HiSilicon SoC uncore Performance Monitoring Unit (PMU)

The HiSilicon SoC chip includes various independent system device PMUs such as L3 cache (L3C), Hydra Home Agent (HHA) and DDRC. These PMUs are independent and have hardware logic to gather statistics and performance information.

The HiSilicon SoC encapsulates multiple CPU and IO dies. Each CPU cluster (CCL) is made up of 4 cpu cores sharing one L3 cache; each CPU die is called Super CPU cluster (SCCL) and is made up of 6 CCLs. Each SCCL has two HHAs (0 - 1) and four DDRCs (0 - 3), respectively.

HiSilicon SoC uncore PMU driver

Each device PMU has separate registers for event counting, control and interrupt, and the PMU driver shall register perf PMU drivers like L3C, HHA and DDRC etc. The available events and configuration options shall be described in the sysfs, see:

/sys/devices/hisi_sccl{X}_<l3c{Y}/hha{Y}/ddrc{Y}>/, or /sys/bus/event_source/devices/hisi_sccl{X}_<l3c{Y}/hha{Y}/ddrc{Y}>. The "perf list" command shall list the available events from sysfs.

Each L3C, HHA and DDRC is registered as a separate PMU with perf. The PMU name will appear in event listing as hisi_sccl<sccl-id>_module<index-id>. where "sccl-id" is the identifier of the SCCL and "index-id" is the index of module.

e.g. hisi_sccl3_l3c0/rd_hit_cpipe is READ_HIT_CPIPE event of L3C index #0 in SCCL ID #3.

e.g. hisi_sccl1_hha0/rx_operations is RX_OPERATIONS event of HHA index #0 in SCCL ID #1.

The driver also provides a "cpumask" sysfs attribute, which shows the CPU core ID used to count the uncore PMU event.

Example usage of perf:

```
$# perf list
hisi_sccl3_l3c0/rd_hit_cpipe/ [kernel PMU event]
------------------------------------------
hisi_sccl3_l3c0/wr_hit_cpipe/ [kernel PMU event]
------------------------------------------
hisi_sccl1_l3c0/rd_hit_cpipe/ [kernel PMU event]
------------------------------------------
hisi_sccl1_l3c0/wr_hit_cpipe/ [kernel PMU event]
------------------------------------------

$# perf stat -a -e hisi_sccl3_l3c0/rd_hit_cpipe/ sleep 5
$# perf stat -a -e hisi_sccl3_l3c0/config=0x02/ sleep 5
```

For HiSilicon uncore PMU v2 whose identifier is 0x30, the topology is the same as PMU v1, but some new functions are added to the hardware.

(a) L3C PMU supports filtering by core/thread within the cluster which can be specified as a bitmap:

```
$# perf stat -a -e hisi_sccl3_l3c0/config=0x02,tt_core=0x3/ sleep 5
```

This will only count the operations from core/thread 0 and 1 in this cluster.

(b) Tracetag allow the user to chose to count only read, write or atomic operations via the tt_req parameeter in perf. The default value counts all operations. tt_req is 3bits, 3'b100 represents read operations, 3'b101 represents write operations, 3'b110 represents atomic store operations and 3'b111 represents atomic non-store operations, other values are reserved:

```
$# perf stat -a -e hisi_sccl3_l3c0/config=0x02,tt_req=0x4/ sleep 5
```

This will only count the read operations in this cluster.

(c) Datasrc allows the user to check where the data comes from. It is 5 bits. Some important codes are as follows: 5'b00001: comes from L3C in this die; 5'b01000: comes from L3C in the cross-die; 5'b01001: comes from L3C which is in another socket; 5'b01110: comes from the local DDR; 5'b01111: comes from the cross-die DDR; 5'b10000: comes from cross-socket DDR; etc, it is mainly helpful to find that the data source is nearest from the CPU cores. If datasrc_cfg is used in the multi-chips, the datasrc_skt shall be configured in perf command:

```
$# perf stat -a -e hisi_sccl3_l3c0/config=0xb9,datasrc_cfg=0xE/,
hisi_sccl3_l3c0/config=0xb9,datasrc_cfg=0xF/ sleep 5
```

(d)Some HiSilicon SoCs encapsulate multiple CPU and IO dies. Each CPU die contains several Compute Clusters (CCLs). The I/O dies are called Super I/O clusters (SICL) containing multiple I/O clusters (ICLs). Each CCL/ICL in the SoC has a unique ID. Each ID is 11bits, include a 6-bit SCCL-ID and 5-bit CCL/ICL-ID. For I/O die, the ICL-ID is followed by: 5'b00000: I/O_MGMT_ICL; 5'b00001: Network_ICL; 5'b00011: HAC_ICL; 5'b10000: PCIe_ICL;

Users could configure IDs to count data come from specific CCL/ICL, by setting srcid_cmd & srcid_msk, and data desitined for specific CCL/ICL by setting tgtid_cmd & tgtid_msk. A set bit in srcid_msk/tgtid_msk means the PMU will not check the bit when matching against the srcid_cmd/tgtid_cmd.

If all of these options are disabled, it can works by the default value that doesn't distinguish the filter condition and ID information and will return the total counter values in the PMU counters.

The current driver does not support sampling. So "perf record" is unsupported. Also attach to a task is unsupported as the events are all uncore.

Note: Please contact the maintainer for a complete list of events supported for the PMU devices in the SoC and its information if needed.