RAPL

Libmsr Version 0.2.1

Before you use RAPL, be sure to call the rapl_init function. See 'general libmsr use' for more details.

Be sure to check out the changelog.txt to see an overview of the latest changes. Significant changes happened to RAPL in release 0.1.15.

Setting a Power Bound

1. Create a rapl_limit struct
2. Set the limits in that struct
3. Call the function to set the limit on the socket and domain you desire, pass in your limit struct
4. Be sure to set limit.bits = 0
5. We recommend that you use RAPL limit 1 for processor power management

Setting RAPL Limits

```
struct rapl_limit limit1, limit2, dramlimit, pp0limit, pp1limit;
unsigned socket = 0;
limit1.watts = 95;
limit1.seconds = 1;

// Leave this as zero, its explanation is beyond the scope of this article

limit1.bits = 0;
limit2.watts = 120;
limit2.seconds = 3;

// Leave this as zero, its explanation is beyond the scope of this article

limit2.bits = 0;

// Set only the lower PKG limit on socket 0

set_pkg_rapl_limit(socket, &limit1, NULL);

// Set only the upper PKG limit on socket 0

set_pkg_rapl_limit(socket, NULL, &limit1);

// Set both PKG limits on socket 0

set_pkg_rapl_limit(socket, &limit1, &limit2);

dramlimit.watts = 50;
dramlimit.seconds = 1;

// Leave this as zero, its explanation is beyond the scope of this article

dramlimit.bits = 0;

// Set the DRAM limit for socket 0

set_dram_rapl_limit(socket, &dramlimit);

pp0limit.watts = 100;
pp0limit.seconds = 5;

// Leave this as zero, its explanation is beyond the scope of this article

pp0limit.bits = 0;
pp1limit.watts = 80;
pp1limit.seconds = 10;

// Leave this as zero, its explanation is beyond the scope of this article
```

```
pp1limit.bits = 0;

// Set the power planes limits for socket 0

set_pp_rapl_limit(socket, &pp0limit, &pp1limit);
```

Reading a Power Bound

This works the same as setting the power bound, but you call the respective 'get' function.

Getting RAPL Limits

```
struct rapl_limit limit1, limit2, dramlimit, pp0limit, pp1limit;
unsigned socket = 1;

// Get both power limits for socket 1


get_pkg_rapl_limit(socket, &limit1, &limit2);

// Get DRAM limit for socket 1

get_dram_rapl_limit(socket, &dramlimit);

// Get the power plane limits for socket 1

get_pp_rapl_limit(socket, &pp0limit, &pp1limit);
```

Reading Used Power

Note: The read_rapl_data function is no longer used for this. Now we use poll_rapl_data, which must be called twice on a socket to calculate watts/deltas.

Reading Used Power

```
// Update the rapl data. Watts/deltas are relative to the last time this function was called

poll_rapl_data();

// Display everything in datai1.
// Since poll_rapl_data has only been called once, these should all be 0


dump_rapl_data(stdout);

// This will calculate Watts/deltas relative to the last poll_rapl_data call

poll_rapl_data();

// Display everything in data1. This time, there should be values for watts

dump_rapl_data(stdout);
```

The rapl_data Struct

This struct contains tons of data.

struct rapl_data

```
// See the msr_rapl.h file for more details. This struct is currently undergoing revisions.
```

There is a centralized rapl_data struct used by RAPL. You can access it by using the rapl_storage function.

rapl_storage

```
struct rapl_data * rapl = NULL;
rapl_storage(&rapl, NULL);
```

Related articles

- PCI Configuration Registers (CSRs)
- The Batch Interface
- RAPL
- Performance Counters
- General LIBMSR Use