Architectural RAPL TPMI Interface

Introduction

TPMI (Topology Aware Register and PM Capsule Interface), planned for future generations, is an architectural, PCIe-standards based model, where feature support is provided cleanly as a driver and not as part of the base OS.

Today's (pre-TPMI) Intel® Xeon® processor platforms feature a RAPL interface that is defined through Model Specific Registers (MSR) and requires SW updates for new CPU generations. Linux kernel updates involving MSRs cannot be made until pertinent information is available publicly which can cause delays and increase resources required for customer programs.

With TPMI, Intel® Xeon® processors are moving to an architectural, driver-based model for RAPL, where feature support is provided cleanly as a driver and not as part of the base OS. The table below shows the benefit of shifting to an architectural implementation.

Today vs. Future (green is preferred vs. red)

Comparison of solutions		
OS Kernel vs. Driver		
	OS Kernel	Driver
Deploy update w/o entire OS upgrade	No	Yes
Deploy update w/o system reboot	No	Yes
MSR vs. MMIO Interface		
	MSR	MMIO
Driver compatible enumeration	Partial	Yes

Driver compatible IRQ	No	Yes
Mappable access from ring 3	No	Yes

The goal of an architectural RAPL interface is to meet the following requirements:

- Enumerability ○ Standard device driver mechanism ○ Ability to enumerate RAPL multiple-domain topology ○ Ability to enumerate RAPL multiple-feature per domain ○ Driver can be reused gen to gen, update is optional ○ New feature can be added w/o confusing legacy driver
- Performance ○ Memory-mapping access at run-time
  - MSR is problematic; often locked by BIOS; ring 3 management daemon needs (slow) system call for MSR
  - ○ Global memory-map includes all devices/domains in system
    - Every address is accessible from every CPU
- Security ○ OEM can configure what part of interface is exposed, R/W or R/O ○ Mapped pages include only RAPL related registers
  - Non-RAPL functions should be mapped & owned by a non-RAPL driver

To deliver the desired properties listed above, a design that uses a standard PCIe device interface with Vendor-Specific Extended Capabilities (VSEC) provides an excellent solution. VSEC is a PCI spec defined optional functionality where vendor specific capabilities can be exposed from a PCI express function. Using VSEC allows this feature to expose multiple capabilities within one PCI function. Consuming a PCI function for each feature/capability desired to expose to the OS would be too expensive for the CPU uncore and mesh resources.

The remainder of this document describes the use of TPMI, which is based on a PCIe VSEC design, to provide a flexible, extendable and software-PCIe-driver-enumerable MMIO interface for RAPL.

Address Mapping

The diagrams below showing example of several TPMI features. PCIe config space hosts the VSEC structure that contains TPMI VSEC_ID, base address pointer and other info. PFS structure describes each PM feature.

The numbers in the tables below are for illustrative purposes only.

PCIe Device (GNR: OOBMSM)	
BAR0	0x100,000
BAR1	0x200,000
BAR2	0x300,000

TPMI Control Interface	
Offset	Register Name
0	TPMI_CONTROL_STATUS
8	TPMI_COMMAND_DATA

VSEC Capability Structure

VSEC_ID	Num Entries	Entry Size	tBIR	Address
PM_Features (0x42)	10	2	1	2000
Telemetry				

PM Feature Structure (PFS)

Num Entries	TPMI_ID	Entry Size	Cap Offset	Attribute
1	RAPL	96	4K	OS
5	PEM	6	8K	OS
1	PMAX	16	12K	BIOS
1	DRC	8	16K	OS
5	SST	182	20K	OS
1	UFS	8	24K	OS
5	CSR_ALL	160	28K	OS
3	CSR_COMPUTE	40	32K	OS
1	CSR_PKG_ROOT	20	36K	OS
3	MISC_REGS	16	40K	OS

SW View - Examples	
BAR[1]	0x200,000
BasePtr of PFS	0x200,000 + 2000
TPMI_ID = RAPL	
BasePtr[RAPL]	0x200,000 + 2000 + 4K
reg_addr[instance, offset]	0x200,000 + 2000 + 4K + instance*96*4+offset , Where instance = [0..NumEntries-1]
TPMI_ID = PMAX	
BasePtr[PMAX]	0x200,000 + 2000 + 12K
reg_addr[instance, offset]	0x200,000 + 2000 + 12K + instance*16*4 + offset
TPMI_ID = CSR_ALL	
BasePtr[CSR_ALL]	0x200,000 + 2000 + 28K
reg_addr[instance, offset]	0x200,000 + 2000 + 28K + instance*160*4 + Toffset
	where Toffset = offset - StartingOffset

	StartingOffset	# registers
CSR_ALL	0x110	160
CSR_COMPUTE	0x1B0	40
CSR_PKG_MASTER	0x1D8	20

TPMI Address mapping

Each TPMI feature may claim a 4KB aligned, could potentially spread into multiple 4KB blocks, region in MMIO address space.

RAPL Discovery and Control

SW and BIOS discover RAPL domains via the per domain HEADER register.

DOMAIN_HEADER.TYPE indicates the domain type, i.e. socket RAPL, DRAM RAPL, and platform RAPL, etc.

DOMAIN_HEADER.FLAGS indicates the feature/register supported within this domain.

SW and BIOS can set the LOCK bit in Power Limit registers to lock out PL and TW settings.

RAPL Domain Registers

This section describes the layout of RAPL TPMI registers. All registers of RAPL domains associated with one Punit are mapped to one MMIO space. The table below is a software view

of the RAPL domain register array. Each domain consists of 128 bytes of registers starting with a 64b Domain Header and followed by a few, up to 15, 64b domain specific registers. The Domain Header specifies the type of domain (e.g. package RAPL, DRAM, system etc.), the domain index of its parent, and whether the specific feature is enabled or disabled. The parent domain index is configured by SoC to support hierarchical power management.

TPMI RAPL Domain Register Map (source)

<table>
<tr><th colspan="4">Array of RAPL Domains</th></tr>
<tr><td>Index</td><td>32b | 32b</td><td>Comments / Usages</td><td>Byte Offset</td></tr>
<tr><td rowspan="5">0</td><td colspan="2">DOMAIN_HEADER 0</td><td>describe the domain type/size/flags/etc.</td><td rowspan="5">0</td></tr>
<tr><td colspan="2">RAPL Register 1</td><td rowspan="4">RAPL register 1 - 15 of this domain</td></tr>
<tr><td colspan="2">RAPL Register 2</td></tr>
<tr><td colspan="2">....</td></tr>
<tr><td colspan="2">RAPL Register 15</td></tr>
<tr><td rowspan="3">1</td><td colspan="2">DOMAIN_HEADER 1</td><td>describe the domain type/size/flags/etc.</td><td rowspan="3">128</td></tr>
<tr><td colspan="2">RAPL Register 1</td><td rowspan="2">RAPL register 1 - 15 of this domain</td></tr>
<tr><td colspan="2">RAPL Register 2</td></tr>
<tr><td rowspan="2"></td><td colspan="2">....</td><td rowspan="2"></td><td rowspan="2"></td></tr>
<tr><td colspan="2">RAPL Register 15</td></tr>
<tr><td></td><td colspan="2">...</td><td></td><td></td></tr>
</table>

	...		
N-1	DOMAIN_HEADER (N-1)	describe the domain type/size/flags/etc.	128*(N-1)
	RAPL Register 1	RAPL register 1 - 15 of this domain	
	RAPL Register 2		
	....		
	RAPL Register 15		

RAPL registers are architecturally-defined structure which are similar to the legacy MSR/CFG RAPL registers with some exceptions. Each domain register type is assigned an unique index as shown in the table below. Some of the domain registers (e.g. power unit, power limit info, etc.) are always present. And some may be ignored if its corresponding bitmask in the DOMAIN_HEADER.FLAGS is 0x0. RAPL domain register of different domain type can have different bitfield definition.

Domain Register Index

RAPL Domain Register Index	
Description	Index
Domain Header	0
Power Unit	1
Power Limit 1	2
Power Limit 2	3
Power Limit 3	4
Power Limit 4	5
PL Offsets	6
Energy Status	7
Perf Status	8
Power Info	9

Domain Info	10
Interrupt	11
Reserved	12
Reserved	13
Reserved	14
Reserved	15

The sections below define the bit fields of each of these RAPL domain registers.

Domain Header (idx 0)

Domain Header

DOMAIN HEADER					
Field Name	**Bits**	**Width**	**Access Type**	**Description**	**Default**
INTERFACE_VERSION	7:0	8	RO	Version number for this interface	1
TYPE	15:8	8	RO	package rapl, dram rapl, platform rapl, etc	0
SIZE	23:16	8	RO	units of 128 bytes	1
PARENT_DOMAIN_INDEX	31:24	8	RO	parent domain index number	0
FLAGS	47:32	16	RO	Bit mask of the supported domain register. Pcode populates default setting.	0

				SW use it to discover which register is valid in the RAPL register bank. 1: the corresponding register is valid. 0: register is invalid.	
RESERVED	63:48	16	RO	Reserved	0

RAPL TPMI supports multiple domain types as listed below.

Domain type is set to 0x0 to if the CPU doesn't support this RAPL domain.

RAPL Domain Type Encoding

domain number	domain name
0	Invalid
1	system
2	package
3	reserved
4	memory
5	reserved
6	reserved
7	reserved
8-255	reserved

Power Unit (idx 1)

Power Unit

OWER UNIT					
Field Name	**Bits**	**Width**	**Access Type**	**Description**	**Default**

PWR_UNIT	3:0	4	RO	Power Units used for power control registers. The actual unit value is calculated by 1 W / Power(2, PWR_UNIT). The default value of 0x3 corresponds to 1/8 W.	3
RSVD1	5:4	2	RO	reserved	0
ENERGY_UNIT	10:6	5	RO	Energy Units used for power control registers. The actual unit value is calculated by 1 J / Power(2,ENERGY_UNIT). The default value of 14 corresponds to Ux.14 number.	0xE
RSVD2	11:11	1	RO	reserved	0
TIME_UNIT	15:12	4	RO	Time Units used for power control registers. The actual unit value is calculated by 1 s / Power(2,TIME_UNIT). The default value of Ah corresponds to 976 usec.	0xA
RSVD3	63:16	48	RO	reserved	0

PL1, PL2 (idx 2,3)

Power Limit Control

POWER LIMIT CONTROL 1 , 2					
Field Name	**Bits**	**Width**	**Access Type**	**Description**	**Default**
PWR_LIM	17:0	18	RW_L	This field indicates the power limitation for the socket RAPL domain. The unit of measurement is defined in POWER_UNIT[PWR_UNIT].	0

TIME_WINDOW	24:18	7	RW_L	Indicates the length of time window over which the power limit will be used by the processor. The time window is floating point number given by 2^Y * (1.0 + X / 4.0) * Time_Unit X = TIME_WINDOW[6:5]; Y = TIME_WINDOW[4:0]; Time_Unit is defined in POWER_UNIT[TIME_UNIT]. The maximal time window is bounded by PL1_PL2_INFO[MAX_TW]. Socket RAPL PL2 TIME_WINDOW is not SW configurable.	0
RSVD	61:25	37	RW_L	Reserved	0
PWR_LIM_EN	62:62	1	RW_L	Enable(1) or Disable(0)	0
LOCK	63:63	1	RW_L	When set, all settings in this register are locked and are treated as Read Only until next reset.	0

PL4 (idx 5)

Power Limit Control

POWER LIMIT CONTROL 4					
Field Name	Bits	Width	Access Type	Description	Default
PWR_LIM	17:0	18	RW_L	This field indicates the power limitation. The default value is loaded from fuses. The unit of measurement is defined in POWER_UNIT[PWR_UNIT].	0
RSVD	61:18	44	RW_L	RESERVED	0

PWR_LIM_EN	62:62	1	RW_L	Enable(1) or Disable(0) power limit	0
LOCK	63:63	1	RW_L	Lock until next reset as needed	0

Energy Status (idx 7)

Energy Status

NERGY STATUS					
Field Name	**Bits**	**Width**	**Access Type**	**Description**	**Default**
ENERGY	31:0	32	RO_V	Total amount of energy consumed since last reset. This is a monotonic increment counter with auto wrap back to zero after overflow.	0
TIME	63:32	32	RO_V	Total time elapsed when the energy was last updated. This is a monotonic increment counter with auto wrap back to zero after overflow. Unit is 10ns.	0

Perf Status (idx 8)

Perf Status

PERF STATUS					
Field Name	**Bits**	**Width**	**Access Type**	**Description**	**Default**

PWR_LIMIT_THROTTLE_CTR	31:0	32	RO_V	Reports the number of times (accumulated throttled time) the Power limiting algorithm had to clip the power limit due to hitting the lowest power state available.	0
RSVD	63:32	32	RO_V	Reserved	0

PL Info (idx 9)

Power Limit Info

ower Limit Info					
Field Name	**Bits**	**Width**	**Access Type**	**Description**	**Default**
MAX_PL1	17:0	18	RWL	Socket RAPL: The TDP package power of the package domain. DRAM RAPL: The typical TDP Power of DRAM Domain. Platform RAPL: The maximum Platform PL1 setting allowed. The units for this value are defined in POWER_UNIT[PWR_UNIT].	0
MIN_PL	35:18	18	RWL	Socket RAPL: The minimal package power setting allowed. DRAM RAPL: The typical min power of DRAM Domain. Platform RAPL: The minimal Platform power setting allowed. The units for this value are defined in POWER_UNIT[PWR_UNIT].	0

MAX_PL2	53:36	18	RWL	Socket RAPL: The maximal package power setting allowed. DRAM RAPL: The typical max power for DRAM Domain. Platform RAPL: The maximal platform PL2 setting allowed. The units for this value are defined in POWER_UNIT[PWR_UNIT].	0
MAX_TW	60:54	7	RWL	The maximal time window allowed. Higher values will be clamped to this value. The timing interval window is floating point number given by 2^Y * (1.0 + X / 4.0) * Time_Unit X = MAX_TW[6:5], Y = MAX_TW[4:0] Time_Unit is defined in POWER_UNIT[TIME_UNIT].	0
RSVD	62:61	2	RWL	Reserved	0
LOCK	63:63	1	RWL	When set, all settings in this register are locked and are treated as Read Only until next reset.	0

Domain Info (idx 10)

Domain Info

DOMAIN INFO					
Field Name	**Bits**	**Width**	**Access Type**	**Description**	**Default**
ROOT	0:0	1	RWL	1=The package is a domain root. 0=not a domain root.	0
DOMAIN_ID	3:1	3	RWL	The domain ID to which this package belongs to.	0

RSVD	62:4	59	RWL	Reserved	0
LOCK	63:63	1	RWL	When set to 1 the register is locked and becomes read-only until next reset. BIOS writes 1 to this bit after initializing the register.	0

Note:

- Pcode for GNR family should only enable DOMAIN_INFO in the RAPL_DOMAIN_HEADER.FLAGS for Psys. i.e. Pcode should set(1) bit10 of RAPL_DOMAIN_HEADER.FLAGS for Psys, and clear(0) it for DRAM and Socket RAPLs.

Register Layout Example

The table below shows the name of RAPL registers and their respective TPMI Offsets of a SOC that supports Socket RAPL, DRAM RAPL and Platform RAPL. Note that the content here is for illustration purpose only, audience should refer to EDS or OSXML documentation for the specific SOC RAPL register layout.

TPMI Register Name	TPMI ID (hex)	TPMI Offset (hex)
SOCKET_RAPL_DOMAIN_HEADER	0	0
SOCKET_RAPL_POWER_UNIT	0	8
SOCKET_RAPL_PL1_CONTROL	0	10
SOCKET_RAPL_PL2_CONTROL	0	18
SOCKET_RAPL_RSVD1	0	20
SOCKET_RAPL_PL4_CONTROL	0	28
SOCKET_RAPL_RSVD2	0	30
SOCKET_RAPL_ENERGY_STATUS	0	38
SOCKET_RAPL_PERF_STATUS	0	40
SOCKET_RAPL_PL_INFO	0	48
SOCKET_RAPL_RSVD3	0	50
SOCKET_RAPL_RSVD4	0	58
SOCKET_RAPL_RSVD5	0	60
SOCKET_RAPL_RSVD6	0	68
SOCKET_RAPL_RSVD7	0	70
SOCKET_RAPL_RSVD8	0	78
DRAM_RAPL_DOMAIN_HEADER	0	80
DRAM_RAPL_POWER_UNIT	0	88
DRAM_RAPL_PL1_CONTROL	0	90
DRAM_RAPL_RSVD1	0	98
DRAM_RAPL_RSVD2	0	A0
DRAM_RAPL_RSVD3	0	A8
DRAM_RAPL_RSVD4	0	B0
DRAM_RAPL_ENERGY_STATUS	0	B8
DRAM_RAPL_PERF_STATUS	0	C0
DRAM_RAPL_PL_INFO	0	C8
DRAM_RAPL_RSVD5	0	D0
DRAM_RAPL_RSVD6	0	D8
DRAM_RAPL_RSVD7	0	E0
DRAM_RAPL_RSVD8	0	E8
DRAM_RAPL_RSVD9	0	F0
DRAM_RAPL_RSVD10	0	F8
PLATFORM_RAPL_DOMAIN_HEADER	0	100
PLATFORM_RAPL_POWER_UNIT	0	108
PLATFORM_RAPL_PL1_CONTROL	0	110
PLATFORM_RAPL_PL2_CONTROL	0	118
PLATFORM_RAPL_RSVD1	0	120
PLATFORM_RAPL_RSVD2	0	128
PLATFORM_RAPL_RSVD3	0	130
PLATFORM_RAPL_ENERGY_STATUS	0	138
PLATFORM_RAPL_PERF_STATUS	0	140
PLATFORM_RAPL_PL_INFO	0	148
PLATFORM_RAPL_DOMAIN_INFO	0	150
PLATFORM_RAPL_RSVD4	0	158
PLATFORM_RAPL_RSVD5	0	160
PLATFORM_RAPL_RSVD6	0	168
PLATFORM_RAPL_RSVD7	0	170
PLATFORM_RAPL_RSVD8	0	178

Reference

• The table below provides a register mapping of legacy RAPL MSR vs. TPMI.

MSR						TPMI	
Socket RAPL		DRAM RAPL		Platform RAPL		RAPL	
Name	Address	Name	Address	Name	Address	Name	Index
						Domain Header	0
PACKAGE_POWER_SKU_UNIT	606h					Power Unit	1
PACKAGE_RAPL_LIMIT	610h	DRAM_PLANE_POWER_LIMIT	618h	PLATFROM_RAPL_LIMIT	65C h	Power Limit	2,3,4,5
PACKAGE_ENERGY_STATUS	611h	DRAM_ENERGY_STATUS	619h	PLATFORM_ENERGY_STATUS	64Dh	Energy Status	7
PACKAGE_RAPL_PERF_STATUS (PACKAGE_OVERFLOW_STATUS)	613h	DRAM_PLANE_OVERFLOW_STATUS	61Bh	PLATFORM_RAPL_PERF_STATUS	666h	Perf Status	8
PACKAGE_POWER_SKU	614h	DRAM_POWER_INFO	61C h	PLATFORM_POWER_INFO	665h	Power Limit Info	9