Supporting RISC-V Performance Counters Through Linux Performance Analysis Tools

João Mário Domingos
INESC-ID, Instituto Superior Técnico
University of Lisbon, Portugal
joao.mario@tecnico.ulisboa.pt

Tiago Rocha
INESC-ID, Instituto Superior Técnico
University of Lisbon, Portugal
tiagolopesrocha@tecnico.ulisboa.pt

Nuno Neves
INESC-ID, Instituto Superior Técnico
University of Lisbon, Portugal
nuno.neves@inesc-id.pt

Nuno Roma
INESC-ID, Instituto Superior Técnico
University of Lisbon, Portugal
nuno.roma@inesc-id.pt

Pedro Tomás
INESC-ID, Instituto Superior Técnico
University of Lisbon, Portugal
pedro.z.tomas@tecnico.ulisboa.pt

Leonel Sousa
INESC-ID, Instituto Superior Técnico
University of Lisbon, Portugal
las@inesc-id.pt

ASAP 2023 34th IEEE International Conference on Application-specific Systems, Architectures and Processors
JULY 19 - 21 2023, PORTO, PORTUGAL

Outline

- Motivation
- RISC-V HPM
- Perf_events, Perf and PAPI
- Proposed approach
- Results
- Next steps

Performance Monitoring:

why it matters for system designers and software developers

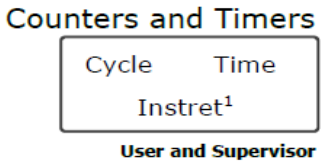
There are multiple ways to improve software performance:

- Execution time profiling (count cycles)
- Hardware simulators (software-based, mixed)
- Hardware Performance Monitors (through Perf and PAPI)
- Mixed/proprietary tools (Intel Advisor, Arm Coresight)

Want to get faster?
Profile and tune your system

RISC-V HPM:

the RISC-V hardware performance monitor



v1.7 - 2015

Perf and PAPI

Perf_events Kernel Driver:

- Communication between user-privilege level and HPM
- Low-level; programmer must know encodings

Perf Application:

- Uses the perf_events
- Low-level
- Default events and raw events

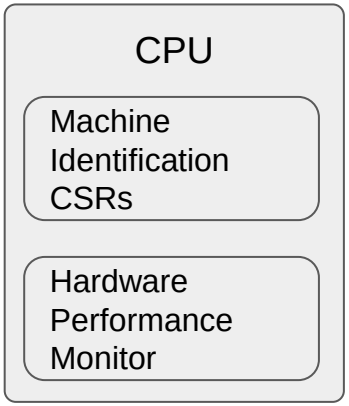
PAPI API:

- Also uses perf_events
- Portability

RISC-V HPM + Linux Perf

ASAP 2023 34th IEEE International Conference on Application-specific Systems, Architectures and Processors
JULY 19 - 21 2023, PORTO, PORTUGAL

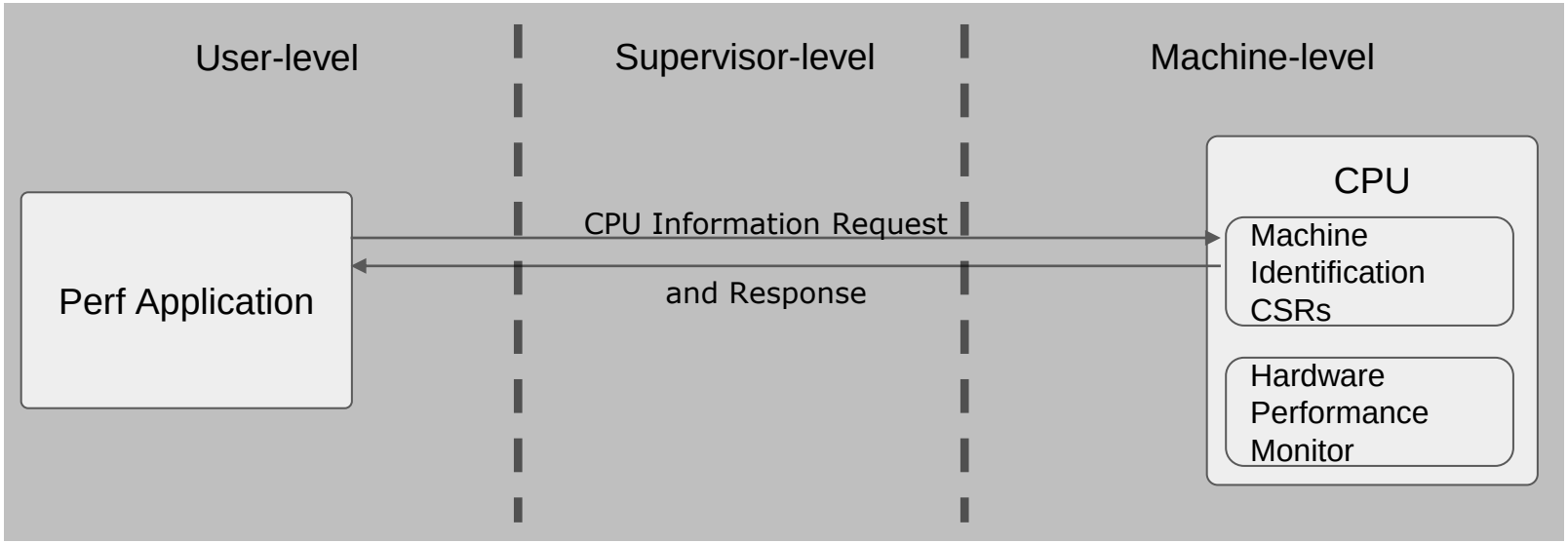
RISC-V HPM + Linux Perf:
Software Overview



- CPU events identification
- Event attribution and counting (through the driver)
- Display results in an uniform way

RISC-V HPM + Linux Perf:

Software Overview

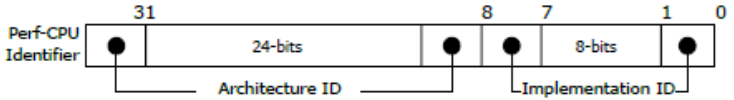


- **CPU events identification** **Contribution**
  - CPU PMU identification through the architecture ID and implementation ID
  - Each CPU unique ID selects the appropriate set of events for that processor

RISC-V HPM + Linux Perf:

CPU specific event selection with Perf pmu_events

Contributions



CPU PMU Unique ID (composed by architecture and implementation IDs)

Used to match the CPU PMU with a set of event description code

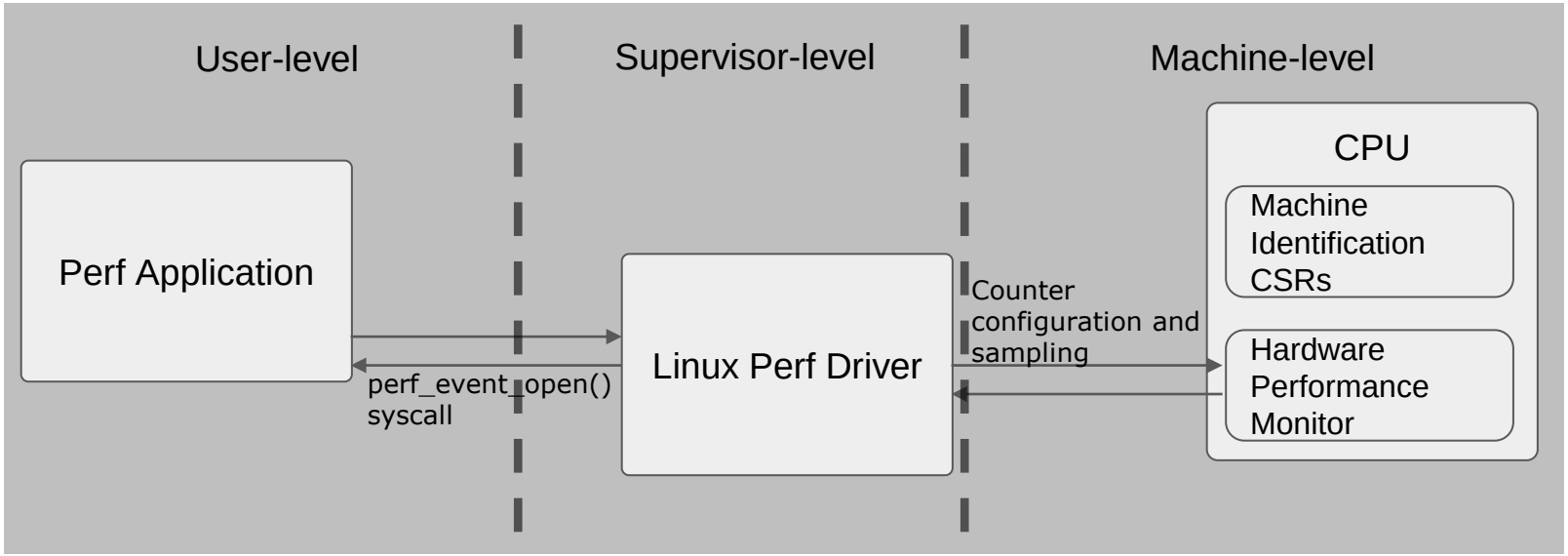
```
CPU Identifier, File Version, Events Filename, Events Type
0x300         , 0           , CVA6           , core
0x500         , 0           , SPIKE          , core
0x200         , 0           , BOOM           , core
...
```

```
{
    "Public Description": "This is an example event,
    for demonstration purposes.",
    "Brief Description": "This is an example event."
    "Event Code" : "0x11",
    "Counter Mask" : "0xF8FF",
    "Event Name" : "EXAMPLE_EVENT",
}
```

Events are simply described in a JSON format.
Each event is identified by the name, and provides an event code and a counter selection mask.

RISC-V HPM + Linux Perf:

Software Overview

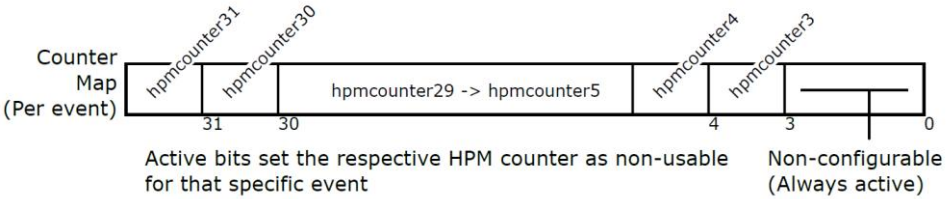


- **Event attribution and counting**
  - Perf Initiates performance counting through a system call
  - The Linux Perf driver setups events and samples counters

RISC-V HPM + Linux Perf:

Counting events with the perf kernel driver

Contributions

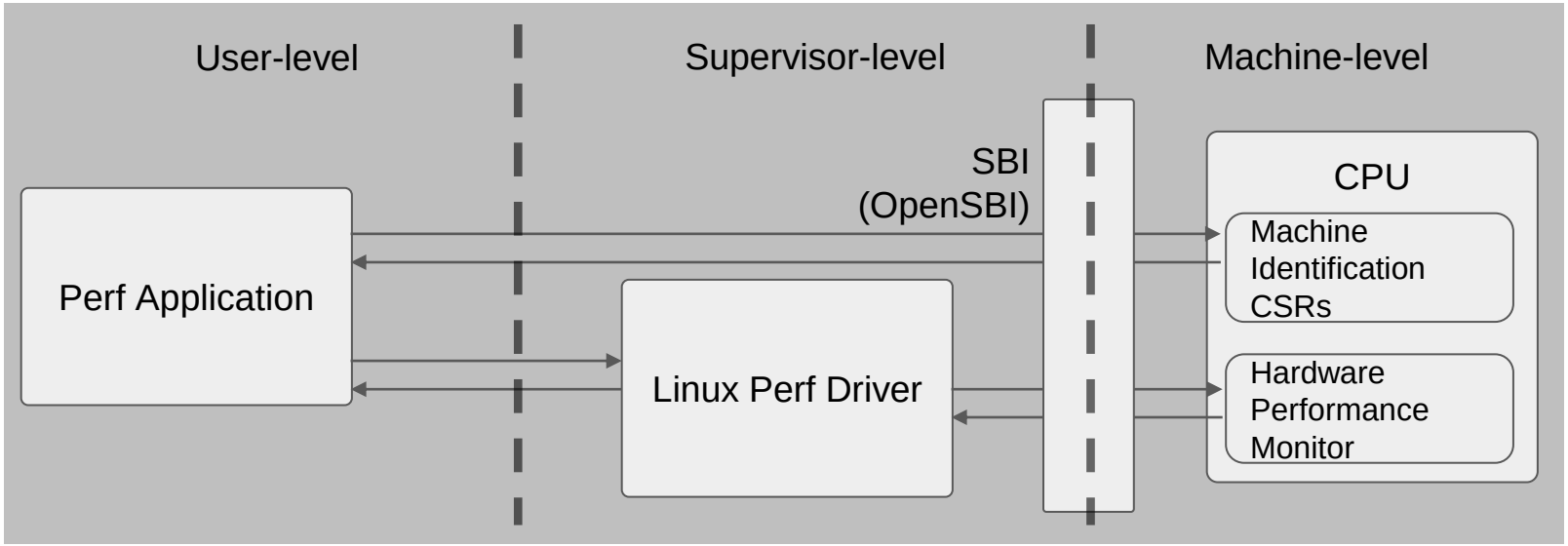


New perf event configuration allows for each event to have a counter mask, providing identification of available counters.

- Improved event configuration through the RISC-V *mhpmevent#* registers.
- Improved support for raw events

RISC-V HPM + Linux Perf:

Software Overview

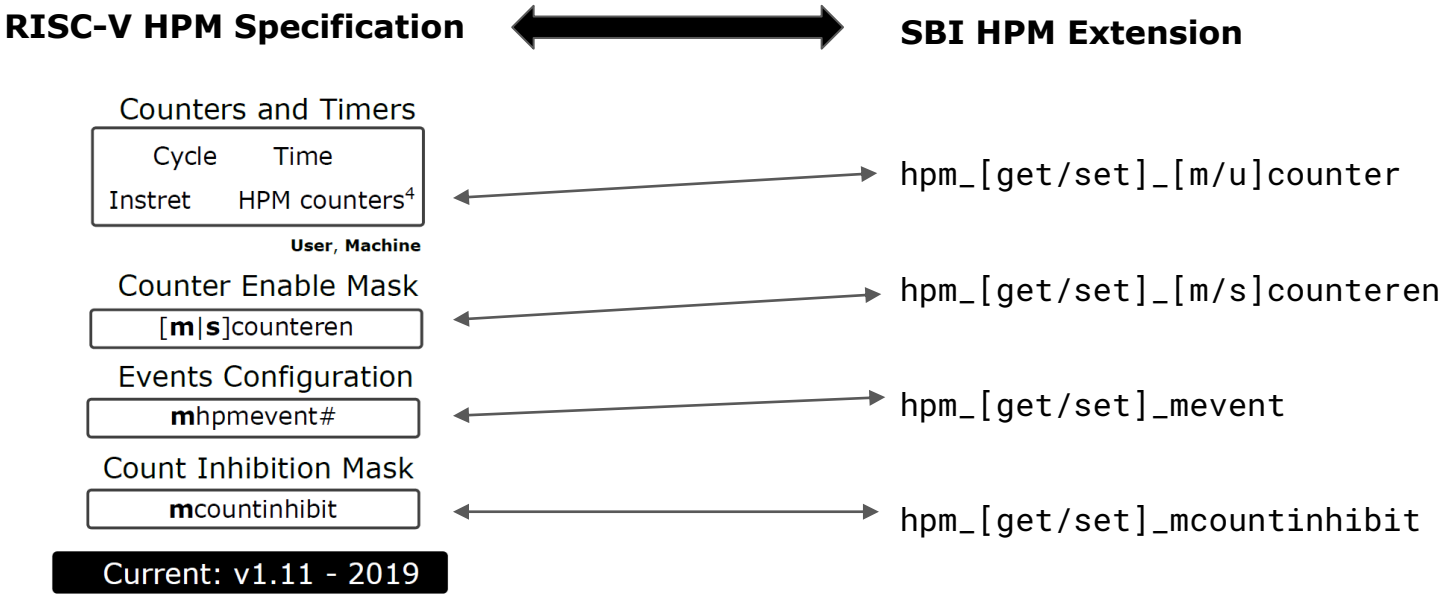


- **Privilege escalation to machine-mode** **Contribution**
  - SBI/OpenSBI extension for HPM counters interaction (configuration, read, write)
  - Added Perf and Perf Driver SBI calls for HPM extension and machine identification CSRs access

RISC-V HPM + Linux Perf:

Privileged access through OpenSBI

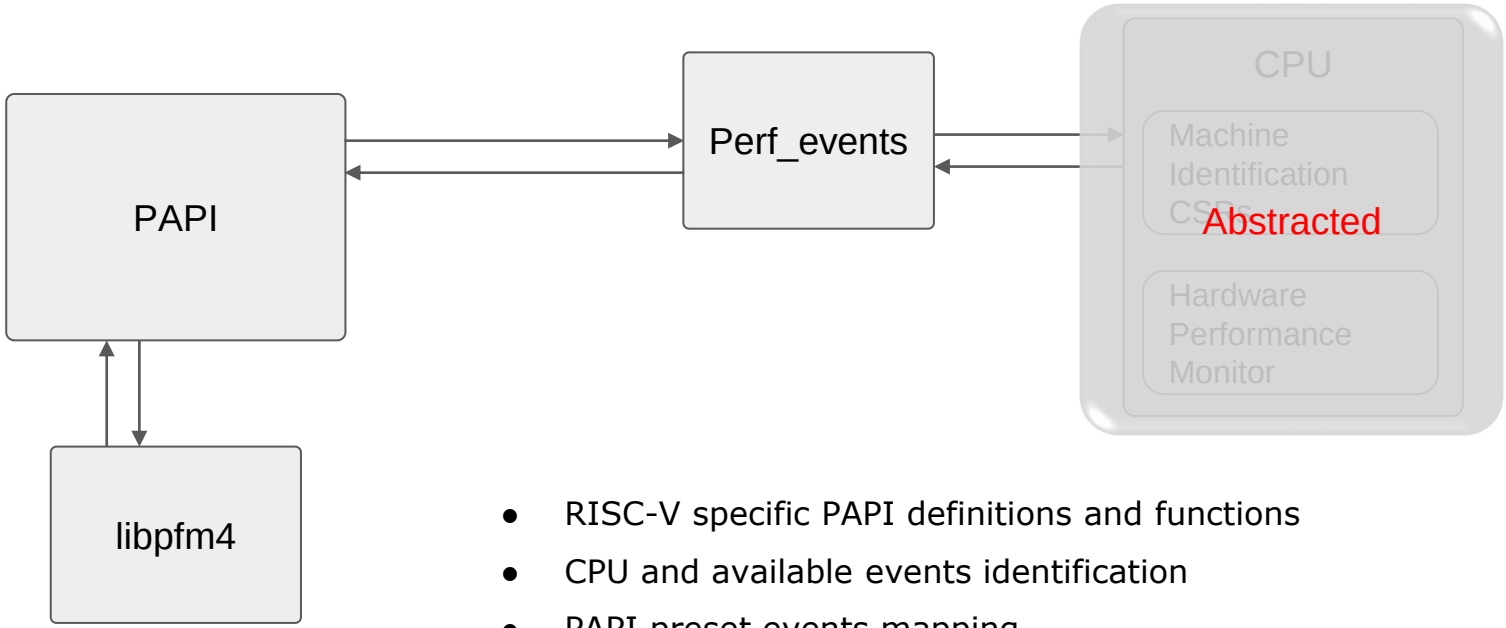
Contributions



RISC-V PAPI

PAPI (+libpfm4) + Perf_events:

Software Overview

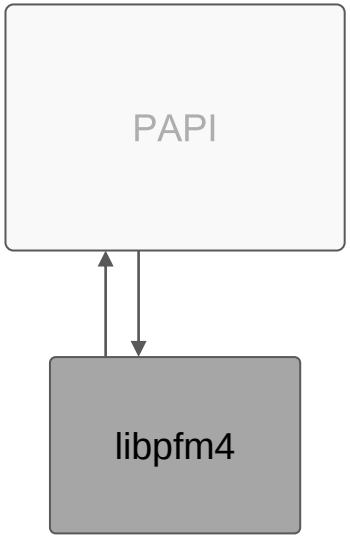


- RISC-V specific PAPI definitions and functions
- CPU and available events identification
- PAPI preset events mapping

PAPI (+libpfm4) + Perf_events:

libpfm4 Changes

Contribution



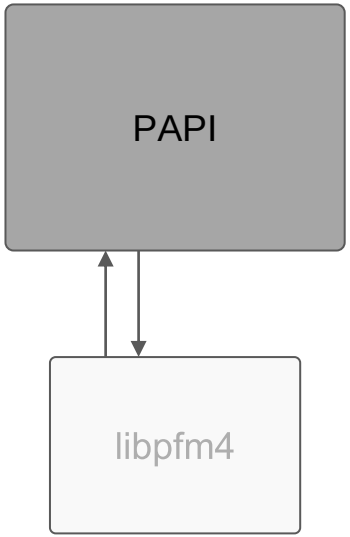
- **Event decoding and event list manipulation**
  - RISC-V event entry structure definition
  - Functions to decode events and produce perf_event control structures and to iterate over event list
  - Available events list
  - HPM model description (#available counters, supported event list, …)

```
{.name = “EXAMPLE_NAME”,
 .code = 0x0000100,
 .desc = “Example description”}
```

PAPI (+libpfm4) + Perf_events:

PAPI Changes

Contribution



- **CPU identification and preset event mapping**
  - CPU identification through /proc/cpuinfo file
  - Inline assembly functions required by PAPI
  - PC context location definition
  - PAPI preset event mapping to available hardware events

Initial Results

Working Perf:

the *perf list* command

```
/ # perf list
  branch-instructions OR branches                    [Hardware event]
  branch-misses                                      [Hardware event]
  cache-misses                                       [Hardware event]
  cache-references                                   [Hardware event]
  cpu-cycles OR cycles                               [Hardware event]
  instructions                                       [Hardware event]
  alignment-faults                                   [Software event]
  bpf-output                                         [Software event]
  context-switches OR cs                             [Software event]
  cpu-clock                                          [Software event]
  cpu-migrations OR migrations                       [Software event]
  dummy                                              [Software event]
  emulation-faults                                   [Software event]
  major-faults                                       [Software event]
  minor-faults                                       [Software event]
  page-faults OR faults                              [Software event]
  task-clock                                         [Software event]
  duration_time                                      [Tool event]
  L1-dcache-load-misses                              [Hardware cache event]
  L1-dcache-loads                                    [Hardware cache event]
  L1-dcache-stores                                   [Hardware cache event]
  L1-icache-load-misses                              [Hardware cache event]
  branch-load-misses                                 [Hardware cache event]
  branch-loads                                       [Hardware cache event]
  dTLB-load-misses                                   [Hardware cache event]
  iTLB-load-misses                                   [Hardware cache event]
```

CVA6 (Ariane) Events

Event	Counter
Cycles	mcycle
Instructions Retired	minstret
ICache Misses	mhpmcounter3
DCache Misses	mhpmcounter4
ITLB Misses	mhpmcounter5
DTLB Misses	mhpmcounter6
Loads	mhpmcounter7
Stores	mhpmcounter8

Event	Counter
Taken Exceptions	mhpmcounter9
Exceptions Returned	mhpmcounter10
Branches and Jumps	mhpmcounter11
Calls	mhpmcounter12
Returns	mhpmcounter13
Mispredicted Branches	mhpmcounter14
Scoreboard Full	mhpmcounter15
Instruction Fetch Empty	mhpmcounter16

Working Perf:

the *perf list* command

```
/ # perf list
  branch-instructions OR branches                    [Hardware event]
  branch-misses                                      [Hardware event]
  cache-misses                                       [Hardware event]
  cache-references                                   [Hardware event]
  cpu-cycles OR cycles                               [Hardware event]
  instructions                                       [Hardware event]
  alignment-faults                                   [Software event]
  bpf-output                                         [Software event]
  context-switches OR cs                             [Software event]
  cpu-clock                                          [Software event]
  cpu-migrations OR migrations                       [Software event]
  dummy                                              [Software event]
  emulation-faults                                   [Software event]
  major-faults                                       [Software event]
  minor-faults                                       [Software event]
  page-faults OR faults                              [Software event]
  task-clock                                         [Software event]
  duration_time                                      [Tool event]
  L1-dcache-load-misses                              [Hardware cache event]
  L1-dcache-loads                                    [Hardware cache event]
  L1-dcache-stores                                   [Hardware cache event]
  L1-icache-load-misses                              [Hardware cache event]
  branch-load-misses                                 [Hardware cache event]
  branch-loads                                       [Hardware cache event]
  dTLB-load-misses                                   [Hardware cache event]
  iTLB-load-misses                                   [Hardware cache event]
```

Supports Perf default Hardware and Hardware Cache events

Not all Ariane Events are supported as Perf default events, we need **raw events**

CVA6 (Ariane) Events

Event	Counter
Cycles	mcycle
Instructions Retired	minstret
ICache Misses	mhpmcounter3
DCache Misses	mhpmcounter4
ITLB Misses	mhpmcounter5
DTLB Misses	mhpmcounter6
Loads	mhpmcounter7
Stores	mhpmcounter8

Event	Counter
Taken Exceptions	mhpmcounter9
Exceptions Returned	mhpmcounter10
Branches and Jumps	mhpmcounter11
Calls	mhpmcounter12
Returns	mhpmcounter13
Mispredicted Branches	mhpmcounter14
Scoreboard Full	mhpmcounter15
Instruction Fetch Empty	mhpmcounter16

Working Perf:

the *perf list* command

```
branch:
  ariane_branch_jump
       [Ariane branches/jumps count]
  ariane_call
       [Ariane calls count]
  ariane_mis_predict
       [Ariane mis-predicted branches count]
  ariane_ret
       [Ariane returns count]

cache:
  ariane_dtlb_miss
       [Ariane data TLB miss]
  ariane_itlb_miss
       [Ariane instruction TLB miss]
  ariane_l1_dcache_miss
       [Ariane data cache misses]
  ariane_l1_icache_miss
       [Ariane instruction cache misses]
  ariane_load
       [Ariane data loads]
  ariane_store
       [Ariane data loads]
```

Events Grouping Supported

Support for all CVA6 events

CVA6 (Ariane) Events

Event	Counter
Cycles	mcycle
Instructions Retired	minstret
ICache Misses	mhpmcounter3
DCache Misses	mhpmcounter4
ITLB Misses	mhpmcounter5
DTLB Misses	mhpmcounter6
Loads	mhpmcounter7
Stores	mhpmcounter8

Event	Counter
Taken Exceptions	mhpmcounter9
Exceptions Returned	mhpmcounter10
Branches and Jumps	mhpmcounter11
Calls	mhpmcounter12
Returns	mhpmcounter13
Mispredicted Branches	mhpmcounter14
Scoreboard Full	mhpmcounter15
Instruction Fetch Empty	mhpmcounter16

Working Perf:

the *perf stat* command

```
CoreMark 1.0 : 173.984251 / GCC10.2.0 -O2 -static    / Heap

 Performance counter stats for './coremark.linux.riscv':

       2370362136      cpu-cycles
       1467434059      instructions              #    0.62  insn per cycle
        236016830      ariane_branch_jump
          5313061      ariane_call
         44041314      ariane_mis_predict
          1406938      ariane_ret
             1103      ariane_dtlb_miss
          6870139      ariane_itlb_miss
          2792754      ariane_l1_dcache_miss
          8444217      ariane_l1_icache_miss
        229115526      ariane_load
         64636669      ariane_store
            22573      ariane_exception
            22573      ariane_exception_ret
        239940589      ariane_if_empty
          9386316      ariane_sb_full

      23.804994040 seconds time elapsed

      23.622965000 seconds user
       0.109596000 seconds sys
```

CoreMark Result: 1.74 points/MHz @ 100MHz

Perf stat outputs event counts

There is support for event multiplexing,
but CVA6 does not have configurable events

CVA6 (Ariane) Events

Event	Counter
Cycles	mcycle
Instructions Retired	minstret
ICache Misses	mhpmcounter3
DCache Misses	mhpmcounter4
ITLB Misses	mhpmcounter5
DTLB Misses	mhpmcounter6
Loads	mhpmcounter7
Stores	mhpmcounter8

Event	Counter
Taken Exceptions	mhpmcounter9
Exceptions Returned	mhpmcounter10
Branches and Jumps	mhpmcounter11
Calls	mhpmcounter12
Returns	mhpmcounter13
Mispredicted Branches	mhpmcounter14
Scoreboard Full	mhpmcounter15
Instruction Fetch Empty	mhpmcounter16

Working PAPI (available components and preset events)

```
Available components and hardware information.
--------------------------------------------------------------------------------
PAPI version             : 7.0.1.0
Operating system         : Linux 5.13.19
Vendor string and code   : RISCV_SIFIVE (9, 0x9)
Model string and code    :  (0, 0x0)
CPU revision             : 0.000000
CPU Max MHz              : -1
CPU Min MHz              : -1
Total cores              : 4
SMT threads per core     : 4
Cores per socket         : 1
Sockets                  : 1
Cores per NUMA region    : 4
NUMA regions             : 0
Running in a VM          : no
Number Hardware Counters : 4
Max Multiplex Counters   : 384
Fast counter read (rdpmc): no
--------------------------------------------------------------------------------

Compiled-in components:
Name:   perf_event              Linux perf_event CPU counters
Name:   perf_event_uncore       Linux perf_event CPU uncore and northbridge
   \-> Disabled: No uncore PMUs or events found
Name:   sysdetect               System info detection component

Active components:
Name:   perf_event              Linux perf_event CPU counters
                                Native: 204, Preset: 22, Counters: 4
                                PMUs supported: perf, perf_raw, riscv_sifive_u74

Name:   sysdetect               System info detection component
                                Native: 0, Preset: 0, Counters: 0
```

```
================================================================================
  PAPI Preset Events
================================================================================
    Name        Code    Avail Deriv Description (Note)
PAPI_L1_DCM  0x80000000  Yes   No   Level 1 data cache misses
PAPI_L1_ICM  0x80000001  Yes   No   Level 1 instruction cache misses
PAPI_L2_DCM  0x80000002  No    No   Level 2 data cache misses
PAPI_L2_ICM  0x80000003  No    No   Level 2 instruction cache misses
PAPI_L3_DCM  0x80000004  No    No   Level 3 data cache misses
PAPI_L3_ICM  0x80000005  No    No   Level 3 instruction cache misses
PAPI_L1_TCM  0x80000006  Yes   Yes  Level 1 cache misses
PAPI_L2_TCM  0x80000007  No    No   Level 2 cache misses
PAPI_L3_TCM  0x80000008  No    No   Level 3 cache misses
PAPI_CA_SNP  0x80000009  No    No   Requests for a snoop
PAPI_CA_SHR  0x8000000a  No    No   Requests for exclusive access to shared cache line
PAPI_CA_CLN  0x8000000b  No    No   Requests for exclusive access to clean cache line
PAPI_CA_INV  0x8000000c  No    No   Requests for cache line invalidation
PAPI_CA_ITV  0x8000000d  No    No   Requests for cache line intervention
PAPI_L3_LDM  0x8000000e  No    No   Level 3 load misses
PAPI_L3_STM  0x8000000f  No    No   Level 3 store misses
PAPI_BRU_IDL 0x80000010  No    No   Cycles branch units are idle
PAPI_FXU_IDL 0x80000011  No    No   Cycles integer units are idle
PAPI_FPU_IDL 0x80000012  No    No   Cycles floating point units are idle
PAPI_LSU_IDL 0x80000013  No    No   Cycles load/store units are idle
PAPI_TLB_DM  0x80000014  Yes   No   Data translation lookaside buffer misses
PAPI_TLB_IM  0x80000015  Yes   No   Instruction translation lookaside buffer misses
PAPI_TLB_TL  0x80000016  Yes   Yes  Total translation lookaside buffer misses
```

Working PAPI (FP Stores test)

```
…
float a, b, c;
…
for (int i = 0; i < 1000; i++) {
    c = a*b;
}
```

```
root@unmatched:~/tiagorocha/RISC-V-PAPI/sifive_u74_tests# ./papi_fp_store
FP Stores: 1000
```

Next Steps...

- Ongoing: submitting perf_events kernel driver patch to the Linux Kernel repository
- Development of more tools that mimic the functionality of proprietary analyzes tools

This project was partially funded by

- FCT under projects UIDB/50021/2020
- European High Performance Computing Joint Undertaking under Framework Partnership Agreement No 800928 and Specific Grant Agreement No 101036168 (EPI SGA2)