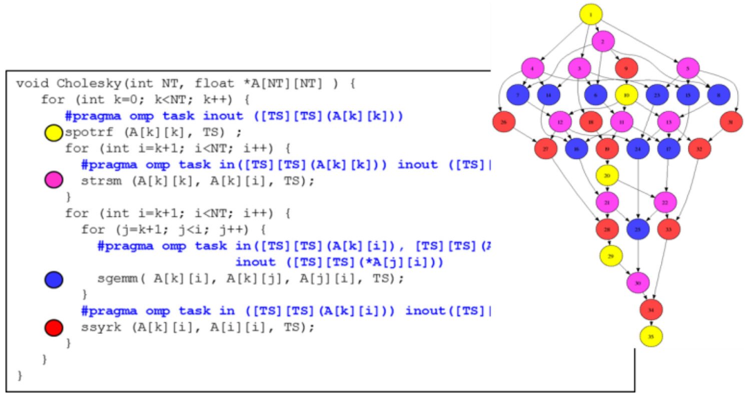
IFS weather code kernel. ECMWF

Task based models

- Single mechanism
  - Concurrency
  - Locality & data management



Task based computational workflows

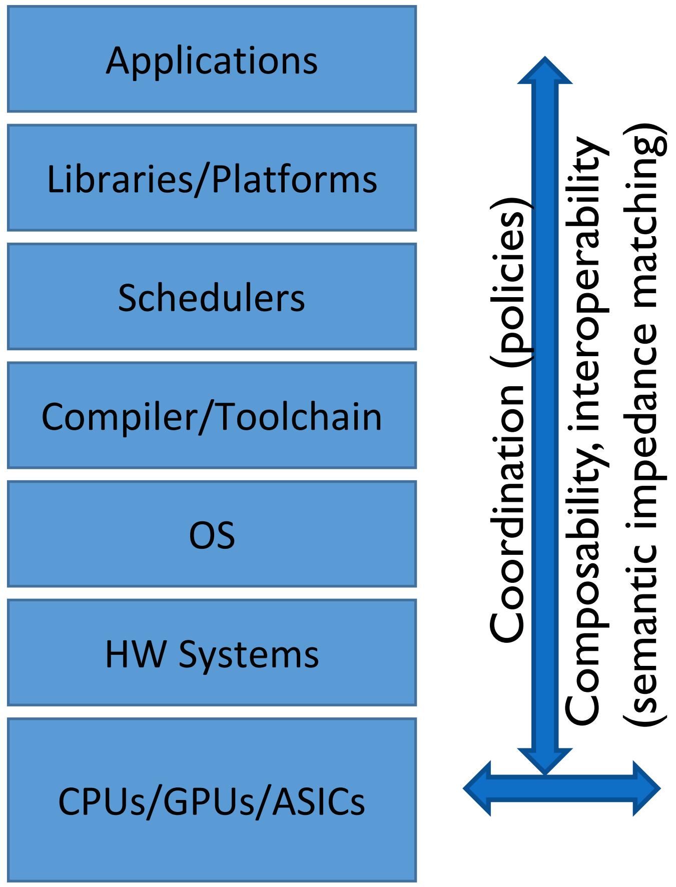


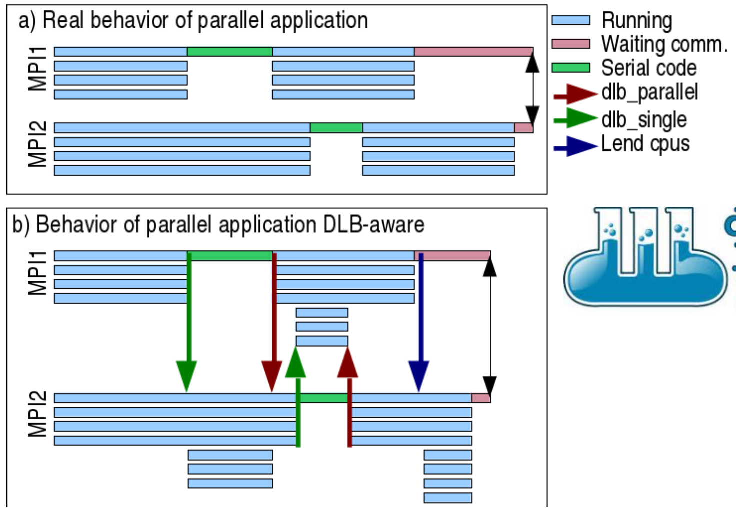
```
void Cholesky(int NT, float *A[NT][NT] ) {
   for (int k=0; k<NT; k++) {
      #pragma omp task inout ([TS][TS](A[k][k]))
      spotrf (A[k][k], TS) ;
      for (int i=k+1; i<NT; i++) {
        #pragma omp task in([TS][TS](A[k][k])) inout ([TS]|
        strsm (A[k][k], A[k][i], TS);
      }
      for (int i=k+1; i<NT; i++) {
        for (j=k+1; j<i; j++) {
          #pragma omp task in([TS][TS](A[k][i]), [TS][TS](/
                                inout ([TS][TS](*A[j][i]))
          sgemm( A[k][i], A[k][j], A[j][i], TS);
        }
        #pragma omp task in ([TS][TS](A[k][i])) inout([TS]|
        ssyrk (A[k][i], A[i][i], TS);
      }
   }
}
```

Long vectors

- decouple Front end – back end
- Convey access pattern semantics to the architecture. Potential to optimize memory throughput:

MALLEABILITY & COORDINATED SCHEDULING




a) Real behavior of parallel application
MPI1
MPI2
Running
Waiting comm.
Serial code
dlb_parallel
dlb_single
Lend cpus
b) Behavior of parallel application DLB-aware
MPI1
MPI2

dynamic load balancing

A wish:
Handoff scheduling

Vector Length Agnostic (VLA) programming and architecture

ISA

RISC-V

Micro architecture decides

```
saxpy:
    vsetvli a4, a0, e32, m8
    vlw.v v0, (a1)
    sub a0, a0, a4
    slli a4, a4, 2
    add a1, a1, a4
    vlw.v v8, (a2)
    vfmacc.vf v8, fa0, v0
    vsw.v v8, (a2)
    add a2, a2, a4
    bnez a0, saxpy
    ret
```

European Processor Initiative

HOMOGENIZING HETEROGENEITY

- Applications
- Libraries/Platforms
- Schedulers
- Compiler/Toolchain
- OS
- HW Systems
- CPUs/GPUs/ASICs

```
void axpy_omp_nest    (double a, double *dx, double *dy, int n) {
   int i, chunk;
   #pragma omp taskloop
   for (i=0; i<n; i+=TS) {
      chunk= n>i+TS? TS : n-i;
      #pragma omp target map(to:dx[i:i+chunk], tofrom:dy[i:i+chunk])
      axpy_omp        (a, &dx[i], &dy[i], chunk);
   }
```

```
void axpy_omp         (double a, double *dx, double *dy, int n) {
   int I, chunk;
   #pragma omp taskloop
   for (i=0; i<n; i+=TS) {
      chunk= n>i+TS? TS : n-i;
      axpy_SIMD       (a, &dx[i], &dy[i], chunk);
   }
}
```

```
void axpy_SIMD        (double a, double *dx, double *dy, int n) {
   int i;
   #pragma omp simd
   for (i=0; i<n; i++) dy[i] += a*dx[i];
}
```

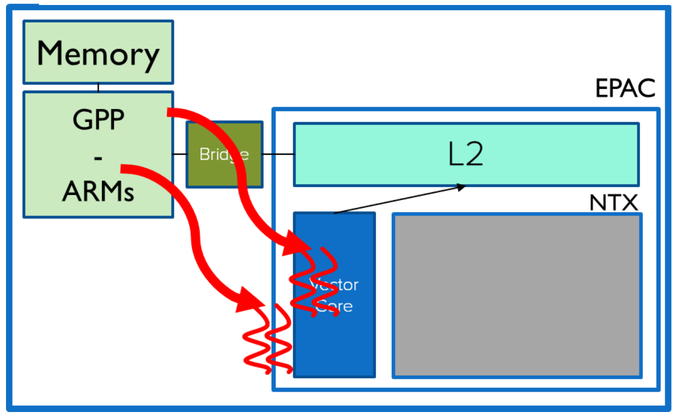
VLA helps homogenize Heterogeneous Performance

- ~ Big – Little cores, ...

Nested tasked/workshared

OpenMP

- Offload regular OpenMP

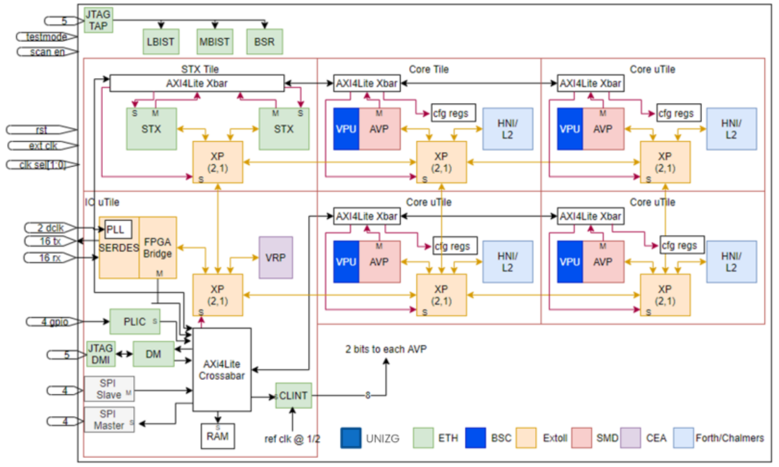


- HW support: IO coherence

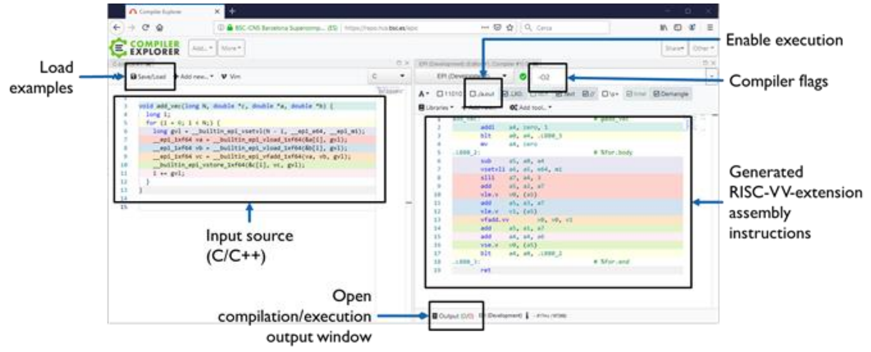
European Processor Initiative

WHERE WE ARE?

EPAC Test chip Tapepout @ 22nm @beginning of 2021



LLVM Vectorizing compiler & intrinsics



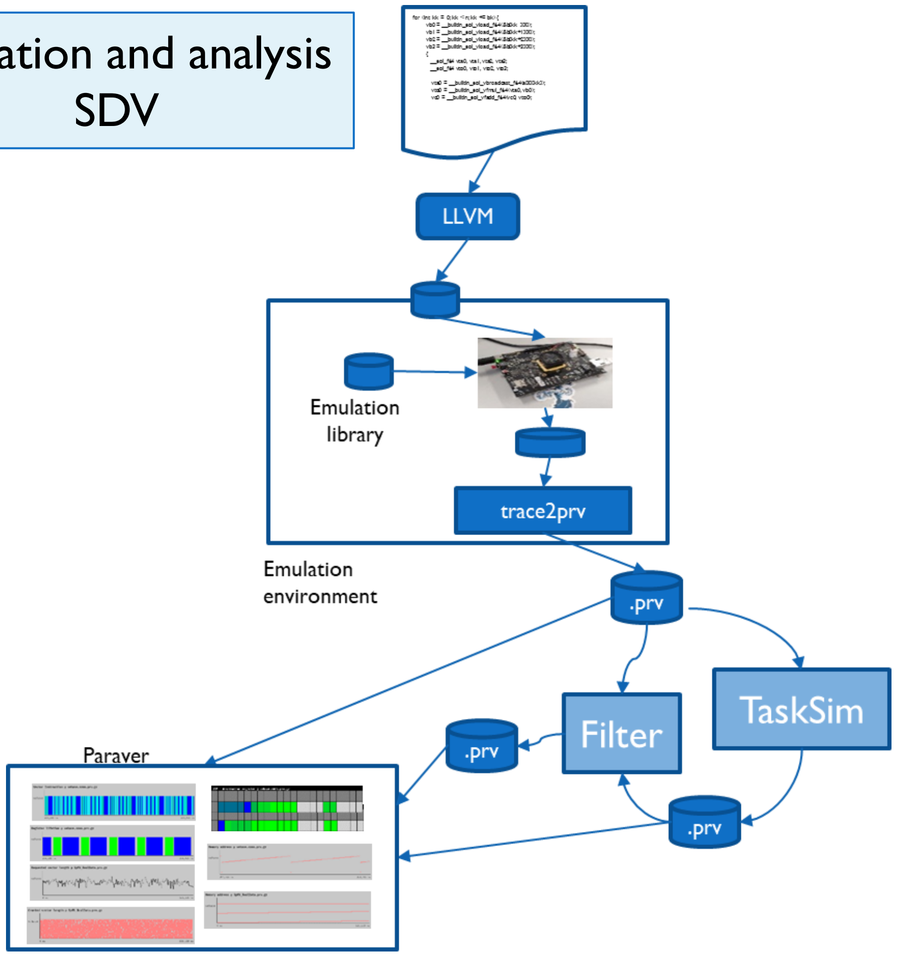
COMPILER EXPLORER
Load examples
Enable execution
Compiler flags
Generated RISC-VV-extension assembly instructions
Input source (C/C++)
Open compilation/execution output window

```
void axpy_SIMD        (double a, double *dx, double *dy, int n) {
   int i;


   #pragma omp simd
   for (i=0; i<n; i++) {
      dy[i] += a*dx[i];
   }
}
```

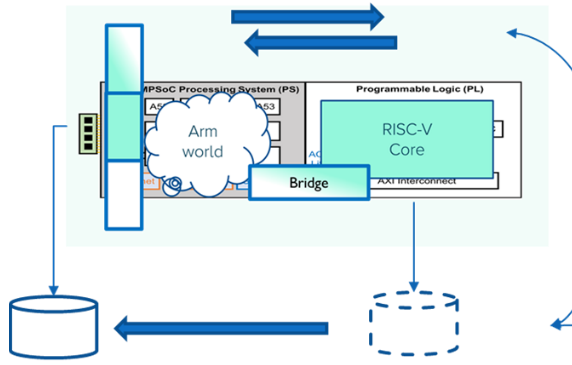
```
void axpy_intrinsics  (double a, double *dx, double *dy, int n) {
   int i;
   int gvl = __builtin_epi_vsetvl(n, __epi_e64, __epi_m1);
    __epi_1xf64 v_a = __builtin_epi_vbroadcast_1xf64(a, gvl);

   for (i=0; i<n; ) {
       gvl = __builtin_epi_vsetvl(n - i, __epi_e64, __epi_m1);
       __epi_1xf64 v_dx = __builtin_epi_vload_1xf64(&dx[i], gvl);
       __epi_1xf64 v_dy = __builtin_epi_vload_1xf64(&dy[i], gvl);
       __epi_1xf64 v_res = __builtin_epi_vfmacc_1xf64(v_dy, v_a, v_dx, gvl);
       __builtin_epi_vstore_1xf64(&dy[i], v_res, gvl);
       i += gvl;
    }
}
```

WHERE WE ARE ?

Emulation and analysis SDV



Heterogeneous System Software SDV



```
#on RISC-V side
$mkfs.ext4 -b 4096 /dev/vda
$mount -t ext4 /dev/vda /mnt/scratchfs
```

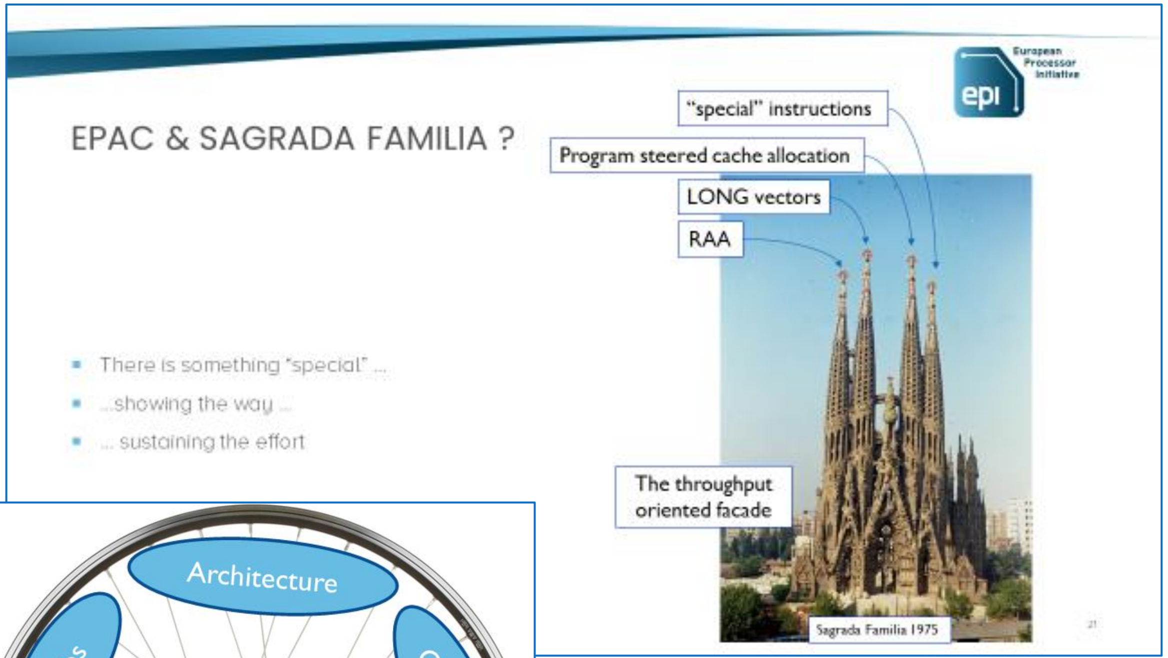
```
void main (…)
{ …
   gemTarget(A, B, C, size);
… }

void gemTarget(double *A, double *B, double *C, long S) {
   #pragma omp target map(to: A[0:S*S], B[0:S*S], S) \
                      map(from: C[0:S*S])
   {    for(int i = 0; i < size; i++)
          for(int j = 0; j < size; j++)
             for(int k = 0; k < size; k++)
                C[i*size + j] += A[i*size + k]*B[k*size + j];
        fp = fopen ("/mnt/scratchfs/output_matrix.txt", "w+");
        for (int i=0;i<size*size; i++) fprintf(fp, "%lf ", C[i]);
   }
}
```

EPAC SDV

EPAC

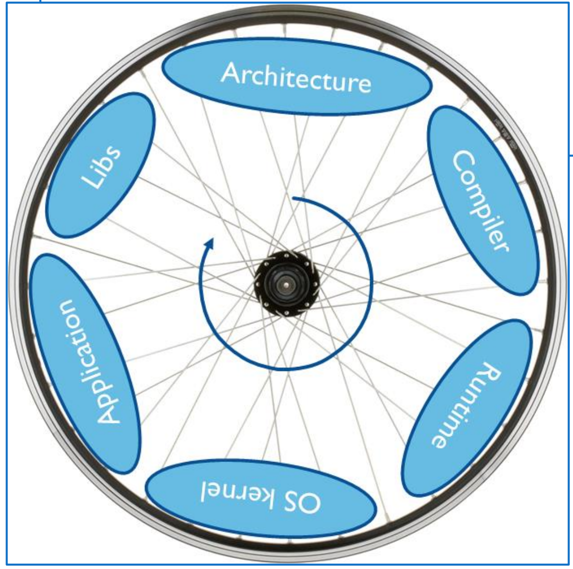
- Holistic throughput oriented vision based on long vectors and task based models
- Hierarchical concurrency and locality exploitation
- Not massive concurrency at a given level
- Push behaviour exploitation to low levels
- Co-ordination between levels
- Make it all look very close to classical sequential programming to ensure productivity

- Contact us if you are interested in evaluating the framework and provide co-design input


EPAC & SAGRADA FAMILIA ?

- There is something "special" ...
- ...showing the way ...
- ... sustaining the effort

"special" instructions
Program steered cache allocation
LONG vectors
RAA
The throughput oriented facade
Sagrada Familia 1975


Architecture
Compiler
Runtime
OS kernel
Application
Libs