skiboot Documentation

Release v5.9-167-g7a2610a145a8

IBM, others

Dec 15, 2017

CONTENTS

CHAPTER

ONE

OVERVIEW

1.1 Skiboot overview

Skiboot is firmware, loaded by the FSP. Along with loading the bootloader, it provides some runtime services to the OS (typically Linux).

1.1.1 Source layout

Directory	Content
asm/	small amount, mainly entry points
ccan/	bits from CCAN
core/	common code among machines.
doc/	not enough here
external/	tools and userspace components
hdata/	Parses HDAT from Hostboot/FSP into Device Tree
hw/	drivers for things & fsp things.
include/	headers!
libc/	tiny libc, originally from SLOF
libfdt/	Manipulate flattened device trees
libflash/	Lib for talking to flash and parsing FFS structs
libpore/	to manipulate PORE[1] engine.
libstb/	See *Secure and Trusted Boot Overview*
libxz/	The xz_embedded library
opal-ci/	Some scripts to help Continuous Integration testing
platforms/	Platform (machine/BMC) specific code
test/	Test scripts and binaries

We have a spinlock implementation in asm/lock.S Entry points are detailed in asm/head.S The main C entry point is in core/init.c: main_cpu_entry()

1.1.2 Binaries

The following binaries are built:

[1] Power On Reset Engine. Used to bring cores out of deep sleep states. For POWER9, this also includes the *p9_stop_api* which manipulates the low level microcode to-reinit certain SPRs on transition out of a state losing STOP state.

File	Purpose
skiboot.lid	Binary for flashing onto systems[2]
skiboot.lid.stb	Secure and Trusted Boot container wrapped skiboot
skiboot.lid.xz	XZ compressed binary[3]
skiboot.lid.xz.stb	STB container wrapped XZ compressed skiboot[4]
skiboot.elf	is the elf binary of it, lid comes from this
skiboot.map	plain map of symbols

1.1.3 Booting

On boot, every thread of execution jumps to a single entry point in skiboot so we need to do some magic to ensure we init things properly and don't stomp on each other. We choose a master thread, putting everybody else into a spinloop.

Essentially, we do this by doing an atomic fetch and inc and whoever gets 0 gets to be the main thread. The main thread will then distribute tasks to secondary threads as needed. We do not (currently) do anything fancy like context switching or scheduling.

When entering skiboot, we enter with one of two data structures describing the system as initialized by Hostboot. There may be a flattened device tree (see https://devicetree.org/), or a HDAT structure. While Device Tree is an industry standard, HDAT comes from IBM POWER. On POWER8, skiboot would get HDAT and a mini-devicetree from an FSP or purely a Device Tree on OpenPOWER systems. On POWER9, it's just HDAT everywhere (that isn't a simulator). The HDAT specification is currently not public. It is purely an interface between Hostboot and skiboot, and is only exposed anywhere else for debugging purposes.

During boot, skiboot will add a lot to the device tree, manipulating what may already be there before exporting this new device tree out to the OS.

The main entry point is main_cpu_entry() in core/init.c, this is a carefully ordered init of things. The sequence is relatively well documented there.

1.1.4 OS interface

OPAL (skiboot) is exclusively called through OPAL calls. The OS has complete controll of *when* OPAL code is executed. The design of all OPAL APIs is that we do not block in OPAL, so as not to introduce jitter.

Skiboot maintains its own stack for each CPU, the running OS does not need to donate or reserve any of its stack space.

With the OPAL API calls and device tree bindings we have the OPAL ABI.

1.1.5 Interrupts

We don't directly handle interrupts in skiboot. The OS is in complete control, and any interrupts we need to process are first received by the OS. The *OPAL_HANDLE_INTERRUPT* call is made by the OS for OPAL to do what's needed.

[2] Practically speaking, this is just IBM FSP based systems now. Since the *skiboot.lid* size is now greater than 1MB, which is the size of the default *PAYLOAD* PNOR partition size on OpenPOWER systems, you will want the *skiboot.lid.xz* or *skiboot.lid.xz.stb* instead.

[3] On OpenPOWER systems, hostboot will read and decompress XZ compressed payloads. This shortens boot time (less data to read), adds a checksum over the *PAYLOAD* and saves valuable PNOR space. If in doubt, use this payload.

[4] If a secure boot system, use this payload.

1.1.6 Memory

We initially occupy a chunk of memory, "heap". We pass to the OS (Linux) a reservation of what we occupy (including stacks).

In the source file include/mem-map.h we include a memory map. This is manually generated, not automatically generated.

We use CCAN for a bunch of helper code, turning on things like DEBUG_LOCKS and DEBUG_MALLOC as these are not a performance issue for us, and we like to be careful.

In include/config.h there are defines for turning on extra tracing. OPAL is what we name the interface from skiboot to OS (Linux).

Each CPU gets a 16k stack, which is probably more than enough. Stack should be used sparingly though.

Important memory locations:

Location	What's there
SKIBOOT_BASE	where skiboot lives, of SKIBOOT_SIZE
HEAP_BASE	Where skiboot heap starts, of HEAP_SIZE

There is also SKIBOOT_SIZE (manually calculated) and DEVICE_TREE_MAX_SIZE, which is largely historical.

1.1.7 Skiboot log

There is a circular log buffer that skiboot maintains. This can be accessed either from the FSP or through /dev/mem or through the sysfs file /sys/firmware/opal/msglog.

1.2 OPAL Specification

DRAFT - VERSION 0.0.1 AT BEST.

COMMENTS ARE WELCOME - and indeed, needed.

If you are reading this, congratulations: you're now reviewing it!

This document aims to define what it means to be OPAL compliant.

While skiboot is the reference implementation, this documentation should be complete enough that (given hardware documentation) create another implementation. It is not recommended that you do this though.

1.2.1 Authors

Stewart Smith <stewart@linux.vnet.ibm.com> : OPAL Architect, IBM

1.2.2 Definitions

Host processor the main POWER CPU (e.g. the POWER8 CPU)

Host OS the operating system running on the host processor.

OPAL OpenPOWER Abstraction Layer.

1.2.3 What is OPAL?

The OpenPower Abstraction Layer (OPAL) is boot and runtime firmware for POWER systems. There are several components to what makes up a firmware image for OpenPower machines.

For example, there may be:

- BMC firmware
  - Firmware that runs purely on the BMC.
  - On IBM systems that have an FSP rather than a BMC, there is FSP firmware
  - While essential to having the machine function, this firmware is not part of the OPAL Specification.
- HostBoot
  - HostBoot (https://github.com/open-power/hostboot) performs all processor, bus and memory initialization within IBM POWER based systems.
- OCC Firmware
  - On Chip Controller (Firmware for OCC - a PPC405 core inside the IBM POWER8 in charge of keeping the system thermally and power safe).
- SkiBoot
  - Boot and runtime services.
- A linux kernel and initramfs incorporating petitboot
  - The bootloader. This is where a user chooses what OS to boot, and petitboot will use kexec to switch to the host Operating System (for example, PowerKVM).

While all of these components may be absolutely essential to power on, boot and operate a specific OpenPower POWER8 system, the majority of the code mentioned above can be thought of as implementation details and not something that should form part of an OPAL Specification.

For an OPAL system, we assume that the hardware is functioning and any hardware management that is specific to a platform is performed by OPAL firmware transparently to the host OS.

The OPAL Specification focus on the interface between firmware and the Operating System. It does not dictate that any specific pieces of firmware code be used, although re-inventing the wheel is strongly discouraged.

The OPAL Specification explicitly allows for:

- A conforming implementation to not use any of the reference implementation code.
- A conforming implementation to use any 64bit POWER ISA conforming processor, and not be limited to the IBM POWER8.
- A conforming implementation to be a simulator, emulator or virtual environment
- A host OS other than Linux

Explicitly not covered in this specification:

- A 32bit OPAL Specification There is no reason this couldn't exist but the current specification is for 64bit POWER systems only.

1.2.4 Boot Services

An OPAL compliant firmware implementation will load and execute a payload capable of booting a Host Operating System.

The reference implementation loads a Linux kernel with an initramfs with a minimal userspace and the petitboot boot loader - collectively referred to as skiroot.

The OPAL Specification explicitly allows variation in this payload.

A requirement of the payload is that it MUST support loading and booting an uncompressed vmlinux Linux kernel. [**TODO**: expand on what this actually means]

An OPAL system MUST pass a device tree to the host kernel. [**TODO**: expand the details, add device-tree section and spec]

An OPAL system MUST provide the host kernel with enough information to know how to call OPAL runtime services. [**TODO**: expand on this.]

Explicitly not covered by the OPAL Specification:

- Kernel module ABI for skiroot kernel
- Userspace environment of skiroot
- That skiroot is Linux.

Explicitly allowed:

- Replacing the payload with something of equal/similar functionality (whether replacing skiroot with an implementation of Zork would be compliant is left as an exercise for the reader)

1.2.5 Payload Environment

The payload is started with:

Register	Value
r3	address of flattened device-tree (fdt)
r8	OPAL base
r9	OPAL entry

1.2.6 Runtime Services

An OPAL Specification compliant system provides runtime services to the host Operating System via a standard interface.

An OPAL call is made by calling opal_entry with:

- r0: OPAL Token
- r2: OPAL Base
- r3..r10: Args (up to 8)

The OPAL API is defined in skiboot/doc/opal-api/

Not all OPAL APIs must be supported for a system to be compliant. When called with an unsupported token, a compliant firmware implementation MUST fail gracefully and not crash. Reporting a warning that an unsupported token was called is okay, as compliant host Operating Systems should use OPAL_CHECK_TOKEN to test for optional functionality.

All parameters to OPAL calls are big endian. Little endian hosts MUST appropriately convert parameters before passing them to OPAL.

Machine state across OPAL calls:

- r1 is preserved
- r12 is scratch
- r13 - 31 preserved
- 64bit HV real mode
- big endian
- external interrupts disabled

1.2.7 Detecting OPAL Support

A Host OS may need to detect the presence of OPAL as it may support booting under other platforms. For example, a single Linux kernel can be built to boot under OPAL and under PowerVM or qemu pseries machine type.

The root node of the device tree MUST have compatible = "ibm,powernv". See *Device Tree* for more details.

The presence of the "/ibm,opal" entry in the device tree signifies running under OPAL. Additionally, the "/ibm,opal" node MUST have a compatibile property listing "ibm,opal-v3".

The "/ibm,opal" node MUST have the following properties:

```
ibm,opal {
          compatible = "ibm,opal-v3";
          opal-base-address = <>;
          opal-entry-address = <>;
          opal-runtime-size = <>;
};
```

The compatible property MAY have other strings, such as a future "ibm,opal-v4". These are reserved for future use.

Some releases of the reference implementation (skiboot) have had compatible contain "ibm,opal-v2" as well as "ibm,opal-v3". Host operating systems MUST NOT rely on "ibm,opal-v2", this is a relic from early OPAL history.

The "ibm,opal" node MUST have a child node named "firmware". It MUST contain the following:

```
firmware {
         compatible = "ibm,opal-firmware";
};
```

It MUST contain one of the following two properties: git-id, version. The git-id property is deprecated, and version SHOULD be used. These are informative and MUST NOT be used by the host OS to determine anything about the firmware environment.

The version property is a textual representation of the OPAL version. For example, it may be "skiboot-4.1" or other versioning described in more detail in *Versioning Scheme of skiboot*.

1.2.8 OPAL log

OPAL implementation SHOULD have an in memory log where informational and error messages are stored. If present it MUST be human readable and text based. There is a separate facility (Platform Error Logs) for machine readable errors.

A conforming implementation MAY also output the log to a serial port or similar. An implementation MAY choose to only output certain log messages to a serial port.

For example, the reference implementation (skiboot) by default filters log messages so that only higher priority log messages go over the serial port while more messages go to the in memory buffer.

[TODO: add device-tree bits here]

CHAPTER

TWO

DEVELOPER GUIDE AND INTERNALS

2.1 SkiBoot Console Log

Skiboot maintains a circular textual log buffer in memory.

It can be accessed using any debugging method that can peek at memory contents. While the debug_descriptor does hold the location of the memory console, we're pretty keen on keeping its location static.

Events are logged in the following format: `[S.T,L] message` where:

S Seconds, which is the timebase divided by 512,000,000. **NOTE**: The timebase is reset during boot, so zero is a few dozen messages into skiboot booting.

T Remaining Timebase. It is *NOT* a fraction of a second, but rather timebase%512000000

L Log level (see below)

Example:

```
[    2.223466021,5] FLASH: Found system flash: Macronix MXxxL51235F id:0
[    3.494892796,7] FLASH: flash subpartition eyecatcher CAPP
```

You should use the new prlog() call for any log message and set the log level/priority appropriately.

printf() is mapped to PR_PRINTF and should be phased out and replaced with prlog() calls.

See timebase.h for full timebase explanation.

2.1.1 Log levels

Define	Value
PR_EMERG	0
PR_ALERT	1
PR_CRIT	2
PR_ERR	3
PR_WARNING	4
PR_NOTICE	5
PR_PRINTF	PR_NOTICE
PR_INFO	6
PR_DEBUG	7
PR_TRACE	8
PR_INSANE	9

The console_log_levels byte in the debug_descriptor controls what messages are written to any console drivers (e.g. fsp, uart) and what level is just written to the in memory console (or not at all).

This enables (advanced) users to vary what level of output they want at runtime in the memory console and through console drivers (fsp/uart)

You can vary two things by poking in the debug descriptor:

1. what log level is printed at all e.g. only turn on PR_TRACE at specific points during runtime
2. what log level goes out the fsp/uart console, defaults to PR_PRINTF

We use two 4bit numbers (1 byte) for this in debug descriptor (saving some space, not needlessly wasting space that we may want in future).

The default is 0x75 (7=PR_DEBUG to in memory console, 5=PR_PRINTF to drivers

If you write 0x77 you will get debug info on uart/fsp console as well as in memory. If you write 0x95 you get PR_INSANE in memory but still only PR_NOTICE through drivers.

People who write something like 0x1f will get a very quiet boot indeed.

2.2 How to log errors on OPAL

Currently the errors reported by OPAL interfaces are in free form, where as errors reported by service processor is in standard Platform Error Log (PEL) format. For out-of band management via IPMI interfaces, it is necessary to push down the errors to service processor via mailbox (reported by OPAL) in PEL format.

PEL size can vary from 2K-16K bytes, fields of which needs to populated based on the kind of event and error that needs to be reported. All the information needed to be reported as part of the error, is passed by user using the error-logging interfaces outlined below. Following which, PEL structure is generated based on the input and then passed on to service processor.

We do create eSEL error log format for some service processors but it's just a wrapper around PEL format. Actual data still stays in PEL format.

2.2.1 Error logging interfaces in OPAL

Interfaces are provided for the user to log/report an error in OPAL. Using these interfaces relevant error information is collected and later converted to PEL format and then pushed to service processor.

Step 1: To report an error, invoke `opal_elog_create()` with required argument.

```
struct errorlog *opal_elog_create(struct opal_err_info *e_info,
uint32_t tag);
```

Parameters:

- `struct opal_err_info *e_info` Struct to hold information identifying error/event source.
- **`uint32_t tag`: Unique value to identify the data.** Ideal to have ASCII value for 4-byte string.

The opal_err_info struct holds several pieces of information to help identify the error/event. The struct can be obtained via the `DEFINE_LOG_ENTRY` macro as below - it only needs to be called once.

```
DEFINE_LOG_ENTRY(OPAL_RC_ATTN, OPAL_PLATFORM_ERR_EVT, OPAL_CHIP,
                 OPAL_PLATFORM_FIRMWARE, OPAL_PREDICTIVE_ERR_GENERAL,
                 OPAL_NA);
```

The various attributes set by this macro are described below.

`uint8_t opal_error_event_type`: Classification of error/events type reported on OPAL.

```
/* Platform Events/Errors: Report Machine Check Interrupt */
#define OPAL_PLATFORM_ERR_EVT           0x01
/* INPUT_OUTPUT: Report all I/O related events/errors */
#define OPAL_INPUT_OUTPUT_ERR_EVT       0x02
/* RESOURCE_DEALLOC: Hotplug events and errors */
#define OPAL_RESOURCE_DEALLOC_ERR_EVT   0x03
/* MISC: Miscellaneous error */
#define OPAL_MISC_ERR_EVT               0x04
```

`uint16_t component_id`: Component ID of OPAL component as listed in `include/errorlog.h`.

`uint8_t subsystem_id`: ID of the sub-system reporting error.

```
/* OPAL Subsystem IDs listed for reporting events/errors */
#define OPAL_PROCESSOR_SUBSYSTEM        0x10
#define OPAL_MEMORY_SUBSYSTEM           0x20
#define OPAL_IO_SUBSYSTEM               0x30
#define OPAL_IO_DEVICES                 0x40
#define OPAL_CEC_HARDWARE               0x50
#define OPAL_POWER_COOLING              0x60
#define OPAL_MISC                       0x70
#define OPAL_SURVEILLANCE_ERR           0x7A
#define OPAL_PLATFORM_FIRMWARE          0x80
#define OPAL_SOFTWARE                   0x90
#define OPAL_EXTERNAL_ENV               0xA0
```

`uint8_t event_severity`: Severity of the event/error to be reported.

```
#define OPAL_INFO                                   0x00
#define OPAL_RECOVERED_ERR_GENERAL                  0x10

/* 0x2X series is to denote set of Predictive Error */
/* 0x20 Generic predictive error */
#define OPAL_PREDICTIVE_ERR_GENERAL                         0x20
/* 0x21 Predictive error, degraded performance */
#define OPAL_PREDICTIVE_ERR_DEGRADED_PERF                   0x21
/* 0x22 Predictive error, fault may be corrected after reboot */
#define OPAL_PREDICTIVE_ERR_FAULT_RECTIFY_REBOOT            0x22
/*
 * 0x23 Predictive error, fault may be corrected after reboot,
 * degraded performance
 */
#define OPAL_PREDICTIVE_ERR_FAULT_RECTIFY_BOOT_DEGRADE_PERF 0x23
/* 0x24 Predictive error, loss of redundancy */
#define OPAL_PREDICTIVE_ERR_LOSS_OF_REDUNDANCY              0x24

/* 0x4X series for Unrecoverable Error */
/* 0x40 Generic Unrecoverable error */
#define OPAL_UNRECOVERABLE_ERR_GENERAL                      0x40
/* 0x41 Unrecoverable error bypassed with degraded performance */
#define OPAL_UNRECOVERABLE_ERR_DEGRADE_PERF                 0x41
/* 0x44 Unrecoverable error bypassed with loss of redundancy */
#define OPAL_UNRECOVERABLE_ERR_LOSS_REDUNDANCY              0x44
/* 0x45 Unrecoverable error bypassed with loss of redundancy
```

```
 * and performance
 */
#define OPAL_UNRECOVERABLE_ERR_LOSS_REDUNDANCY_PERF         0x45
/* 0x48 Unrecoverable error bypassed with loss of function */
#define OPAL_UNRECOVERABLE_ERR_LOSS_OF_FUNCTION             0x48

#define OPAL_ERROR_PANIC                                    0x50
```

`uint8_t event_subtype`: Event Sub-type

```
#define OPAL_NA                                         0x00
#define OPAL_MISCELLANEOUS_INFO_ONLY                    0x01
#define OPAL_PREV_REPORTED_ERR_RECTIFIED                0x10
#define OPAL_SYS_RESOURCES_DECONFIG_BY_USER             0x20
#define OPAL_SYS_RESOURCE_DECONFIG_PRIOR_ERR            0x21
#define OPAL_RESOURCE_DEALLOC_EVENT_NOTIFY              0x22
#define OPAL_CONCURRENT_MAINTENANCE_EVENT               0x40
#define OPAL_CAPACITY_UPGRADE_EVENT                     0x60
#define OPAL_RESOURCE_SPARING_EVENT                     0x70
#define OPAL_DYNAMIC_RECONFIG_EVENT                     0x80
#define OPAL_NORMAL_SYS_PLATFORM_SHUTDOWN               0xD0
#define OPAL_ABNORMAL_POWER_OFF                         0xE0
```

`uint8_t opal_srctype`: SRC type, value should be OPAL_SRC_TYPE_ERROR. SRC refers to System Reference Code. It is 4 byte hexa-decimal number that reflects the current system state. Eg: BB821010,

- 1st byte -> BB -> SRC Type
- 2nd byte -> 82 -> Subsystem
- 3rd, 4th byte -> Component ID and Reason Code

SRC needs to be generated on the fly depending on the state of the system. All the parameters needed to generate a SRC should be provided during reporting of an event/error.

`uint32_t reason_code`: Reason for failure as stated in `include/errorlog.h` for OPAL. Eg: Reason code for code-update failures can be

- `OPAL_RC_CU_INIT` -> Initialisation failure
- `OPAL_RC_CU_FLASH` -> Flash failure

Step 2: Data can be appended to the user data section using the either of the below two interfaces:

```
void log_append_data(struct errorlog *buf, unsigned char *data,
                     uint16_t size);
```

Parameters:

`struct opal_errorlog *buf`: `struct opal_errorlog` pointer returned by `opal_elog_create()` call.

`unsigned char *data`: Pointer to the dump data

`uint16_t size`: Size of the dump data.

`void log_append_msg(struct errorlog *buf, const char *fmt, ...);`

Parameters:

`struct opal_errorlog *buf`: pointer returned by `opal_elog_create()` call.

`const char *fmt`: Formatted error log string.

Additional user data sections can be added to the error log to separate data (eg. readable text vs binary data) by calling `log_add_section()`. The interfaces in Step 2 operate on the 'last' user data section of the error log.

```
void log_add_section(struct errorlog *buf, uint32_t tag);
```

Parameters:

`struct opal_errorlog *buf`: pointer returned by `opal_elog_create()` call.

`uint32_t tag`: Unique value to identify the data. Ideal to have ASCII value for 4-byte string.

Step 3: There is a platform hook for the OPAL error log to be committed on any service processor(Currently used for FSP and BMC based machines).

Below is snippet of the code of how this hook is called.

```
void log_commit(struct errorlog *elog)
{
        ....
        ....
        if (platform.elog_commit) {
                rc = platform.elog_commit(elog);
                if (rc)
                        prerror("ELOG: Platform commit error %d"
                                "\n", rc);
                return;
        }
        ....
        ....
}
```

Step 3.1 FSP:

```
.elog_commit            = elog_fsp_commit
```

Once all the data for an error is logged in, the error needs to be committed in FSP.

In the process of committing an error to FSP, log info is first internally converted to PEL format and then pushed to the FSP. FSP then take cares of sending all logs including its own and OPAL's one to the POWERNV.

OPAL maintains timeout field for all error logs it is sending to FSP. If it is not logged within allotted time period (e.g if FSP is down), in that case OPAL sends those logs to POWERNV.

Step 3.2 BMC:

```
.elog_commit            = ipmi_elog_commit
```

In case of BMC machines, error logs are first converted to eSEL format. i.e:

```
eSEL = SEL header + PEL data
```

SEL header contains below fields.

```
struct sel_header {
        uint16_t id;
        uint8_t record_type;
        uint32_t timestamp;
        uint16_t genid;
        uint8_t evmrev;
```

```
        uint8_t sensor_type;
        uint8_t sensor_num;
        uint8_t dir_type;
        uint8_t signature;
        uint8_t reserved[2];
}
```

After filling up the SEL header fields, OPAL copies the error log PEL data after the header section in the error log buffer. Then using IPMI interface, eSEL gets logged in BMC.

If the user does not intend to dump various user data sections, but just log the error with some amount of description around that error, they can do so using just the simple error logging interface.

```
log_simple_error(uint32_t reason_code, char *fmt, ...);
```

For example:

```
log_simple_error(OPAL_RC_SURVE_STATUS,
                 "SURV: Error retrieving surveillance status: %d\n",
                                                        err_len);
```

Using the reason code, an error log is generated with the information derived from the look-up table, populated and committed to service processor. All of it is done with just one call.

2.3 Error logging retrieval from FSP:

FSP sends error log notification to OPAL via mailbox protocol.

OPAL maintains below lists:

- Free list : List of free nodes.
- Pending list : List of nodes which is yet to be read by the POWERNV.
- Processed list : List of nodes which has been read but still waiting for acknowledgement.

Below is the structure of the node:

```
struct fsp_log_entry {
        uint32_t log_id;
        size_t log_size;
        struct list_node link;
};
```

OPAL maintains a state machine which has following states.

```
enum elog_head_state {
        ELOG_STATE_FETCHING,    /*In the process of reading log from FSP. */
        ELOG_STATE_FETCHED_INFO,/* Indicates reading log info is completed */
        ELOG_STATE_FETCHED_DATA,/* Indicates reading log is completed */
        ELOG_STATE_HOST_INFO,   /* Host read log info */
        ELOG_STATE_NONE,        /* Indicates to fetch next log */
        ELOG_STATE_REJECTED,    /* resend all pending logs to linux */
};
```

Initially, state of the state machine is `ELOG_STATE_NONE`. When OPAL gets the notification about the error log, it takes out the node from free list and put it into pending list and update the state machine to fetching state (`ELOG_STATE_FETCHING`). It also gives response back to FSP about the received error log notification.

It then queue mailbox message to get the error log data in OPAL error log buffer, once it is done state machine gets into fetched state (`ELOG_STATE_FETCHED_DATA`). After that, OPAL notifies POWERNV host to fetch new error log.

POWERNV uses the OPAL interface to get the error log info(elogid, elog_size, elog_type) first then it reads the error log data in its buffer that moves the pending error log to processed list. After reading, the state machine moves to `ELOG_STATE_NONE` state.

It acknowledges the error log id after reading error log data by sending the call to OPAL, which in turn sends the acknowledgement mbox message to FSP and moves error log id from processed list to again back to free node list and this process goes on every FSP error log.

2.3.1 Design constraints:

```
#define ELOG_READ_MAX_RECORD            128
```

Currently, the number of error logs from FSP, OPAL can hold is limited to 128. If OPAL run out of free node in the list for the new error log, it sends 'Discarded by OPAL' message to the FSP. At some point in the future, it is upto FSP when it notifies again to OPAL about the discarded error log.

```
#define ELOG_WRITE_MAX_RECORD           64
```

There is also limitation on the number of OPAL error logs OPAL can hold is 64. If it is run out of the buffers in the pool, it will log the message saying 'Failed to get the buffer'.

2.3.2 Note

- For more information regarding error logging and PEL format refer to PAPR doc and P7 PEL and SRC PLDD document.
- Refer to `include/errorlog.h` for all the error logging interface parameters and `include/pel.h` for PEL structures.

2.3.3 Sample error logging

```
DEFINE_LOG_ENTRY(OPAL_RC_ATTN, OPAL_PLATFORM_ERR_EVT, OPAL_ATTN,
                OPAL_PLATFORM_FIRMWARE, OPAL_PREDICTIVE_ERR_GENERAL,
                OPAL_NA);

void report_error(int index)
{
       struct errorlog *buf;
       char data1[] = "This is a sample user defined data section1";
       char data2[] = "Error logging sample. These are dummy errors. Section 2";
       char data3[] = "Sample error Sample error Sample error Sample error \
                        Sample error abcdefghijklmnopqrstuvwxyz";
       int tag;

       printf("ELOG: In machine check report error index: %d\n", index);

       /* To report an error, create an error log with relevant information
        * opal_elog_create(). Call returns a pre-allocated buffer of type
        * 'struct errorlog' buffer with relevant fields updated.
        */
```

```
        /* tag -> unique ascii tag to identify a particular data dump section */
        tag = 0x4b4b4b4b;
        buf = opal_elog_create(&e_info(OPAL_RC_ATTN), tag);
        if (buf == NULL) {
                printf("ELOG: Error getting buffer.\n");
                return;
        }

        /* Append data or text with log_append_data() or log_append_msg() */
        log_append_data(buf, data1, sizeof(data1));

        /* In case of user wanting to add multiple sections of various dump data
         * for better debug, data sections can be added using this interface
         * void log_add_section(struct errorlog *buf, uint32_t tag);
         */
        tag = 0x4c4c4c4c;
        log_add_section(buf, tag);
        log_append_data(buf, data2, sizeof(data2));
        log_append_data(buf, data3, sizeof(data3));

        /* Once all info is updated, ready to be sent to FSP */
        printf("ELOG:commit to FSP\n");
        log_commit(buf);
}
```

2.3.4 Sample output PEL dump got from FSP

```
$ errl -d -x 0x533C9B37
|   00000000     50480030  01004154  20150728  02000500     PH.0..AT ..(....   |
|   00000010     20150728  02000566  4B000107  00000000      ..(...fK.......   |
|   00000020     00000000  00000000  B0000002  533C9B37     ............S..7   |
|   00000030     55480018  01004154  80002000  00000000     UH....AT.. .....   |
|   00000040     00002000  01005300  50530050  01004154     .. ...S.PS.P..AT   |
|   00000050     02000008  00000048  00000080  00000000     .......H........   |
|   00000060     00000000  00000000  00000000  00000000     ................   |
|   00000070     00000000  00000000  42423832  31343130     ........BB821410   |
|   00000080     20202020  20202020  20202020  20202020                        |
|   00000090     20202020  20202020  4548004C  01004154             EH.L..AT   |
|   000000A0     38323836  2D343241  31303738  34415400     8286-42A10784AT.   |
|   000000B0     00000000  00000000  00000000  00000000     ................   |
|   000000C0     00000000  00000000  00000000  00000000     ................   |
|   000000D0     00000000  00000000  20150728  02000500     ........ ..(....   |
|   000000E0     00000000  4D54001C  01004154  38323836     ....MT....AT8286   |
|   000000F0     2D343241  31303738  34415400  00000000     -42A10784AT.....   |
|   00000100     5544003C  01004154  4B4B4B4B  00340000     UD....ATKKKK.4..   |
|   00000110     54686973  20697320  61207361  6D706C65     This is a sample   |
|   00000120     20757365  72206465  66696E65  64206461      user defined da   |
|   00000130     74612073  65637469  6F6E3100  554400A7     ta section1.UD..   |
|   00000140     01004154  4C4C4C4C  009F0000  4572726F     ..ATLLLL....Erro   |
|   00000150     72206C6F  6767696E  67207361  6D706C65     r logging sample   |
|   00000160     2E205468  65736520  61726520  64756D6D     . These are dumm   |
|   00000170     79206572  726F7273  2E205365  6374696F     y errors. Sectio   |
|   00000180     6E203200  53616D70  6C652065  72726F72     n 2.Sample error   |
|   00000190     2053616D  706C6520  6572726F  72205361      Sample error Sa   |
|   000001A0     6D706C65  20657272  6F722053  616D706C     mple error Sampl   |
```

```
|   000001B0     65206572  726F7220  09090953  616D706C     e error ...Sampl   |
|   000001C0     65206572  726F7220  61626364  65666768     e error abcdefgh   |
|   000001D0     696A6B6C  6D6E6F70  71727374  75767778     ijklmnopqrstuvwx   |
|   000001E0     797A00                                     yz.                |
|-------------------------------------------------------------------------------|
|                       Platform Event Log - 0x533C9B37                         |
|-------------------------------------------------------------------------------|
|                                Private Header                                 |
|-------------------------------------------------------------------------------|
| Section Version          : 1                                                  |
| Sub-section type         : 0                                                  |
| Created by               : 4154                                               |
| Created at               : 07/28/2015 02:00:05                                |
| Committed at             : 07/28/2015 02:00:05                                |
| Creator Subsystem        : OPAL                                               |
| CSSVER                   :                                                    |
| Platform Log Id          : 0xB0000002                                         |
| Entry Id                 : 0x533C9B37                                         |
| Total Log Size           : 483                                                |
|-------------------------------------------------------------------------------|
|                                 User Header                                   |
|-------------------------------------------------------------------------------|
| Section Version          : 1                                                  |
| Sub-section type         : 0                                                  |
| Log Committed by         : 4154                                               |
| Subsystem                : Platform Firmware                                  |
| Event Scope              : Unknown - 0x00000000                               |
| Event Severity           : Predictive Error                                   |
| Event Type               : Not Applicable                                     |
| Return Code              : 0x00000000                                         |
| Action Flags             : Report Externally                                  |
| Action Status            : Sent to Hypervisor                                 |
|-------------------------------------------------------------------------------|
|                        Primary System Reference Code                          |
|-------------------------------------------------------------------------------|
| Section Version          : 1                                                  |
| Sub-section type         : 0                                                  |
| Created by               : 4154                                               |
| SRC Format               : 0x80                                               |
| SRC Version              : 0x02                                               |
| Virtual Progress SRC     : False                                              |
| I5/OS Service Event Bit  : False                                              |
| Hypervisor Dump Initiated: False                                              |
| Power Control Net Fault  : False                                              |
|                                                                               |
| Valid Word Count         : 0x08                                               |
| Reference Code           : BB821410                                           |
| Hex Words 2 - 5          : 00000080 00000000 00000000 00000000                |
| Hex Words 6 - 9          : 00000000 00000000 00000000 00000000                |
|                                                                               |
|-------------------------------------------------------------------------------|
|                             Extended User Header                              |
|-------------------------------------------------------------------------------|
| Section Version          : 1                                                  |
| Sub-section type         : 0                                                  |
| Created by               : 4154                                               |
| Reporting Machine Type   : 8286-42A                                           |
| Reporting Serial Number  : 10784AT                                            |
```

```
| FW Released Ver          :                                                  |
| FW SubSys Version        :                                                  |
| Common Ref Time          : 07/28/2015 02:00:05                              |
| Symptom Id Len           : 0                                                |
| Symptom Id               :                                                  |
|-----------------------------------------------------------------------------|
|                    Machine Type/Model & Serial Number                       |
|-----------------------------------------------------------------------------|
| Section Version          : 1                                                |
| Sub-section type         : 0                                                |
| Created by               : 4154                                             |
| Machine Type Model       : 8286-42A                                         |
| Serial Number            : 10784AT                                          |
|-----------------------------------------------------------------------------|
|                             User Defined Data                               |
|-----------------------------------------------------------------------------|
| Section Version          : 1                                                |
| Sub-section type         : 0                                                |
| Created by               : 4154                                             |
|                                                                             |
|   00000000     4B4B4B4B  00340000  54686973  20697320     KKKK.4..This is    |
|   00000010     61207361  6D706C65  20757365  72206465     a sample user de   |
|   00000020     66696E65  64206461  74612073  65637469     fined data secti   |
|   00000030     6F6E3100                                   on1.               |
|                                                                             |
|-----------------------------------------------------------------------------|
|                             User Defined Data                               |
|-----------------------------------------------------------------------------|
| Section Version          : 1                                                |
| Sub-section type         : 0                                                |
| Created by               : 4154                                             |
|                                                                             |
|   00000000     4C4C4C4C  009F0000  4572726F  72206C6F     LLLL....Error lo   |
|   00000010     6767696E  67207361  6D706C65  2E205468     gging sample. Th   |
|   00000020     65736520  61726520  64756D6D  79206572     ese are dummy er   |
|   00000030     726F7273  2E205365  6374696F  6E203200     rors. Section 2.   |
|   00000040     53616D70  6C652065  72726F72  2053616D     Sample error Sam   |
|   00000050     706C6520  6572726F  72205361  6D706C65     ple error Sample   |
|   00000060     20657272  6F722053  616D706C  65206572      error Sample er   |
|   00000070     726F7220  09090953  616D706C  65206572     ror ...Sample er   |
|   00000080     726F7220  61626364  65666768  696A6B6C     ror abcdefghijkl   |
|   00000090     6D6E6F70  71727374  75767778  797A00       mnopqrstuvwxyz.    |
|                                                                             |
|-----------------------------------------------------------------------------|
```

2.4 OPAL <–> BMC interactions

This document provides information about some of the user-visible interactions that skiboot performs with the BMC.

2.4.1 IPMI sensors

OPAL will interact with a few IPMI sensors during the boot process. These are:

- Boot Count [type 0xc3: OEM reserved]

- FW Boot progress [type 0x0f: System Firmware Progress]

Boot Count: assertion type. When OPAL reaches a late stage of boot, it sets the boot count sensor to 0x02. This is intended to allow the BMC detect a failed or aborted boot, for switching to a known-good firmware image.

FW Boot Progress: assertion type. During boot, skiboot will update this sensor to one of the IPMI-defined progress codes. The codes use by skiboot are:

- **PCI Resource configuration (0x01)**
  - asserted as the PCI devices have been probed and resources allocated
- **Motherboard init (0x14)**
  - asserted as the platform-specific components have been initialised
- **OS boot (0x13)**
  - asserted after skiboot has loaded the PAYLOAD image, and is about to boot it.

2.4.2 Chassis control messages

OPAL uses chassis control messages to instruct the BMC to remove power from the host. These messages are sent during graceful reboot and shutdown processes initiated by the host.

For a BMC-initiated graceful power-down (or reboot), the BMC is expected to send an OEM-defined SEL message, using a SMS_ATN to trigger a BMC-to-host notification. This SEL has a type of 0xc0, and command of 0x04. The data0 field of the SEL indicates shutdown (0x0) or reboot (0x1).

2.4.3 Watchdog support

OPAL supports a BMC watchdog during the boot process. This will be disabled before entering the OS.

2.4.4 Real-time clock

On platforms where a real-time-clock is not available, skiboot may use the IPMI SEL Time as a real-time-clock device.

2.5 GCOV for skiboot

2.5.1 Unit tests

All unit tests are built+run with gcov enabled.

```
make coverage-report
```

will generate a unit test coverage report like: http://open-power.github.io/skiboot/coverage-report/

2.5.2 Skiboot

You can now build Skiboot itself with gcov support, boot it on a machine, do things, and then extract out gcda files to generate coverage reports from real hardware (or a simulator).

2.5.3 Building Skiboot with GCOV

```
SKIBOOT_GCOV=1 make
```

You may need to `make clean` first.

This will build a skiboot lid roughly *twice* the size.

Flash/Install the skiboot.lid and boot.

2.5.4 Extracting GCOV data

The way we extract the gcov data from a system is by dumping the contents of skiboot memory and then parsing the data structures in user space with the extract-gcov utility in the skiboot repo.

mambo:

```
mysim memory fwrite 0x30000000 0x240000 skiboot.dump
```

FSP:

```
getmemproc 30000000 3407872 -fb skiboot.dump
```

linux (e.g. petitboot environment):

```
dd if=/proc/kcore skip=1572864 count=6656  of=skiboot.dump
```

You basically need to dump out the first 3MB of skiboot memory.

Then you need to find out where the gcov data structures are:

```
perl -e "printf '0x%x', 0x30000000 + 0x`grep gcov_info_list skiboot.map|cut -f 1 -d '
↪'`"
```

That address needs to be supplied to the extract-gcov utility:

```
./extract-gcov skiboot.dump 0x3023ec40
```

Once you've run extract-gcov, it will have extracted the gcda files from the skiboot memory image.

You can then run lcov:

```
lcov -b . -q -c -d . -o skiboot-boot.info
--gcov-tool
/opt/cross/gcc-4.8.0-nolibc/powerpc64-linux/bin/powerpc64-linux-gcov
```

IMPORTANT you should point lcov to the gcov for the compiler you used to build skiboot, otherwise you're likely to get errors.

2.6 Memory in skiboot

There are regions of memory we statically allocate for firmware as well as a HEAP region for boot and runtime allocations.

A design principle of skiboot is to attempt not to allocate memory at runtime, or at least keep it to a minimum, and not do so in any critical code path for the system to remain running.

At no point during runtime should a skiboot memory allocation failure cause the system to stop functioning.

2.6.1 HEAP

Dynamic memory allocations go in a single heap. This is identified as Region ibm,firmware-heap and appears as a reserved section in the device tree.

Originally, it was 12582912 bytes in size (declared in mem_map.h). Now, it is 13631488 bytes after being bumped as part of the GCOV work.

We increased heap size as on larger systems, we were getting close to using all the heap once skiboot became 2MB with GCOV.

Heap usage is printed before running the payload.

For example, as of writing, on a dual socket Tuleta:

```
[45215870591,5] SkiBoot skiboot-5.0.1-94-gb759ce2 starting...
[3680939340,5] CUPD: T side MI Keyword = SV830_027
[3680942658,5] CUPD: T side ML Keyword = FW830.00
[15404383291,5] Region ibm,firmware-heap free: 5378072
```

and on a palmetto:

```
[24748502575,5] SkiBoot skiboot-5.0.1-94-gb759ce2 starting...
[9870429550,5] Region ibm,firmware-heap free: 10814856
```

Our memory allocator is simple, a use pattern of:

```
A = malloc();
B = malloc();
free(A);
```

is likely to generate fragmentation, so it should generally be avoided where possible.

2.7 OPAL/Skiboot Nvlink Interface Documentation

2.7.1 Overview

NV-Link is a high speed interconnect that is used in conjunction with a PCI-E connection to create an interface between chips that provides very high data bandwidth. The PCI-E connection is used as the control path to initiate and report status of large data transfers. The data transfers themselves are sent over the NV-Link.

On IBM Power systems the NV-Link hardware is similar to our standard PCI hardware so to maximise code reuse the NV-Link is exposed as an emulated PCI device through system firmware (OPAL/skiboot). Thus each NV-Link capable device will appear as two devices on a system, the real PCI-E device and at least one emulated PCI device used for the NV-Link.

Presently the NV-Link is only capable of data transfers initiated by the target, thus the emulated PCI device will only handle registers for link initialisation, DMA transfers and error reporting (EEH).

2.7.2 Emulated PCI Devices

Each link will be exported as an emulated PCI device with a minimum of two emulated PCI devices per GPU. Emulated PCI devices are grouped per GPU.

The emulated PCI device will be exported as a standard PCI device by the Linux kernel. It has a standard PCI configuration space to expose necessary device parameters. The only functionality available is related to the setup of DMA windows.

Configuration Space Parameters

Vendor ID	0x1014	(IBM)
Device ID	0x04ea	
Revision ID	0x00	
Class	0x068000 / 0x068001	(Bridge Device Other, ProgIf = 0x0 / 0x1)
BAR0/1	TL/DL Registers	
BAR2/3	GEN-ID Registers	(Only for rev-id = 0x1)

TL/DL Registers

Each link has 128KB of TL/DL registers. These will always be mapped to 64-bit BAR#0 of the emulated PCI device configuration space.

```
BAR#0 + 128K +-----------+
             | NTL (64K) |
BAR#0 + 64K  +-----------+
             | DL (64K)  |
BAR#0        +-----------+
```

Generation Registers (GEN-ID)

On POWER9 each link has 64K of generation ID registers for the relaxed ordering mode syncronisation. Refer to the programming guide for details of the register layout in this BAR.

Relaxed ordering mode will be disabled by default as it requires device driver support. Device drivers will need to request relaxed ordering mode through some yet to be designed mechanism.

Vendor Specific Capabilities

```
+----------------+----------------+----------------+----------------+
|  Version (0x02) |   Cap Length   |  Next Cap Ptr  |  Cap ID (0x09) |
+----------------+----------------+----------------+----------------+
|                      Procedure Status Register                     |
+--------------------------------------------------------------------+
|                      Procedure Control Register                    |
+--------------------------------------------------+-----------------+
|             Reserved             |   PCI Dev Flag  |   Link Number  |
+--------------------------------------------------+-----------------+
```

Version

> This refers to the version of the NPU config space. Used by device drivers to determine which fields of the config space they can expect to be available.

Procedure Control Register

Used to start hardware procedures.

Writes will start the corresponding procedure and set bit 31 in the procedure status register. This register must not be written while bit 31 is set in the status register. Performing a write while another procudure is already in progress will abort that procedure.

Reads will return the in progress procedure or the last completed procedure number depending on the procedure status field.

Procedure Numbers:

0. Abort in-progress procedure
1. NOP
2. Unsupported procedure
3. Unsupported procedure
4. Naples PHY - RESET
5. Naples PHY - TX_ZCAL
6. Naples PHY - RX_DCCAL
7. Naples PHY - TX_RXCAL_ENABLE
8. Naples PHY - TX_RXCAL_DISABLE
9. Naples PHY - RX_TRAINING
10. Naples NPU - RESET
11. Naples PHY - PHY preterminate
12. Naples PHY - PHY terminated
13. Witherspoon TL credit validation

Procedure 5 (TX_ZCAL) should only be run once. System firmware will ensure this so device drivers may call this procedure mutiple times.

Procedure Status Register

The procedure status register is used to determine when execution of the procedure number in the control register is complete and if it completed successfully.

This register must be polled frequently to allow system firmware to execute the procedures.

Fields: Bit 31 - Procedure in progress Bit 30 - Procedure complete Bit 3-0 - Procedure completion code

Procedure completion codes: 0 - Procedure completed successfully. 1 - Transient failure. Procedure should be rerun. 2 - Permanent failure. Procedure will never complete successfully. 3 - Procedure aborted. 4 - Unsupported procedure.

PCI Device Flag

Bit 0 - set if the GPU PCIe device associated with this nvlink was found. bit 1 - set if the DL has been taken out of reset.

Link Number

Physical link number this emulated PCI device is associated with. One of 0, 1, 4 or 5 (links 2 & 3 do not exist on Naples).

Reserved

These fields must be ignored and no value should be assumed.

Interrupts

Each link has a single DL/TL interrupt assigned to it. These will be exposed as an LSI via the emulated PCI device. There are 4 links consuming 4 LSI interrupts. The 4 remaining interrupts supported by the corresponding PHB will be routed to OS platform for the purpose of error reporting.

2.7.3 Device Tree Bindings

See *Nvlink Device Tree Bindings*

2.8 Skiboot stable tree rules and releases

If you're at all familiar with the Linux kernel stable trees, this should seem fairly familiar.

The purpose of a -stable tree is to give vendors a stable base to create firmware releases from and to incorporate into service packs. New stable releases contain critical fixes only.

As a general rule, on the most recent skiboot release gets a maintained -stable tree. If you wish to maintain an older tree, speak up! For example, with my IBMer hat on, we'll maintain branches that we ship in products.

2.8.1 What patches are accepted?

- Patches must be obviously correct and tested
  - A Tested-by signoff is *important*
- A patch must fix a real bug
- No trivial patches, such fixups belong in main branch
- Not fix a purely theoretical problem unless you can prove how it's exploitable
- The patch, or an equivalent one, must already be in master
  - Submitting to both at the same time is okay, but backporting is better

2.8.2 HOWTO submit to stable

Two ways: 1. Send patch to the skiboot@ list with "[PATCH stable]" in subject

- This targets the patch *ONLY* to the stable branch.
  - Such commits will *NOT* be merged into master.
- Use this when:
  1. cherry-picking a fix from master
  2. fixing something that is only broken in stable
  3. fix in stable needs to be completely different than in master

  If b or c: explain why.
- If cherry-picking, include the following at the top of your commit message:

```
commit <sha1> upstream.
```

- If the patch has been modified, explain why in description.

2. Add "Cc: stable" above your Signed-off-by line when sending to skiboot@
   - This targets the patch to master and stable.
   - You can target a patch to a specific stable tree with:

     ```
     Cc: stable # 5.1.x
     ```

     and that will target it to the 5.1.x branch.
   - You can ask for prerequisites to be cherry-picked:

     ```
     Cc: stable # 5.1.x 55ae15b Ensure we run pollers in cpu_wait_job()
     Cc: stable # 5.1.x
     ```

     Which means:

     (a) please `git cherry-pick 55ae15b`

     (b) then apply this patch to 5.1.x".

2.8.3 Trees

- https://github.com/open-power/skiboot/ (or via ssh at `git@github.com:open-power/skiboot.git`)
  - (branches are skiboot-X.Y.x - e.g. skiboot-5.1.x)
- Some stable versions may last longer than others
  - So there may be skiboot-5.1.x and skiboot-5.2.x actively maintained and skiboot-5.1.x could possibly outlast skiboot-5.2.x

2.9 Versioning Scheme of skiboot

2.9.1 History

For roughly the first six months of public life, skiboot just presented a git SHA1 as a version "number". This was "user visible" in two places:

1. `/sys/firmware/opal/msglog` the familiar `SkiBoot 71664fd-dirty starting...` message
2. device tree: `/proc/device-tree/ibm,opal/firmware/git-id`

Builds were also referred to by date and by corresponding PowerKVM release. Clearly, this was unlikely to be good practice going forward.

As of skiboot-4.0, this scheme has changed and we now present a version string instead. This better addresses the needs of everybody who is building OpenPower systems.

2.9.2 Current practice

The version string is constructed from a few places and is designed to be *highly* informative about what you're running. For the most part, it should be automatically constructed by the skiboot build system. The only times you need to do something is if you are a) making an upstream skiboot release or b) building firmware to release for your platform(s).

OPAL/skiboot has several consumers, for example:

- IBM shipping POWER8 systems with an FSP (FW810.XX and future)
- OpenPower
- OpenPower partners manufacturing OpenPower systems
- developers, test and support needing to understand what code a system is running

and there are going to be several concurrent maintained releases in the wild, likely build by different teams of people at different companies.

tl;dr; is you're likely going to see version numbers like this (for the hypothetical platforms 'ketchup' and 'mustard'):

- `skiboot-4.0-ketchup-0`
- `skiboot-4.0-ketchup-1`
- `skiboot-4.1-mustard-4`
- `skiboot-4.1-ketchup-0`

If you see *extra* things on the end of the version, then you're running a custom build from a developer (e.g. `skiboot-4.0-1-g23f147e-stewart-dirty-f42fc40` means something to us - explained below).

If you see less, for example `skiboot-4.0`, then you're running a build directly out of the main git tree. Those producing OPAL builds for users must *not* ship like this, even if the tree is identical.

Here are the components of the version string from master:

```
skiboot-4.0-1-g23f147e-debug-occ-stewart-dirty-f42fc40
^        ^^^ ^  ^^^^^^^ ^_______^    ^       ^   ^^^^^^^
|         |  |     |         |        |       |      |
|         |  |     |         |         \     /        - 'git diff|sha1sum'
|         |  |     |         |          \  /
|         |  |     |         |            - built from a dirty tree of $USER
|         |  |     |         |
|         |  |     |          - $EXTRA_VERSION (optional)
|         |  |     |
|         |  |      - git SHA1 of commit built
|         |  |
|         |   - commits head of skiboot-4.0 tag
|         |
|          - skiboot version number ---\
|                                      >--  from  the 'skiboot-4.0' git tag
 - product name (always skiboot)   ---/
```

When doing a release for a particular platform, you are expected to create and tag a branch from master. For the (hypothetical) ketchup platform which is going to do a release based on skiboot-4.0, you would create a tag 'skiboot-4.0-ketchup-0' pointing to the same revision as the 'skiboot-4.0' tag and then make any additional modifications to skiboot that were not in the 4.0 release. So, you could ship a skiboot with the following version string:

```
skiboot-4.0-ketchup-1
^        ^^^ ^       ^
|         |  |       |
|         |  |        - revision for this platform
|         |  |
|         |  |
|         |   - Platform name/version
|         |
|          - skiboot version number
|
 - product name (always skiboot)
```

This version string tells your users to expect what is in skiboot-4.0 plus some revisions for your platform.

2.9.3 Practical Considerations

You MUST correctly tag your git tree for sensible version numbers to be generated. Look at the (generated) version.c file to confirm you're building the correct version number. You will need annotated tags (git tag -a).

If your build infrastructure does *not* build skiboot from a git tree, you should specify SKIBOOT_VERSION as an environment variable (following this versioning scheme), otherwise the build will fail.

2.10 PCI

WARNING: This documentation **urgently needs updating** and is *woefully* incomplete.

2.10.1 IODA PE Setup Sequences

(**WARNING**: this was rescued from old internal documentation. Needs verification)

To setup basic PE mappings, the host performs this basic sequence:

For ibm,opal-ioda2, prior to allocating PHB resources to PEs, the host must allocate memory for PE structures and then calls `opal_pci_set_phb_table_memory( phb_id, rtt_addr, ivt_addr, ivt_len, rrba_addr, peltv_addr)` to define them to the PHB. OPAL returns `OPAL_UNSUPPORTED` status for `ibm, opal-ioda` PHBs.

The host calls `opal_pci_set_pe( phb_id, pe_number, bus, dev, func, validate_mask, bus_mask, dev_mask, func mask)` to map a PE to a PCI RID or range of RIDs in the same PE domain.

The host calls `opal_pci_set_peltv(phb_id, parent_pe, child_pe, state)` to set a parent PELT vector bit for the child PE argument to 1 (a child of the parent) or 0 (not in the parent PE domain).

2.10.2 IODA MMIO Setup Sequences

(**WARNING**: this was rescued from old internal documentation. Needs verification)

The host calls `opal_pci_phb_mmio_enable( phb_id, window_type, window_num, 0x0)` to disable the MMIO window.

The host calls `opal_pci_set_phb_mmio_window( phb_id, mmio_window, starting_real_address, starting_pci_address, segment_size)` to change the MMIO window location in PCI and/or processor real address space, or to change the size – and corresponding window size – of a particular MMIO window.

The host calls `opal_pci_map_pe_mmio_window( pe_number, mmio_window, segment_number)` to map PEs to window segments, for each segment mapped to each PE.

The host calls `opal_pci_phb_mmio_enable( phb_id, window_type, window_num, 0x1)` to enable the MMIO window.

2.10.3 IODA MSI Setup Sequences

(**WARNING**: this was rescued from old internal documentation. Needs verification)

To setup MSIs:

1. For ibm,opal-ioda PHBs, the host chooses an MVE for a PE to use and calls `opal_pci_set_mve( phb_id, mve_number, pe_number,)` to setup the MVE for the PE number. HAL treats this call as a NOP and returns hal_success status for ibm,opal-ioda2 PHBs.
2. The host chooses an XIVE to use with a PE and calls a. `opal_pci_set_xive_pe( phb_id, xive_number, pe_number)` to authorize that PE to signal that XIVE as an interrupt. The host must call this function for each XIVE assigned to a particular PE, but may use this call for all XIVEs prior to calling `opel_pci_set_mve()` to bind the PE XIVEs to an MVE. For MSI conventional, the host must bind a unique MVE for each sequential set of 32 XIVEs. b. The host forms the interrupt_source_number from the combination of the device tree MSI property base BUID and XIVE number, as an input to `opal_set_xive(interrupt_source_number, server_number, priority)` and `opal_get_xive(interrupt_source_number, server_number, priority)` to set or return the server and priority numbers within an XIVE. c. `opal_get_msi_64[32](phb_id, mve_number, xive_num, msi_range, msi_address, message_data)` to determine the MSI DMA address (32 or 64 bit) and message data value for that xive.

   For MSI conventional, the host uses this for each sequential power of 2 set of 1 to 32 MSIs, to determine the MSI DMA address and starting message data value for that MSI range. For MSI-X, the host calls this uniquely for each MSI interrupt with an msi_range input value of 1.
3. For `ibm,opal-ioda` PHBs, once the MVE and XIVRs are setup for a PE, the host calls `opal_pci_set_mve_enable( phb_id, mve_number, state)` to enable that MVE to be a valid target of MSI DMAs. The host may also call this function to disable an MVE when changing PE domains or states.

2.10.4 IODA DMA Setup Sequences

(**WARNING**: this was rescued from old internal documentation. Needs verification)

To Manage DMA Windows :

1. The host calls `opal_pci_map_pe_dma_window( phb_id, dma_window_number, pe_number, tce_levels, tce_table_addr, tce_table_size, tce_page_size, utin64_t* pci_start_addr )` to setup a DMA window for a PE to translate through a TCE table structure in KVM memory.
2. The host calls `opal_pci_map_pe_dma_window_real( phb_id, dma_window_number, pe_number, mem_low_addr, mem_high_addr)` to setup a DMA window for a PE that is translated (but validated by the PHB as an untranlsated address space authorized to this PE).

2.11 PCI Slots

The PCI slots are instantiated to represent their associated properties and operations. The slot properties are exported to OS through the device tree node of the corresponding parent PCI device. The slot operations are used to accomodate requests from OS regarding the indicated PCI slot:

- PCI slot reset
- PCI slot property retrival

The PCI slots are expected to be created by individual platforms based on the given templates, which are classified to PHB slot or normal one currently. The PHB slot is instantiated based on PHB types like P7IOC and PHB3. However, the normal PCI slots are created based on general RC (Root Complex), PCIE switch ports, PCIE-to-PCIx bridge. Individual platform may create PCI slot, which doesn't have existing template.

The PCI slots are created at different stages according to their types. PHB slots are expected to be created once the PHB is register (struct platform::pci_setup_phb()) because the PHB slot reset operations are required at early stage of PCI enumeration. The normal slots are populated after their parent PCI devices are instantiated at struct platform::pci_get_slot_info().

The operation set supplied by the template might be overrided and reimplemented, or partially. It's usually done according to the VPD figured out by individual platforms.

2.11.1 PCI Slot Operations

The following operations are supported to one particular PCI slot. More details could be found from the definition of struct pci_slot_ops:

Operation	Definition
get_presence_state	Check if any adapter connected to slot
get_link_state	Retrieve PCIE link status: up, down, link width
get_power_state	Retrieve the power status: on, off
get_attention_state	Retrieve attention status: on, off, blinking
get_latch_state	Retrieve latch status
set_power_state	Configure the power status: on, off
set_attention_state	Configure attention status: on, off, blinking
prepare_link_change	Prepare PCIE link status change
poll_link	Poll PCIE link until it's up or down permanently
creset	Complete reset, only available to PHB slot
freset	Fundamental reset
hreset	Hot reset
poll	Interface for OPAL API to drive internal state machine
add_properties	Additional PCI slot properties seen by platform

2.12 PCI Slot Properties

The following PCI slot properties have been exported through PCI device tree node for a root port, a PCIE switch port, or a PCIE to PCIx bridge. If the individual platforms (e.g. Firenze and Apollo) have VPD for the PCI slot, they should extract the PCI slot properties from VPD and export them accordingly.

Property	Definition
ibm,reset-by-firmware	Boolean indicating whether the slot reset should be done in firmware
ibm,slot-pluggable	Boolean indicating whether the slot is pluggable
ibm,slot-surprise-pluggable	Boolean indicating whether the slot supports surprise hotplug
ibm,slot-broken-pdc	Boolean indicating whether PDC (Presence Detection Change) is broken
ibm,slot-power-ctl	Boolean indicating whether the slot has power control
ibm,slot-wired-lanes	The number of hardware lanes that are wired
ibm,slot-pwr-led-ctl	Presence of slot power led, and controlling entity
ibm,slot-attn-led-ctl	Presence of slot ATTN led, and controlling entity

2.12.1 PCI Hotplug

The implementation of PCI slot hotplug heavily relies on its power state. Initially, the slot is powered off if there are no adapters behind it. Otherwise, the slot should be powered on.

In hot add scenario, the adapter is physically inserted to PCI slot. Then the PCI slot is powered on by OPAL API opal_pci_set_power_state(). The power is supplied to the PCI slot, the adapter behind the PCI slot is probed and the device sub-tree (for hot added devices) is populated. A OPAL message is sent to OS on completion. The OS needs retrieve the device sub-tree through OPAL API opal_get_device_tree(), unflatten it and populate the device sub-tree. After that, the adapter behind the PCI slot should be probed and added to the system.

On the other hand, the OS removes the adapter behind the PCI slot before calling opal_pci_set_power_state(). Skiboot cuts off the power supply to the PCI slot, removes the adapter behind the PCI slot and the corresponding device sub-tree. A OPAL message (OPAL_MSG_ASYNC_COMP) is sent to OS. The OS removes the device sub-tree for the adapter behind the PCI slot.

The OPAL message used in PCI hotplug is comprised of 4 dwords in sequence: asychronous token from OS, PCI slot device node's phandle, OPAL_PCI_SLOT_POWER_{ON, OFF}, OPAL_SUCCESS or errcode.

The states OPAL_PCI_SLOT_OFFLINE and OPAL_PCI_SLOT_ONLINE are used for removing or adding devices behind the slot. The device nodes in the device tree are removed or added accordingly, without actually changing the slot's power state. The API call will return OPAL_SUCCESS immediately and no further asynchronous message will be sent.

2.12.2 PCI Slot on Apollo and Firenze

On IBM's Apollo and Firenze platform, the PCI VPD is fetched from dedicated LID, which is organized in so-called 1004, 1005, or 1006 format. 1006 mapping format isn't supported currently. The PCI slot properties are figured out from the VPD. On the other hand, there might have external power management entity hooked to I2C buses for one PCI slot. The fundamental reset operation of the PCI slot should be implemented based on the external power management entity for that case.

On Firenze platform, PERST pin is accessible through bit#10 of PCI config register (offset: 0x80) for those PCI slots behind some PLX switch downstream ports. For those PCI slots, PERST pin is utilized to implement fundamental reset if external power management entity doesn't exist.

For Apollo and Firenze platform, following PCI slot properties are exported through PCI device tree node except those generic properties (as above):

Property	Definition
ibm,slot-location-code	System location code string for the slot connector
ibm,slot-label	Slot label, part of "ibm,slot-location-code"

2.13 XSCOM Bindings

2.13.1 XSCOM regions

The top-level xscom nodes specify the mapping range from the 64-bit address space into the PCB address space.

There's one mapping range per chip xscom, therefore one node per mapping range.

```
/
/xscom@<chip-base-address-0>/
/xscom@<chip-base-address-1>/
...
/xscom@<chip-base-address-n>/
```

- where <chip-base-address-n> is the xscom base address with the gcid-specific bits (for chip n) OR-ed in.

Each xscom node has the following properties:

- #address-cells = 1
- #size-cells = 1
- reg = <base-address[#parent-address-cells] size[#parent-size-cells]>
- ibm,chip-id = gcid
- compatible = "ibm,xscom", "ibm,power8-scom" / "ibm,power7-xscom"
- ecid = <Electronic Chip ID, applicable for POWER9 onwards>
- wafer-id = <wafer ID, applicable for POWER9 onwards>
- wafer-location = <wafer location, applicable for POWER9 onwards>

2.13.2 ECID

Electronic Chip ID (ECID) is a process by which the wafer number, chip location (i.e. X,Y) and other optional data items are electrically encoded directly on the chip. wafer-id property represents wafer number and wafer-location property represents chip location (both X and Y location).

2.13.3 Chiplet endpoints

One sub-node per endpoint. Endpoints are defined by their (port, endpoint-address) data on the PCB, and are named according to their endpoint types:

```
/xscom@<chip-base-address>/
/xscom@<chip-base-address>/chiptod@<endpoint-addr>
/xscom@<chip-base-address>/lpc@<endpoint-addr>
```

- where the <endpoint-addr> is a single address (as distinct from the current (gcid,base) format), consisting of the SCOM port and SCOM endpoint bits in their 31-bit address format.

Each endpoint node has the following properties:

- reg = <endpoint-address[#parent-address-cells] size[#parent-size-cells]>
- compatible - depends on endpoint type, eg "ibm,power8-chiptod"

The endpoint address specifies the address on the PCB. So, to calculate the MMIO address for a PCB register:

```
mmio_addr  = <xscom-base-addr> | (pcb_addr[1:27] << 4)
                               | (pcb_addr[28:31] << 3)
```

Where:

- xscom-base-addr is the address from the first two cells of the parent node's reg property
- pcb_addr is the first cell of the endpoint's reg property

2.14 P9 XIVE Exploitation

2.14.1 I - Device-tree updates

1. The existing OPAL `/interrupt-controller@0` node remains

This node represents both the emulated XICS source controller and an abstraction of the virtualization engine. This represents the fact thet OPAL set_xive/get_xive functions are still supported though they don't provide access to the full functionality.

It is still the parent of all interrupts in the device-tree.

New or modified properties:

- `compatible` : This is extended with a new value `ibm,opal-xive-vc`

2. The new `/interrupt-controller@<addr>` node

   This node represents both the emulated XICS presentation controller and the new XIVE presentation layer.

   Unlike the traditional XICS, there is only one such node for the whole system.

   New or modified properties:

   - `compatible` : This contains at least the following strings:
     - `ibm,opal-intc` : This represents the emulated XICS presentation facility and might be the only property present if the version of OPAL doesn't support XIVE exploitation.
     - `ibm,opal-xive-pe` : This represents the XIVE presentation engine.
   - `ibm,xive-eq-sizes` : One cell per size supported, contains log2 of size, in ascending order.
   - `ibm,xive-#priorities` : One cell, the number of supported priorities (the priorities will be 0...n)
   - `ibm,xive-provision-page-size` : Page size (in bytes) of the pages to pass to OPAL for provisioning internal structures (see opal_xive_donate_page). If this is absent, OPAL will never require additional provisioning. The page must be naturally aligned.
   - `ibm,xive-provision-chips` : The list of chip IDs for which provisioning is required. Typically, if a VP allocation return OPAL_XIVE_PROVISIONING, opal_xive_donate_page() will need to be called to donate a page to *each* of these chips before trying again.
   - `reg` property contains the addresses & sizes for the register ranges corresponding respectively to the 4 rings:
     - Ultravisor level
     - Hypervisor level
     - Guest OS level
     - User level

     For any of these, a size of 0 means this level is not supported.
   - `single-escalation-support` (option). When present, indicatges that the "single escalation" feature is supported, thus enabling the use of the OPAL_XIVE_VP_SINGLE_ESCALATION flag.

3. Interrupt descriptors

   The interrupt descriptors (aka "interrupts" properties and parts of "interrupt-map" properties) remain 2 cells. The first cell is a global interrupt number which represents a unique interrupt source in the system and is an abstraction provided by OPAL.

   The default configuration for all sources in the IVT/EAS is to issue that number (it's internally a combination of the source chip and per-chip interrupt number but the details of that combination are not exposed and subject to change).

   The second cell remains as usual "0" for an edge interrupt and "1" for a level interrupts.

4. IPIs

   Each `cpu` node now contains an `interrupts` property which has one entry (2 cells per entry) for each thread on that core containing the interrupt number for the IPI targeted at that thread.

5. Interrupt targets

   Targetting of interrupts uses processor targets and priority numbers. The processor target encoding depends on which API is used:

   - The legacy opal_set/get_xive() APIs only support the old "mangled" (ie. shifted by 2) HW processor numbers.
   - The new opal_xive_set/get_irq_config API (and other exploitation mode APIs) use a "token" VP number which is described in II-2. Unmodified HW processor numbers are valid VP numbers for those APIs.

2.14.2 II - General operations

Most configuration operations are abstracted via OPAL calls, there is no direct access or exposure of such things as real HW interrupt or VP numbers.

OPAL sets up all the physical interrupts and assigns them numbers, it also allocates enough virtual interrupts to provide an IPI per physical thread in the system.

All interrupts are pre-configured masked and must be set to an explicit target before first use. The default interrupt number is programmed in the EAS and will remain unchanged if the targetting/unmasking is done using the legacy set_xive() interface.

An interrupt "target" is a combination of a target processor number and a priority.

Processor numbers are in a single domain that represents both the physical processors and any virtual processor or group allocated using the interfaces defined in this specification. These numbers are an OPAL maintained abstraction and are only partially related to the real VP numbers:

In order to maintain the grouping ability, when VPs are allocated in blocks of naturally aligned powers of 2, the underlying HW numbers will respect this alignment.

Note: The block group mode extension makes the numbering scheme a bit more tricky than simple powers of two however, see below.

---

1. Interrupt numbering and allocation

   As specified in the device-tree definition, interrupt numbers are abstracted by OPAL to be a 30-bit number. All HW interrupts are "allocated" and configured at boot time along with enough IPIs for all processor threads.

   Additionally, in order to be compatible with the XICS emulation, all interrupt numbers present in the device-tree (ie all physical sources or pre-allocated IPIs) will fit within a 24-bit number space.

   Interrupt sources that are only usable in exploitation mode, such as escalation interrupts, can have numbers covering the full 30-bit range. The same is true of interrupts allocated dynamically.

   The hypervisor can allocate additional blocks of interrupts, in which case OPAL will return the resulting abstracted global numbers. They will have to be individually configured to map to a given number at the target and be routed to a given target and priority using opal_xive_set_irq_config(). This call is semantically equivalent to the old opal_set_xive() which is still supported with the addition that opal_xive_set_irq_config() can also specify the logical interrupt number.

2. VP numbering and allocation

A VP number is a 64-bit number. The internal make-up of that number is opaque to the OS. However, it is a discrete integer that will be a naturally aligned power of two when allocating a chunk of VPs representing the "base" number of that chunk, the OS will do basic arithmetic to get to all the VPs in the range.

Groups, when supported, will also be numbers in that space.

The physical processors numbering uses the same number space.

The underlying HW VP numbering is hidden from the OS, the APIs uses the system processor numbers as presented in the `ibm,ppc-interrupt-server#s` which corresponds to the PIR register content to represent physical processors within the same number space as dynamically allocated VPs.

Note: Note about block group mode:

The block group mode shall as much as possible be handled transparently by OPAL.

For example, on a 2-chips machine, a request to allocate 2^n VPs might result in an allocation of 2^(n-1) VPs per chip allocated accross 2 chips. The resulting VP numbers will encode the order of the allocation allowing OPAL to reconstitute which bits are the block ID bits and which bits are the index bits in a way transparent to the OS. The overall range of numbers passed to Linux will still be contiguous.

That implies however a limitation: We can only allocate within power-of-two number of blocks. Thus the VP allocator will limit itself to the largest power of two that can fit in the number of available chips in the machine: A machine with 3 good chips will only be able to allocate VPs from 2 of them.

---

3. Group numbering and allocation

   The group numbers are in the *same* number space as the VP numbers. OPAL will internally use some bits of the VP number to encode the group geometry.

   [TBD] OPAL may or may not allocate a default group of all physical processors, per-chip groups or per-core groups. This will be represented in the device-tree somewhat...

   [TBD] OPAL will provide interfaces for allocating groups

Note: Note about P/Q bit operation on sources:

opal_xive_get_irq_info() returns a certain number of flags which define the type of operation supported. The following rules apply based on what those flags say:

- The Q bit isn't functional on an LSI interrupt. There is no garantee that the special combination "01" will work for an LSI (and in fact it will not work on the PHB LSIs). However just setting P to 1 is sufficient to mask an LSI (just don't EOI it while masked).
- The recommended setting for a masked interrupt that is temporarily masked by a driver is "10". This means a new occurrence while masked will be recorded and a "StoreEOI" will replay it appropriately.

2.14.3 III - Event queues

Each virtual processor or group has a certain number of event queues associated with it. Each correspond to a given priority. The number of supported priorities is provided in the device-tree (`ibm,xive-#priorities` property of the xive node).

By default, OPAL populates at least one queue for every physical thread in the system. The number of queues and the size used is implementation specific. If the OS wants to re-use these to save memory, it can query the VP configuration.

The opal_xive_get_queue_info() and opal_xive_set_queue_info() can be used to query a queue configuration (ie, to obtain the current page and size for the queue itself, but also to collect some configuration flags for that queue such as

whether it coalesces notifications etc...) and to obtain the MMIO address of the queue EOI page (in the case where coalescing is enabled).

2.14.4 IV - OPAL APIs

Warning: *All* the calls listed below may return OPAL_BUSY unless explicitely documented not to. In that case, the call should be performed again. The OS is allowed to insert a delay though no minimum nor maxmimum delay is specified. This will typically happen when performing cache update operations in the XIVE, if they result in a collision.

Warning: Calls that are expected to be called at runtime simultaneously without conflicts such as getting/setting IRQ info or queue info are fine to do so concurrently.

However, there is no internal locking to prevent races between things such as freeing a VP block and getting/setting queue infos on that block.

These aren't fully specified (yet) but common sense shall apply.

OPAL_XIVE_RESET

```
int64_t opal_xive_reset(uint64_t version)
```

The OS should call this once when starting up to re-initialize the XIVE hardware and the OPAL XIVE related state back to all defaults.

It can call it a second time before handing over to another (ie. kexec) to re-enable XICS emulation.

The "version" argument should be set to 1 to enable the XIVE exploitation mode APIs or 0 to switch back to the default XICS emulation mode.

Future versions of OPAL might allow higher versions than 1 to represent newer versions of this API. OPAL will return an error if it doesn't recognize the requested version.

Any page of memory that the OS has "donated" to OPAL, either backing store for EQDs or VPDs or actual queue buffers will be removed from the various HW maps and can be re-used by the OS or freed after this call regardless of the version information. The HW will be reset to a (mostly) clean state.

It is the responsibility of the caller to ensure that no other XIVE or XICS emulation call happens simultaneously to this. This basically should happen on an otherwise quiescent system. In the case of kexec, it is recommended that all processors CPPR is lowered first.

Note: This call always executes fully synchronously, never returns OPAL_BUSY and will work regardless of whether VPs and EQs are left enabled or disabled. It *will* spend a significant amount of time inside OPAL and as such is not suitable to be performed during normal runtime.

OPAL_XIVE_GET_IRQ_INFO

```
int64_t opal_xive_get_irq_info(uint32_t girq,
                               uint64_t *out_flags,
                               uint64_t *out_eoi_page,
```

```
                                uint64_t *out_trig_page,
                                uint32_t *out_esb_shift,
                                uint32_t *out_src_chip);
```

Returns info about an interrupt source. This call never returns OPAL_BUSY.

- out_flags returns a set of flags. The following flags are defined in the API (some bits are reserved, so any bit not defined here should be ignored):
  - OPAL_XIVE_IRQ_TRIGGER_PAGE

    Indicate that the trigger page is a separate page. If that bit is clear, there is either no trigger page or the trigger can be done in the same page as the EOI, see below.
  - OPAL_XIVE_IRQ_STORE_EOI

    Indicates that the interrupt supports the "Store EOI" option, ie a store to the EOI page will move Q into P and retrigger if the resulting P bit is 1. If this flag is 0, then a store to the EOI page will do a trigger if OPAL_XIVE_IRQ_TRIGGER_PAGE is also 0.
  - OPAL_XIVE_IRQ_LSI

    Indicates that the source is a level sensitive source and thus doesn't have a functional Q bit. The Q bit may or may not be implemented in HW but SW shouldn't rely on it doing anything.
  - OPAL_XIVE_IRQ_SHIFT_BUG

    Indicates that the source has a HW bug that shifts the bits of the "offset" inside the EOI page left by 4 bits. So when this is set, us 0xc000, 0xd000... instead of 0xc00, 0xd00... as offets in the EOI page.
  - OPAL_XIVE_IRQ_MASK_VIA_FW

    Indicates that a FW call is needed (either opal_set_xive() or opal_xive_set_irq_config()) to succesfully mask and unmask the interrupt. The operations via the ESB page aren't fully functional.
  - OPAL_XIVE_IRQ_EOI_VIA_FW

    Indicates that a FW call to opal_xive_eoi() is needed to successfully EOI the interrupt. The operation via the ESB page isn't fully functional.

    * out_eoi_page and out_trig_page outputs will be set to the EOI page physical address (always) and the trigger page address (if it exists). The trigger page may exist even if OPAL_XIVE_IRQ_TRIGGER_PAGE is not set. In that case out_trig_page is equal to out_eoi_page. If the trigger page doesn't exist, out_trig_page is set to 0.
    * out_esb_shift contains the size (as an order, ie 2^n) of the EOI and trigger pages. Current supported values are 12 (4k) and 16 (64k). Those cannot be configured by the OS and are set by firmware but can be different for different interrupt sources.
    * out_src_chip will be set to the chip ID of the HW entity this interrupt is sourced from. It's meant to be informative only and thus isn't guaranteed to be 100% accurate. The idea is for the OS to use that to pick up a default target processor on the same chip.

OPAL_XIVE_EOI

```
int64_t opal_xive_eoi(uint32_t girq);
```

Performs an EOI on the interrupt. This should only be called if OPAL_XIVE_IRQ_EOI_VIA_FW is set as otherwise direct ESB access is preferred.

Note: This is the *same* opal_xive_eoi() call used by OPAL XICS emulation. However the XIRR parameter is re-purposed as "GIRQ".

The call will perform the appropriate function depending on whether OPAL is in XICS emulation mode or native XIVE exploitation mode.

OPAL_XIVE_GET_IRQ_CONFIG

```
int64_t opal_xive_get_irq_config(uint32_t girq, uint64_t *out_vp,
                                 uint8_t *out_prio, uint32_t *out_lirq);
```

Returns current the configuration of an interrupt source. This is the equivalent of opal_get_xive() with the addition of the logical interrupt number (the number that will be presented in the queue).

- girq: The interrupt number to get the configuration of as provided by the device-tree.
- out_vp: Will contain the target virtual processor where the interrupt is currently routed to. This can return 0xffffffff if the interrupt isn't routed to a valid virtual processor.
- out_prio: Will contain the priority of the interrupt or 0xff if masked
- out_lirq: Will contain the logical interrupt assigned to the interrupt. By default this will be the same as girq.

OPAL_XIVE_SET_IRQ_CONFIG

```
int64_t opal_xive_set_irq_config(uint32_t girq, uint64_t vp, uint8_t prio,
                                 uint32_t lirq);
```

This allows configuration and routing of a hardware interrupt. This is equivalent to opal_set_xive() with the addition of the ability to configure the logical IRQ number (the number that will be presented in the target queue).

- girq: The interrupt number to configure of as provided by the device-tree.
- vp: The target virtual processor. The target VP/Prio combination must already exist, be enabled and populated (ie, a queue page must be provisioned for that queue).
- prio: The priority of the interrupt.
- lirq: The logical interrupt number assigned to that interrupt

Note: Note about masking:

If the prio is set to 0xff, this call will cause the interrupt to be masked (*). This function will not clobber the source P/Q bits (**). It will however set the IVT/EAS "mask" bit if the prio passed is 0xff which means that interrupt events from the ESB will be discarded, potentially leaving the ESB in a stale state. Thus care must be taken by the caller to "cleanup" the ESB state appropriately before enabling an interrupt with this.

(*) Escalation interrupts cannot be masked via this function

(**) The exception to this rule is interrupt sources that have the OPAL_XIVE_IRQ_MASK_VIA_FW flag set. For such sources, the OS should make no assumption as to the state of the ESB and this function *will* perform all the necessary masking and unmasking.

Note: This call contains an implicit opal_xive_sync() of the interrupt source (see OPAL_XIVE_SYNC below)

It is recommended for an OS exploiting the XIVE directly to not use this function for temporary driver-initiated masking of interrupts but to directly mask using the P/Q bits of the source instead.

Masking using this function is intended for the case where the OS has no handler registered for a given interrupt anymore or when registering a new handler for an interrupt that had none. In these case, losing interrupts happening while no handler was attached is considered fine.

OPAL_XIVE_GET_QUEUE_INFO

```
int64_t opal_xive_get_queue_info(uint64_t vp, uint32_t prio,
                                 uint64_t *out_qpage,
                                 uint64_t *out_qsize,
                                 uint64_t *out_qeoi_page,
                                 uint32_t *out_escalate_irq,
                                 uint64_t *out_qflags);
```

This returns informations about a given interrupt queue associated with a virtual processor and a priority.

- out_qpage: will contain the physical address of the page where the interrupt events will be posted or 0 if none has been configured yet.
- out_qsize: will contain the log2 of the size of the queue buffer or 0 if the queue hasn't been populated. Example: 12 for a 4k page.
- out_qeoi_page: will contain the physical address of the MMIO page used to perform EOIs for the queue notifications.
- out_escalate_irq: will contain a girq number for the escalation interrupt associated with that queue.

Warning: The "escalate_irq" is a special interrupt number, depending on the implementation it may or may not correspond to a normal XIVE source. Those interrupts have no triggers, and will not be masked by opal_set_irq_config() with a prio of 0xff.

..note:: The state of the OPAL_XIVE_VP_SINGLE_ESCALATION flag passed to
opal_xive_set_vp_info() can change the escalation irq number, so make sure you only retrieve this after having set the flag to the desired value. When set, all priorities will have the same escalation interrupt.

- out_qflags: will contain flags defined as follow:
  - OPAL_XIVE_EQ_ENABLED

    This must be set for the queue to be enabled and thus a valid target for interrupts. Newly allocated queues are disabled by default and must be disabled again before being freed (allocating and freeing of queues currently only happens along with their owner VP).

    Note: A newly enabled queue will have the generation set to 1 and the queue pointer to 0. If the OS wants to "reset" a queue generation and pointer, it thus must disable and re-enable the queue.

  - OPAL_XIVE_EQ_ALWAYS_NOTIFY

    When this is set, the HW will always notify the VP on any new entry in the queue, thus the queue own P/Q bits won't be relevant and using the EOI page will be unnecessary.

- OPAL_XIVE_EQ_ESCALATE

  When this is set, the EQ will escalate to the escalation interrupt when failing to notify.

OPAL_XIVE_SET_QUEUE_INFO

```
int64_t opal_xive_set_queue_info(uint64_t vp, uint32_t prio,
                                 uint64_t qpage,
                                 uint64_t qsize,
                                 uint64_t qflags);
```

This allows the OS to configure the queue page for a given processor and priority and adjust the behaviour of the queue via flags.

- qpage: physical address of the page where the interrupt events will be posted. This has to be naturally aligned.
- qsize: log2 of the size of the above page. A 0 here will disable the queue.
- qflags: Flags (see definitions in opal_xive_get_queue_info)

Note: This call will reset the generation bit to 1 and the queue production pointer to 0.

Note: The PQ bits of the escalation interrupts and of the queue notification will be set to 00 when OPAL_XIVE_EQ_ENABLED is set, and to 01 (masked) when disabling it.

Note: This must be called at least once on a queue with the flag OPAL_XIVE_EQ_ENABLED in order to enable it after it has been allocated (along with its owner VP).

Note: When the queue is disabled (flag OPAL_XIVE_EQ_ENABLED cleared) all other flags and arguments are ignored and the queue configuration is wiped.

OPAL_XIVE_DONATE_PAGE

```
int64_t opal_xive_donate_page(uint32_t chip_id, uint64_t addr);
```

This call is used to donate pages to OPAL for use by VP/EQ provisioning.

The pages must be of the size specified by the "ibm,xive-provision-page-size" property and naturally aligned.

All donated pages are forgotten by OPAL (and thus returned to the OS) on any call to opal_xive_reset().

The chip_id should be the chip on which the pages were allocated or -1 if unspecified. Ideally, when a VP allocation request fails with the OPAL_XIVE_PROVISIONING error, the OS should allocate one such page for each chip in the system and hand it to OPAL before trying again.

Note: It is possible that the provisioning ends up requiring more than one page per chip. OPAL will keep returning the above error until enough pages have been provided.

OPAL_XIVE_ALLOCATE_VP_BLOCK

```
int64_t opal_xive_alloc_vp_block(uint32_t alloc_order);
```

This call is used to allocate a block of VPs. It will return a number representing the base of the block which will be aligned on the alloc order, allowing the OS to do basic arithmetic to index VPs in the block.

The VPs will have queue structures reserved (but not initialized nor provisioned) for all the priorities defined in the "ibm,xive-#priorities" property

This call might return OPAL_XIVE_PROVISIONING. In this case, the OS must allocate pages and provision OPAL using opal_xive_donate_page(), see the documentation for opal_xive_donate_page() for details.

The resulting VPs must be individudally enabled with opal_xive_set_vp_info below with the OPAL_XIVE_VP_ENABLED flag set before use.

For all priorities, the corresponding queues must also be individually provisioned and enabled with opal_xive_set_queue_info.

OPAL_XIVE_FREE_VP_BLOCK

```
int64_t opal_xive_free_vp_block(uint64_t vp);
```

This call is used to free a block of VPs. It must be called with the same *base* number as was returned by opal_xive_alloc_vp() (any index into the block will result in an OPAL_PARAMETER error).

The VPs must have been previously all disabled with opal_xive_set_vp_info below with the OPAL_XIVE_VP_ENABLED flag cleared before use.

All the queues must also have been disabled.

Failure to do any of the above will result in an OPAL_XIVE_FREE_ACTIVE error.

OPAL_XIVE_GET_VP_INFO

```
int64_t opal_xive_get_vp_info(uint64_t vp,
                              uint64_t *flags,
                              uint64_t *cam_value,
                              uint64_t *report_cl_pair,
                              uint32_t *chip_id);
```

This call returns information about a VP:

- flags:
  - OPAL_XIVE_VP_ENABLED

    Returns the enabled state of the VP
  - OPAL_XIVE_VP_SINGLE_ESCALATION (if available)

    Returns whether single escalation mode is enabled for this VP (see opal_xive_set_vp_info()).
- cam_value: This is the value to program into the thread management area to dispatch that VP (ie, an encoding of the block + index).
- report_cl_pair: This is the real address of the reporting cache line pair for that VP (defaults to 0, ie disabled)
- chip_id: The chip that VCPU was allocated on

OPAL_XIVE_SET_VP_INFO

```
int64_t opal_xive_set_vp_info(uint64_t vp,
                              uint64_t flags,
                              uint64_t report_cl_pair);
```

This call configures a VP:

- flags:
  - OPAL_XIVE_VP_ENABLED

    This must be set for the VP to be usable and cleared before freeing it.

    Note: This can be used to disable the boot time VPs though this isn't recommended. This must be used to enable allocated VPs.

  - OPAL_XIVE_VP_SINGLE_ESCALATION (if available)

    If this is set, the queues are configured such that all priorities turn into a single escalation interrupt. This results in the loss of priority 7 which can no longer be used. This this needs to be set before any interrupt is routed to that priority and queue 7 must not have been already enabled.

    This feature is available if the "single-escalation-property" is present in the xive device-tree node.

    Warning: When enabling single escalation, and pre-existing routing and configuration of the individual queues escalation is lost (except queue 7 which is the new merged escalation). When further disabling it, the previous value is not retrieved and the field cleared, escalation is disabled on all the queues.

- report_cl_pair: This is the real address of the reporting cache line pair for that VP or 0 to disable.

  Note: When disabling a VP, all other VP settings are lost.

OPAL_XIVE_ALLOCATE_IRQ

```
int64_t opal_xive_allocate_irq(uint32_t chip_id);
```

This call allocates a software IRQ on a given chip. It returns the interrupt number or a negative error code.

OPAL_XIVE_FREE_IRQ

```
int64_t opal_xive_free_irq(uint32_t girq);
```

This call frees a software IRQ that was allocated by opal_xive_allocate_irq. Passing any other interrupt number will result in an OPAL_PARAMETER error.

OPAL_XIVE_SYNC

```
int64_t opal_xive_sync(uint32_t type, uint32_t id);
```

This call is uses to synchronize some HW queues to ensure various changes have taken effect to the point where their effects are visible to the processor.

- type: Type of synchronization:
  - XIVE_SYNC_EAS: Synchronize a source. "id" is the girq number of the interrupt. This will ensure that any change to the PQ bits or the interrupt targetting has taken effect.
  - XIVE_SYNC_QUEUE: Synchronize a target queue. "id" is the girq number of the interrupt. This will ensure that any previous occurrence of the interrupt has reached the in-memory queue and is visible to the processor.

    Note: XIVE_SYNC_EAS and XIVE_SYNC_QUEUE can be used together (ie. XIVE_SYNC_EAS | XIVE_SYNC_QUEUE) to completely synchronize the path of an interrupt to its queue.

- id: Depends on the synchronization type, see above

OPAL_XIVE_DUMP

```
int64_t opal_xive_dump(uint32_t type, uint32_t id);
```

This is a debugging call that will dump in the OPAL console various state information about the XIVE.

- type: Type of info to dump:
  - **XIVE_DUMP_TM_HYP: Dump the TIMA area for hypervisor physical thread** "id" is the PIR value of the thread
  - **XIVE_DUMP_TM_POOL: Dump the TIMA area for the hypervisor pool** "id" is the PIR value of the thread
  - **XIVE_DUMP_TM_OS: Dump the TIMA area for the OS** "id" is the PIR value of the thread
  - **XIVE_DUMP_TM_USER: Dump the TIMA area for the "user" area (unsupported)** "id" is the PIR value of the thread
  - **XIVE_DUMP_VP: Dump the state of a VP structure** "id" is the VP id
  - **XIVE_DUMP_EMU: Dump the state of the XICS emulation for a thread** "id" is the PIR value of the thread

2.15 OPAL/Skiboot In-Memory Collection (IMC) interface Documentation

2.15.1 Overview:

In-Memory-Collection (IMC) is performance monitoring infrastrcuture for counters that (once started) can be read from memory at any time by an operating system. Such counters include those for the Nest and Core units, enabling continuous monitoring of resource utilisation on the chip.

The API is agnostic as to how these counters are implemented. For the Nest units, they're implemented by having microcode in an on-chip microcontroller and for core units, they are implemented as part of core logic to gather data and periodically write it to the memory locations.

2.15.2 Nest (On-Chip, Off-Core) unit:

Nest units have dedicated hardware counters which can be programmed to monitor various chip resources such as memory bandwidth, xlink bandwidth, alink bandwidth, PCI, NVlink and so on. These Nest unit PMU counters can be programmed in-band via scom. But alternatively, programming of these counters and periodically moving the counter data to memory are offloaded to a hardware engine part of OCC (On-Chip Controller).

Microcode, starts to run at system boot in OCC complex, initialize these Nest unit PMUs and periodically accumulate the nest pmu counter values to memory. List of supported events by the microcode is packages as a DTS and stored in IMA_CATALOG partition.

2.15.3 Core unit:

Core IMC PMU counters are handled in the core-imc unit. Each core has 4 Core Performance Monitoring Counters (CPMCs) which are used by Core-IMC logic. Two of these are dedicated to count core cycles and instructions. The 2 remaining CPMCs have to multiplex 128 events each.

Core IMC hardware does not support interrupts and it peridocially (based on sampling duration) fetches the counter data and accumulate to main memory. Memory to accumulate counter data are refered from "PDBAR" (per-core scom) and "LDBAR" per-thread spr.

2.15.4 OPAL APIs:

The OPAL API is simple: a call to init a counter type, and calls to start and stop collection. The memory locations are described in the device tree.

See *OPAL_IMC_COUNTERS_INIT* and *IMC Device Tree Bindings*

CHAPTER

THREE

OPAL ABI

3.1 Secure and Trusted Boot Overview

Just as a quick reference:

```
Secure boot:  verify and enforce.
Trusted boot: measure and record.
```

Secure boot seeks to protect system integrity from execution of malicious code during boot. The authenticity and integrity of every code is verified by its predecessor code before it is executed. If the verification fails, the boot process is aborted.

Trusted boot does not perform enforcement. Instead it creates artifacts during system boot to prove that a particular chain of events have happened during boot. Interested parties can subsequently assess the artifacts to check whether or not only trusted events happened and then make security decisions. These artifacts comprise a log of measurements and the digests extended into the TPM PCRs. Platform Configuration Registers (PCRs) are registers in the Trusted Platform Module (TPM) that are shielded from direct access by the CPU.

Trusted boot measures and maintains in an Event Log a record of all boot events that may affect the security state of the platform. A measurement is calculated by hashing the data of a given event. When a new measurement is added to the Event Log, the same measurement is also sent to the TPM, which performs an extend operation to incrementally update the existing digest stored in a PCR.

PCR extend is an operation that uses a hash function to combine a new measurement with the existing digest saved in the PCR. Basically, it concatenates the existing PCR value with the received measurement, and then stores the hash of this string in the PCR.

The TPM may maintain multiple banks of PCRs, where a PCR bank is a collection of PCRs that are extended with the same hash algorithm. TPM 2.0 has a SHA1 bank and a SHA256 bank with 24 PCRs each.

When the system boot is complete, each non-zero PCR value represents one or more events measured during the boot in chronological order. Interested parties can make inferences about the system's state by using an attestation tool to remotely compare the PCR values of a TPM against known good values, and also identify unexpected events by replaying the Event Log against known good Event Log entries.

3.1.1 Implementation in skiboot

Libstb implements an API for secure and trusted boot, which is used to verify and measure images retrieved from PNOR. The libstb interface is documented in `libstb/stb.h`

The example below shows how libstb can be used to add secure and trusted boot support for a platform:

```
stb_init();
    start_preload_resource(RESOURCE_ID_CAPP, 0, capp_ucode_info.lid, &capp_ucode_info.
↪size);
        sb_verify(id, buf, len);
        tb_measure(id, buf, len);
    start_preload_resource(RESOURCE_ID_KERNEL, 0, KERNEL_LOAD_BASE, &kernel_size);
        sb_verify(id, buf, len);
        tb_measure(id, buf, len);
stb_final();
```

First, `stb_init()` must be called to initialize libstb. Basically, it reads both secure mode and trusted mode flags and loads drivers accordingly. In P8, secure mode and trusted mode are read from the *ibm,secureboot* device tree node (see *ibm,secureboot*).

If either secure mode or trusted mode is on, `stb_init()` loads a driver (romcode driver) to access the verification and SHA512 functions provided by the code stored in the secure ROM at manufacture time. Both secure boot and trusted boot depends on the romcode driver to access the ROM code. If trusted mode is on, `stb_init()` loads a TPM device driver compatible with the tpm device tree node and also initializes the existing event log in skiboot. For device tree bindings for the TPM, see *Trusted Platform Module (TPM)*.

Once libstb is initialized in the platform, `sb_verify()` and `tb_measure()` can used as shown in the example above to respectively verify and measure images retrieved from PNOR. If a platform claims secure and trusted boot support, then `sb_verify()` and `tb_measure()` is called for all images retrieved from PNOR.

`sb_verify()` and `tb_measure()` do nothing if libstb is not initialized in the platform since both secure mode and trusted mode are off by default.

Finally, `stb_final()` must be called when no more images need to be retrieved from PNOR in order to indicate that secure boot and trusted boot have completed in skiboot. When stb_final() is called, basically it records eight *EV_SEPARATOR* events in the event log (one for each PCR through 0 to 7) and extends the PCR through 0 to 7 of both SHA1 and SHA256 PCR banks with the digest of *0xFFFFFFFF*. Additionally, `stb_final()` also frees resources allocated for secure boot and trusted boot.

Verifying an image

If secure mode is on, `sb_verify()` verifies the integrity and authenticity of an image by calling the `ROM_verify()` function from the ROM code via romcode driver. In general terms, this verification will pass only if the following conditions are satisfied. Otherwise the boot process is aborted.

1. Secure boot header is properly built and attached to the image. When `sb_verify()` is called, the ROM code verifies all the secure boot header fields, including the keys, hashes and signatures. The secure boot header and the image are also collectively referred to as secure boot container, or just container. As the secure boot header is the container header and the image is the container payload.
2. The public hardware keys of the container header match with the hw-key-hash read from the device tree. The way that secure boot is designed, this assertion ensures that only images signed by the owner of the hw-key-hash will pass the verification. The hw-key-hash is a hash of three hardware public keys stored in *SEEPROM* at manufacture time and written to the device tree at boot time.

Measuring an image

`tb_measure()` measures an image retrieved from PNOR if trusted mode is on, but only if the provided image is included in the *resource_map* whitelist. This whitelist defines for each expected image to what PCR the measurement must be recorded and extended. `tb_measure()` returns an error if the provided image is not included in the *resource_map* whitelist.

For the sake of simplicity we say that `tb_measure()` measures an image, but calculating the digest of a given image is just one of the steps performed by `tb_measure()`.

Steps performed by `tb_measure()` if trusted mode is on:

1. Measure the provided image for each PCR bank: SHA1 and SHA256. If secure mode is on and the image is a container, parse the container header to get the SHA512 hash of the container payload (*sw-payload-hash* field). Otherwise, call the ROM code via romcode driver to calculate the SHA512 hash of the image at boot time. In both cases, the SHA512 hash is truncated to match the size required by each PCR bank: SHA1 bank PCRs are 20 bytes and SHA256 bank PCRs are 32 bytes.
2. Record a new event in the event log for the mapped PCR. Call the tpmLogMgr API to generate a new event and record it in the event log. The new event is generated for the mapped PCR and it also contains a digest list with both SHA1 and SHA256 measurements obtained in step 1.
3. Extend the measurements into the mapped PCR. Call the TCG Software Stack (TSS) API to extend both measurements obtained in step 1 into the mapped PCR number. The SHA1 measurement is extended to the SHA1 PCR bank and the SHA256 measurement is extended to the SHA256 PCR bank. However, they are extended to the same PCR number on each bank. Since this TSS implementation supports multibank, it does the marshalling of both SHA1 and SHA256 measurements into a single TPM extend command and then it sends the command to the TPM device via TPM device driver.

Both TSS and tpmLogMgr APIs are implemented by hostboot, but their source code are added to skiboot. The TSS and tpmLogMgr interfaces are defined in `libstb/tss/trustedbootCmds.H` and `libstb/tss/tpmLogMgr.H`, respectively.

3.2 Device Tree

General notes on the Device Tree produced by skiboot. This chapter **needs updating**.

3.2.1 General comments

- skiboot does not require nodes to have phandle properties, but if you have them then *all* nodes must have them including the root of the device-tree (currently a HB bug !). It is recommended to have them since they are needed to represent the cache levels.
- **NOTE**: The example tree below only has phandle properties for nodes that are referenced by other nodes. This is *not* correct and is purely done for keeping this document smaller, make sure to follow the rule above.
- Only the "phandle" property is required. Sapphire also generates a "linux,phandle" for backward compatibility but doesn't require it as an input
- Any property not specifically documented must be put in "as is"
- All ibm,chip-id properties contain a HW chip ID which correspond on P8 to the PIR value shifted right by 7 bits, ie. it's a 6-bit value made of a 3-bit node number and a 3-bit chip number.
- Unit addresses (@xxxx part of node names) should if possible use lower case hexadecimal to be consistent with what skiboot does and to help some stupid parsers out there...

3.2.2 Reserve Map

Here are the reserve map entries. They should exactly match the reserved-ranges property of the root node (see documentation of that property)

```
/dts-v1/;
/memreserve/  0x00000007fe600000 0x0000000000100000;
/memreserve/  0x00000007fe200000 0x0000000000100000;
/memreserve/  0x0000000031e00000 0x00000000003e0000;
/memreserve/  0x0000000031000000 0x0000000000e00000;
/memreserve/  0x0000000030400000 0x0000000000c00000;
/memreserve/  0x0000000030000000 0x0000000000400000;
/memreserve/  0x0000000400000000 0x0000000000600450;
```

3.2.3 Root Node

Root node of device tree. There are a few required and a few optional properties that sit in the root node. They're described here.

compatible

The "compatible" properties are string lists indicating the overall compatibility from the more specific to the least specific.

The root node compatible property *must* contain "ibm,powernv" for Linux to have the powernv platform match the machine.

Each distinct platform *MUST* also add a more precise property (first in order) indicating the board type.

The standard naming is "vendor,name". For example: *compatible = "goog,rhesus","ibm,powernv";* would work. Or even better: *compatible = "goog,rhesus-v1","goog,rhesus","ibm,powernv";*.

The bare *ibm,powernv* should be reserved for bringup/testing:

```
/dts-v1/;
/ {
        compatible = "ibm,powernv";
  };
```

Example

```
/dts-v1/;
/ {
        compatible = "ibm,powernv";

        /* mandatory */
        #address-cells = <0x2>;
        #size-cells = <0x2>;

        /* User visible board name (will be shown in /proc/cpuinfo) */
        model = "Machine Name";

        /*
         * The reserved-names and reserve-names properties work hand in hand. The␣
↪first one
         * is a list of strings providing a "name" for each entry in the second one␣
↪using
         * the traditional "vendor,name" format.
         *
```

```
        * The reserved-ranges property contains a list of ranges, each in the form of␣
↪2 cells
        * of address and 2 cells of size (64-bit x2 so each entry is 4 cells)␣
↪indicating
        * regions of memory that are reserved and must not be overwritten by skiboot␣
↪or
        * subsequently by the Linux Kernel.
        *
        * Corresponding entries must also be created in the "reserved map" part of␣
↪the flat
        * device-tree (which is a binary list in the header of the fdt).
        *
        * Unless a component (skiboot or Linux) specifically knows about a region␣
↪(usually
        * based on its name) and decides to change or remove it, all these regions are
        * passed as-is to Linux and to subsequent kernels across kexec and are kept
        * preserved.
        *
        * NOTE: Do *NOT* copy the entries below, they are just an example and are␣
↪actually
        * created by skiboot itself. They represent the SLW image as "detected" by␣
↪reading
        * the PBA BARs and skiboot own memory allocations.
        *
        * I would recommend that you put in there the SLW and OCC (or HOMER as one␣
↪block
        * if that's how you use it) and any additional memory you want to preserve␣
↪such
        * as FW log buffers etc...
        */

       reserved-names = "ibm,slw-image", "ibm,slw-image", "ibm,firmware-stacks", "ibm,
↪firmware-data", "ibm,firmware-heap", "ibm,firmware-code", "memory@400000000";
       reserved-ranges = <0x7 0xfe600000 0x0 0x100000 0x7 0xfe200000 0x0 0x100000 0x0␣
↪0x31e00000 0x0 0x3e0000 0x0 0x31000000 0x0 0xe00000 0x0 0x30400000 0x0 0xc00000 0x0␣
↪0x30000000 0x0 0x400000 0x4 0x0 0x0 0x600450>;

       /* Mandatory */
       cpus {
               #address-cells = <0x1>;
               #size-cells = <0x0>;

               /*
                * The following node must exist for each *core* in the system. The␣
↪unit
                * address (number after the @) is the hexadecimal HW CPU number (PIR␣
↪value)
                * of thread 0 of that core.
                */
               PowerPC,POWER8@20 {
                       /* mandatory/standard properties */
                       device_type = "cpu";
                       64-bit;
                       32-64-bridge;
                       graphics;
                       general-purpose;

                       /*
```

```
                         * The "status" property indicate whether the core is␣
↪functional. It's
                         * a string containing "okay" for a good core or "bad" for a␣
↪non-functional
                         * one. You can also just ommit the non-functional ones from␣
↪the DT
                         */
                        status = "okay";

                        /*
                         * This is the same value as the PIR of thread 0 of that core
                         * (ie same as the @xx part of the node name)
                         */
                        reg = <0x20>;

                        /* same as above */
                        ibm,pir = <0x20>;

                        /* chip ID of this core */
                        ibm,chip-id = <0x0>;

                        /*
                         * interrupt server numbers (aka HW processor numbers) of all␣
↪threads
                         * on that core. This should have 8 numbers and the first one␣
↪should
                         * have the same value as the above ibm,pir and reg properties
                         */
                        ibm,ppc-interrupt-server#s = <0x20 0x21 0x22 0x23 0x24 0x25␣
↪0x26 0x27>;

                        /*
                         * This is the "architected processor version" as defined in␣
↪PAPR. Just
                         * stick to 0x0f000004 for P8 and things will be fine
                         */
                        cpu-version = <0x0f000004>;

                        /*
                         * These are various definitions of the page sizes and segment␣
↪sizes
                         * supported by the MMU, those values are fine for P8 for now
                         */
                        ibm,processor-segment-sizes = <0x1c 0x28 0xffffffff 0xffffffff>
↪;
                        ibm,processor-page-sizes = <0xc 0x10 0x18 0x22>;
                        ibm,segment-page-sizes = <0xc 0x0 0x3 0xc 0x0 0x10 0x7 0x18␣
↪0x38 0x10 0x110 0x2 0x10 0x1 0x18 0x8 0x18 0x100 0x1 0x18 0x0 0x22 0x120 0x1 0x22␣
↪0x3>;

                        /*
                         * Similarly that might need to be reviewed later but will do␣
↪for now...
                         */
                        ibm,pa-features = [0x6 0x0 0xf6 0x3f 0xc7 0x0 0x80 0xc0];

                        /* SLB size, use as-is */
                        ibm,slb-size = <0x20>;
```

```
                        /* VSX support, use as-is */
                        ibm,vmx = <0x2>;

                        /* DFP support, use as-is */
                        ibm,dfp = <0x2>;

                        /* PURR/SPURR support, use as-is */
                        ibm,purr = <0x1>;
                        ibm,spurr = <0x1>;

                        /*
                         * Old-style core clock frequency. Only create this property␣
↪if the frequency fits
                         * in a 32-bit number. Do not create it if it doesn't
                         */
                        clock-frequency = <0xf5552d00>;

                        /*
                         * mandatory: 64-bit version of the core clock frequency,␣
↪always create this
                         * property.
                         */
                        ibm,extended-clock-frequency = <0x0 0xf5552d00>;

                        /* Timebase freq has a fixed value, always use that */
                        timebase-frequency = <0x1e848000>;

                        /* Same */
                        ibm,extended-timebase-frequency = <0x0 0x1e848000>;

                        /* Use as-is, values might need to be adjusted but that will␣
↪do for now */
                        reservation-granule-size = <0x80>;
                        d-tlb-size = <0x800>;
                        i-tlb-size = <0x0>;
                        tlb-size = <0x800>;
                        d-tlb-sets = <0x4>;
                        i-tlb-sets = <0x0>;
                        tlb-sets = <0x4>;
                        d-cache-block-size = <0x80>;
                        i-cache-block-size = <0x80>;
                        d-cache-size = <0x10000>;
                        i-cache-size = <0x8000>;
                        i-cache-sets = <0x4>;
                        d-cache-sets = <0x8>;
                        performance-monitor = <0x0 0x1>;

                        /*
                         * optional: phandle of the node representing the L2 cache for␣
↪this core,
                         * note: it can also be named "next-level-cache", Linux will␣
↪support both
                         * and Sapphire doesn't currently use those properties, just␣
↪passes them
                         * along to Linux
                         */
                        l2-cache = < 0x4 >;
```

```
                };

                /*
                 * Cache nodes. Those are siblings of the processor nodes under /cpus
↪and
                 * represent the various level of caches.
                 *
                 * The unit address (and reg property) is mostly free-for-all as long
↪as
                 * there is no collisions. On HDAT machines we use the following
↪encoding
                 * which I encourage you to also follow to limit surprises:
                 *
                 * L2   :  (0x20 << 24) | PIR (PIR is PIR value of thread 0 of core)
                 * L3   :  (0x30 << 24) | PIR
                 * L3.5 :  (0x35 << 24) | PIR
                 *
                 * In addition, each cache points to the next level cache via its
                 * own "l2-cache" (or "next-level-cache") property, so the core node
                 * points to the L2, the L2 points to the L3 etc...
                 */

                l2-cache@20000020 {
                        phandle = <0x4>;
                        device_type = "cache";
                        reg = <0x20000020>;
                        status = "okay";
                        cache-unified;
                        d-cache-sets = <0x8>;
                        i-cache-sets = <0x8>;
                        d-cache-size = <0x80000>;
                        i-cache-size = <0x80000>;
                        l2-cache = <0x5>;
                };

                l3-cache@30000020 {
                        phandle = <0x5>;
                        device_type = "cache";
                        reg = <0x30000020>;
                        status = "bad";
                        cache-unified;
                        d-cache-sets = <0x8>;
                        i-cache-sets = <0x8>;
                        d-cache-size = <0x800000>;
                        i-cache-size = <0x800000>;
                };

        };

        /*
         * Interrupt presentation controller (ICP) nodes
         *
         * There is some flexibility as to how many of these are presents since
         * a given node can represent multiple ICPs. When generating from HDAT we
         * chose to create one per core
         */
        interrupt-controller@3ffff80020000 {
                /* Mandatory */
```

```
                compatible = "IBM,ppc-xicp", "IBM,power8-icp";
                interrupt-controller;
                #address-cells = <0x0>;
                device_type = "PowerPC-External-Interrupt-Presentation";

                /*
                 * Range of HW CPU IDs represented by that node. In this example
                 * the core starting at PIR 0x20 and 8 threads, which corresponds
                 * to the CPU node of the example above. The property in theory
                 * supports multiple ranges but Linux doesn't.
                 */
                ibm,interrupt-server-ranges = <0x20 0x8>;

                /*
                 * For each server in the above range, the physical address of the
                 * ICP register block and its size. Since the root node #address-cells
                 * and #size-cells properties are both "2", each entry is thus
                 * 2 cells address and 2 cells size (64-bit each).
                 */
                reg = <0x3ffff 0x80020000 0x0 0x1000 0x3ffff 0x80021000 0x0 0x1000␣
↪0x3ffff 0x80022000 0x0 0x1000 0x3ffff 0x80023000 0x0 0x1000 0x3ffff 0x80024000 0x0␣
↪0x1000 0x3ffff 0x80025000 0x0 0x1000 0x3ffff 0x80026000 0x0 0x1000 0x3ffff␣
↪0x80027000 0x0 0x1000>;
        };

        /*
         * The "memory" nodes represent physical memory in the system. They
         * do not represent DIMMs, memory controllers or Centaurs, thus will
         * be expressed separately.
         *
         * In order to be able to handle affinity properly, we require that
         * a memory node is created for each range of memory that has a different
         * "affinity", which in practice means for each chip since we don't
         * support memory interleaved across multiple chips on P8.
         *
         * Additionally, it is *not* required that one chip = one memory node,
         * it is perfectly acceptable to break down the memory of one chip into
         * multiple memory nodes (typically skiboot does that if the two MCs
         * are not interlaved).
         */
        memory@0 {
                device_type = "memory";

                /*
                 * We support multiple entries in the ibm,chip-id property for
                 * memory nodes in case the memory is interleaved across multiple
                 * chips but that shouldn't happen on P8
                 */
                ibm,chip-id = <0x0>;

                /* The "reg" property is 4 cells, as usual for a child of
                 * the root node, 2 cells of address and 2 cells of size
                 */
                reg = <0x0 0x0 0x4 0x0>;
        };

        /*
         * The XSCOM node. This is the closest thing to a "chip" node we have.
```

```
         * there must be one per chip in the system (thus a DCM has two) and
         * while it represents the "parent" of various devices on the PIB/PCB
         * that we want to expose, it is also used to store all sort of
         * miscellaneous per-chip information on HDAT based systems (such
         * as VPDs).
         */
        xscom@3fc0000000000 {
                /* standard & mandatory */
                #address-cells = <0x1>;
                #size-cells = <0x1>;
                scom-controller;
                compatible = "ibm,xscom", "ibm,power8-xscom";

                /* The chip ID as usual ... */
                ibm,chip-id = <0x0>;

                /* The base address of xscom for that chip */
                reg = <0x3fc00 0x0 0x8 0x0>;

                /*
                 * This comes from HDAT and I *think* is the raw content of the
                 * module VPD eeprom (and thus doesn't have a standard ASCII keyword
                 * VPD format). We don't currently use it though ...
                 */
                ibm,module-vpd = < /* ... big pile of binary data ... */ >;

                /* PSI host bridge XSCOM register set */
                psihb@2010900 {
                        reg = <0x2010900 0x20>;
                        compatible = "ibm,power8-psihb-x", "ibm,psihb-x";
                };

                /* Chip TOD XSCOM register set */
                chiptod@40000 {
                        reg = <0x40000 0x34>;
                        compatible = "ibm,power-chiptod", "ibm,power8-chiptod";

                        /*
                         * Create that property with no value if this chip has
                         * the Primary TOD in the topology. If it has the secondary
                         * one (backup master ?) use "secondary".
                         */
                        primary;
                };

                /* NX XSCOM register set */
                nx@2010000 {
                        reg = <0x2010000 0x4000>;
                        compatible = "ibm,power-nx", "ibm,power8-nx";
                };

                /*
                 * PCI "PE Master" XSCOM register set for each active PHB
                 *
                 * For now, do *not* create these if the PHB isn't connected,
                 * clocked, or the PHY/HSS not configured.
                 */
                pbcq@2012000 {
```

```
                reg = <0x2012000 0x20 0x9012000 0x5 0x9013c00 0x15>;
                compatible = "ibm,power8-pbcq";

                /* Indicate the PHB index on the chip, ie, 0,1 or 2 */
                ibm,phb-index = <0x0>;

                /* Create that property to use the IBM-style "A/B" dual input
                 * slot presence detect mechanism.
                 */
                ibm,use-ab-detect;

                /*
                 * TBD: Lane equalization values. Not currently used by
                 * skiboot but will have to be sorted out
                 */
                ibm,lane_eq = <0x0>;
        };

        pbcq@2012400 {
                reg = <0x2012400 0x20 0x9012400 0x5 0x9013c40 0x15>;
                compatible = "ibm,power8-pbcq";
                ibm,phb-index = <0x1>;
                ibm,use-ab-detect;
                ibm,lane_eq = <0x0>;
        };

        /*
         * Here's the LPC bus. Ideally each chip has one but in
         * practice it's ok to only populate the ones actually
         * used for something. This is not an exact representation
         * of HW, in that case we would have eccb -> opb -> lpc,
         * but instead we just have an lpc node and the address is
         * the base of the ECCB register set for it
         *
         * Devices on the LPC are represented as children nodes,
         * see example below for a standard UART.
         */
        lpc@b0020 {
                /*
                 * Empty property indicating this is the primary
                 * LPC bus. It will be used for the default UART
                 * if any and this is the bus that will be used
                 * by Linux as the virtual 64k of IO ports
                 */
                primary;

                /*
                 * 2 cells of address, the first one indicates the
                 * address type, see below
                 */
                #address-cells = <0x2>;
                #size-cells = <0x1>;
                reg = <0xb0020 0x4>;
                compatible = "ibm,power8-lpc";

                /*
                 * Example device: a UART on IO ports.
                 *
```

```
                        * LPC address have 2 cells. The first cell is the
                        * address type as follow:
                        *
                        *   0 : LPC memory space
                        *   1 : LPC IO space
                        *   2:  LPC FW space
                        *
                        * (This corresponds to the OPAL_LPC_* arguments
                        * passed to the opal_lpc_read/write functions)
                        *
                        * The unit address follows the old ISA convention
                        * for open firmware which prefixes IO ports with "i".
                        *
                        * (This is not critical and can be 1,3f8 if that's
                        * problematic to generate)
                        */
                       serial@i3f8 {
                               reg = <0x1 0x3f8 8>;
                               compatible = "ns16550", "pnpPNP,501";

                               /* Baud rate generator base frequency */
                               clock-frequency = < 1843200 >;

                               /* Default speed to use */
                               current-speed = < 115200 >;

                               /* Historical, helps Linux */
                               device_type = "serial";

                               /*
                                * Indicate which chip ID the interrupt
                                * is routed to (we assume it will always
                                * be the "host error interrupt" (aka
                                * "TPM interrupt" of that chip).
                                */
                                ibm,irq-chip-id = <0x0>;
                       };
               };
       };
};
```

3.3 Device Tree

Device Tree for OPAL. Please refer to Device Tree Spec.

3.3.1 ibm,firmware-versions node

The *ibm,firmware-versions* node contains information on the versions of various firmware components as they were **during boot**. It **does not** change if there are pending or runtime updates. It represents (to the best of boot firmware's ability) what versions of firmware were during this boot.

Property	Required	Value
version	POWER9	See below
skiboot	N	component version number
occ	N	component version number
buildroot	N	component version number
capp-ucode	N	component version number
petitboot	N	component version number
open-power	N	component version number
hostboot-binaries	N	component version number
MACHINE-xml	N	MACHINE (e.g. habanero) machine XML version
hostboot	N	component version number
linux	N	component version number

`version` property

This property **must** exist on POWER9 and above systems. It **may** exist on POWER8 systems.

If this property exists, it **must** conform to this specification. It's a single version number of the firmware image. In the event of a system supporting multiple firmware sides, this represents the **default** boot side. That is, the version that is applicable when determining if a machine requires a firmware update.

Examples (for three different platforms):

- `IBM-sandwich-20170217`
- `open-power-habanero-v1.14-45-g78d89280c3f9-dirty`
- `open-power-SUPERMICRO-P8DTU-V2.00.GA2-20161028`

To compare two versions (for the purpose of determining if the current installed firmware is in need of updating to the one being compared against) we need a defined set of rules on how to do this comparison.

Version numbers are **not** intended to be compared across platforms.

The version string may include a description at the start of it. This description can contain any set of characters but **must not** contain a '-' followed by a digit. It also **must not** contain '-v' or '-V' followed by a digit.

Each part of the version string is separated by a '-' character. Leading sections are ignored, until one starts with a digit (0-9) or a 'v' or 'V', followed by a digit. Where there is a leading 'v' or 'V', it is also stripped.

For the above three examples, we'd be left with:

- `20170217`
- `1.14-45-g78d89280c3f9-dirty`
- `2.00.GA2-20161028`

Each section is now compared until a difference is found. All comparisons are done *lexically*. The lexical comparison sorts in this order: tilde (~), all letters, non-letters. The tilde is special and sorts before an end of part. This allows the common usage of designating pre-release builds by a tailing section beginning with a '~'.

For example: "1.0~20170217", "1.0~rc4" and "1.0~beta1" all sort **before** "1.0"

Note that "1.0beta" sorts **after** "1.0"

The start of the version string contains an optional *epoch*. If not present, it is zero. This allows a reset of versioning schemes. All versions with an epoch of N+1 are greater than those with epoch N, no matter what the version strings would compare. For example "0:4.0" is **less** than "1:1.0". Increasing the epoch should **not** be a regular occurance.

For the remainder of the version strings, each part (separated by '.' or '-') is compared lexically. There are two exceptions: any part beginning with "-g" or "-p" followed by a hexadecimal string is compared as a string, and if they are different the versions are determined to be different. For example, the sections "-g78d89280c3f9" and "-g123456789abc" differ and for all comparisons (less than, greater than, equal) the result should be true.

For those who have been paying attention, this scheme should look very familiar to those who are familiar with RPM and Debian package versioning.

The below table shows comparisons between versions and what the result should be:

A	B	Result
1.14-45-g78d89280c3f9-dirty	1.14-45-g78d89280c3f9-dirty	Equal
1.14-45-g78d89280c3f9-dirty	1.14-45-g78d89280c3f9	A > B
1.14-45-g78d89280c3f9-dirty	1.14-45-g123456789abc	A < B, A > B, A != B
1.14-45-g78d89280c3f9-dirty	1.14-46	A < B
1.14-45-g78d89280c3f9-dirty	1.15	A < B
1.14-45-g78d89280c3f9-dirty	1:1.0	A < B
1.0	1.0~daily20170201	A > B
1.0.1	1.0~daily20170201	A > B
1.0	1.0.1	A < B
1.0	1.0beta	A < B

Examples

New style (required for POWER9 and above):

```
ibm,firmware-versions {
              version = "open-power-habanero-v1.14-45-g78d89280c3f9-dirty";
              skiboot = "5.4.0";
              occ = "d7efe30";
              linux = "4.4.32-openpower1";
};
```

Old-style:

```
ibm,firmware-versions {
         occ = "d7efe30-opdirty";
         skiboot = "5.4.0-opdirty";
         buildroot = "211bd05";
         capp-ucode = "1bb7503-opdirty";
         petitboot = "v1.3.1-opdirty-d695626";
         open-power = "habanero-f7b8f65-dirty";
         phandle = <0x1000012e>;
         hostboot-binaries = "56532f5-opdirty";
         habanero-xml = "6a78496-opdirty-526ff79";
         hostboot = "09cfacb-opdirty";
         linux = "4.4.32-openpower1-opdirty-85cf528";
};
```

3.3.2 ibm,opal

ibm,opal/diagnostics device tree entries

The diagnostics node under ibm,opal describes a userspace-to-firmware interface, supporting the runtime processor recovery diagnostics functions.

Note: Some systemd init scripts look for the presence of the path `/ibm,opal/diagnostics` in order to run the opal-prd daemon.

The properties of a prd node are:

```
/ {
   ibm,opal {
     diagnostics {
       compatible = "ibm,opal-prd";
   };
  };
};
```

System Firmware

The 'firmware' node under 'ibm,opal' lists system and OPAL firmware version.

```
firmware {
      symbol-map = <0x0 0x300ac650 0x0 0x1b3f5>;
      compatible = "ibm,opal-firmware";
      ml-version = [4d 4c 20 46 57 37 37 30 2e 32 30 20 46 57 37 37 30 2e 32 30 20 46␣
↪57 37 37 30 2e 32 30];
      mi-version = <0x4d49205a 0x4c373730 0x5f303735 0x205a4c37 0x37305f30 0x3735205a␣
↪0x4c373730 0x5f303735>;
      version = "skiboot-5.0-rc2";
      phandle = <0x8e>;
      linux,phandle = <0x8e>;
};
```

compatible property describes OPAL compatibility.

symbol-map property describes OPAL symbol start address and size.

version property describes OPAL version. Replaces 'git-id', so may not be present. On POWER9 and above, it is always present.

mi-version property describes Microcode Image. Only on IBM FSP systems. Will (likely) not be present on POWER9 systems.

ml-version property describes Microcode Level. Only on IBM FSP systems. Will (likely) not be present on POWER9 systems.

MI/ML format

```
<ML/MI> <T side version> <P side version> <boot side version>
```

ibm,opal/flash device tree entries

The flash@<n> nodes under ibm,opal describe flash devices that can be accessed through the OPAL_FLASH_{READ,ERASE,WRITE} interface.

These interfaces take an 'id' parameter, which corresponds to the ibm,opal-id property of the node.

The properties under a flash node are:

- `compatible = "ibm,opal-flash"`

ibm,opal-id = <id> provides the index used for the OPAL_FLASH_XXX calls to reference this flash device

reg = <0 size> the offset and size of the flash device

ibm,flash-block-size the read/write/erase block size for the flash interface. Calls to read/write/erase must be aligned to the block size.

#address-cells = <1>, #size-cells = <1> flash devices are currently 32-bit addressable

If valid partitions are found on the flash device, then `partition@<offset>` sub-nodes are added to the flash node. These match the Linux binding for flash partitions; the reg parameter contains the offset and size of the partition.

Example:

```
flash@0 {
  reg = <0x0 0x4000000>;
  compatible = "ibm,opal-flash";
  ibm,opal-id = <0x0>;
  ibm,flash-block-size = <0x1000>;
  #address-cells = <0x1>;
  phandle = <0x100002bf>;
  #size-cells = <0x1>;
};
```

Service Indicators (LEDS)

The 'leds' node under 'ibm,opal' lists service indicators available in the system and their capabilities.

```
leds {
      compatible = "ibm,opal-v3-led";
      phandle = <0x1000006b>;
      linux,phandle = <0x1000006b>;
      led-mode = "lightpath";

      U78C9.001.RST0027-P1-C1 {
              led-types = "identify", "fault";
              phandle = <0x1000006f>;
              linux,phandle = <0x1000006f>;
      };
      /* Other LED nodes like the above one */
};
```

compatible property describes LEDs compatibility.

led-mode property describes service indicator mode (lightpath/guidinglight).

Each node under 'leds' node describes location code of FRU/Enclosure.

The properties under each node:

led-types Supported indicators (attention/identify/fault).

These LEDs can be accessed through OPAL_LEDS_{GET/SET}_INDICATOR interfaces. Refer to *Service Indicators (LEDS)* for interface details.

Operator Panel (oppanel)

```
oppanel {
      compatible = "ibm,opal-oppanel";
      #lines = <0x2>;
      #length = <0x10>;
};
```

The Operator Panel is a device for displaying small amounts of textual data to an administrator. On IBM POWER8 systems with an FSP, this is a small 16x2 LCD panel that can be viewed either from the Web UI of the FSP (known as ASM) or by physically going to the machine and looking at the panel.

The operator panel does not have to be present.

If it is, there are OPAL calls to read and write to it.

The device tree entry is so that the host OS knows the size of the panel and can pass buffers of the appropriate size to the OPAL calls.

ibm,opal/power-mgt device tree entries

ibm,opal/power-mgt/occ device tree entries

This node exports the per-chip pstate table properties to kernel.

Example:

```
occ@7fddf8000 {
       ibm,pstate-vdds = [45 45 46 46 46 47 48 49 4a 4b 4c 4d 4f 50 51 52 53 54 55 57␣
↪58 59 5a 5b 5c 5d 5e 5f 5f 60 61 62 63 64 65 65 66 67 68 69 6a 6a 6b 6c 6d 6e 6f␣
↪70 70 71];
       ibm,chip-id = <0x1>;
       phandle = <0x100003b8>;
       ibm,pstate-vcss = [3b 3d 3f 41 42 44 45 46 47 48 49 4a 4b 4c 4d 4e 4f 50 50␣
↪51 52 53 54 55 56 56 57 57 58 58 59 59 5a 5a 5b 5b 5c 5c 5d 5d 5e 5e 5f 5f 60 60␣
↪61 61 62 62];
       reg = <0x7f 0xfddf8000 0xb98>;
};
```

ibm,chip-id

This property denotes the ID of chip to which OCC belongs to.

reg

This tuple gives the statring address of the OPAL data in HOMER and the size of the OPAL data.

The top-level /ibm,opal/power-mgt contains :

```
#size-cells = <1>
#address-cells = <2>
```

ibm,pstate-vcss ibm,pstate-vdds

These properties list a voltage-identifier of each of the pstates listed in ibm,pstate-ids for the Vcs and Vdd values used for that pstate in that chip. Each VID is a single byte.

power-mgt/powercap

The powercap sensors are populated in this node. Each child node in the "powercap" node represents a power-cappable component.

For example :

```
system-powercap/
```

The OPAL_GET_POWERCAP and OPAL_SET_POWERCAP calls take a handle for what powercap property to get/set which is defined in the child node.

The compatible property for the linux driver which will be "ibm,opal-powercap"

Each child node has below properties:

powercap-current Handle to indicate the current powercap

powercap-min Minimum possible powercap

powercap-max Maximum possible powercap

Powercap handle uses the following encoding:

```
| Class |    Reserved    | Attribute |
|-------|----------------|-----------|
```

Note: The format of the powercap handle is `NOT` ABI and may change in the future.

```
power-mgt {
  powercap {
     compatible = "ibm,opal-powercap";

     system-powercap {
             name = "system-powercap";
             powercap-current = <0x00000002>;
             powercap-min = <0x00000000>;
             powercap-max = <0x00000001>;
     };
  };
 };
```

power-mgt/psr

Some systems allow modification of how power consumption throttling is balanced between entities in a system. A typical one may be how the power management complex should balance throttling CPU versus the GPU. An OPAL call can be used to set these ratios, which are described in the device tree.

In the future, there may be more available settings than just CPU versus GPU.

Each child node in the "psr" node represents a configurable psr sensor.

For example [::] cpu-to-gpu@1

The compatible property is set to "ibm,opal-power-shift-ratio".

Each child node has below properties:

handle Handle to indicate the type of psr

label Name of the psr sensor

The format of the handle is internal, and `not` ABI, although currently it uses the following encoding

```
| Class |Reserved|  RID | Type |
|-------|--------|------|------|
```

```
power-mgt {
  psr {
     compatible = "ibm,opal-power-shift-ratio";

     cpu-to-gpu@0 {
             name = "cpu-to-gpu";
             handle = <0x00000000>;
             label = "cpu_to_gpu_0";
     };

     cpu-to-gpu@1 {
             name = "cpu-to-gpu";
             handle = <0x00000100>;
             label = "cpu_to_gpu_1";
     };
  };
 };
```

All available CPU idle states are listed in ibm,cpu-idle-state-names

For example:

```
power-mgt {
  ibm,cpu-idle-state-names = "nap", "fastsleep_", "winkle";
  ibm,cpu-idle-state-residency-ns = <0x1 0x2 0x3>;
  ibm,cpu-idle-state-latencies-ns = <0x1 0x2 0x3>;
};
```

The idle states are characterized by latency and residency numbers which determine the breakeven point for entry into them. The latency is a measure of the exit overhead from the idle state and residency is the minimum amount of time that a CPU must be predicted to be idle so as to reap the powersavings from entering into that idle state.

These numbers are made use of by the cpuidle governors in the kernel to arrive at the appropriate idle state that a CPU must enter into when there is no work to be done. The values in ibm,cpu-idle-state-latencies-ns are the the measured latency numbers for the idle states. The residency numbers have been arrived at experimentally after ensuring that the performance of latency sensitive workloads do not regress while allowing deeper idle states to be entered into during low load situations. The kernel is expected to use these values for optimal power efficiency.

Example:

```
/ {
  ibm,opal {
    power-mgt {
             ibm,pstate-frequencies-mhz = <0xda3 0xd82 0xd60 0xd3f 0xd1e 0xcfd 0xcdb␣
↪0xcba 0xc99 0xc78 0xc56 0xc35 0xc14 0xbf3 0xbd1 0xbb0 0xb8f 0xb6e 0xb4c 0xb2b 0xb0a␣
↪0xae9 0xac7 0xaa6 0xa85 0xa64 0xa42 0xa21 0xa00 0x9df 0x9bd 0x99c 0x97b 0x95a 0x938␣
↪0x917 0x8f6 0x8d5 0x8b3 0x892 0x871 0x850 0x82e 0x80d>;
```

```
            ibm,cpu-idle-state-latencies-ns = <0xfa0 0x9c40 0x989680>;
            ibm,cpu-idle-state-flags = <0x11000 0x81003 0x47003>;
            ibm,cpu-idle-state-names = "nap", "fastsleep_", "winkle";
            ibm,cpu-idle-state-pmicr = <0x0 0x0 0x20 0x0 0x0 0x0>;
            ibm,pstate-nominal = <0xffffffef>;
            ibm,cpu-idle-state-residency-ns = <0x186a0 0x11e1a300 0x3b9aca00>;
            ibm,cpu-idle-state-pmicr-mask = <0x0 0x0 0x30 0x0 0x0 0x0>;
            phandle = <0x100002a0>;
            ibm,pstate-ids = <0x0 0xffffffff 0xfffffffe 0xfffffffd 0xfffffffc␣
↪0xfffffffb 0xfffffffa 0xfffffff9 0xfffffff8 0xfffffff7 0xfffffff6 0xfffffff5␣
↪0xfffffff4 0xfffffff3 0xfffffff2 0xfffffff1 0xfffffff0 0xffffffef 0xffffffee␣
↪0xffffffed 0xffffffec 0xffffffeb 0xffffffea 0xffffffe9 0xffffffe8 0xffffffe7␣
↪0xffffffe6 0xffffffe5 0xffffffe4 0xffffffe3 0xffffffe2 0xffffffe1 0xffffffe0␣
↪0xffffffdf 0xffffffde 0xffffffdd 0xffffffdc 0xffffffdb 0xffffffda 0xffffffd9␣
↪0xffffffd8 0xffffffd7 0xffffffd6 0xffffffd5>;
            ibm,pstate-max = <0x0>;
            ibm,pstate-min = <0xffffffd5>;
    };
  };
};
```

ibm,cpu-idle-state-pmicr ibm,cpu-idle-state-pmicr-mask

In POWER8, idle states sleep and winkle have 2 modes- fast and deep. In fast mode, idle state puts the core into threshold voltage whereas deep mode completely turns off the core. Choosing fast vs deep mode for an idle state can be done either via PM_GP1 scom or by writing to PMICR special register. If using the PMICR path to choose fast/deep mode then ibm,cpu-idle-state-pmicr and ibm,cpu-idle-state-pmicr-mask properties expose relevant PMICR bits and values for corresponding idle states.

ibm,cpu-idle-state-psscr ibm,cpu-idle-state-psscr-mask

In POWER ISA v3, there is a common instruction 'stop' to enter any idle state and SPR PSSCR is used to specify which idle state needs to be entered upon executing stop instruction. Properties ibm,cpu-idle-state-psscr and ibm,cpu-idle-state-psscr-mask expose the relevant PSSCR bits and values for corresponding idle states.

ibm,cpu-idle-state-flags

These flags are used to describe the characteristics of the idle states like the kind of core state loss caused. These flags are used by the kernel to save/restore appropriate context while using the idle states.

ibm,pstate-ids

This property lists the available pstate identifiers, as signed 32-bit big-endian values. While the identifiers are somewhat arbitrary, these define the order of the pstates in other ibm,pstate-* properties.

ibm,pstate-frequencies-mhz

This property lists the frequency, in MHz, of each of the pstates listed in the ibm,pstate-ids file. Each frequency is a 32-bit big-endian word.

ibm,pstate-max ibm,pstate-min ibm,pstate-nominal

These properties give the maximum, minimum and nominal pstate values, as an id specified in the ibm,pstate-ids file.

ibm,pstate-ultra-turbo ibm,pstate-turbo

These properties are added when ultra-turbo(WOF) is enabled. These properties give the max turbo and max ultra-turbo pstate.

Example:

```
power-mgt {
      ibm,pstate-core-max = <0x0 0x0 0x0 0x0 0x0 0x0 0x0>;
      ibm,pstate-turbo = <0xffffffb>
      ibm,pstate-ultra-turbo = <0x0>;
};
```

ibm,pstate-core-max

This property is added when ultra_turbo(WOF) is enabled. This property gives the list of max pstate for each 'n' number of active cores in the chip.

ibm,opal/sensor-groups

This node contains all sensor groups defined in the system. Each child node here represents a sensor group.

For example [::] occ-csm@1c00020/

The compatible property is set to "ibm,opal-sensor-group"

Each child node has below properties:

type string to indicate the sensor group

sensor-group-id Uniquely identifies a sensor group.

ibm,chip-id This property is added if the sensor group is chip specific

sensors Phandles of all sensors belonging to this sensor group

ops Array of opal call numbers to indicate the available sensor group operations

```
ibm,opal {
  sensor-groups {
     compatible = "ibm,opal-sensor-group";

     occ-csm@1c00020 {
             name = "occ-csm";
             type = "csm";
             sensor-group-id = <0x01c00020>;
             ibm,chip-id = <0x00000008>;
             ops = <0x9c>;
             sensors = <0x00000175 0x00000176 0x00000177 0x00000178 0x00000179␣
↪0x0000017a 0x0000017b 0x0000017c>;
     };
  };
 };
```

ibm,opal/sensors/ device tree nodes

All sensors of a POWER8 system are made available to the OS in the ibm,opal/sensors/ directory. Each sensor is identified with a node which name follows this pattern :

```
<resource class name>@<resource identifier>/
```

For example :

```
core-temp@20/
```

Each node has a minimum set of properties describing the sensor :

- a "compatible" property which should be "ibm,opal-sensor"
- a "sensor-type" property, which can be "temp", "fan", "power". More will be added when new resources are supported. This type is used "as is" by the Linux driver to map sensors in the sysfs interface of the hwmon framework of Linux.
- a "sensor-data" property giving a unique handler for the OPAL_SENSOR_READ call to be used by Linux to get the value of a sensor attribute. This value is opaque to the OS but is *currently* constructed using the following encoding :

  ```
  |  Attr. |Fam|Res. |   Resource     |
  | Number |ily|Class|      Id        |
  |--------|---|-----|----------------|
  ```

  The sensor family (FSP, DTS, etc) is used to dispatch the call to the appriopriate skiboot component.
- a "sensor-status" property giving the state of the sensor. The status bits have the slightly meanings depending on the resource type but testing against 0x6 should raise an alarm.
- an optional "label" property

Each node can have some extra properties depending on the resource they represent. See the tree below for more information.

```
ibm,opal {
  sensors {

      /*
       * Core temperatures (DTS) nodes.
       *
       * We use the PIR of the core as a resource identifier.
       */
      core-temp@20 {
              compatible = "ibm,opal-sensor";
              name = "core-temp";
              sensor-type = "temp";

              /* Status bits :
               *
               * 0x0003       FATAL
               * 0x0002       CRITICAL
               * 0x0001       WARNING
               */
              sensor-data = <0x00800020>;

              /*
               * These are extra properties to help Linux output.
```

```
                 */
                ibm,pir = <0x20>;
                label = "Core";
        };

        /*
         * Centaur temperatures (DTS) nodes. Open Power only.
         *
         * We use the PIR of the core as a resource identifier.
         */
        mem-temp@1 {
                compatible = "ibm,opal-sensor";
                name = "mem-temp";
                sensor-type = "temp";

                /* Status bits :
                 *
                 * 0x0003       FATAL
                 * 0x0002       CRITICAL
                 * 0x0001       WARNING
                 */
                sensor-data = <0x00810001>;

                /*
                 * These are extra properties to help Linux output.
                 */
                ibm,chip-id = <0x80000001>;
                label = "Centaur";
        };
    };
};
```

Top level ibm,opal node

```
ibm,opal {
              #address-cells = <0x0>;
              #size-cells = <0x0>;
              compatible = "ibm,opal-v2", "ibm,opal-v3";

/* v2 is maintained for possible compatibility with very, very old kernels
 * it will go away at some point in the future. Detect and rely on ibm,opal-v3
 * ibm,opal-v2 is *NOT* present on POWER9 and above.
 */

              ibm,associativity-reference-points = <0x4 0x3, 0x2>;
              ibm,heartbeat-ms = <0x7d0>;

/* how often any OPAL call needs to be made to avoid a watchdog timer on BMC
 * from kicking in
 */

              ibm,opal-memcons = <0x0 0x3007a000>;

/* location of in memory OPAL console buffer. */

              ibm,opal-trace-mask = <0x0 0x3008c3f0>;
```

```
            ibm,opal-traces = <0x0 0x3007b010 0x0 0x10077 0x0 0x3b001010 0x0␣
↪0x1000a7 0x0 0x3b103010 0x0 0x1000a7 0x0 0x3b205010 0x0 0x1000a7 0x0 0x3b307010 0x0␣
↪0x1000a7 0x0 0x3b409010 0x0 0x1000a7 0x10 0x1801010 0x0 0x1000a7 0x10 0x1903010 0x0␣
↪0x1000a7 0x10 0x1a05010 0x0 0x1000a7 0x10 0x1b07010 0x0 0x1000a7 0x10 0x1c09010 0x0␣
↪0x1000a7 0x10 0x1d0b010 0x0 0x1000a7 0x10 0x1e0d010 0x0 0x1000a7 0x10 0x1f0f010 0x0␣
↪0x1000a7 0x10 0x2011010 0x0 0x1000a7 0x10 0x2113010 0x0 0x1000a7 0x10 0x2215010 0x0␣
↪0x1000a7 0x10 0x2317010 0x0 0x1000a7 0x10 0x2419010 0x0 0x1000a7 0x10 0x251b010 0x0␣
↪0x1000a7 0x10 0x261d010 0x0 0x1000a7>;

/* see docs on tracing */

            linux,phandle = <0x10000003>;
            opal-base-address = <0x0 0x30000000>;
            opal-entry-address = <0x0 0x300050c0>;
            opal-interrupts = <0x10 0x11 0x12 0x13 0x14 0x20010 0x20011 0x20012␣
↪0x20013 0x20014 0xffe 0xfff 0x17fe 0x17ff 0x2ffe 0x2fff 0x37fe 0x37ff 0x20ffe␣
↪0x20fff 0x22ffe 0x22fff 0x237fe 0x237ff>;
            opal-msg-async-num = <0x8>;
            opal-msg-size = <0x48>;
            opal-runtime-size = <0x0 0x9a00000>;
            phandle = <0x10000003>;
 };
```

3.3.3 ibm,secureboot

Secure boot and trusted boot relies on a code stored in the secure ROM at manufacture time to verify and measure other codes before they are executed. This ROM code is also referred to as ROM verification code.

On POWER8, the presence of the ROM code is announced to skiboot (by Hostboot) by the `ibm,secureboot` device tree node.

If the system is booting up in secure mode, the ROM code is called for secure boot to verify the integrity and authenticity of an image before it is executed.

If the system is booting up in trusted mode, the ROM code is called for trusted boot to calculate the SHA512 hash of an image only if the image is not a secure boot container or the system is not booting up in secure mode.

For further information about secure boot and trusted boot please refer to *Secure and Trusted Boot Overview*.

Required properties

```
compatible:         ibm,secureboot version. It is related to the ROM code version.

hash-algo:          hash algorithm used for the hw-key-hash. Aspects such as the size
                    of the hw-key-hash can be infered from this property.

secure-enabled:     this property exists if the system is booting in secure mode.

trusted-enabled:    this property exists if the system is booting in trusted mode.

hw-key-hash:        hash of three concatenated hardware public key. This is required
                    by the ROM code to verify images.
```

Example

For the first version `ibm,secureboot-v1`, the ROM code expects the *hw-key-hash* to be a SHA512 hash.

```
ibm,secureboot {
    compatible = "ibm,secureboot-v1";
    hash-algo = "sha512";
    secure-enabled;
    trusted-enabled;
    hw-key-hash = <0x40d487ff 0x7380ed6a 0xd54775d5 0x795fea0d 0xe2f541fe
                   0xa9db06b8 0x466a42a3 0x20e65f75 0xb4866546 0x17d907
                   0x515dc2a5 0xf9fc5095 0x4d6ee0c9 0xb67d219d 0xfb708535
                   0x1d01d6d1>;
    phandle = <0x100000fd>;
    linux,phandle = <0x100000fd>;
};
```

3.3.4 IMC Device Tree Bindings

See *OPAL/Skiboot In-Memory Collection (IMC) interface Documentation* for general In-Memory Collection (IMC) counter information.

imc-counters top-level node

```
imc-counters {
  compatible = "ibm,opal-in-memory-counters";
  #address-cells = <0x1>;
  #size-cells = <0x1>;
  phandle = <0x1000023a>;
  version-id = <0xd>;
  /* Denote IMC Events Catalog version used to build this DTS file. */

};
```

IMC device/units bindings

```
mcs3 {
        compatible = "ibm,imc-counters";
        events-prefix = "PM_MCS3_"; /* denotes event name to be prefixed to get␣
↪complete event name supported by this device */

        phandle = <0x10000241>;
        events = <0x10000242>; /* phandle of the events node supported by this device␣
↪*/

        unit = "MiB";
        scale = "4"; /* unit and scale for all the events for this device */

        reg = <0x118 0x8>; /* denotes base address for device event updates */
        type = <0x10>;
        size = 0x40000;
        offset = 0x180000;
        base_addr = <Base address of the counter in reserve memory>
```

```
        /* This is per-chip memory field and OPAL files it based on the no of chip in␣
↪the system */
        /* base_addr property also indicates (or hints) kernel whether to memory */
        /* should be mmapped or allocated at system start for the counters */
        chipids = <chip-id for the base_addr >
};
```

IMC device event bindings

```
nest-mcs-events {
        #address-cells = <0x1>;
        #size-cells = <0x1>;
        phandle = <0x10000242>;

        event@98 {
              desc = "Total Write Bandwidth seen on both MCS"; /* event description */

              phandle = <0x1000023d>;
              reg = <0x98 0x8>; /* event offset,when added with (nest-offset-address␣
↪+ device reg) will point to actual counter memory */

              event-name = "DOWN_128B_DATA_XFER"; /* denotes the actual event name */

        };

        /* List of events supported */

};
```

3.3.5 Nvlink Device Tree Bindings

See *OPAL/Skiboot Nvlink Interface Documentation* for general Nvlink information.

NPU bindings:

```
xscom@3fc0000000000 {
  npu@8013c00 {
    reg = <0x8013c00 0x2c>;
    compatible = "ibm,power8-npu";
    ibm,npu-index = <0x0>;
    ibm,npu-links = <0x4>; /* Number of links wired up to this npu. */

    phandle = <0x100002bc>;
    linux,phandle = <0x100002bc>;

    link@0 {
      ibm,npu-pbcq = <0x1000000b>; /* phandle to the pbcq which connects to the GPU.␣
↪*/
      ibm,npu-phy = <0x80000000 0x8010c3f>; /* SCOM address of the IBM PHY␣
↪controlling this link. */
      compatible = "ibm,npu-link";
      ibm,npu-lane-mask = <0xff>; /* Mask specifying which IBM PHY lanes are used for␣
↪this link. */

      phandle = <0x100002bd>;
```

```
      ibm,npu-link-index = <0x0>; /* Hardware link index. Naples systems
                                   * contain links at index 0,1,4 & 5.
                                   * Used to calculate various address offsets. */

      linux,phandle = <0x100002bd>;
    };

    link@1 {
      ibm,npu-pbcq = <0x1000000b>;
      ibm,npu-phy = <0x80000000 0x8010c3f>;
      compatible = "ibm,npu-link";
      ibm,npu-lane-mask = <0xff00>;
      phandle = <0x100002be>;
      ibm,npu-link-index = <0x1>;
      linux,phandle = <0x100002be>;
    };

    link@4 {
      ibm,npu-pbcq = <0x1000000a>;
      ibm,npu-phy = <0x80000000 0x8010c7f>;
      compatible = "ibm,npu-link";
      ibm,npu-lane-mask = <0xff00>;
      phandle = <0x100002bf>;
      ibm,npu-link-index = <0x4>;
      linux,phandle = <0x100002bf>;
    };

    link@5 {
      ibm,npu-pbcq = <0x1000000a>;
      ibm,npu-phy = <0x80000000 0x8010c7f>;
      compatible = "ibm,npu-link";
      ibm,npu-lane-mask = <0xff>;
      phandle = <0x100002c0>;
      ibm,npu-link-index = <0x5>;
      linux,phandle = <0x100002c0>;
    };
  };
};
```

GPU memory bindings

```
memory@100000000 {
        device_type = "memory"
        compatible = "ibm,coherent-device-memory";
        linux,usable-memory = <0x0 0x100000000 0x0 0x0>;

; denotes a region of unplugged system memory

        reg = <0x0 0x100000000 0x0 0x80000000>;
        ibm,associativity = <0x4 0x0 0x0 0x0 0x64>;

; numa associativity for the memory once it is hotplugged

        phandle = <0x10000abc>;
        linux,phandle = <0x10000abc>;
};
```

Emulated PCI device bindings

```
pciex@3fff000400000 {
        ibm,npcq = <0x100002bc>; /* phandle to the NPU node. Used to find associated
↪PCI GPU devices. */
        compatible = "ibm,power8-npu-pciex", "ibm,ioda2-npu-phb";

        pci@0 {
                reg = <0x0 0x0 0x0 0x0 0x0>;
                revision-id = <0x0>;
                interrupts = <0x1>;
                device-id = <0x4ea>;
                ibm,pci-config-space-type = <0x1>;
                vendor-id = <0x1014>;
                ibm,gpu = <0x100002f7>; /* phandle pointing the associated GPU PCI
↪device node */
                memory-region = <0x10000abc>; /* phandle pointing to the GPU memory
↪*/
                ibm,nvlink-speed = <0x1>;

        ; Denotes the speed the link is running at:
        ; 0x3 == 20 Gbps, 0x8 = 25.78125 Gbps, 0x9 == 25.00000 Gbps

                phandle = <0x100002fc>;
        };

        pci@1 {
                reg = <0x800 0x0 0x0 0x0 0x0>;
                revision-id = <0x0>;
                interrupts = <0x1>;
                device-id = <0x4ea>;
                ibm,pci-config-space-type = <0x1>;
                vendor-id = <0x1014>;
                ibm,gpu = <0x100002f5>;
                memory-region = <0x10000def>;
                phandle = <0x100002fe>;
                class-code = <0x60400>;
                linux,phandle = <0x100002fe>;
        };

        pci@0,1 {
                reg = <0x100 0x0 0x0 0x0 0x0>;
                revision-id = <0x0>;
                interrupts = <0x2>;
                device-id = <0x4ea>;
                ibm,pci-config-space-type = <0x1>;
                vendor-id = <0x1014>;
                ibm,gpu = <0x100002f7>;
                memory-region = <0x10000abc>;
                phandle = <0x100002fd>;
                class-code = <0x60400>;
                linux,phandle = <0x100002fd>;
        };

        pci@1,1 {
               reg = <0x900 0x0 0x0 0x0 0x0>;
                revision-id = <0x0>;
                interrupts = <0x2>;
```

```
                device-id = <0x4ea>;
                ibm,pci-config-space-type = <0x1>;
                vendor-id = <0x1014>;
                ibm,gpu = <0x100002f5>;
                memory-region = <0x10000def>;
                phandle = <0x100002ff>;
                class-code = <0x60400>;
                linux,phandle = <0x100002ff>;
        };
};
```

3.3.6 Nest (NX) Accelerator Coprocessor

The NX coprocessor is present in P7+ or later processors. Each NX node represents a unique NX coprocessor. The nodes are located under an xscom node, as:

```
/xscom@<xscom_addr>/nx@<nx_addr>
```

With unique xscom and nx addresses. Their compatible node contains "ibm,power-nx".

NX Compression Coprocessor

This is the memory compression coprocessor. which uses the IBM proprietary 842 compression algorithm and format. Each NX node contains an 842 engine.

```
ibm,842-coprocessor-type     : CT value common to all 842 coprocessors
ibm,842-coprocessor-instance : CI value unique to all 842 coprocessors
```

Access to the coprocessor requires using the ICSWX instruction, which uses a specific format including a Coprocessor Type (CT) and Coprocessor Instance (CI) value to address each request to the right coprocessor. The driver should use the CT and CI values for a particular node to communicate with it. For all 842 coprocessors in the system, the CT value will (should) be the same, while each will have a different CI value. The driver can use CI 0 to allow the hardware to automatically select which coprocessor instance to use.

On P9, this compression coprocessor also supports standard GZIP/ZLIB compression algorithm and format. Virtual Accelerator Swirchboard (VAS) is used to access this coprocessor. VAS writes each request to receive FIFOs (RXFIFO) which are either high or normal priority and these FIFOs are bound to coprocessor types (842 and gzip).

VAS distinguishes NX requests for the target engines based on logical partition ID (lpid), process ID (pid) and Thread ID (tid). So (lpid, pid, tid) combination has to be unique in the system. Each NX node contains high and normal FIFOs for each 842 and GZIP engines.

```
/ibm,842-high-fifo            : High priority 842 RxFIFO
/ibm,842-normal-fifo          : Normal priority 842 RxFIFO
/ibm,gzip-high-fifo           : High priority gzip RxFIFO
/ibm,gzip-normal-fifo         : Normal priority gzip RxFIFO
```

Each RxFIFO node contains:

```
compatible              : ibm,p9-nx-842 or ibm,p9-nx-gzip
priority                : High or Normal
rx-fifo-address         : RxFIFO buffer address
rx-fifo-size            : RxFIFO size
lpid                    : 0xfff (1's for 12 bits in UMAC notify match
                          register)
```

```
pid                     : Coprocessor type (either 842 or gzip)
tid                     : counter in each coprocessor type
```

During initialization, the driver invokes VAS interface for each coprocessor type (842 and gzip) to configure the RxFIFO with rx_fifo_address, lpid, pid and tid for high and nornmal priority FIFOs.

NX RNG Coprocessor

This is the Random Number Generator (RNG) coprocessor, which is a part of each NX coprocessor. Each node represents a unique RNG coprocessor. Its nodes are not under the main nx node, they are located at:

```
/hwrng@<addr>           : RNG at address <addr>
ibm,chip-id             : chip id where the RNG is
reg                     : address of the register to read from
```

Each read from the RNG register will provide a new random number.

3.3.7 reserved-memory device tree nodes

OPAL exposes reserved memory through a top-level reserved-memory node, containing subnodes that represent each reserved memory region.

This follows the Linux specification for the /reserved-memory node, described in the kernel source tree, in:

> Documentation/devicetree/bindings/reserved-memory/reserved-memory.txt

The top-level /reserved-memory node contains:

```
#size-cells = <2>
#address-cells = <2>
```

Addresses and sizes are all 64-bits.

ranges the empty ranges node indicates no translation of physical addresses in the subnodes.

The sub-nodes under the /reserved-memory node contain:

reg = <address size> the address and size of the reserved memory region. The address and size values are two cells each, as signified by the top-level `#{address,size}-cells`

ibm,prd-label = "string" a string token for use by the prd system. Specific ranges may be used by prd - those will be referenced by this label.

3.3.8 Trusted Platform Module (TPM)

The tpm node describes a TPM device present in the platform. It also includes event log information.

Required properties

All these properties are added by hostboot and consumed by skiboot and the linux kernel (tpm and vtpm codes)

```
compatible :        manufacturer,model

linux,sml-base:     64-bit base address of the reserved memory allocated for firmware␣
↪event log.
```

```
                    sml stands for shared memory log.

linux,sml-size:     size of the memory allocated for firmware event log.
```

Optional properties

```
status:             indicates whether the device is enabled or disabled. "okay" for
                    enabled and "disabled" for disabled.
```

Example

```
tpm@57 {
    reg = <0x57>;
    compatible = "nuvoton,npct650", "nuvoton,npct601";
    linux,sml-base = <0x7f 0xfd450000>;
    linux,sml-size = <0x10000>;
    status = "okay";
    phandle = <0x10000017>;
    linux,phandle = <0x10000017>;
};
```

3.3.9 Virtual Accelerator Switchboard (VAS)

VAS is present in P9 or later processors. In P9, each chip has one instance of VAS. Each instance of VAS is represented as a "platform device" i.e as a node in root of the device tree:

```
/vas@<vas_addr>
```

with unique VAS address which also represents the Hypervisor window context address for the instance of VAS.

Each VAS node contains:

```
compatible: "ibm,power9-vas", "ibm,vas"

ibm,chip-id: Chip-id of the chip containing this instance of VAS.

ibm,vas-id: unique identifier for each instance of VAS in the system.

reg: contains 8 64-bit fields.

      Fields [0] and [1] represent the Hypervisor window context BAR
      (start and length). Fields [2] and [3] represent the OS/User
      window context BAR (start and length). Fields [4] and [5]
      contain the start and length of paste power bus address region
      for this chip. Fields [6] and [7] represent the bit field (start
      bit and number of bits) where the window id of the window should
      be encoded when computing the paste address for the window.
```

3.3.10 VPD (Vital Product Data)

VPD provides the information about the FRUs (Field Replaceable Unit) present in the system and each vpd node in the device tree represents a FRU. These properties are passed to skiboot in the form of HDAT structure. Skiboot

parses these structures and adds respective nodes in the device tree. These properties are available in all system except POWER8 BMC based system.

```
vpd {                       /* VPD root node */
  fru-name@rsrc-id {        /* Node name formatted as such */
  ibm,vpd = <               /* VPD data binary blob */ >;
    ccin = "524D";          /* Customer Card Identification Number */
    fru-type = [ 41 56 ]; /* FRU type label (2 bytes ASCII character) */
    fru-number     =        "FRU stocking part number";
    ibm,loc-code   =        "Location code";
    part-number    =        "ABC123456";
    serial-number  =        "ABC123456";
    ibm,chip-id    = <0x0>; /* If part is a chip, Processor Id */
    size = "0032768";       /* DIMM size (applicable for DIMM VPD only) */
    ibm,memory-bus-frequency = <0x0 0x0>; /* DIMM frequency (applicable for DIMM VPD␣
↪only) */
    vendor = <vendor name> /* Vendor name */
    build-date = <build date> /* Manufacturing build date (applicate for P9 BMC␣
↪systems only) */
  };
};
```

The VPD tree in the device tree depicts the hierarchial structure of the FRUs having parent-child relationship.

```
root-node-vpd@a000
   |-- enclosure@1e00
   |    |-- air-mover@3a00
   |    |-- air-mover@3a01
   |    |-- backplane@800
   |    |    |-- anchor-card@500
   |    |    |-- backplane-extender@900
   |    |    |    |-- serial-connector@2a00
   |    |    |    |-- usb-connector@2900
   |    |    |    `-- usb-connector@2901
   |    |    |-- ethernet-connector@2800
   |    |    |-- ethernet-connector@2801
   |    |    |-- ms-dimm@d002
   |    |    |-- ms-dimm@d003
   |    |    |-- processor@1000
   |    |    |-- processor@1001
   |    |    |-- usb-connector@2902
   |    |    |-- usb-connector@2903
   |    |    |-- usb-connector@2904
   |    |    `-- usb-connector@2905
   |    |-- dasd-backplane@2400
   |    |-- dasd-backplane@2401
   |    |-- power-supply@3103
   |    `-- service-processor@200
   |-- enclosure-fault-led@a300
   |-- enclosure-led@a200
   |-- root-node-vpd@a001
   `-- system-vpd@1c00
```

Example vpd node:

```
anchor-card@500 {
       ccin = "52FE";
       fru-number = "00E2147";
```

```
        description = "System Anchor Card - IBM Power 824";
        ibm,loc-code = "U78C9.001.WZS007X-P1-C13";
        serial-number = "YL10113BJ001";
        ibm,vpd = <0x84cc0052 0x54045649 0x4e494452 0x10414e43 0x484f5220 0x20202020␣
↪0x20202020 0x20434501 0x31565a02 0x3031464e 0x7303045 0x32313437 0x504e0730␣
↪0x30453231 0x3438534e 0xc594c31 0x30313133 0x424a3030 0x31434304 0x35324645␣
↪0x50520881 0x300000 0x48 0x45043030 0x31304354 0x440b400 0x485702 0x14233 0x6000000␣
↪0x142 0x34010042 0x370c0000 0x0 0x0 0x4239 0x3c435333 0x22071917 0xd1569c53␣
↪0x50973c87 0x71f9c40 0x1d4d3142 0x985e80f1 0x5cb3614d 0x32a902cb 0xd9d714ab␣
↪0x164d3322 0xdda4f986 0x5a618f4d 0x340b157c 0x2cac0a94 0x6504603 0x78 0x0>;
        fru-type = [41 56];
        part-number = "00E2148";
        phandle = <0x8d>;
        linux,phandle = <0x8d>;
};
```

3.4 OPAL API Documentation

The OPAL API is the interface between an Operating System and OPAL.

3.4.1 OPAL_CEC_POWER_DOWN

```
#define OPAL_CEC_POWER_DOWN                    5

int64 opal_cec_power_down(uint64 request)
```

Arguments

```
uint64 request values as follows:
 0 - Power down normally
 1 - Power down immediately
```

This OPAL call requests OPAL to power down the system. The exact difference between a normal and immediate shutdown is platform specific.

Current Linux kernels just use power down normally (0). It is valid for a platform to only support some types of power down operations.

Return Values

OPAL_SUCCESS the power down request was successful. This may/may not result in immediate power down. An OS should spin in a loop after getting *OPAL_SUCCESS* as it is likely that there will be a delay before instructions stop being executed.

OPAL_BUSY unable to power down, try again later.

OPAL_BUSY_EVENT Unable to power down, call *opal_run_pollers* and try again.

OPAL_PARAMETER a parameter was incorrect

OPAL_INTERNAL_ERROR Something went wrong, and waiting and trying again is unlikely to be successful. Although, considering that in a shutdown code path, there's unlikely to be any other valid option to take, retrying is perfectly valid.

In older OPAL versions (prior to skiboot v5.9), on IBM FSP systems, this return code was returned erroneously instead of OPAL_BUSY_EVENT during an FSP Reset/Reload.

OPAL_UNSUPPORTED this platform does not support being powered off.

3.4.2 OPAL_CEC_REBOOT and OPAL_CEC_REBOOT2

```
#define OPAL_CEC_REBOOT               6
#define OPAL_CEC_REBOOT2     116
```

There are two opal calls to invoke system reboot.

OPAL_CEC_REBOOT Used for normal reboot by Linux host.

OPAL_CEC_REBOOT2 Newly introduced to handle abnormal system reboots. The Linux kernel will make this OPAL call when it has to terminate abruptly due to an anomalous condition. The kernel will push some system state context to OPAL, which will in turn push it down to the BMC for further analysis.

OPAL_CEC_REBOOT

Syntax:

```
int64_t opal_cec_reboot(void)
```

System reboots normally.

OPAL_CEC_REBOOT2

Syntax:

```
int64_t opal_cec_reboot2(uint32_t reboot_type, char *diag)
```

Input parameters

reboot_type Type of reboot. (see below)

diag Null-terminated string.

Depending on reboot type, this call will carry out additional steps before triggering reboot.

Supported reboot types:

OPAL_REBOOT_NORMAL = 0 Behavior is as similar to that of opal_cec_reboot()

OPAL_REBOOT_PLATFORM_ERROR = 1 Log an error to the BMC and then trigger a system checkstop, using the information provided by 'ibm,sw-checkstop-fir' property in the device-tree. Post the checkstop trigger, OCC/BMC will collect relevant data for error analysis and trigger a reboot.

In absence of 'ibm,sw-checkstop-fir' device property, this function will return with OPAL_UNSUPPORTED and no reboot will be triggered.

OPAL_REBOOT_FULL_IPL = 2 Force a full IPL reboot rather than using fast reboot.

On platforms that don't support fast reboot, this is equivalent to a normal reboot.

Unsupported Reboot type For unsupported reboot type, this function will return with OPAL_UNSUPPORTED and no reboot will be triggered.

3.4.3 OPAL_CHECK_TOKEN

This OPAL call allows the host OS to determine if a particular OPAL call is present on a system. This allows for simple compatibility between OPAL versions and different OPAL implementations/platforms.

One parameter is accepted: the OPAL token number.

`OPAL_CHECK_TOKEN` will return:

```
enum OpalCheckTokenStatus {
  OPAL_TOKEN_ABSENT = 0,
  OPAL_TOKEN_PRESENT = 1
};
```

indicating the presence/absence of the particular OPAL_CALL.

`OPAL_CHECK_TOKEN` is REQUIRED to be implemented by a conformant OPAL implementation.

For skiboot, only positively ancient internal-to-IBM versions were missing OPAL_CHECK_TOKEN. In this case, OPAL_PARAMETER would be returned. There is no reason for a host OS to support this behaviour.

3.4.4 Code Update on FSP based machine

There are three OPAL calls for code update on FSP based machine:

```
#define OPAL_FLASH_VALIDATE    76
#define OPAL_FLASH_MANAGE      77
#define OPAL_FLASH_UPDATE      78
```

OPAL_FLASH_VALIDATE

Validate new image is valid for this platform or not. We do below validation in OPAL:

- We do below sys parameters validation to confirm inband update is allowed. - Platform is managed by HMC or not?. - Code update policy (inband code update allowed?).
- We parse candidate image header (first 4k bytes) to perform below validations. - Image magic number. - Image version to confirm image is valid for this platform.

Input

buffer First 4k bytes of new image

size Input buffer size

Output

buffer Output result (current and new image version details)

size Output buffer size

result Token to identify what will happen if update is attempted See hw/fsp/fsp-codeupdate.h for token values.

Return value

Validation status

OPAL_FLASH_MANAGE

Commit/Reject image.

- We can commit new image (T -> P), if system is running with T side image.
- We can reject T side image, if system is running with P side image.

Note: If a platform is running from a T side image when an update is to be applied, then the platform may automatically commit the current T side image to the P side to allow the new image to be updated to the temporary image area.

Input

op Operation (1 : Commit /0 : Reject)

Return value Commit operation status (0 : Success)

OPAL_FLASH_UPDATE

Update new image. It only sets the flag, actual update happens during system reboot/shutdown.

Host splits FW image to scatter/gather list and sends it to OPAL. OPAL parse the image to get indivisual LID and passes it to FSP via MBOX command.

FW update flow :

- **if (running side == T)** Swap P & T side
- Start code update
- Delete T side LIDs
- Write LIDs
- Code update complete
- Deep IPL

Input

list Real address of image scatter/gather list of the FW image

Return value: Update operation status (0: update requested)

3.4.5 OPAL Console calls

There are four OPAL calls relating to the OPAL console:

```
#define OPAL_CONSOLE_WRITE                      1
#define OPAL_CONSOLE_READ                       2
#define OPAL_CONSOLE_WRITE_BUFFER_SPACE                 25
#define OPAL_CONSOLE_FLUSH                      117
```

The OPAL console calls can support multiple consoles. Each console MUST be represented in the device tree.

A conforming implementation SHOULD have at least one console. It is valid for it to simply be an in-memory buffer and only support writing.

[TODO: details on device tree specs for console]

OPAL_CONSOLE_WRITE

Parameters:

```
int64_t term_number
int64_t *length,
const uint8_t *buffer
```

Returns:

```
OPAL_SUCCESS
OPAL_PARAMETER - invalid term_number
OPAL_CLOSED - console device closed
OPAL_BUSY_EVENT - unable to write any of buffer
```

`term_number` is the terminal number as represented in the device tree. `length` is a pointer to the length of buffer.

A conforming implementation SHOULD try to NOT do partial writes, although partial writes and not writing anything are valid.

OPAL_CONSOLE_WRITE_BUFFER_SPACE

Parameters:

```
int64_t term_number
int64_t *length
```

Returns:

```
OPAL_SUCCESS
OPAL_PARAMETER - invalid term_number
```

Returns the available buffer length for OPAL_CONSOLE_WRITE in `length`. This call can be used to help work out if there is sufficient buffer space to write your full message to the console with OPAL_CONSOLE_WRITE.

OPAL_CONSOLE_READ

Parameters:

```
int64_t term_number
int64_t *length
uint8_t *buffer
```

Returns:

```
OPAL_SUCCESS
OPAL_PARAMETER - invalid term_number
OPAL_CLOSED
```

Use OPAL_POLL_EVENTS for how to determine

OPAL_CONSOLE_FLUSH

Parameters:

```
int64_t term_number
```

Returns:

```
OPAL_SUCCESS
OPAL_UNSUPPORTED - the console does not implement a flush call
OPAL_PARAMETER - invalid term_number
OPAL_PARTIAL - more to flush, call again
OPAL_BUSY - nothing was flushed this call
```

3.4.6 OPAL_ELOG: Error logging

OPAL provides an abstraction to platform specific methods of storing and retrieving error logs. Some service processors may be able to store information in the Platform Error Log (PEL) format. These may be generated at runtime by the service processor or OPAL in reaction to certain events. For example, an IPL failure could be recorded in an error log, as could the reason and details of an unexpected shut-down/reboot (e.g. hard thermal limits, check-stop).

There are five OPAL calls from host to OPAL on error log:

```
#define OPAL_ELOG_READ          71
#define OPAL_ELOG_WRITE         72
#define OPAL_ELOG_ACK           73
#define OPAL_ELOG_RESEND        74
#define OPAL_ELOG_SIZE          75
```

Note: `OPAL_ELOG_WRITE` (72) Unused for now, can be used in future.

Not all platforms support these calls, so it's important for a host Operating System to use the OPAL_CHECK_TOKEN call first. If `OPAL_ELOG_READ`, `OPAL_ELOG_ACK`, `OPAL_ELOG_RESEND`, or `OPAL_ELOG_SIZE` is present, then the rest of that group is also present. The presence of `OPAL_ELOG_WRITE` must be checked separately.

TODO: we need a good explanation of the notification mechanism and in what order and *when* to call each of the OPAL APIs.

OPAL_ELOG_READ

The `OPAL_ELOG_READ` call will copy the error log identified by `id` into the `buffer` of size `size`.

`OPAL_ELOG_READ` accepts 3 parameters:

```
uint64_t *elog_buffer
uint64_t elog_size
uint64_t elog_id
```

Returns:

OPAL_WRONG_STATE When there are no error logs to read, or OPAL_ELOG calls are done in the wrong order.

OPAL_PARAMETER The id does not match the log id that is available.

OPAL_SUCCESS Error log is copied to buffer.

Other generic OPAL error codes may also be returned and should be treated like OPAL_INTERNAL_ERROR.

OPAL_ELOG_ACK

Acknowledging (ACKing) an error log tells OPAL and the service processor that the host operating system has dealt with the error log successfully. This allows OPAL and the service processor to delete the error log from their memory/storage.

OPAL_ELOG_ACK accepts 1 parameter:

```
uint64_t ack_id
```

Returns:

OPAL_INTERNAL_ERROR OPAL failed to send acknowledgement to the error log creator.

OPAL_SUCCESS Success!

Other generic OPAL error codes may also be returned, and should be treated like OPAL_INTERNAL_ERROR.

OPAL_ELOG_RESEND

The OPAL_ELOG_RESEND call will cause OPAL to resend notification to the host operating system of all outstanding error logs. This is commonly used (although doesn't have to be) in a kexec scenario.

The call registered with this token accepts no parameter and returns type is void.

OPAL_ELOG_SIZE

The OPAL_ELOG_SIZE call retrieves information about an error log.

Here, type specifies error log format. Supported types are :

```
0 -> Platform Error Log
```

OPAL_ELOG_SIZE accepts 3 parameters:

```
uint64_t *elog_id
uint64_t *elog_size
uint64_t *elog_type
```

Returns:

OPAL_WRONG_STATE There is no error log to fetch information about.

OPAL_SUCCESS Success.

Other general OPAL errors may be returned.

3.4.7 OPAL Flash calls

There are three OPAL calls for interacting with flash devices:

```
#define OPAL_FLASH_READ         110
#define OPAL_FLASH_WRITE        111
#define OPAL_FLASH_ERASE        112
```

Multiple flash devices are supported by OPAL - each of these calls takes an id parameter, which must match an ID found in the corresponding `ibm,opal/flash@n` device tree node. See *ibm,opal/flash device tree entries* for details of the device tree bindings.

All operations on the flash device must be aligned to the block size of the flash. This applies to both offset and size arguments.

This interface is asynchronous; all calls require a 'token' argument. On success, the calls will return `OPAL_ASYNC_COMPLETION`, and an opal_async_completion message will be sent (with the appropriate token argument) when the operation completes.

All calls share the same return values:

`OPAL_ASYNC_COMPLETION` operation started, an async completion will be triggered with the `token` argument

`OPAL_PARAMETER` invalid flash id

`OPAL_PARAMETER` invalid size or offset (alignment, or access beyond end of device)

`OPAL_BUSY` flash in use

`OPAL_HARDWARE` error accessing flash device

OPAL_FLASH_READ

Parameters:

```
uint64_t id
uint64_t offset
uint64_t buffer
uint64_t size
uint64_t token
```

Reads from the specified flash id, at the specified offset, into the buffer. Will trigger an async completion with token when completed.

OPAL_FLASH_ERASE

Parameters:

```
uint64_t id
uint64_t offset
uint64_t size
uint64_t token
```

Erases the specified flash id, at the specified offset and size. Will trigger an async completion with token when completed.

OPAL_FLASH_WRITE

Parameters:

```
uint64_t id
uint64_t offset
uint64_t buffer
uint64_t size
uint64_t token
```

Writes buffer to the specified flash id, at the specified offset and size. The flash must be erased before being written. Will trigger an async completion with token when completed.

3.4.8 OPAL_GET_DEVICE_TREE

Get device sub-tree.

Parameters:

```
uint32_t phandle: root device node phandle of the device sub-tree
uint64_t buf: FDT blob buffer or NULL
uint64_t len: length of the FDT blob buffer
```

Calling:

Retrieve device sub-tree. The root node's phandle is identified by @phandle. The typical use is for the kernel to update its device tree following a change in hardware (e.g. PCI hotplug).

Return Codes:

FDT blob size returned FDT blob buffer size when `buf` is NULL

OPAL_SUCCESS FDT blob is created successfully

OPAL_PARAMETER invalid argument @phandle or @len

OPAL_INTERNAL_ERROR failure creating FDT blob when calculating its size

OPAL_NO_MEM not enough room in buffer for device sub-tree

OPAL_EMPTY failure creating FDT blob

3.4.9 OPAL_GET_MSG

OPAL_GET_MSG will get the next pending OPAL Message (see *OPAL_MESSAGE*).

Parameters:

```
buffer to copy message into
sizeof buffer to copy message into
```

The maximum size of an opal message is specified in the device tree passed to the host OS:

```
ibm,opal {
          opal-msg-size = <0x48>;
}
```

It is ALWAYS at least 72 bytes. In the future, OPAL may have messages larger than 72 bytes. Naturally, a HOST OS will only be able to interpret these if it correctly uses opal-msg-size. Any OPAL message > 72 bytes, a host OS may safely ignore.

A host OS *SHOULD* always supply a buffer to OPAL_GET_MSG of either 72 bytes or opal-msg-size. It MUST NOT supply a buffer of < 72 bytes.

Return values

OPAL_RESOURCE no available message.

OPAL_PARAMETER buffer is NULL or size is < 72 bytes. If buffer size < 72 bytes, the message will NOT be discarded by OPAL.

OPAL_PARTIAL If pending opal message is greater than supplied buffer. In this case the message is *DISCARDED* by OPAL. This is to keep compatibility with host Operating Systems with a hard coded opal-msg-size of 72 bytes. **NOT CURRENTLY IMPLEMENTED**. Specified so that host OS can prepare for the possible future with either a sensible error message or by gracefully ignoring such OPAL messages.

OPAL_SUCCESS message successfully copied to buffer.

3.4.10 OPAL_GET_MSI_32 and OPAL_GET_MSI_64

```
#define OPAL_GET_MSI_32                                 39
#define OPAL_GET_MSI_64                                 40
```

WARNING: the following documentation is from old sources, and is possibly not representative of OPALv3 as implemented by skiboot. This should be used as a starting point for full documentation.

OPAL PHBs encode MVE and XIVE specifiers in MSI DMA and message data values. The host calls these functions to determine the PHB MSI DMA address and message data to program into a PE PCIE function for a particular MVE and XIVE. The msi_address parameter returns the MSI DMA address and the msi_data parameter returns the MSI DMA message data value the PE uses to signal that interrupt.

phb_id The `phb_id` parameter is the value from the PHB node `ibm,opal-phbid` property.

mve_number The `mve_number` is the index of an MVE used to authorize this PE to this MSI. For `ibm,opal-ioda2` PHBs, the MVE number argument is ignored.

xive_number The `xive_number` is the index of an XIVE that corresponds to a particular DMA address and message data value this PE will signal as an MSI ro MSI-X.

msi_range The msi_range parameter specifies the number of MSIs associated with the in put MVE and XIVE, primarily for MSI-conventional Multiple Message Enable > 1 MSI. MSI requires consecutive MSIs per MSI address, and each MSI DMA address must be unique for any given consecutive power of 2 set of 32 message data values,. which in turn select particular PHB XIVEs. This value must be a power of 2 value in the range of 0 to 32. OPAL returns opal_parameter for values outside of this range.

For MSI conventional, the MSI address and message data returned apply to a power of 2 sequential set of XIVRs starting from the xive_number for the power of 2 msi_range input argument. The message data returned represents the power of 2 aligned starting message data value of the first interrupt number in that sequential range. Valid msi_range input values are from 1 to 32. Non-power of 2 values result in a return code of opal_PARAMETER .

An msi_range value of 0 or 1 signifies that OPAL should return the message data and message address for exactly one MSI specified by the input XIVE number. For MSI conventional, the host should specify either a value of 0 or 1, for an MSI Capability MME value of 1 MSI. For MSI-X XIVRs, the host should specify a value of '1' for the msi_range argument and call this function for each MSI-X uniquely.

3.4.11 OPAL_GET_XIVE

```
#define OPAL_GET_XIVE                                   20
```

The host calls this function to return the configuration of an interrupt source. See OPAL_SET_XIVE for details.

Parameters

isn The `isn` is the global interrupt number being queried

server_number the `server_number` returns the mangled server (processor) that is set to receive that interrupt.

priority the `priority` returns the current interrupt priority setting for that interrupt.

3.4.12 OPAL_GET_XIVE_SOURCE

```
This function validates the given ``xive_num`` and sets the
``interrupt_source_number``. Then returns the proper return code.
```

Parameters

phb_id The `phb_id` parameter is the value from the PHB node `ibm,opal-phbid` property.

xive_num The `xive_num` is the index of an XIVE that corresponds to a particular interrupt.

interrupt_source_number The `interrupt_source_number` is a value formed by the combination of the device tree MSI property base BUID and `xive_num`

Return Codes

OPAL_PARAMETER The indicated `phb_id` not found

OPAL_UNSUPPORTED Presence retrieval not supported on the `phb_id`

OPAL_SUCCESS Indicates Success!

3.4.13 OPAL_HANDLE_INTERRUPT

The host OS must pass all interrupts in the *opal-interrupts* property of *ibm,opal* in the device tree to OPAL.

An example dt snippet is:

```
ibm,opal {
        opal-interrupts = <0x10 0x11 0x12 0x13 0x14 0x20010 0x20011 0x20012 0x20013␣
↪0x20014 0xffe 0xfff 0x17fe 0x17ff 0x2ffe 0x2fff 0x37fe 0x37ff 0x20ffe 0x20fff␣
↪0x217fe 0x217ff 0x22ffe 0x22fff 0x237fe 0x237ff>;
};
```

When the host OS gets any of these interrupts, it must call `OPAL_HANDLE_INTERRUPT`.

The `OPAL_HANDLE_INTERRUPT` call takes two parameters, one input and one output.

uint32_t isn the interrupt

uint64_t *outstanding_event_mask returns outstanding events for host OS to handle

The host OS should then handle any outstanding events.

See *OPAL_POLL_EVENTS* for documentation on events.

3.4.14 OPAL_IMC_COUNTERS_INIT

OPAL call interface to initialize In-memory collection infrastructure. Call does multiple scom writes on each incavation for Core IMC initialization. And for the Nest IMC, at this point, call is a no-op and returns OPAL_SUCCESS. Incase of kexec, OS driver should first stop the engine via OPAL_IMC_COUNTER_STOP(and then free the memory if allocated, for nest memory is mmapped). Incase of kdump, OS driver should stop the engine via OPAL_IMC_COUNTER_STOP.

OPAL does sanity checks to detect unknown or unsupported IMC device type and nest units. check_imc_device_type() function removes unsupported IMC device type. disable_unavailable_units() removes unsupported nest units by the microcode. This way OPAL can lock down and advertise only supported device type and nest units.

Parameters

uint32_t type This parameter specifies the imc counter domain. The value should be 'OPAL_IMC_COUNTERS_CORE'

uint64_t addr This parameter must have a non-zero value. This value must be a physical address of the core.

uint64_t cpu_pir This parameter specifices target cpu pir

Returns

OPAL_PARAMETER - In case of unsupported `type` OPAL_HARDWARE - If any error in setting up the hardware. OPAL_SUCCESS - On succesfully initialized or even if init operation is a no-op.

3.4.15 OPAL_IMC_COUNTERS_START

OPAL call interface for starting the In-Memory Collection counters for a specified domain (NEST/CORE).

Parameters

uint32_t type This parameter specifies the imc counter domain. The value can be either 'OPAL_IMC_COUNTERS_NEST' or 'OPAL_IMC_COUNTERS_CORE'

uint64_t cpu_pir This parameter specifices target cpu pir

Returns

OPAL_PARAMETER - In case of Unsupported `type` OPAL_HARDWARE - If any error in setting up the hardware. OPAL_SUCCESS - On successful execution of the operation for the given `type`.

3.4.16 OPAL_IMC_COUNTERS_STOP

OPAL call interface for stoping In-Memory Collection counters for a specified domain (NEST/CORE). STOP should always be called after a related START. While STOP *may* run successfully without an associated START call, this is not gaurenteed.

Parameters

uint32_t type This parameter specifies the imc counter domain. The value can be either 'OPAL_IMC_COUNTERS_NEST' or 'OPAL_IMC_COUNTERS_CORE'

uint64_t cpu_pir This parameter specifices target cpu pir

Returns

OPAL_PARAMETER - In case of Unsupported `type` OPAL_HARDWARE - If any error in setting up the hardware. OPAL_SUCCESS - On successful execution of the operation for the given `type`.

3.4.17 OPAL_INT_EOI

```
static int64_t opal_xive_eoi(uint32_t xirr)
```

Not yet implemented.

Modelled on the `H_EOI` PAPR call.

This can return a positive value, which means more interrupts are queued for that CPU/priority and must be fetched as the XIVE is not guaranteed to assert the CPU external interrupt line again until the pending queue for the current priority has been emptied.

For P9 and above systems where host doesn't know about interrupt controller. An OS can instead make OPAL calls for XICS emulation.

For an OS to use this OPAL call, an `ibm,opal-intc` compatible device must exist in the device tree. If OPAL does not create such a device, the host OS MUST NOT use this call.

3.4.18 OPAL_INT_GET_XIRR

```
int64_t opal_xive_get_xirr(uint32_t *out_xirr, bool just_poll)
```

Not yet implemented.

Modelled on the PAPR call.

For P9 and above systems where host doesn't know about interrupt controller. An OS can instead make OPAL calls for XICS emulation.

For an OS to use this OPAL call, an `ibm,opal-intc` compatible device must exist in the device tree. If OPAL does not create such a device, the host OS MUST NOT use this call.

3.4.19 OPAL_INT_SET_CPPR

```
static int64_t opal_xive_set_cppr(uint8_t cppr)
```

Not yet implemented.

Modelled on the `H_CPPR` PAPR call.

For P9 and above systems where host doesn't know about interrupt controller. An OS can instead make OPAL calls for XICS emulation.

For an OS to use this OPAL call, an `ibm,opal-intc` compatible device must exist in the device tree. If OPAL does not create such a device, the host OS MUST NOT use this call.

3.4.20 OPAL_INT_SET_MFRR

```
static int64_t opal_xive_set_mfrr(uint32_t cpu, uint8_t mfrr)
```

Not yet implemented.

Modelled on the `H_IPI` PAPR call.

For P9 and above systems where host doesn't know about interrupt controller. An OS can instead make OPAL calls for XICS emulation.

For an OS to use this OPAL call, an `ibm,opal-intc` compatible device must exist in the device tree. If OPAL does not create such a device, the host OS MUST NOT use this call.

3.4.21 OPAL_INVALID_CALL

An OPAL call of `-1` will always return `OPAL_PARAMETER`. It is always ivalid.

It exists purely for testing.

3.4.22 OPAL_IPMI_SEND

```
#define OPAL_IPMI_SEND                          107
```

`OPAL_IPMI_SEND` call will send an IPMI message to the service processor.

Parameters

```
uint64_t interface
struct opal_ipmi_msg *opal_ipmi_msg
uint64_t msg_len
```

`interface` `interface` parameter is the value from the ipmi interface node `ibm,ipmi-interface-id`

`opal_ipmi_msg` `opal_ipmi_msg` is the pointer to below structure `opal_ipmi_msg`

```
struct opal_ipmi_msg {
     uint8_t version;
     uint8_t netfn;
     uint8_t cmd;
     uint8_t data[];
};
```

`msg_len` ipmi message request size

Return Values

`OPAL_SUCCESS` `msg` queued successfully

`OPAL_PARAMETER` invalid ipmi message request length `msg_len`

OPAL_HARDWARE backend support is not present as block transfer/service processor ipmi routines are not initialized which are used for communication

OPAL_UNSUPPORTED in-correct opal ipmi message format version `opal_ipmi_msg->version`

OPAL_RESOURCE insufficient resources to create `ipmi_msg` structure

3.4.23 OPAL_IPMI_RECV

```
#define OPAL_IPMI_RECV                          108
```

`OPAL_IPMI_RECV` call reads an ipmi message of type `ipmi_msg` from ipmi message queue `msgq` into host OS structure `opal_ipmi_msg`.

Parameters

```
uint64_t interface
struct opal_ipmi_msg *opal_ipmi_msg
uint64_t *msg_len
```

interface `interface` parameter is the value from the ipmi interface node `ibm,ipmi-interface-id`

opal_ipmi_msg `opal_ipmi_msg` is the pointer to below structure `opal_ipmi_msg`

```
struct opal_ipmi_msg {
     uint8_t version;
     uint8_t netfn;
     uint8_t cmd;
     uint8_t data[];
};
```

msg_len `msg_len` is the pointer to ipmi message response size

Return Values

OPAL_SUCCESS ipmi message dequeued from `msgq` queue and memory taken by it got released successfully

OPAL_EMPTY `msgq` list is empty

OPAL_PARAMETER invalid ipmi `interface` value

OPAL_UNSUPPORTED in-correct opal ipmi message format version `opal_ipmi_msg->version`

3.4.24 Service Indicators (LEDS)

The service indicator is one element of an overall hardware service strategy where end user simplicity is a high priority. The goal is system firmware or operating system code to isolate hardware failures to the failing FRU and automatically activate the fault indicator associated with the failing FRU. The end user then needs only to look for the FRU with the active fault indicator to know which part to replace.

Different types of indicators handled by LED code:

- **System attention indicator (Check log indicator)** Indicates there is a problem with the system that needs attention.
- **Identify** Helps the user locate/identify a particular FRU or resource in the system.

- **Fault** Indicates there is a problem with the FRU or resource at the location with which the indicator is associated.

All LEDs are defined in the device tree (see *Service Indicators (LEDS)*).

LED Design

When it comes to implementation we can classify LEDs into two categories:

1. Hypervisor (OPAL) controlled LEDs (All identify & fault indicators) During boot, we read/cache these LED details in OPAL (location code, state, etc). We use cached data to serve read request from FSP/Host. And we use SPCN passthrough MBOX command to update these LED state.
2. Service processor (FSP) controlled LEDs (System Attention Indicator) During boot, we read/cache this LED info using MBOX command. Later anytime FSP updates this LED, it sends update system parameter notification MBOX command. We use that data to update cached data. LED update request is sent via set/reset attn MBOX command.

LED update request: Both FSP and Host will send LED update requests. We have to serialize SPCN passthrough command. Hence we maintain local queue.

Note:

- For more information regarding service indicator refer to PAPR spec (Service Indicators chapter).

There are two OPAL calls relating to LED operations.

OPAL_LEDS_GET_INDICATOR

Returns LED state for the given location code.

OPAL_LEDS_SET_INDICATOR

Sets LED state for the given location code.

See hw/fsp/fsp-leds.c for more deatails.

3.4.25 OPAL_LPC_READ

```
This function related to Low Pin Count (LPC) bus. This function reads the
data from IDSEL register for ``chip_id``, which has LPC information.
From ``addr`` for ``addr_type`` with read size ``sz`` bytes in to a
variable named ``data``.
```

Parameters

`chip_id` The `chip_id` parameter contains value of the chip number identified at boot time.

`addr_type` The `addr_type` is one of the LPC supported address types. Supported address types are - LPC memory, LPC IO and LPC firmware.

`addr` The `addr` from which the data has to be read.

`data` The `data` will be used to store the read data.

`sz` How many `sz` bytes to be read in to `data`.

Return Codes

OPAL_PARAMETER Indicates either `chip_id` not found or `chip_id` doesn't contain LPC information.

OPAL_SUCCESS Indicates Success!

3.4.26 OPAL_LPC_WRITE

```
This function related to Low Pin Count (LPC) bus. This function writes the
``data`` in to  ECCB register for ``chip_id``, which has LPC information.
From ``addr`` for ``addr_type`` with write size ``sz`` bytes.
```

Parameters

chip_id The `chip_id` parameter contains value of the chip number identified at boot time.

addr_type The `addr_type` is one of the address types LPC supported. Supported address types are - LPC memory, LPC IO and LPC firmware.

addr The `addr` to where the `data` need to be written.

data The `data` for writing.

sz How many `sz` bytes to write.

Return Codes

OPAL_PARAMETER Indicates either `chip_id` not found or `chip_id` doesn't contain LPC information.

OPAL_SUCCESS Indicates Success!

3.4.27 OPAL_MESSAGE

The host OS can use OPAL_GET_MSG to retrive messages queued by OPAL. The messages are defined by enum opal_msg_type. The host is notified of there being messages to be consumed by the OPAL_EVENT_MSG_PENDING bit being set.

An opal_msg is:

```
struct opal_msg {
      __be32 msg_type;
      __be32 reserved;
      __be64 params[8];
};
```

The data structure is ALWAYS at least this size (4+4+8*8 = 72 bytes). Some messages define fewer than eight parameters. For messages that do not define all eight parameters, the value in the undefined parameters is undefined, although can safely be memcpy()d or otherwise moved.

In the device tree, there's an opal-msg-size property of the OPAL node that says the size of a struct opal-msg. In the future, OPAL may support larger messages. See `OPAL_GET_MESSAGE` documentation for details.

```
ibm,opal {
          opal-msg-size = <0x48>;
}
```

OPAL_MSG_ASYNC_COMP

```
params[0] = token
params[1] = rc
```

Additional parameters are function-specific.

OPAL_MSG_MEM_ERR

OPAL_MSG_EPOW

Used by OPAL to issue environmental and power warnings to host OS for conditions requiring an earlier poweroff. A few examples of these are high ambient temperature or system running on UPS power with low UPS battery. Host OS can query OPAL via `GET_EPOW_STATUS` API to obtain information about EPOW conditions present. Refer include/opal-api.h for description of all supported EPOW events. OPAL_SYSPOWER_CHNG, OPAL_SYSPOWER_FAIL and OPAL_SYSPOWER_INC events don't require system poweroff.

Host OS should look for 'ibm,opal-v3-epow' string as compatible property for 'epow' node under OPAL device-tree to determine epow support.

OPAL_MSG_SHUTDOWN

Used by OPAL to inform the host OS it must imitate a graceful shutdown. Uses the first parameter to indicate weather the system is going down for shutdown or a reboot.

```
params[0] = 0x01 reboot, 0x00 shutdown
```

OPAL_MSG_HMI_EVT

Used by OPAL to sends the OPAL HMI Event to the host OS that reports a summary of HMI error and whether it was successfully recovered or not.

HMI is a Hypervisor Maintenance Interrupt usually reports error related to processor recovery/checkstop, NX checkstop and Timer facility. Hypervisor then takes this opportunity to analyze and recover from some of these errors. Hypervisor takes assistance from OPAL layer to handle and recover from HMI. After handling HMI, OPAL layer sends the summary of error report and status of recovery action using HMI event structure shown below.

The HMI event structure uses version numbering to allow future enhancement to accommodate additional members. The version start from V1 onward. Version 0 is invalid version and unsupported.

The current version of HMI event structure V2 and is backward compatible to V1 version.

Notes:

- When adding new structure to the union in future, the version number must be bumped.
- All future versions must be backward compatible to all its older versions.
- Size of this structure should not exceed that of struct opal_msg.

```
struct OpalHMIEvent {
     uint8_t         version;        /* 0x00 */
     uint8_t         severity;       /* 0x01 */
     uint8_t         type;           /* 0x02 */
     uint8_t         disposition;    /* 0x03 */
     uint8_t         reserved_1[4];  /* 0x04 */
```

```
    __be64          hmer;
    /* TFMR register. Valid only for TFAC and TFMR_PARITY error type. */
    __be64          tfmr;

    /* version 2 and later */
    union {
            /*
             * checkstop info (Core/NX).
             * Valid for OpalHMI_ERROR_MALFUNC_ALERT.
             */
            struct {
                    uint8_t xstop_type;     /* enum OpalHMI_XstopType */
                    uint8_t reserved_1[3];
                    __be32 xstop_reason;
                    union {
                            __be32 pir;       /* for CHECKSTOP_TYPE_CORE */
                            __be32 chip_id; /* for CHECKSTOP_TYPE_NX */
                    } u;
            } xstop_error;
    } u;
};
```

OPAL_MSG_DPO

Delayed poweroff where OPAL informs host OS that a poweroff has been requested and a forced shutdown will happen in future. Host OS can use OPAL_GET_DPO_STATUS API to query OPAL the number of seconds remaining before a forced poweroff will occur.

OPAL_MSG_PRD

This message is a OPAL-to-HBRT notification, and contains a struct opal_prd_msg:

```
enum opal_prd_msg_type {
        OPAL_PRD_MSG_TYPE_INIT = 0,     /* HBRT --> OPAL */
        OPAL_PRD_MSG_TYPE_FINI,         /* HBRT --> OPAL */
        OPAL_PRD_MSG_TYPE_ATTN,         /* HBRT <-- OPAL */
        OPAL_PRD_MSG_TYPE_ATTN_ACK,     /* HBRT --> OPAL */
        OPAL_PRD_MSG_TYPE_OCC_ERROR,    /* HBRT <-- OPAL */
        OPAL_PRD_MSG_TYPE_OCC_RESET,    /* HBRT <-- OPAL */
};

struct opal_prd_msg {
        uint8_t         type;
        uint8_t         pad[3];
        __be32          token;
        union {
                struct {
                        __be64  version;
                        __be64  ipoll;
                } init;
                struct {
                        __be64  proc;
                        __be64  ipoll_status;
                        __be64  ipoll_mask;
```

```
                } attn;
                struct {
                        __be64  proc;
                        __be64  ipoll_ack;
                } attn_ack;
                struct {
                        __be64  chip;
                } occ_error;
                struct {
                        __be64  chip;
                } occ_reset;
        };
};
```

Responses from the kernel use the same message format, but are passed through the opal_prd_msg call.

OPAL_MSG_OCC

This is used by OPAL to inform host about OCC events like OCC reset, OCC load and throttle status change by OCC which can indicate the host the reason for frequency throttling/unthrottling.

```
#define OCC_RESET                       0
#define OCC_LOAD                        1
#define OCC_THROTTLE                    2
#define OCC_MAX_THROTTLE_STATUS                 5
/*
 * struct opal_occ_msg:
 * type: OCC_RESET, OCC_LOAD, OCC_THROTTLE
 * chip: chip id
 * throttle status: Indicates the reason why OCC may have limited
 * the max Pstate of the chip.
 * 0x00 = No throttle
 * 0x01 = Power Cap
 * 0x02 = Processor Over Temperature
 * 0x03 = Power Supply Failure (currently not used)
 * 0x04 = Over current (currently not used)
 * 0x05 = OCC Reset (not reliable as some failures will not allow for
 * OCC to update throttle status)
 */
struct opal_occ_msg {
      __be64 type;
      __be64 chip;
      __be64 throttle_status;
};
```

Host should read opal_occ_msg.chip and opal_occ_msg.throttle_status only when `opal_occ_msg.type = OCC_THROTTLE`. If host receives `OCC_THROTTLE` after an `OCC_RESET` then this throttle message will have a special meaning which indicates that all the OCCs have become active after a reset. In such cases `opal_occ_msg.chip` and `opal_occ_msg.throttle_status` will be set to 0 and host should not use these values.

If `opal_occ_msg.type > 2` then host should ignore the message for now, new events can be defined for `opal_occ_msg.type` in the future versions of OPAL.

3.4.28 OPAL NPU2 calls

There are three OPAL calls for interacting with NPU2 devices:

```
#define OPAL_NPU_INIT_CONTEXT                   146
#define OPAL_NPU_DESTROY_CONTEXT                147
#define OPAL_NPU_MAP_LPAR                       148
```

These are used to setup and configure address translation services (ATS) for a given NVLink2 device. Note that in some documentation this is also referred to as extended translation services (XTS).

Each NVLink2 supports multiple processes running on a GPU which issues requests for address translation. The NPU2 is responsible for completing the request by forwarding it to the Nest MMU (NMMU) along with the appropriate translation context (MSR/LPCR) bits. These bits are keyed off a 20-bit process ID (PASID/PID) which is identical to the PID used on the processor.

The OPAL calls documented here are used to setup/destroy the appropriate context for a given process on a given NVLink2 device.

OPAL_NPU_INIT_CONTEXT

Parameters:

```
uint64_t phb_id
int pasid
uint64_t msr
uint64_t lpid
```

Allocates a new context ID and sets up the given PASID/PID to be associated with the supplied MSR on for the given LPID. MSR should only contain bits set requried for NPU2 address lookups - ie. MSR DR/HV/PR/SF.

Returns the context ID on success or `OPAL_RESOURCE` if no more contexts are available or `OPAL_UNSUPPORTED` in the case of unsupported MSR bits.

OPAL_NPU_DESTROY_CONTEXT

Parameters:

```
uint64_t phb_id
uint64_t id
```

Destroys a previously allocated context ID. This may cause further translation requests from the GPU to fail.

OPAL_NPU_MAP_LPAR

Parameters:

```
uint64_t phb_id
uint64_t bdf
uint64_t lparid
uint64_t lpcr
```

Associates the given GPU BDF with a particular LPAR and LPCR bits. Hash mode ATS is currently unsupported so lpcr should be set to 0.

3.4.29 OPAL_READ_NVRAM

```
#define OPAL_READ_NVRAM                         7
```

OPAL_READ_NVRAM call requests OPAL to read the data from system NVRAM memory into a memory buffer. The data at offset from nvram_image will be copied to memory buffer of size size.

Parameters

```
uint64_t buffer
uint64_t size
uint64_t offset
```

buffer the data from nvram will be copied to buffer

size the data of size size will be copied

offset the data will be copied from address equal to base nvram_image plus offset

Return Values

OPAL_SUCCESS data from nvram to memory buffer copied successfully

OPAL_PARAMETER a parameter offset or size was incorrect

OPAL_HARDWARE either nvram is not initialized or permanent error related to nvram hardware.

3.4.30 OPAL_WRITE_NVRAM

```
#define OPAL_WRITE_NVRAM                        8
```

OPAL_WRITE_NVRAM call requests OPAL to write the data to actual system NVRAM memory from memory buffer at offset, of size size

Parameters

```
uint64_t buffer
uint64_t size
uint64_t offset
```

buffer data from buffer will be copied to nvram

size the data of size size will be copied

offset the data will be copied to address which is equal to base nvram_image plus offset

Return Values

OPAL_SUCCESS data from memory buffer to actual nvram_image copied successfully

OPAL_PARAMETER a parameter offset or size was incorrect

OPAL_HARDWARE either nvram is not initialized or permanent error related to nvram hardware.

3.4.31 OPAL_PCI_GET_PHB_DIAG_DATA2

Get PCI diagnostic data from a given PHB

Parameters

uint64_t phb_id the ID of the PHB you want to retrieve data from

void *diag_buffer an allocated buffer to store diag data in

uint64_t diag_buffer_len size in bytes of the diag buffer

Calling

Retrieve the PHB's diagnostic data. The diagnostic data is stored in the buffer pointed by @diag_buffer. Different PHB versions will store different diagnostics, defined in include/opal-api.h as `struct OpalIo<PHBVer>ErrorData`.

`OPAL_PCI_GET_PHB_DIAG_DATA` is deprecated and `OPAL_PCI_GET_PHB_DIAG_DATA2` should be used instead.

Return Codes

OPAL_SUCCESS Diagnostic data has been retrieved and stored successfully

OPAL_PARAMETER The given buffer is too small to store the diagnostic data

OPAL_HARDWARE The PHB is in a broken state and its data cannot be retreived

OPAL_UNSUPPORTED Diagnostic data is not implemented for this PHB type

3.4.32 OPAL_PCI_GET_POWER_STATE

Get PCI slot power state

Parameter

uint64_t id PCI slot ID

uint64_t data memory buffer pointer for power state

Calling

Retrieve PCI slot's power state. The retrieved power state is stored in buffer pointed by @data.

Return Codes

OPAL_SUCCESS PCI slot's power state is retrieved successfully

OPAL_PARAMETER The indicated PCI slot isn't found

OPAL_UNSUPPORTED Power state retrieval not supported on the PCI slot

3.4.33 OPAL_PCI_GET_PRESENCE_STATE

Get PCI slot presence state

Parameters

uint64_t id PCI slot ID

uint64_t data memory buffer pointer for presence state

Calling

Retrieve PCI slot's presence state. The detected presence means there are adapters inserted to the PCI slot. Otherwise, the PCI slot is regarded as an empty one. The typical use is to ensure there are adapters existing before probing the PCI slot in PCI hot add path. The retrieved presence state is stored in buffer pointed by @data.

Return Codes

OPAL_SUCCESS PCI slot's presence state is retrieved successfully

OPAL_PARAMETER The indicated PCI slot isn't found

OPAL_UNSUPPORTED Presence retrieval not supported on the PCI slot

3.4.34 OPAL_PCI_GET_XIVE_REISSUE and OPAL_PCI_SET_XIVE_REISSUE

```
static int64_t opal_pci_get_xive_reissue(uint64_t phb_id __unused,
                                         uint32_t xive_number __unused,
                                         uint8_t *p_bit __unused,
                                         uint8_t *q_bit __unused)

static int64_t opal_pci_set_xive_reissue(uint64_t phb_id __unused,
                                         uint32_t xive_number __unused,
                                         uint8_t p_bit __unused,
                                         uint8_t q_bit __unused)
```

Both of these calls are remnants from previous OPAL versions, calling either of them shall return `OPAL_UNSUPPORTED`.

3.4.35 OPAL_PCI_MAP_PE_DMA_WINDOW

```
#define OPAL_PCI_MAP_PE_DMA_WINDOW           44

static int64_t opal_pci_map_pe_dma_window(uint64_t phb_id,
                                          uint64_t pe_number,
                                          uint16_t window_id,
                                          uint16_t tce_levels,
                                          uint64_t tce_table_addr,
                                          uint64_t tce_table_size,
                                          uint64_t tce_page_size)
```

WARNING: following documentation is from old sources, and is possibly not representative of OPALv3 as implemented by skiboot. This should be used as a starting point for full documentation.

The host calls this function to create a DMA window and map it to a PE. This call returns the address in PCI memory that corresponds to the specified DMA window, which in part may depend on the particular PHB DMA window used. An address that is all zeros in the upper 32 bits reflects a DMA window enabled for 32-bit DMA addresses.

The overall size of the DMA window in PCI memory is determined by the number of tce_levels times the tce_table_size times the tce_page_size.

phb_id is the value from the PHB node `ibm,opal-phbid` property.

dma_window_number specifies the DMA window

For ibm,opal-ioda PHBs the dma_window_number is an index from 0 to the PHB total number of windows minus 1. For ibm,opal-ioda2 PHBs the DMA window_number is an index from 0 to n-1, where n is the number of windows per window set, within the window set associated with the specified PE number.

pe_number is the index of the PE that is authorized to DMA to this window address space in PCI memory,

tce_levels is the number of TCE table levels in the translation hiearchy, from 1 to ibm,opal-dmawins property <translation levels>.

tce_table_addr is the 64-bit system real address of the first level (root, for mult-level) TCE table in the translation hiearchy.

tce_table_size is the size, in bytes, of each TCE table in the translation hierarchy. A value of '0' indicates to disable this DMA window.

For ibm,opal-ioda, this must be a value in the range from 128MB / tce_page_size to 256TB / tce_page_size, and must be in the format and matching a value in the tce_table ranges property that is minimally 256KB for 4K pages.

A particular PE may be mapped to multiple DMA windows, each spanning a DMA window size corresponding to the win_size32 or win_size_64 specified in the ibm,opal-dmawins<> property. However, the TCE table base address must be unique for each window unless it is intended that the same page address in each DMA window is mapped through the same TCE table entry. Generally, when mapping the same PE to multiple DMA windows, so as to create a larger overall DMA window, it is recommended to use consecutive DMA windows and each DMA window should use a TCE table address that is offset by the win_size value of predecessor DMA window.

tce_page_size is the size of PCI memory pages mapped to system real pages through all TCE tables in the translation hierarchy. This must be the same format as and match a value from the ibm,opal-dmawins property <dma-page-sizes>. This page size applies to all TCE tables in the translation hierarchy.

pci_start_addr returns the starting address in PCI memory that corresponds to this DMA window based on the input translation parameter values.

pci_mem_type selects whether this DMA window should be created in 32-bit or 64-bit PCI memory. The input values correspond to the same PCI memory space locators as MMIO spaces in the ranges<> property – 0x2 indicated 32-bit PCI memory and 0x3 indicates 64-bit memory.

Window 0 for both ibm,opal-ioda and ibm,opal-ioda2 PHBs must be within 32-bit PCI memory and this call return opal_parameter for calls that specify window 0 in 64-bit PCI memory.

The DMA win_size property for 32 bit DMA windows limits the number of ibm,opal-ioda PHB windows that can map32-bit address space. For example, with a win_size_32 = 256MB, only 16 DMA windows (and therefore no more than 16 distinct PEs) can map the 4GB of 32-bit PCI memory for DMA. OPAL does not police this limitation.

Return value:

```
if (!phb)
        return OPAL_PARAMETER;
if (!phb->ops->map_pe_dma_window)
        return OPAL_UNSUPPORTED;
```

3.4.36 OPAL_PCI_MAP_PE_DMA_WINDOW_REAL

```
#define OPAL_PCI_MAP_PE_DMA_WINDOW_REAL              45
```

WARNING: following documentation is from old sources, and is possibly not representative of OPALv3 as implemented by skiboot. This should be used as a starting point for full documentation.

The host calls this function to initialize the specified DMA window for untranslated DMA addresses. This allows a PE to DMA directly to system memory without TCE translation. The DMA window PCI memory address is equal to the system memory real address. The PHB passes PCI address bits 04:63 directly to system real address bits 04:63 when PCI address bits 04:39 are within the region specified by mem_addr t0 mem_addr + window_size.

The addresses must be 16MB aligned and a multiple of 16MB in size.

phb_id is the value from the PHB node ibm,opal-phbid property.

dma_window_number specifies the DMA window

For ibm,opal-ioda PHBs the dma_window_number is an index from 0 to the PHB total number of windows minus 1. For ibm,opal-ioda2 PHBs the DMA window_number is an index from 0 to n-1, where n is the number of windows per window set, within the window set associated with the specified PE number.

pe_number is the index of the PE that is authorized to DMA to this window address space in PCI memory,

mem_addr is the starting 64-bit system real address mapped directly to the starting address in PCI memory. Addresses below 4GB are zero in bits above bit 32. This value must be aligned on a 16MB boundary; OPAL returns OPAL_PARAMETER for any value that is not a multiple of 16MB.

window_size is the size, in bytes, of the address range defined by this window. This value must be a multiple of 16MB; OPAL returns OPAL_PARAMETER for any value that is not a multiple of 16MB. A value of '0' indicates to disable this DMA window.

3.4.37 OPAL_PCI_MAP_PE_MMIO_WINDOW

```
#define OPAL_PCI_MAP_PE_MMIO_WINDOW          29

static int64_t opal_pci_map_pe_mmio_window(uint64_t phb_id,
                                           uint64_t pe_number,
                                           uint16_t window_type,
                                           uint16_t window_num,
                                           uint16_t segment_num)
```

WARNING: following documentation is from old sources, and is possibly not representative of OPALv3 as implemented by skiboot. This should be used as a starting point for full documentation.

The host calls this function to map a segment of MMIO address space to a PE.

phb_id is the value from the PHB node ibm,opal-phbid property.

window_type specifies 32-bit or 64-bit PCI memory

'0' selects PCI IO Space. ibm,opal-ioda2 PHBs do not support IO space, and OPAL returns opal_unsupported if called for IO windows.

'1' selects 32-bit PCI memory space

'2' selects 64 bit PCI memory space

window_num is the MMIO window number within the specified PCI memory space

segment_num is an index from 0 to the number of segments minus 1 defined or this window, and selects a particular segment within the specified window.

Return value:

```
if (!phb)
        return OPAL_PARAMETER;
if (!phb->ops->map_pe_mmio_window)
        return OPAL_UNSUPPORTED;
```

3.4.38 OPAL_PCI_PHB_MMIO_ENABLE

```
#define OPAL_PCI_PHB_MMIO_ENABLE            27

static int64_t opal_pci_phb_mmio_enable(uint64_t phb_id, uint16_t window_type,
                                        uint16_t window_num, uint16_t enable)
```

WARNING: following documentation is from old sources, and is possibly not representative of OPALv3 as implemented by skiboot. This should be used as a starting point for full documentation.

The host calls this function to enable or disable PHB decode of the PCI IO and Memory address spaces below that PHB. Window_num selects an mmio window within that address space. Enable set to '1' enables the PHB to decode and forward system real addresses to PCI memory, while enable set to '0' disables PHB decode and forwarding for the address range defined in a particular MMIO window.

Not all PHB hardware may support disabling some or all MMIO windows. OPAL returns OPAL_UNSUPPORTED if called to disable an MMIO window for which hardware does not support disable. KVM may call this function for all MMIO windows and ignore the opal_unsuppsorted return code so long as KVM has disabled MMIO to all downstream PCI devices and assured that KVM and OS guest partitions cannot issue CI loads/stores to these address spaces from the processor (e.g.,via HPT).

OPAL returns OPAL_SUCCESS for calls to OPAL to enable them for PHBs that do not support disable.

phb_id is the value from the PHB node ibm,opal-phbid property.

window_type specifies 32-bit or 64-bit PCI memory

'0' selects PCI IO Space

'1' selects 32-bit PCI memory space

'2' selects 64 bit PCI memory space

window_num is the MMIO window number within the specified PCI memory space

enable specifies to enable or disable this MMIO window.

3.4.39 OPAL_PCI_SET_MVE

```
#define OPAL_PCI_SET_MVE                    33

static int64_t opal_pci_set_mve(uint64_t phb_id, uint32_t mve_number,
                                uint64_t pe_number)
```

WARNING: following documentation is from old sources, and is possibly not representative of OPALv3 as implemented by skiboot. This should be used as a starting point for full documentation.

The host calls this function to bind a PE to an MSI Validation Table Entry (MVE) in the PHB. The MVE compares the MSI requester (RID) to a PE RID, including within the XIVE, to validate that the requester is authorized to signal an interrupt to the associated DMA address for a message value that selects a particular XIVE.

phb_id is the value from the PHB node ibm,opal-phbid property.

mve_number is the index, from 0 to ibm,opal,ibm-num-msi-ports minus1

pe_number is the index of a PE, from 0 to ibm,opal-num-pes minus 1.

This call maps an MVE to a PE and PE RID domain. OPAL uses the PELT to determine the PE domain. OPAL treats this call as a NOP for IODA2 PHBs and returns a status of OPAL_SUCCESS.

Return value:

```
if (!phb)
        return OPAL_PARAMETER;
if (!phb->ops->set_mve)
        return OPAL_UNSUPPORTED;
```

3.4.40 OPAL_PCI_SET_MVE_ENABLE

```
#define OPAL_PCI_SET_MVE_ENABLE                         34

static int64_t opal_pci_set_mve_enable(uint64_t phb_id, uint32_t mve_number,
                                       uint32_t state)

enum OpalMveEnableAction {
     OPAL_DISABLE_MVE = 0,
     OPAL_ENABLE_MVE = 1
};
```

WARNING: following documentation is from old sources, and is possibly not representative of OPALv3 as implemented by skiboot. This should be used as a starting point for full documentation.

The host calls this function to enable or disable an MVE to respond to an MSI DMA address and message data value.

phb_id is the value from the PHB node ibm,opal-phbid property.

mve_number is the index, from 0 to ibm,opal,ibm-num-msi-ports minus1

state A '1' value of the state parameter indicates to enable the MVE and a '0' value indicates to disable the MVE.

This call sets the MVE to an enabled (1) or disabled (0) state.

Return value:

```
if (!phb)
        return OPAL_PARAMETER;
if (!phb->ops->set_mve_enable)
        return OPAL_UNSUPPORTED;
```

3.4.41 OPAL_PCI_SET_P2P

```
#define OPAL_PCI_SET_P2P                        157

int64_t opal_pci_set_p2p(uint64_t phbid_init, uint64_t phbid_target,
                             uint64_t desc, uint16_t pe_number)
```

```
/* PCI p2p descriptor */
#define OPAL_PCI_P2P_ENABLE          0x1
#define OPAL_PCI_P2P_LOAD            0x2
#define OPAL_PCI_P2P_STORE           0x4
```

The host calls this function to enable PCI peer-to-peer on the PHBs.

Parameters

```
uint64_t phbid_init
uint64_t phbid_target
uint64_t desc
uint16_t pe_number
```

phbid_init is the value from the PHB node ibm,opal-phbid property for the device initiating the p2p operation

phbid_target is the value from the PHB node ibm,opal-phbid property for the device targeted by the p2p operation

desc tells whether the p2p operation is a store (OPAL_PCI_P2P_STORE) or load (OPAL_PCI_P2P_LOAD). Can be both. OPAL_PCI_P2P_ENABLE enables/disables the setting

pe_number PE number for the initiating device

Return Values

OPAL_SUCCESS Configuration was successful

OPAL_PARAMETER Invalid PHB or mode parameter

OPAL_UNSUPPORTED Not supported by hardware

3.4.42 OPAL_PCI_SET_PE

```
#define OPAL_PCI_SET_PE                                 31
```

NOTE: The following two paragraphs come from some old documentation and have not been checked for accuracy. Same goes for bus_compare, dev_compare and func_compare documentation. Do *NOT* assume this documentation is correct without checking the source.

A host OS calls this function to map a PCIE function (RID), or range of function bus/dev/funcs (RIDs), to a PHB PE. The bus, device, func, and compare parameters define a range of bus, device, or function numbers to define a range of RIDs within this domain. A value of "7" for the bus_compare, and non-zero for the dev_compare and func_compare, define exactly one function RID to be a PE (within a PE number domain).

This must be called prior to ALL other OPAL calls that take a PE number argument, for OPAL to correlate the RID (bus/dev/func) domain of the PE. If a PE domain is changed, the host must call this to reset the PE bus/dev/func domain and then call all other OPAL calls that map PHB IODA resources to update those domains within PHB facilities.

```
static int64_t opal_pci_set_pe(uint64_t phb_id, uint64_t pe_number,
                               uint64_t bus_dev_func, uint8_t bus_compare,
                               uint8_t dev_compare, uint8_t func_compare,
                               uint8_t pe_action)
```

phb_id is the value from the PHB node ibm,opal-phbid property.

pe_number is the index of a PE, from 0 to ibm,opal-num-pes minus 1.

bus_compare is a value from 0 to 7 indicating which bus number bits define the range of buses in a PE domain:

0 = do not validate against RID bus number (PE = all bus numbers)

2 = compare high order 3 bits of RID bus number to high order 3 bits of PE bus number

3 = compare high order 4 bits of RID bus number to high order 4 bits of PE bus number

6 = compare high order 7 bits of RID bus number to high order 7 bits of PE bus number

7 = compare all bits of RID bus number to all bits of PE bus number

dev_compare indicates to compare the RID device number to the PE device number or not. '0' signifies that the RID device number is not compared – essentially all device numbers within the bus and function number range of this PE are also within this PE. Non-zero signifies to compare the RID device number to the PE device number, such that only that device number is in the PE domain, for all buses and function numbers in the PE domain.

func_compare indicates to compare the RID function number to the PE function number or not. '0' signifies that the RID function number is not compared – essentially all function numbers within the bus and device number range of this PE are also within this PE. Non-zero signifies to compare the RID function number to the PE function number, such that only that function number is in the PE domain, for all buses and device numbers in the PE domain.

pe_action is one of:

```
enum OpalPeAction {
    OPAL_UNMAP_PE = 0,
    OPAL_MAP_PE = 1
};
```

Return value:

OPAL_PARAMETER If one of the following: - invalid phb - invalid pe_action - invalid bus_dev_func - invalid bus_compare

OPAL_UNSUPPORTED PHB does not support set_pe operation

OPAL_SUCCESS if opreation was successful

3.4.43 OPAL_PCI_SET_PELTV

```
#define OPAL_PCI_SET_PELTV                      32
```

WARNING: This documentation comes from an old source and is possibly not up to date with OPALv3. Rely on this documentation only as a starting point, use the source (and update the docs).

```
static int64_t opal_pci_set_peltv(uint64_t phb_id, uint32_t parent_pe,
                                  uint32_t child_pe, uint8_t state)
```

This call sets the PELTV of a parent PE to add or remove a PE number as a PE within that parent PE domain. The host must call this function for each child of a parent PE.

phb_id is the value from the PHB node ibm,opal-phbid property

parent_pe is the PE number of a PE that is higher in the PCI hierarchy to other PEs, such that an error involving this parent PE should cause a collateral PE freeze for PEs below this PE in the PCI hierarchy. For example a

switch upstream bridge is a PE that is parent to PEs reached through that upstream bridge such that an error involving the upstream bridge (e.g, ERR_FATAL) should cause the PHB to freeze all other PEs below that upstream bridge (e.g., a downstream bridge, or devices below a downstream bridge).

child_pe is the PE number of a PE that is lower in the PCI hierarchy than another PE, such that an error involving that other PE should cause a collateral PE freeze for this child PE. For example a device below a downstream bridge of a PCIE switch is a child PE that downstream bridge PE and the upstream bridge PE of that switch – an ERR_Fatal from either bridge should result in a collateral freeze of that device PE.

```
enum OpalPeltvAction {
     OPAL_REMOVE_PE_FROM_DOMAIN = 0,
     OPAL_ADD_PE_TO_DOMAIN = 1
};
```

OPAL Implementation Note: WARNING TODO: *CHECK IF THIS IS CORRECT FOR skiboot:* For ibm,opal-ioda2, OPAL sets the PELTV bit in all RTT entries for the parent PE when the state argument is '1'. OPAL clears the PELTV bit in all RTT entries for the parent PE when the state argument is '0' and setting the child PE bit in the parent PELTV results in an all-zeros value for that PELTV.

Return value:

```
if (!phb)
        return OPAL_PARAMETER;
if (!phb->ops->set_peltv)
        return OPAL_UNSUPPORTED;
```

3.4.44 OPAL_PCI_SET_PHB_MEM_WINDOW

```
#define OPAL_PCI_SET_PHB_MEM_WINDOW             28

static int64_t opal_pci_set_phb_mem_window(uint64_t phb_id,
                                           uint16_t window_type,
                                           uint16_t window_num,
                                           uint64_t addr,
                                           uint64_t pci_addr,
                                           uint64_t size)
```

WARNING: following documentation is from old sources, and is possibly not representative of OPALv3 as implemented by skiboot. This should be used as a starting point for full documentation.

The host calls this function to set the PHB PCI memory window parameters for PHBs. OPAL sets IO space for P7IOC and KVM cannot relocate this. KVM should changes these windows only while all devices below the PHB are disabled for PCI memory ops, and with the target window in disabled state (where supported by PHB hardware).

phb_id is the value from the PHB node ibm,opal-phbid property.

window_type specifies 32-bit or 64-bit PCI memory

'0' selects IO space, and is not supported for relocation. OPAL returns OPAL_UNSUPPORTED for this value.

'1' selects 32-bit PCI memory space

'2' selects 64 bit PCI memory space

window_num is the MMIO window number within the specified PCI memory space

starting_real_address specifies the location within sytsem (processor)real address space this MMIO window starts. This must be a location within the IO Hub or PHB node ibm,opal-mmio-real property.

starting_pci_address specifies the location within PCI 32 or 64-bit address space that this MMIO window starts. For 64-bit PCI memory, this must be within the low order 60 bit (1 Exabyte) region of PCI memory. Addresses above 1EB are reserved to IODA definitions.

segment_size defines the segment size of this window, in the same format as and a matching value from the ibm,opal-memwin32/64 <segment_size> property. The window total size, in bytes, is the segment_size times the ibm,opal-memwin32/64 <num_segments> property and must not extend beyond the ibm,opal-mmio-real property range within system real address space. The total MMIO window size is the segment_size times the num_segments supported for the specifice window. The host must assure that the cumulative address space for all enabled windows does not exceed the total PHB 32-bit or 64-bit real address window space, or extend outside these address ranges, and that no windows overlap each other in real or PCI address space. OPAL does not validate those conditions.

A segment size of '0' indicates to disable this MMIO window. If the PHB hardware does not support disabling a window, OPAL returns OPAL_UNSUPPORTED status.

The size of the system real and PCI memory spaces are equal and defined by segment_size times the number of segments within this MMIO window.

The host must set PHB memory windows to be within the system real address ranges indicated in the PHB parent HDT hub node ibm,opal-mmio-real property.

Return value:

```
if (!phb)
        return OPAL_PARAMETER;
if (!phb->ops->set_phb_mem_window)
        return OPAL_UNSUPPORTED;
```

3.4.45 OPAL_PCI_SET_POWER_STATE

Set PCI slot power state

Parameters

uint64_t async_token Token of asynchronous message to be sent on completion of OPAL_PCI_SLOT_POWER_{OFF, ON}. It is ignored when @data is OPAL_PCI_SLOT_{OFFLINE, ONLINE}.

uint64_t id PCI slot ID

uint64_t data memory buffer pointer for the power state which can be one of OPAL_PCI_SLOT_POWER_{OFF, ON, OFFLINE, ONLINE}.

Calling

Set PCI slot's power state. The power state is stored in buffer pointed by @data. The typical use is to hot add or remove adapters behind the indicated PCI slot (by @id) in PCI hotplug path.

User will receive an asychronous message after calling the API. The message contains the API completion status: event (Power off or on), device node's phandle identifying the PCI slot, errcode (e.g. OPAL_SUCCESS). The API returns OPAL_ASYNC_COMPLETION for the case.

The states OPAL_PCI_SLOT_OFFLINE and OPAL_PCI_SLOT_ONLINE are used for removing or adding devices behind the slot. The device nodes in the device tree are removed or added accordingly, without actually changing the

slot's power state. The API call will return OPAL_SUCCESS immediately and no further asynchronous message will be sent.

Return Codes

OPAL_SUCCESS PCI hotplug on the slot is completed successfully

OPAL_ASYNC_COMPLETION PCI hotplug needs further message to confirm

OPAL_PARAMETER The indicated PCI slot isn't found

OPAL_UNSUPPORTED Setting power state not supported on the PCI slot

3.4.46 OPAL_PCI_SET_XIVE_PE

```
static int64_t opal_pci_set_xive_pe(uint64_t phb_id, uint64_t pe_number,
                                    uint32_t xive_num)
```

WARNING: following documentation is from old sources, and is possibly not representative of OPALv3 as implemented by skiboot. This should be used as a starting point for full documentation.

The host calls this function to bind a PE to an XIVE. Only that PE may then signal an MSI that selects this XIVE.

`phb_id` is the value from the PHB node ibm,opal-phbid property.

`pe_number` is the index of a PE, from 0 to ibm,opal-num-pes minus 1.

`xive_number` is the index, from 0 to ibm,opal,ibm-num-msis minus (num_lsis+1)

This call maps the XIVR indexed by xive_num to the PE specified by pe_number. For ibm,opal-ioda HW, the pe_number must match the pe_number set in the MVE.

Return value:

```
if (!phb)
        return OPAL_PARAMETER;
if (!phb->ops->set_xive_pe)
        return OPAL_UNSUPPORTED;
```

3.4.47 OPAL_PCI_TCE_KILL

```
int64_t opal_pci_tce_kill(uint64_t phb_id,
                          uint32_t kill_type,
                          uint64_t pe_number,
                          uint32_t tce_size,
                          uint64_t dma_addr,
                          uint32_t npages)
```

An abstraction around TCE kill. This allows host OS kernels to use an OPAL call if they don't know the model specific invalidation method.

Where kill_type is one of:

```
enum {
   OPAL_PCI_TCE_KILL_PAGES,
   OPAL_PCI_TCE_KILL_PE,
```

```
   OPAL_PCI_TCE_KILL_ALL,
};
```

Not all PHB types currently support this abstraction. It is supported in PHB4, which means from POWER9 onwards it will be present.

Returns

OPAL_PARAMETER if phb_id is invalid (or similar)

OPAL_UNSUPPORTED if PHB model doesn't support this call. This is likely true for systems before POWER9/PHB4. Do *NOT* rely on this call existing for systems prior to POWER9 (i.e. PHB4).

Example code (from linux/arch/powerpc/platforms/powernv/pci-ioda.c)

```
static inline void pnv_pci_ioda2_tce_invalidate_pe(struct pnv_ioda_pe *pe)
{
      struct pnv_phb *phb = pe->phb;

      if (phb->model == PNV_PHB_MODEL_PHB3 && phb->regs)
         pnv_pci_phb3_tce_invalidate_pe(pe);
      else
         opal_pci_tce_kill(phb->opal_id, OPAL_PCI_TCE_KILL_PE,
                           pe->pe_number, 0, 0, 0);
}
```

and

```
struct pnv_phb *phb = pe->phb;
unsigned int shift = tbl->it_page_shift;
if (phb->model == PNV_PHB_MODEL_PHB3 && phb->regs)
      pnv_pci_phb3_tce_invalidate(pe, rm, shift,
                                  index, npages);
else
      opal_pci_tce_kill(phb->opal_id,
                        OPAL_PCI_TCE_KILL_PAGES,
                        pe->pe_number, 1u << shift,
                        index << shift, npages);
```

3.4.48 OPAL_POLL_EVENTS

Poll for outstanding events.

Fills in a bitmask of pending events.

Current events are:

OPAL_EVENT_OPAL_INTERNAL = 0x1

Currently unused.

OPAL_EVENT_NVRAM = 0x2

Unused

OPAL_EVENT_RTC = 0x4

TODO: clean this up, this is just copied from hw/fsp/fsp-rtc.c:

```
* Because the RTC calls can be pretty slow, these functions will shoot
* an asynchronous request to the FSP (if none is already pending)
*
* The requests will return OPAL_BUSY_EVENT as long as the event has
* not been completed.
*
* WARNING: An attempt at doing an RTC write while one is already pending
* will simply ignore the new arguments and continue returning
* OPAL_BUSY_EVENT. This is to be compatible with existing Linux code.
*
* Completion of the request will result in an event OPAL_EVENT_RTC
* being signaled, which will remain raised until a corresponding call
* to opal_rtc_read() or opal_rtc_write() finally returns OPAL_SUCCESS,
* at which point the operation is complete and the event cleared.
*
* If we end up taking longer than rtc_read_timeout_ms millieconds waiting
* for the response from a read request, we simply return a cached value (plus
* an offset calculated from the timebase. When the read request finally
* returns, we update our cache value accordingly.
*
* There is two separate set of state for reads and writes. If both are
* attempted at the same time, the event bit will remain set as long as either
* of the two has a pending event to signal.
```

OPAL_EVENT_CONSOLE_OUTPUT = 0x8

TODO

OPAL_EVENT_CONSOLE_INPUT = 0x10

TODO

OPAL_EVENT_ERROR_LOG_AVAIL = 0x20

TODO

OPAL_EVENT_ERROR_LOG = 0x40

TODO

OPAL_EVENT_EPOW = 0x80

TODO

OPAL_EVENT_LED_STATUS = 0x100

TODO

OPAL_EVENT_PCI_ERROR = 0x200

TODO

OPAL_EVENT_DUMP_AVAIL = 0x400

Signifies that there is a pending system dump available. See OPAL_DUMP suite of calls for details.

```
OPAL_EVENT_MSG_PENDING = 0x800,
```

3.4.49 OPAL_GET_POWER_SHIFT_RATIO

OPAL call to read the power-shifting-ratio using a handle to identify the type (e.g CPU vs. GPU, CPU vs. MEM) which is exported via device-tree.

The call can be asynchronus, where the token parameter is used to wait for the completion.

Parameters

u32	handle
int	token
u32	*ratio

Returns

OPAL_SUCCESS Success

OPAL_PARAMETER Invalid ratio pointer

OPAL_UNSUPPORTED No support for reading psr

OPAL_HARDWARE Unable to procced due to the current hardware state

OPAL_ASYNC_COMPLETION Request was sent and an async completion message will be sent with token and status of the request.

3.4.50 OPAL_SET_POWER_SHIFT_RATIO

OPAL call to set power-shifting-ratio using a handle to identify the type of PSR which is exported in device-tree. This call can be asynchronus where the token parameter is used to wait for the completion.

Parameters

:: u32 handle int token u32 ratio

Returns

OPAL_SUCCESS Success

OPAL_PARAMETER Invalid ratio requested

OPAL_UNSUPPORTED No support for changing the ratio

OPAL_PERMISSION Hardware cannot take the request

OPAL_ASYNC_COMPLETION Request was sent and an async completion message will be sent with token and status of the request.

OPAL_HARDWARE Unable to procced due to the current hardware state

OPAL_BUSY Previous request in progress

OPAL_INTERNAL_ERROR Error in request response

OPAL_TIMEOUT Timeout in request completion

3.4.51 OPAL_GET_POWERCAP

The OPAL_GET_POWERCAP call retreives current information on the power cap.

For each entity that can be power capped, the device tree binding indicates what handle should be passed for each of the power cap properties (minimum possible, maximum possible, current powercap).

The current power cap must be between the minimium possible and maximum possible power cap. The minimum and maximum values are dynamic to allow for them possibly being changed by other factors or entities (e.g. service processor).

The call can be asynchronus, where the token parameter is used to wait for the completion.

Parameters

u32	handle
int	token
u32	*pcap

Returns

OPAL_SUCCESS Success

OPAL_PARAMETER Invalid pcap pointer

OPAL_UNSUPPORTED No support for reading powercap sensor

OPAL_HARDWARE Unable to procced due to the current hardware state

OPAL_ASYNC_COMPLETION Request was sent and an async completion message will be sent with token and status of the request.

3.4.52 OPAL_SET_POWERCAP

The OPAL_SET_POWERCAP call sets a power cap.

For each entity that can be power capped, the device tree binding indicates what handle should be passed for each of the power cap properties (minimum possible, maximum possible, current powercap).

The current power cap must be between the minimium possible and maximum possible power cap.

You cannot currently set the minimum or maximum power cap, and thus OPAL_PERMISSION will be returned if it is attempted to set. In the future, this may change - but for now, the correct behaviour for an Operating System is to not attempt to set them.

Parameters

:: u32 handle int token u32 pcap

Returns

OPAL_SUCCESS Success

OPAL_PARAMETER Invalid powercap requested beyond powercap limits

OPAL_UNSUPPORTED No support for changing the powercap

OPAL_PERMISSION Hardware cannot take the request

OPAL_ASYNC_COMPLETION Request was sent and an async completion message will be sent with token and status of the request.

OPAL_HARDWARE Unable to procced due to the current hardware state

OPAL_BUSY Previous request in progress

OPAL_INTERNAL_ERROR Error in request response

OPAL_TIMEOUT Timeout in request completion

3.4.53 OPAL_PRD_MSG

The OPAL_PRD_MSG call is used to pass a struct opal_prd_msg from the HBRT code into opal, and is paired with the OPAL_PRD_MSG message type.

Parameters

`struct opal_msg *msg` Passes an opal_msg, of type OPAL_PRD_MSG, from the OS to OPAL.

3.4.54 OPAL_QUIESCE

The host OS can use OPAL_QUIESCE to ensure CPUs under host control are not executing OPAL. This is useful in crash or shutdown scenarios to try to ensure that CPUs are not holding locks, and is intended to be used with OPAL_SIGNAL_SYSTEM_RESET, for example.

Arguments

```
uint32_t quiesce_type
  QUIESCE_HOLD          Wait for all target(s) currently executing OPAL to
                        return to the host. Any new OPAL call that is made
                        will be held off until QUIESCE_RESUME.
  QUIESCE_REJECT        Wait for all target(s) currently executing OPAL to
                        return to the host. Any new OPAL call that is made
                        will fail with OPAL_BUSY until QUIESCE_RESUME.
  QUIESCE_LOCK_BREAK    After QUIESCE_HOLD or QUIESCE_REJECT is successful,
                        the CPU can call QUIESCE_LOCK_BREAK to skip all
                        locking in OPAL to give the best chance of making
                        progress in the crash/debug paths. The host should
                        ensure all other CPUs are stopped (e.g., with
                        OPAL_SIGNAL_SYSTEM_RESET) before this call is made, to
                        avoid concurrency.
  QUIESCE_RESUME        Undo the effects of QUIESCE_HOLD/QUIESCE_REJECT and
                        QUIESCE_LOCK_BREAK calls.
  QUIESCE_RESUME_FAST_REBOOT
                        As above, but also reset the tracking of OS calls
                        into firmware as part of fast reboot (secondaries
                        will never return to OS, but instead be released
                        into a new OS boot).

int32_t target_cpu
  cpu_nr >= 0         The cpu server number of the target cpu to reset.
  -1                  All cpus except the current one should be quiesced.
```

Returns

OPAL_SUCCESS The quiesce call was successful.

OPAL_PARTIAL Some or all of the CPUs executing OPAL when the call was made did not return to the host after a timeout of 1 second. This is a best effort at quiescing OPAL, and QUIESCE_RESUME must be called to resume normal firmware operation.

OPAL_PARAMETER A parameter was incorrect.

OPAL_BUSY This CPU was not able to complete the operation, either because another has concurrently started quiescing the system, or because it has not successfully called QUIESCE_HOLD or QUIESCE_REJECT before attempting QUIESCE_LOCK_BREAK or QUIESCE_RESUME.

3.4.55 OPAL_READ_TPO and OPAL_WRITE_TPO

TPO is a Timed Power On facility.

It is an OPTIONAL part of the OPAL spec.

If a platform supports Timed Power On (TPO), the RTC node in the device tree (itself under the "ibm,opal" node will have the has-tpo property:

```
rtc {
   compatible = "ibm,opal-rtc";
   has-tpo;
};
```

If the "has-tpo" proprety is *NOT* present then OPAL does *NOT* support TPO.

3.4.56 OPAL_REGISTER_DUMP_REGION

This call is used to register regions of memory for a service processor to capture when the host crashes.

e.g. if an assert is hit in OPAL, a service processor will copy

This is an OPTIONAL feature that may be unsupported, the host OS should use an OPAL_CHECK_TOKEN call to find out if OPAL_REGISTER_DUMP_REGION is supported.

OPAL_REGISTER_DUMP_REGION accepts 3 parameters:

- region ID
- address
- length

There is a range of region IDs that can be used by the host OS. A host OS should start from OPAL_DUMP_REGION_HOST_END and work down if it wants to add a not well defined region to dump. Currently the only well defined region is for the host OS log buffer (e.g. dmesg on linux).

```
/*
 * Dump region ID range usable by the OS
 */
 #define OPAL_DUMP_REGION_HOST_START          0x80
 #define OPAL_DUMP_REGION_LOG_BUF             0x80
 #define OPAL_DUMP_REGION_HOST_END            0xFF
```

OPAL_REGISTER_DUMP_REGION will return OPAL_UNSUPPORTED if the call is present but the system doesn't support registering regions to be dumped.

In the event of being passed an invalid region ID, OPAL_REGISTER_DUMP_REGION will return OPAL_PARAMETER.

Systems likely have a limit as to how many regions they can support being dumped. If this limit is reached, OPAL_REGISTER_DUMP_REGION will return OPAL_INTERNAL_ERROR.

BUGS

Some skiboot versions incorrectly returned OPAL_SUCCESS in the case of OPAL_REGISTER_DUMP_REGION being supported on a platform (so the call was present) but the call being unsupported for some reason (e.g. on an IBM POWER7 machine).

See also: OPAL_UNREGISTER_DUMP_REGION

3.4.57 OPAL_REINIT_CPUS

```
static int64_t opal_reinit_cpus(uint64_t flags);
```

This OPAL call reinitializes some bit of CPU state across *ALL* CPUs. Consequently, all CPUs must be in OPAL for this call to succeed (either at boot time or after OPAL_RETURN_CPU is called)

Arguments

Currently, possible flags are:

```
enum {
      OPAL_REINIT_CPUS_HILE_BE         = (1 << 0),
      OPAL_REINIT_CPUS_HILE_LE         = (1 << 1),
      OPAL_REINIT_CPUS_MMU_HASH        = (1 << 2),
      OPAL_REINIT_CPUS_MMU_RADIX       = (1 << 3),
      OPAL_REINIT_CPUS_TM_SUSPEND_DISABLED = (1 << 4),
};
```

Extra flags may be added in the future, so other bits *must* be 0.

On POWER7 CPUs, only OPAL_REINIT_CPUS_HILE_BE is supported. All other flags will return OPAL_UNSUPPORTED.

On POWER8 CPUs, only OPAL_REINIT_CPUS_HILE_BE and OPAL_REINIT_CPUS_HILE_LE are support and other bits *MUST NOT* be set.

On POWER9 CPUs, all options including OPAL_REINIT_CPUS_MMU_HASH and OPAL_REINIT_CPUS_MMU_RADIX.

OPAL_REINIT_CPUS_TM_SUSPEND_DISABLED

This flag requests that CPUs be configured with TM (Transactional Memory) suspend mode disabled. This may only be supported on some CPU versions.

Returns

OPAL_SUCCESS Success!

OPAL_UNSUPPORTED Processor does not suport reinit flags.

3.4.58 OPAL_RETURN_CPU

```
int64_t opal_return_cpu(void);
```

When OPAL first starts the host, all secondary CPUs are spinning in OPAL. To start them, one must call OPAL_START_CPU (you may want to OPAL_REINIT_CPUS to set the HILE bit first).

In cases where you need OPAL to do something for you across all CPUs, such as OPAL_REINIT_CPUS, (on some platforms) a firmware update or get the machine back into a similar state as to when the host OS was started (e.g. for kexec) you may also need to return control of the CPU to OPAL.

Returns

This call does **not return**. You need to OPAL_START_CPU.

3.4.59 OPAL_RTC_READ

Read the Real Time Clock.

Parameters

uint32_t* year_month_day the year, month and day formatted as follows:

- bits 0-15 is bcd formatted year (0100-9999)
- bits 16-23 is bcd formatted month (01-12)
- bits 24-31 is bcd formatted day (01-31)

uint64_t* hour_minute_second_millisecond the hour, minute, second and millisecond formatted as follows:

- bits 0-16 is reserved
- bits 17-24 is bcd formatted hour (00-23)
- bits 25-31 is bcd formatted minute (00-59)
- bits 32-39 is bcd formatted second (00-60)
- bits 40-63 is bcd formatted milliseconds (000000-999999)

Calling

Since RTC calls can be pretty slow, OPAL_RTC_READ is likely to first return OPAL_BUSY_EVENT, requiring the caller to wait until the OPAL_EVENT_RTC event has been signaled. Once the event has been signaled, a subsequent OPAL_RTC_READ call will retrieve the time. Since the OPAL_EVENT_RTC event is used for both reading and writing the RTC, callers must be able to handle the event being signaled for a concurrent in flight OPAL_RTC_WRITE rather than this read request.

The following code is one way to correctly issue and then wait for a response:

```
int rc = OPAL_BUSY_EVENT;
while (rc == OPAL_BUSY_EVENT) {
      rc = opal_rtc_read(&y_m_d, &h_m_s_ms);
      if (rc == OPAL_BUSY_EVENT)
         opal_poll_events(NULL);
}
```

Although as of writing all OPAL_RTC_READ backends are asynchronous, there is no requirement for them to be - it is valid for OPAL_RTC_READ to immediately return the retreived value rather than OPAL_BUSY_EVENT.

TODO: describe/document format of arguments.

Return codes

OPAL_SUCCESS parameters now contain the current time, or one read from cache.

OPAL_HARDWARE error in retrieving the time. May be transient error, may be permanent.

OPAL_PARAMETER year_month_day or hour_minute_second_millisecond parameters are NULL

OPAL_INTERNAL_ERROR something went wrong, Possibly reported in error log.

OPAL_BUSY_EVENT request is in flight

3.4.60 OPAL_RTC_WRITE

OPAL_RTC_WRITE is much like OPAL_RTC_READ in that it can be asynchronous.

If multiple WRITES are issued before the first one completes, subsequent writes are ignored. There can only be one write in flight at any one time.

Format of the time is the same as for OPAL_RTC_READ.

3.4.61 OPAL_SENSOR_GROUP_CLEAR

OPAL call to clear the sensor groups data using a handle to identify the type of sensor group which is exported via DT.

The call can be asynchronus, where the token parameter is used to wait for the completion.

Parameters

:: u32 handle int token

Returns

OPAL_SUCCESS Success

OPAL_UNSUPPORTED No support for clearing the sensor group

OPAL_HARDWARE Unable to procced due to the current hardware state

OPAL_PERMISSION Hardware cannot take the request

OPAL_ASYNC_COMPLETION Request was sent and an async completion message will be sent with token and status of the request.

OPAL_BUSY Previous request in progress

OPAL_INTERNAL_ERROR Error in request response

OPAL_TIMEOUT Timeout in request completion

3.4.62 OPAL_SENSOR_READ

The OPAL sensor call reads a sensor data using a unique handler to identity the targeted sensor. The *sensor_handler* is provided via the device tree and is opaque to the OS (although we currently do use an encoding scheme).

This call can be asynchronous, when a message needs to be sent to a service processor for example. In this case, the call will return OPAL_ASYNC_COMPLETION and the token parameter will be used to wait for the completion of the request.

The OPAL API doesn't enforce alimit on the number of sensor calls that can be in flight.

Parameters

```
uint32_t sensor_handler
int      token
uint32_t *sensor_data
```

Return values

OPAL_SUCCESS Success!

OPAL_PARAMETER invalid sensor handler

OPAL_UNSUPPORTED platform does not support reading sensors.

Some sensors may have to be read asynchronously (e.g. because OPAL must communicate with a service processor). One example is sensors provided by the FSP on IBM FSP systems.

OPAL_ASYNC_COMPLETION a request was sent and an async completion will be triggered with the @token argument

OPAL_PARTIAL the request completed but the data returned is invalid

OPAL_BUSY_EVENT a previous request is still pending

OPAL_NO_MEM allocation failed

OPAL_INTERNAL_ERROR communication failure with the FSP

OPAL_HARDWARE FSP is not available

3.4.63 OPAL_SET_XIVE

```
#define OPAL_SET_XIVE                                   19
```

The host calls this function to set the server (target processor) and priority parameters of an interrupt source.

This can be also used to mask or unmask the interrupt (by changing the priority to 0xff one masks an interrupt).

WARNINGS:

- For MSIs or generally edge sensitive interrupts, OPAL provides no guarantee as to whether the interrupt will be latched if it occurs while masked and replayed on unmask. It may or may not. The OS needs to be aware of this. The current implementation will *not* replay, neither on P8 nor on P9 XICS emulation.
- When masking, there is no guarantee that the interrupt will not still occur after this call returns. The reason is that it might already be on its way past the source controller and latched into one of the presenters. There is however a guarantee that it won't replay indefinitely so it's acceptable for the OS to simply ignore it.

Parameters

`isn` This is a global interrupt number as obtained from the device-tree "interrupts" or "interrupt-map" properties.

`server_number` is the mangled server (processor) that is to receive the interrupt request. The mangling means that the actual processor number is shifted left by 2 bits, the bottom bits representing the "link". However links aren't supported in OPAL so the bottom 2 bits should be 0.

`priority` is the interrupt priority value applied to the interrupt (0=highest, 0xFF = lowest/disabled).

3.4.64 OPAL_SIGNAL_SYSTEM_RESET

```
int64_t signal_system_reset(int32_t cpu_nr);
```

This OPAL call causes the specified cpu(s) to be reset to the system reset exception handler (0x100).

The SRR1 register will indicate a power-saving wakeup when appropriate, and the wake reason will be System Reset (see Power ISA).

This interrupt may not be recoverable in some cases (e.g., if it is raised when the target has MSR[RI]=0), so it should not be used in normal operation, but only for crashing, debugging, and similar exceptional cases.

OPAL_SIGNAL_SYSTEM_RESET can pull CPUs out of OPAL, which may be undesirable in a crash or shutdown situation (e.g., because they may hold locks which are required to access the console, or may be halfway through setting hardware registers), so OPAL_QUIESCE can be used before OPAL_SIGNAL_SYSTEM_RESET to (attempt to) ensure all CPUs are out of OPAL before being interrupted.

Arguments

```
int32_t cpu_nr
  cpu_nr >= 0        The cpu server number of the target cpu to reset.
  SYS_RESET_ALL (-1) All cpus should be reset.
  SYS_RESET_ALL_OTHERS (-2) All but the current cpu should be reset.
```

Returns

OPAL_SUCCESS The system reset requests to target CPU(s) was successful. This returns asynchronously without acknowledgement from targets that system reset interrupt processing has completed or even started.

OPAL_PARAMETER A parameter was incorrect.

OPAL_HARDWARE Hardware indicated failure during reset, some or all of the target CPUs may have the system reset delivered.

OPAL_CONSTRAINED Platform does not support broadcast operations.

OPAL_PARTIAL Platform can not reset sibling threads on the same core as requested. None of the specified CPUs are reset in this case.

OPAL_UNSUPPORTED This processor/platform is not supported.

3.4.65 OPAL_SLW_SET_REG

```
int64_t opal_slw_set_reg(uint64_t cpu_pir, uint64_t sprn, uint64_t val)
```

OPAL_SLW_SET_REG is used to inform low-level firmware to restore a given value of SPR when there is a state loss. The actual set of SPR that is supported is platform dependent.

In Power 8, it uses p8_pore_gen_cpufreq_fixed(), api provided by pore engine, to inform the spr with their corresponding values with which they must be restored.

In Power 9, it uses p9_stop_save_cpureg(), api provided by self restore code, to inform the spr with their corresponding values with which they must be restored.

Parameters

`uint64_t cpu_pir` This parameter specifies the pir of the cpu for which the call is being made.

`uint64_t sprn` This parameter specifies the spr number as mentioned in p9_stop_api.H for Power9 and p8_pore_table_gen_api.H for Power8.

`uint64_t val` This parameter specifices value with which the spr should be restored.

Returns

`OPAL_INTERNAL_ERROR` On failure. The actual error code from the platform specific code is logged in the OPAL logs

`OPAL_UNSUPPORTED` In power8 only, if spr restore is not supported by pore engine.

`OPAL_SUCCESS` On success

3.4.66 OPAL_SYNC_HOST_REBOOT

```
static int64_t opal_sync_host_reboot(void)
```

This OPAL call halts asynchronous operations in preparation for something like kexec. It will halt DMA as well notification of some events (such as a new error log being available for retreival).

It's meant to be called in a loop until OPAL_SUCCESS is returned.

Returns

OPAL_SUCCESS Success!

OPAL_BUSY_EVENT not yet complete, call opal_sync_host_reboot() again, possibly with a short delay.

OPAL_BUSY Call opal_poll_events() and then retry opal_sync_host_reboot

3.4.67 OPAL_TEST

OPAL_TEST is a REQUIRED call for OPAL and conforming implementations MUST have it.

It is designed to test basic OPAL call functionality.

Token:

```
#define OPAL_TEST                               0
```

Arguments

```
uint64_t     arg
```

Returns

```
0xfeedf00d
```

Function

OPAL_TEST MAY print a string to the OPAL log with the value of argument.

For example, the reference implementation (skiboot) implements OPAL_TEST as:

```
static uint64_t opal_test_func(uint64_t arg)
{
      printf("OPAL: Test function called with arg 0x%llx\n", arg);

      return 0xfeedf00d;
}
```

3.4.68 OPAL_UNREGISTER_DUMP_REGION

While OPAL_REGISTER_DUMP_REGION registers a region, OPAL_UNREGISTER_DUMP_REGION will unregister a region by region ID.

OPAL_UNREGISTER_DUMP_REGION takes one argument: the region ID.

A host OS should check OPAL_UNREGISTER_DUMP_REGION is supported through a call to OPAL_CHECK_TOKEN.

If OPAL_UNREGISTER_DUMP_REGION is called on a system where the call is present but unsupported, it will return OPAL_UNSUPPORTED.

BUGS

Some skiboot versions incorrectly returned OPAL_SUCCESS in the case of OPAL_UNREGISTER_DUMP_REGION being supported on a platform (so the call was present) but the call being unsupported for some reason (e.g. on an IBM POWER7 machine).

3.4.69 OPAL_XSCOM_READ and OPAL_XSCOM_WRITE

These low level calls will read/write XSCOM values directly.

They should only be used by low level manufacturing/debug tools. "Normal" host OS kernel code should not know about XSCOM.

each takes three parameters:

```
int xscom_read(uint32_t partid, uint64_t pcb_addr, uint64_t *val)
int xscom_write(uint32_t partid, uint64_t pcb_addr, uint64_t val)
```

Returns

OPAL_SUCCESS Success!

OPAL_HARDWARE if operation failed

OPAL_WRONG_STATE if CPU is asleep

3.4.70 POWER9 Changes to OPAL API

This document is a summary of POWER9 changes to the OPAL API over what it was for POWER7 and POWER8. As the POWER series of processors (at least up to POWER9) require changes in the hypervisor to work on a new processor generation, this gives us an opportunity with POWER9 to clean up several parts of the OPAL API.

Eventually, when the kernel drops support for POWER8 and before, we can then remove the associated kernel code too.

OPAL_REINIT_CPUS

Can now be extended beyond HILE BE/LE bits. If invalid flags are set on POWER9, OPAL_UNSUPPORTED will be returned.

Device Tree

- `/ibm,opal/` compatible property now just lists `ibm,opal-v3` and no longer `ibm,opal-v2` (power9 and above only)
- Rename `fsp-ipl-side` as `sp-ipl-side` in `/ipl-params`

TODO

Things we still have to do for POWER9:

- PCI to use async API rather than returning delays
- deprecate/remove v1 APIs where there's a V2
- Fix this FWTS warning:

```
FAILED [MEDIUM] DeviceTreeBaseDTCWarnings: Test 3, dtc reports warnings from
device tree: Warning (reg_format): "reg" property in /ibm,opal/flash@0 has
invalid length (8 bytes) (#address-cells == 0, #size-cells == 0)
```

- Remove mi-version / ml-version from `/ibm,opal/firmware` and replace with something better and more portable

3.4.71 OPAL API Return Codes

All OPAL calls return an integer relaying the success/failure of the OPAL call.

Success is typically indicated by OPAL_SUCCESS. Failure is always indicated by a negative return code.

Conforming host Operating Systems MUST handle return codes other than those listed here. In future OPAL versions, additional return codes may be added.

In the reference implementation (skiboot) these are all in include/opal.h.

The core set of return codes are:

OPAL_SUCCESS

```
#define OPAL_SUCCESS            0
```

Success!

OPAL_PARAMETER

```
#define OPAL_PARAMETER          -1
```

A parameter was invalid. This will also be returned if you call an invalid OPAL call. To determine if a specific OPAL call is supported or not, OPAL_CHECK_TOKEN should be called rather than relying on OPAL_PARAMETER being returned for an invalid token.

OPAL_BUSY

```
#define OPAL_BUSY               -2
```

Try again later. Related to *OPAL_BUSY_EVENT*, but *OPAL_BUSY* indicates that the caller need not call *OPAL_POLL_EVENTS* itself. **TODO** Clarify current situation.

OPAL_PARTIAL

```
#define OPAL_PARTIAL            -3
```

The operation partially succeeded.

OPAL_CONSTRAINED

```
#define OPAL_CONSTRAINED        -4
```

FIXME

OPAL_CLOSED

```
#define OPAL_CLOSED             -5
```

FIXME document these

OPAL_HARDWARE

```
#define OPAL_HARDWARE                   -6
```

FIXME document these

OPAL_UNSUPPORTED

```
#define OPAL_UNSUPPORTED      -7
```

Unsupported operation. Non-fatal.

OPAL_PERMISSION

```
#define OPAL_PERMISSION               -8
```

Inadequate permission to perform the operation.

OPAL_NO_MEM

```
#define OPAL_NO_MEM           -9
```

Indicates a temporary or permanent lack of adequate memory to perform the operation. Ideally, this should never happen. Skiboot reserves a small amount of memory for its heap and some operations (such as I2C requests) are allocated from this heap.

If this is ever hit, you should likely file a bug.

OPAL_RESOURCE

```
#define OPAL_RESOURCE                 -10
```

FIXME

OPAL_INTERNAL_ERROR

```
#define OPAL_INTERNAL_ERROR   -11
```

FIXME

OPAL_BUSY_EVENT

```
#define OPAL_BUSY_EVENT               -12
```

The same as *OPAL_BUSY* but signals that the OS should call *OPAL_POLL_EVENTS* as that may be required to get into a state where the call will succeed.

OPAL_HARDWARE_FROZEN

```
#define OPAL_HARDWARE_FROZEN -13
```

OPAL_WRONG_STATE

```
#define OPAL_WRONG_STATE      -14
```

OPAL_ASYNC_COMPLETION

```
#define OPAL_ASYNC_COMPLETION         -15
```

For asynchronous calls, successfully queueing/starting executing the command is indicated by the OPAL_ASYNC_COMPLETION return code. pseudo-code for an async call:

```
token = opal_async_get_token();
rc = opal_async_example(foo, token);
if (rc != OPAL_ASYNC_COMPLETION)
    handle_error(rc);
rc = opal_async_wait(token);
// handle result here
```

OPAL_EMPTY

```
#define OPAL_EMPTY            -16
```

I2C Calls

Added for I2C, only applicable to I2C calls:

```
#define OPAL_I2C_TIMEOUT      -17
#define OPAL_I2C_INVALID_CMD  -18
#define OPAL_I2C_LBUS_PARITY  -19
#define OPAL_I2C_BKEND_OVERRUN        -20
#define OPAL_I2C_BKEND_ACCESS -21
#define OPAL_I2C_ARBT_LOST    -22
#define OPAL_I2C_NACK_RCVD    -23
#define OPAL_I2C_STOP_ERR     -24
```

CHAPTER

FOUR

SKIBOOT RELEASE NOTES

4.1 Release Notes

4.1.1 skiboot-5.1.0

skiboot-5.1.0 was released on August 17th, 2015.

skiboot-5.1.0 is the first stable release of 5.1.0 following two beta releases. This new stable release replaces skiboot-5.0 as the current stable skiboot release (5.0 was released April 14th 2015).

Skiboot 5.1.0 contains all fixes from skiboot-5.0 stable branch up to skiboot-5.0.5 and everything from 5.1.0-beta1 and 5.1.0-beta2.

Over skiboot-5.1.0-beta2, we have the following changes:

- opal_prd now supports multiple socket systems
- fix compiler warnings in gard and libflash

Below are the changes introduced in previous skiboot-5.1.0 releases over the previous stable release, skiboot-5.0:

New features

- Add Naples chip (CPU, PHB, LPC serial interrupts) support
- Added qemu platform
- improvements to FSI error handling
- improvements in chip TOD failover (some only on FSP systems)
- Set Relative Priority Register (RPR) to recommended value
  - this affects thread priority in SMT modes
- greatly reduce memory consumption by CPU stacks for non-present CPUs
  - Previously we would reserve enough memory for max PIR for each CPU type.
  - This fix frees up 77MB of RAM on a typical P8 system.
- increased OPAL API documentation
- Asynchronous preloading of resources from FSP/flash
  - improves boot time on some systems
- Basic Garrison platform support
- Add Mambo platform (P8 Functional Simulator, systemsim)

  - includes fake NVRAM, RTC
- Support building with GCOV, increasing memory for skiboot binary to 2MB
  - includes boot code coverage testing
- Increased skiboot HEAP size.
  - We are not aware of any system where you would run out, but on large systems it was getting closer than we liked.
- add boot_tests.sh for helping automate boot testing on FSP and BMC machines
- Versioning of pflash and gard utilities to help Linux (or other OS) distributions with packaging.
- OCC throttle status messages to host
- CAPP timebase sync ("ibm,capp-timebase-sync" in DT to indicate CAPP timebase was synced by OPAL)
- opal-api: Add OPAL call to handle abnormal reboots.

`OPAL_CEC_REBOOT2` currently supports two reboot types:

0. normal reboot, that will behave similar to that of opal_cec_reboot() call
1. platform error reboot.

Long term, this is designed to replace OPAL_CEC_REBOOT.

New features for FSP based machines

- in-band IPMI support
- ethernet adaptor location codes
- add DIMM frequency information to device tree
- improvements in FSP error log code paths
- fix some boot time memory leaks
  - harmless to end user

New features for AMI BMC based machines

- PCIe power workaround for K80
- Added support for Macronix 128Mbit flash chips
- Initial PRD support for Firestone platform
- improved reliability when BMC reboots

The following bugs have been fixed

- Increase PHB3 timeout for electrical links coming up to 2 seconds.
  - fixes issues with some Mellanox cards
- Hang in opal_reinit_cpus() that could prevent kdump from functioning
- PHB3: fix crash in phb3_init
- PHB3: fix crash with fenced PHB in phb3_init_hw()

- Fix bugs in hw/bt.c (interface for IPMI on BMC machines) that could possibly lead to a crash (dereferencing invalid address, deadlock)
- ipmi/sel: fix use-after-free
- Bug fixes in EEH handling
  - opal_pci_next_error() cleared OPAL_EVENT_PCI_ERROR unconditionally, possibly leading to missed errors.
- external/opal-prd: Only map each PRD range once
  - could eventually lead to failing to map PRD ranges
- On skiboot crash, don't try to print symbol when we didn't find one
  - makes backtrace prettier
- On skiboot crash, dump hssr0 and hsrr1 registers correctly.
- Better support old and biarch compilers
  - test "new" compiler flags before using them
  - Specify -mabi=elfv1 if supported (which means it's needed)
- fix boot-coverage-report makefile target
- ipmi: Fix the opal_ipmi_recv() call to handle the error path
- Could make kernel a sad panda when in continues with other IPMI commands
- IPMI: truncate SELs at 2kb
  - it's the limit of the astbmc. We think.
- IPMI/SEL/PEL:
  - As per PEL spec, we should log events with severity >= 0x22 and "service action flag" is "on". But in our case, all logs OPAL originagted logs are makred as report externally. We now only report logs with severity >= 0x22
- IPMI: fixes to eSEL logging
- hw/phb3: Change reserved PE to 255
  - Currently, we have reserved PE#0 to which all RIDs are mapped prior to PE assignment request from kernel. The last M64 BAR is configured to have shared mode. So we have to cut off the first M64 segment, which corresponds to reserved PE#0 in kernel. If the first BAR (for example PF's IOV BAR) requires huge alignment in kernel, we have to waste huge M64 space to accommodate the alignment. If we have reserved PE#256, the waste of M64 space will be avoided.

FSP-specific bugs fixed

- (also fixed in skiboot-5.0.2) Fix race in firenze_get_slot_info() leading to assert() with many PCI cards

  With many PCI cards, we'd hit a race where calls to firenze_add_pcidev_to_fsp_inventory would step on each other leading to memory corruption and finally an assert() in the allocator being hit during boot.
- PCIe power workaround for K80 cards
- /ibm,opal/led renamed to /ibm,opal/leds in Device Tree
  - compatible change as no FSP based systems shipped with skiboot-5.0

General improvements

- Preliminary Centaur i2c support
  - lays framework for supporting Centaur i2c
- don't run pollers on non-boot CPUs in time_wait
- improvements to opal-prd, pflash, libflash
  - including new blocklevel interface in libflash
- many minor fixes to issues found by static analysis
- improvements in FSP error log code paths
- code cleanup in memory allocator
- Don't expose individual nvram partitions in the device tree, just the whole flash device.
- build improvements for building on ppc64el host
- improvements in cpu_relax() for idle threads, needed for GCOV on large machines.
- Optimized memset() for POWER8, greatly reducing number of instructions executed for boot, which helps boot time in simulators.
- Major improvements in hello_world kernel
  - Bloat of huge 17 instruction test case reduced to 10.
- Disable bust_locks for general calls of abort()
  - Should enable better error messages during abort() when other users of LPC bus exist (e.g. flash)
- unified version numbers for bundled utilities
- external/boot_test/boot_test.sh
  - better usable for automated boot testing

Contributors

Since skiboot-5.0, we've had the following changesets:

Processed 372 csets from 27 developers 2 employers found A total of 15868 lines added, 3359 removed (delta 12509)

Developers with the most changesets

Developer	Changesets
Stewart Smith	117 (31.5%)
Jeremy Kerr	37 (9.9%)
Cyril Bur	33 (8.9%)
Vasant Hegde	32 (8.6%)
Benjamin Herrenschmidt	32 (8.6%)
Kamalesh Babulal	22 (5.9%)
Joel Stanley	12 (3.2%)
Mahesh Salgaonkar	12 (3.2%)
Alistair Popple	12 (3.2%)
Neelesh Gupta	9 (2.4%)
Gavin Shan	8 (2.2%)
Cédric Le Goater	8 (2.2%)
Ananth N Mavinakayanahalli	8 (2.2%)
Vipin K Parashar	6 (1.6%)
Michael Neuling	6 (1.6%)
Samuel Mendoza-Jonas	3 (0.8%)
Frederic Bonnard	3 (0.8%)
Andrew Donnellan	2 (0.5%)
Vaidyanathan Srinivasan	2 (0.5%)
Philippe Bergheaud	1 (0.3%)
Shilpasri G Bhat	1 (0.3%)
Daniel Axtens	1 (0.3%)
Hari Bathini	1 (0.3%)
Michael Ellerman	1 (0.3%)
Andrei Warkentin	1 (0.3%)
Dan Horák	1 (0.3%)
Anton Blanchard	1 (0.3%)

Developers with the most changed lines

Stewart Smith	4499 (27.3%)
Benjamin Herrenschmidt	3782 (22.9%)
Jeremy Kerr	1887 (11.4%)
Cyril Bur	1654 (10.0%)
Vasant Hegde	959 (5.8%)
Mahesh Salgaonkar	886 (5.4%)
Neelesh Gupta	473 (2.9%)
Samuel Mendoza-Jonas	387 (2.3%)
Vipin K Parashar	332 (2.0%)
Philippe Bergheaud	171 (1.0%)
Shilpasri G Bhat	165 (1.0%)
Alistair Popple	151 (0.9%)
Joel Stanley	105 (0.6%)
Cédric Le Goater	89 (0.5%)
Gavin Shan	83 (0.5%)
Frederic Bonnard	76 (0.5%)
Kamalesh Babulal	65 (0.4%)
Michael Neuling	46 (0.3%)
Daniel Axtens	31 (0.2%)
Andrew Donnellan	22 (0.1%)
Ananth N Mavinakayanahalli	20 (0.1%)
Anton Blanchard	3 (0.0%)
Vaidyanathan Srinivasan	2 (0.0%)
Hari Bathini	2 (0.0%)
Michael Ellerman	1 (0.0%)
Andrei Warkentin	1 (0.0%)
Dan Horák	1 (0.0%)

Developers with the most lines removed

Michael Neuling	24 (0.7%)
Hari Bathini	1 (0.0%)

Developers with the most signoffs (total 253)

Stewart Smith	249 (98.4%)
Mahesh Salgaonkar	4 (1.6%)

Developers with the most reviews (total 24)

Vasant Hegde	9 (37.5%)
Joel Stanley	3 (12.5%)
Gavin Shan	2 (8.3%)
Kamalesh Babulal	2 (8.3%)
Samuel Mendoza-Jonas	2 (8.3%)
Alistair Popple	2 (8.3%)
Stewart Smith	1 (4.2%)
Andrei Warkentin	1 (4.2%)
Preeti U Murthy	1 (4.2%)
Ananth N Mavinakayanahalli	1 (4.2%)

Developers with the most test credits (total 1)

Chad Larson	1 (100.0%)

Developers who gave the most tested-by credits (total 1)

Gavin Shan	1 (100.0%)

Developers with the most report credits (total 4)

Benjamin Herrenschmidt	2 (50.0%)
Chad Larson	1 (25.0%)
Andrei Warkentin	1 (25.0%)

Developers who gave the most report credits (total 4)

Stewart Smith	3 (75.0%)
Gavin Shan	1 (25.0%)

Top changeset contributors by employer

IBM	369 (99.2%)
(Unknown)	3 (0.8%)

Top lines changed by employer

IBM	16497 (100.0%)
(Unknown)	3 (0.0%)

Employers with the most signoffs (total 253)

IBM	253 (100.0%)

Employers with the most hackers (total 27)

IBM	24 (88.9%)
(Unknown)	3 (11.1%)

4.1.2 skiboot-5.1.0-beta1

skiboot-5.1.0-beta1 was released on July 21st, 2015.

skiboot-5.1.0-beta1 is the first beta release of skiboot 5.1, which will become a new stable release, replacing skiboot-5.0 (released April 14th 2015)

Skiboot 5.1-beta1 contains all fixes from skiboot-5.0 stable branch up to skiboot-5.0.5.

New features

Over skiboot-5.0, the following features have been added:

- Centaur i2c support
- Add Naples chip (CPU, PHB, LPC serial interrupts) support
- Added qemu platform
- improvements to FSI error handling
- improvements in chip TOD failover (some only on FSP systems)
- Set Relative Priority Register (RPR) to recommended value
  - this affects thread priority in SMT modes
- greatly reduce memory consumption by CPU stacks for non-present CPUs
  - Previously we would reserve enough memory for max PIR for each CPU type.
  - This fix frees up 77MB of RAM on a typical P8 system.
- increased OPAL API documentation
- Asynchronous preloading of resources from FSP/flash
  - improves boot time on some systems
- Basic Garrison platform support
- Add Mambo platform (P8 Functional Simulator, systemsim)
  - includes fake NVRAM, RTC
- Support building with GCOV, increasing memory for skiboot binary to 2MB
  - includes boot code coverage testing
- Increased skiboot HEAP size.
  - We are not aware of any system where you would run out, but on large systems it was getting closer than we liked.

- add boot_tests.sh for helping automate boot testing on FSP and BMC machines
- Versioning of pflash and gard utilities to help Linux (or other OS) distributions with packaging.
- OCC throttle status messages to host
- CAPP timebase sync ("ibm,capp-timebase-sync" in DT to indicate CAPP timebase was synced by OPAL)

New features for FSP based machines

- in-band IPMI support
- ethernet adaptor location codes
- add DIMM frequency information to device tree
- improvements in FSP error log code paths
- fix some boot time memory leaks
  - harmless to end user

New features for AMI BMC based machines

- PCIe power workaround for K80
- Added support for Macronix 128Mbit flash chips
- Initial PRD support for Firestone platform
- improved reliability when BMC reboots

Bug Fixes

The following bugs have been fixed:

- Increase PHB3 timeout for electrical links coming up to 2 seconds.
  - fixes issues with some Mellanox cards
- Hang in opal_reinit_cpus() that could prevent kdump from functioning
- PHB3: fix crash in phb3_init
- PHB3: fix crash with fenced PHB in phb3_init_hw()
- Fix bugs in hw/bt.c (interface for IPMI on BMC machines) that could possibly lead to a crash (dereferencing invalid address, deadlock)
- ipmi/sel: fix use-after-free
- Bug fixes in EEH handling
  - opal_pci_next_error() cleared OPAL_EVENT_PCI_ERROR unconditionally, possibly leading to missed errors.

FSP-specific bugs fixed:

- (also fixed in skiboot-5.0.2) Fix race in firenze_get_slot_info() leading to assert() with many PCI cards

  With many PCI cards, we'd hit a race where calls to firenze_add_pcidev_to_fsp_inventory would step on each other leading to memory corruption and finally an assert() in the allocator being hit during boot.

- PCIe power workaround for K80 cards
- /ibm,opal/led renamed to /ibm,opal/leds in Device Tree
  - compatible change as no FSP based systems shipped with skiboot-5.0

General improvements:

- don't run pollers on non-boot CPUs in time_wait
- improvements to opal-prd, pflash, libflash
  - including new blocklevel interface in libflash
- many minor fixes to issues found by static analysis
- improvements in FSP error log code paths
- code cleanup in memory allocator
- Don't expose individual nvram partitions in the device tree, just the whole flash device.
- build improvements for building on ppc64el host
- improvements in cpu_relax() for idle threads, needed for GCOV on large machines.
- Optimized memset() for POWER8, greatly reducing number of instructions executed for boot, which helps boot time in simulators.
- Major improvements in hello_world kernel
  - Bloat of huge 17 instruction test case reduced to 10.
- Disable bust_locks for general calls of abort()
  - Should enable better error messages during abort() when other users of LPC bus exist (e.g. flash)

Contributors

Thanks to everyone who has made skiboot-5.1.0-beta1 happen!

Processed 321 csets from 25 developers 3 employers found A total of 13696 lines added, 2754 removed (delta 10942)

Developers with the most changesets

Developer	Changesets
Stewart Smith	101 (31.5%)
Benjamin Herrenschmidt	32 (10.0%)
Cyril Bur	31 (9.7%)
Vasant Hegde	28 (8.7%)
Jeremy Kerr	27 (8.4%)
Kamalesh Babulal	19 (5.9%)
Alistair Popple	12 (3.7%)
Mahesh Salgaonkar	12 (3.7%)
Neelesh Gupta	8 (2.5%)
Cédric Le Goater	8 (2.5%)
Joel Stanley	8 (2.5%)
Ananth N Mavinakayanahalli	8 (2.5%)
Gavin Shan	6 (1.9%)
Michael Neuling	6 (1.9%)
Frederic Bonnard	3 (0.9%)
Vipin K Parashar	2 (0.6%)
Vaidyanathan Srinivasan	2 (0.6%)
Philippe Bergheaud	1 (0.3%)
Shilpasri G Bhat	1 (0.3%)
Daniel Axtens	1 (0.3%)
Hari Bathini	1 (0.3%)
Michael Ellerman	1 (0.3%)
Andrei Warkentin	1 (0.3%)
Dan Horák	1 (0.3%)
Anton Blanchard	1 (0.3%)

Developers with the most changed lines

Developer	Changed Lines
Stewart Smith	3987 (27.9%)
Benjamin Herrenschmidt	3811 (26.6%)
Cyril Bur	1918 (13.4%)
Jeremy Kerr	1307 (9.1%)
Mahesh Salgaonkar	886 (6.2%)
Vasant Hegde	764 (5.3%)
Neelesh Gupta	473 (3.3%)
Vipin K Parashar	176 (1.2%)
Alistair Popple	175 (1.2%)
Philippe Bergheaud	171 (1.2%)
Shilpasri G Bhat	165 (1.2%)
Cédric Le Goater	89 (0.6%)
Frederic Bonnard	78 (0.5%)
Gavin Shan	73 (0.5%)
Joel Stanley	65 (0.5%)
Kamalesh Babulal	63 (0.4%)
Michael Neuling	47 (0.3%)
Daniel Axtens	31 (0.2%)
Ananth N Mavinakayanahalli	22 (0.2%)
Anton Blanchard	3 (0.0%)
Vaidyanathan Srinivasan	2 (0.0%)
Hari Bathini	2 (0.0%)
Michael Ellerman	1 (0.0%)
Andrei Warkentin	1 (0.0%)
Dan Horák	1 (0.0%)

Developers with the most lines removed:

Vipin K Parashar	105 (3.8%)
Michael Neuling	24 (0.9%)
Hari Bathini	1 (0.0%)

Developers with the most signoffs (total 214)

Stewart Smith	214 (100.0%)

Developers with the most reviews (total 21)

Vasant Hegde	7 (33.3%)
Joel Stanley	3 (14.3%)
Gavin Shan	2 (9.5%)
Kamalesh Babulal	2 (9.5%)
Alistair Popple	2 (9.5%)
Stewart Smith	1 (4.8%)
Andrei Warkentin	1 (4.8%)
Preeti U Murthy	1 (4.8%)
Samuel Mendoza-Jonas	1 (4.8%)
Ananth N Mavinakayanahalli	1 (4.8%)

Developers with the most test credits (total 1)

Chad Larson	1 (100.0%)

Developers who gave the most tested-by credits (total 1)

Gavin Shan	1 (100.0%)

Developers with the most report credits (total 4)

Benjamin Herrenschmidt	2 (50.0%)
Chad Larson	1 (25.0%)
Andrei Warkentin	1 (25.0%)

Developers who gave the most report credits (total 4)

Stewart Smith	3 (75.0%)
Gavin Shan	1 (25.0%)

Top changeset contributors by employer

IBM	319 (99.4%)
dan@danny.cz	1 (0.3%)
andrey.warkentin@gmail.com	1 (0.3%)

Top lines changed by employer

IBM	14309 (100.0%)
dan@danny.cz	1 (0.0%)
andrey.warkentin@gmail.com	1 (0.0%)

Employers with the most signoffs (total 214)

IBM	214 (100.0%)

Employers with the most hackers (total 25)

IBM	23 (92.0%)
dan@danny.cz	1 (4.0%)
andrey.warkentin@gmail.com	1 (4.0%)

4.1.3 skiboot-5.1.0-beta2

skiboot-5.1.0-beta2 was released on August 14th, 2015.

skiboot-5.1.0-beta2 is the second beta release of skiboot 5.1, which will become a new stable release, replacing skiboot-5.0 (released April 14th 2015)

Skiboot 5.1.0-beta2 contains all fixes from skiboot-5.0 stable branch up to skiboot-5.0.5 and everything from 5.1.0-beta1.

New Features

Over skiboot-5.1.0-beta1, the following features have been added:

- **opal-api: Add OPAL call to handle abnormal reboots.** OPAL_CEC_REBOOT2 Currently it will support two reboot types (0). normal reboot, that will behave similar to that of opal_cec_reboot() call, and (1). platform error reboot.

  Long term, this is designed to replace OPAL_CEC_REBOOT.

Bug fixes

Over skiboot-5.1.0-beta1, the following bugs have been fixed:

- external/opal-prd: Only map each PRD range once
  - could eventually lead to failing to map PRD ranges
- On skiboot crash, don't try to print symbol when we didn't find one
  - makes backtrace prettier
- On skiboot crash, dump hssr0 and hsrr1 registers correctly.
- Better support old and biarch compilers
  - test "new" compiler flags before using them
  - Specify -mabi=elfv1 if supported (which means it's needed)
- fix boot-coverage-report makefile target
- ipmi: Fix the opal_ipmi_recv() call to handle the error path
  - Could make kernel a sad panda when in continues with other IPMI commands
- IPMI: truncate SELs at 2kb
  - it's the limit of the astbmc. We think.
- IPMI/SEL/PEL:
  - As per PEL spec, we should log events with severity >= 0x22 and "service action flag" is "on". But in our case, all logs OPAL originagted logs are makred as report externally. We now only report logs with severity >= 0x22
- IPMI: fixes to eSEL logging
- hw/phb3: Change reserved PE to 255
  - Currently, we have reserved PE#0 to which all RIDs are mapped prior to PE assignment request from kernel. The last M64 BAR is configured to have shared mode. So we have to cut off the first M64 segment, which corresponds to reserved PE#0 in kernel. If the first BAR (for example PF's IOV BAR) requires huge

alignment in kernel, we have to waste huge M64 space to accommodate the alignment. If we have reserved PE#256, the waste of M64 space will be avoided.

Other changes

- unified version numbers for bundled utilities
- external/boot_test/boot_test.sh
  - better usable for automated boot testing

4.1.4 skiboot-5.1.1

skiboot-5.1.1 was released on August 18th, 2015.

skiboot-5.1.1 is the send stable release of 5.1, it follows skiboot-5.1.0.

Skiboot 5.1.1 contains all fixes from skiboot-5.1.0 and is a minor bugfix release.

Changes

Over skiboot-5.1.0, we have the following changes:

- Fix detection of compiler options on ancient GCC (e.g. gcc 4.4, shipped with RHEL6)
- ensure the GNUC version defines for GCOV are coming from target CC rather than host CC for extract-gcov
- phb3: Continue CAPP setup even if PHB is already in CAPP mode This fixes a critical bug in CAPI support.

  CAPI requires that all faults are escalated into a fence, not a freeze. This is done by setting bits in a number of MMIO registers. phb3_set_capi_mode() calls phb3_init_capp_errors() to do this. However, if the PHB is already in CAPP mode - for example in the recovery case - phb3_set_capi_mode() will bail out early, and those registers will not be set.

  This is quite easy to verify. PCI config space access errors, for example, normally cause a freeze. On a CAPI-mode PHB, they should cause a fence. Say we have a CAPI card on PHB 0, and we inject a PCI config space error:

```
echo 0x8000000000000000 > /sys/kernel/debug/powerpc/PCI0000/err_injct_inboundA;
lspci;
```

  The first time we inject this, the PHB will fence and recover, but won't reset the registers. Therefore, the second time we inject it, we will incorrectly freeze, not fence.

  Worse, the recovery for the resultant EEH freeze event interacts poorly with the CAPP, triggering an EEH recovery of the PHB. The combination of the two attempted recoveries will get the PHB into an inoperable state.

4.1.5 skiboot-5.1.10

skiboot-5.1.10 was released on Friday November 13th, 2015.

skiboot-5.1.10 is the 11th stable release of 5.1, it follows skiboot-5.1.9 (which was released October 30th, 2015).

Skiboot 5.1.10 contains all fixes from skiboot-5.1.9 and is a minor bug fix release.

Over skiboot-5.1.9, we have the following change:

IBM FSP machines

- FSP: Handle Delayed Power Off initiated CEC shutdown with FSP in Reset/Reload

  In a scenario where the DPO has been initiated, but the FSP then went into reset before the CEC power down came in, OPAL may not give up the link since it may never see the PSI interrupt. So, if we are in dpo_pending and an FSP reset is detected via the DISR, give up the PSI link voluntarily.

Generic

- sensor: add a compatible property OPAL needs an extra compatible property "ibm,opal-sensor" to make module autoload work smoothly in Linux for ibmpowernv driver.
- console: Completely flush output buffer before power down and reboot Completely flush the output buffer of the console driver before power down and reboot. Implements the flushing function for uart consoles, which includes the astbmc and rhesus platforms.

  This fixes an issue where some console output is sometimes lost before power down or reboot in uart consoles. If this issue is also prevalent in other console types then it can be fixed later by adding a .flush to that driver's con_ops.

4.1.6 skiboot-5.1.11

skiboot-5.1.11 was released on Friday November 13th, 2015.

Since it was Friday 13th, we had to find a bug right after we tagged and released skiboot-5.1.10.

skiboot-5.1.11 is the 12th stable release of 5.1, it follows skiboot-5.1.10 (which was released November 13th, 2015).

Skiboot 5.1.11 contains one additional bug fix over skiboot-5.1.10.

It is:

- On IBM FSP machines, if IPMI/Serial console is not connected during shutdown or reboot, machine would enter termination state rather than shut down.

4.1.7 skiboot-5.1.12

skiboot-5.1.12 was released on Friday December 4th, 2015.

skiboot-5.1.12 is the 13th stable release of 5.1, it follows skiboot-5.1.11 (which was released November 13th, 2015).

Skiboot 5.1.12 contains bug fixes and a performance improvement.

opal-prd

- Display an explict and obvious message if running on a system that does not support opal-prd, such as an IBM FSP based POWER system, where the FSP takes on the role of opal-prd.

pflash

- Fix a missing (C) header - cherry-picked from master.

General

- Don't link with libgcc - On some toolchains, we don't have libgcc available.

POWER8 PHB (PCIe) specific

- **hw/phb3: Flush cache line after updating P/Q bits** When doing an MSI EOI, we update the P and Q bits in the IVE. That causes the corresponding cache line to be dirty in the L3 which will cause a subsequent update by the PHB (upon receiving the next MSI) to get a few retries until it gets flushed.

  We improve the situation (and thus performance) by doing a dcbf instruction to force a flush of the update we do in SW.

  This improves interrupt performance, reducing latency per interrupt. The improvement will vary by workload.

IBM FSP based machines

- FSP: Give up PSI link on shutdown This clears up some erroneous SRCs (error logs) in some situations.
- **Correctly report back Real Time Clock errors to host** Under certain rare error conditions, we could return an error code to the host OS that would cause current Linux kernels to get stuck in an infinite loop during boot. This was introduced in skiboot-5.0-rc1.

4.1.8 skiboot-5.1.13

skiboot-5.1.13 was released on Wed January 27th, 2016.

skiboot-5.1.13 is the 14th stable release of 5.1, it follows skiboot-5.1.12 (which was released December 4th, 2015). This release contains bug fixes.

General

- core/device.c: Sort nodes with name@unit names by unit
  - This gives predictable device tree ordering to the payload (usually petitboot)
  - This means that utilities such as "lspci" will always return the same ordering.
- Add OPAL_CONSOLE_FLUSH to the OPAL API uart consoles only flush output when polled. The Linux kernel calls these pollers frequently, except when in a panic state. As such, panic messages are not fully printed unless the system is configured to reboot after panic.

  This patch adds a new call to the OPAL API to flush the buffer. If the system has a uart console (i.e. BMC machines), it will incrementally flush the buffer, returning if there is more to be flushed or not. If the system has a different console, the function will have no effect. This will allow the Linux kernel to ensure that panic message have been fully printed out.

CAPI

- hmi: Identify the phb upon CAPI malfunction alert Previously, any error on a CAPI adapter would assume PHB0. This could cause issues on Firestone machines.

gard utility

- Fix displaying 'cleared' gard records When a garded component is replaced hostboot detects this and updates the gard partition.

  Previously, there was ambiguity on if the gard record ID or the whole gard record needed to be erased. This fix makes gard and hostboot agree.

firestone platform

- fix spacing in slot name The other SlotN names have no space.

4.1.9 skiboot-5.1.14

skiboot-5.1.14 was released on Wed March 9th, 2016.

skiboot-5.1.14 is the 15th stable release of 5.1, it follows skiboot-5.1.13 (which was released January 27th, 2016). This release contains a spelling fix in a log message and an added device tree property to enable older kernels (with bootloader support) to use a framebuffer that is redirected to the BMC VGA port.

As such, skiboot-5.1.14 has no advantage over skiboot-5.1.13 unless you are wanting the neat offb framebuffer trick.

Changes are:

- fsp: fix spelling of "advertise" in log message See: https://www.youtube.com/watch?v=8Gv0H-vPoDc
- Explicit 1:1 mapping in ranges properties have been added to PCI bridges. This allows a neat trick with offb and VGA ports that should probably not be told to young children.

4.1.10 skiboot-5.1.15

skiboot-5.1.15 was released on Wed March 16th, 2016.

skiboot-5.1.15 is the 16th stable release of 5.1, it follows skiboot-5.1.14 (which was released March 9th, 2016). This release contains one bug fix, a fix for a memory leak in an error path for AMI BMC based systems when logging non-severe errors. As such, it is a minor bug fix update.

4.1.11 skiboot-5.1.16

skiboot-5.1.16 was released on Friday April 29th, 2016.

skiboot-5.1.16 is the 17th stable release of 5.1, it follows skiboot-5.1.15 (which was released March 16th, 2016).

This release contains a few bug fixes and is a recommended upgrade.

Changes

PHB3 (all POWER8 platforms)

- hw/phb3: Ensure PQ bits are cleared in the IVC when masking IRQ When we mask an interrupt, we may race with another interrupt coming in from the hardware. If this occurs, the P and/or Q bit may end up being set but we never EOI/clear them. This could result in a lost interrupt or the next interrupt that comes in after re-enabling never being presented.

This fixes a bug seen with some CAPI workloads which have lots of interrupt masking at the same time as high interrupt load. The fix is not specific to CAPI though.

- **hw/phb3: Fix potential race in EOI** When we EOI we need to clear the present (P) bit in the Interrupt Vector Cache (IVC). We must clear P ensuring that any additional interrupts that come in aren't lost while also maintaining coherency with the Interrupt Vector Table (IVT).

  To do this, the hardware provides a conditional update bit in the IVC. This bit ensures that generation counts between the IVT and the IVC updates are synchronised.

  Unfortunately we never set this the bit to conditionally update the P bit in the IVC based on the generation count. Also, we didn't set what we wanted the new generation count to be if the update was successful.

FSP platforms

- OPAL:Handle mbox response with bad status:0x24 during FSP termination OPAL committed a predictive log with SRC BB822411 in some situations.

Generic

- hmi: Fix a bug where partial hmi event was reported to host. This bug fix ensures the CPU PIR is reported correctly:

```
  [  305.628283] Fatal Hypervisor Maintenance interrupt [Not recovered]
  [  305.628341]  Error detail: Malfunction Alert
  [  305.628388]      HMER: 8040000000000000
- [  305.628423]       CPU PIR: 00000000
+ [  200.123021]     CPU PIR: 000008e8
  [  305.628458]  [Unit: VSU] Logic core check stop
```

4.1.12 skiboot-5.1.17

skiboot-5.1.17 was released on Thursday 21st July 2016.

skiboot-5.1.17 is the 18th stable release of 5.1, it follows skiboot-5.1.16 (which was released April 29th, 2016).

This release contains a few minor bug fixes.

Changes

All platforms:

- Fix a few typos in user visible (OPAL log) strings
- pci: Do a dummy config write to devices to establish bus number
- Make the XSCOM engine code more resilient to errors: - hw/xscom: Reset XSCOM engine after querying sleeping core FIR - hw/xscom: Reset XSCOM engine after finite number of retries when busy - xscom: Return OPAL_WRONG_STATE on XSCOM ops if CPU is asleep

4.1.13 skiboot-5.1.18

skiboot-5.1.18 was released on Friday 26th August 2016.

skiboot-5.1.18 is the 19th stable release of 5.1, it follows skiboot-5.1.17 (which was released July 21st, 2016).

This release contains a few minor bug fixes.

Changes are:

All platforms:

- opal/hmi: Fix a TOD HMI failure during a race condition. Rare race condition which meant we wouldn't recover from TOD error
- hw/phb3: Update capi initialization sequence The capi initialization sequence was revised in a circumvention document when a 'link down' error was converted from fatal to Endpoint Recoverable. Other, non-capi, register setup was corrected even before the initial open-source release of skiboot, but a few capi-related registers were not updated then, so this patch fixes it. The point is that a link-down error detected by the UTL logic will lead to an AIB fence, so that the CAPP unit can detect the error.

FSP platforms:

- FSP/ELOG: Fix OPAL generated elog resend logic
- FSP/ELOG: Fix possible event notifier hangs
- FSP/ELOG: Disable event notification if list is not consistent
- FSP/ELOG: Fix OPAL generated elog event notification
- FSP/ELOG: Disable event notification during kexec

4.1.14 skiboot-5.1.19

skiboot-5.1.19 was released on Monday 16th January 2017.

skiboot-5.1.19 is the 20th stable release of 5.1, it follows skiboot-5.1.18 (which was released 26th August 2016).

This release contains a few minor bug fixes.

Changes are:

Generic:

- Makefile: Disable stack protector due to gcc problems
- stack: Don't recurse into __stack_chk_fail
- Makefile: Use -ffixed-r13 We did not find evidence of this ever being a problem, but this fix is good and preventative.
- Limit number of "Poller recursion detected" errors to display In some error conditions, we could spiral out of control on this and spend all of our time printing the exact same backtrace. Limit it to 16 times, because 16 is a nice number.

FSP based Systems:

- fsp: Don't recurse pollers in ibm_fsp_terminate If we were to terminate in a poller, we'd call op_display() which called pollers which hit the recursive poller warning, which ended in not much fun at all.

PCI:

- hw/phb3: set PHB retry state correctly when fresetting during a creset

- **phb3: Lock the PHB on set_xive callbacks** Those are called by the interrupts core and thus skip the locking implicit in the PCI opal calls.
- hw/{phb3, p7ioc}: Return success for freset on empty PHB OPAL_CLOSED is returned when fundamental reset is issued on the PHB who doesn't have subordinate devices (root port excluded). The kernel raises an error message, which is unnecessary. This returns OPAL_SUCCESS for this case to avoid the error message.
- hw/phb3: fix error handling in complete reset During a complete reset, when we get a timeout waiting for pending transaction in state PHB3_STATE_CRESET_WAIT_CQ, we mark the PHB as broken and return OPAL_PARAMETER. Change the return code to OPAL_HARDWARE which is way more sensible, and set the state to PHB3_STATE_FENCED so that the kernel can retry the complete reset.

4.1.15 skiboot-5.1.2

skiboot-5.1.2 was released on September 9th, 2015.

skiboot-5.1.2 is the third stable release of 5.1, it follows skiboot-5.1.1 (which was released August 18th, 2015).

Skiboot 5.1.2 contains all fixes from skiboot-5.1.1 and is a minor bugfix release.

Changes

Over skiboot-5.1.1, we have the following changes:

- phb3: Handle fence in phb3_pci_msi_check_q to fix hang

  If the PHB is fenced during phb3_pci_msi_check_q, it can get stuck in an infinite loop waiting to lock the FFI. Further, as the phb lock is held during this function it will prevent any other CPUs from dealing with the fence, leading to the entire system hanging.

  If the PHB_FFI_LOCK returns all Fs, return immediately to allow the fence to be dealt with.
- phb3: Continue CAPP setup even if PHB is already in CAPP mode This fixes a critical bug in CAPI support.
- Platform hook for terminate call
  - on assert() or other firmware failure, we will make a SEL callout on ASTBMC platforms
  - (slight) refactor of code for IBM-FSP platforms
- refactor slot naming code
- Slot names for Habanero platform
- misc improvements in userspace utilities (incl pflash, gard)
- build improvements
  - fixes for two compiler warnings were squashed in 5.1.1 commit, re-introduce the fixes.
  - misc compiler/static analysis warning fixes
- gard utility:
  - If gard tool detects the GUARD PNOR partition is corrupted, it will pro-actively re-initialize it. Modern Hostboot is more sensitive to the content of the GUARD partition in order to boot.
  - Update record clearing to match Hostboots expectations We now write ECC bytes throughout the whole partition. Without this fix, hostboot may not bring up the machine.
  - In the event of a corrupted GUARD partition so that even the first entry cannot be read, the gard utility now provides the user with the option to wipe the entirety of the GUARD partition to attempt recovery.

- opal_prd utility:
  - Add run command to pass through commands to HostBoot RunTime (HBRT)
    - this is for OpenPower firmware developers only.
  - Add htmght-passthru command.
    - this is for OpenPower firmware developers only.
  - Add override interface to pass attribute-override information to HBRT.
  - Server sends response in error path, so that client doesn't block forever
- external/mambo tcl scripts
  - Running little-endian kernels in mambo requires HILE to be set properly, which requires a bump in the machine's pvr value to a DD2.x chip.

Stats

For skiboot-5.1.0 to 5.1.2: Processed 67 csets from 11 developers 1 employers found A total of 2258 lines added, 784 removed (delta 1474)

Developers with the most changesets

Stewart Smith	24 (35.8%)
Cyril Bur	18 (26.9%)
Vasant Hegde	8 (11.9%)
Neelesh Gupta	5 (7.5%)
Benjamin Herrenschmidt	5 (7.5%)
Daniel Axtens	2 (3.0%)
Samuel Mendoza-Jonas	1 (1.5%)
Vaidyanathan Srinivasan	1 (1.5%)
Vipin K Parashar	1 (1.5%)
Ian Munsie	1 (1.5%)
Michael Neuling	1 (1.5%)

Developers with the most changed lines

Cyril Bur	969 (42.5%)
Neelesh Gupta	433 (19.0%)
Benjamin Herrenschmidt	304 (13.3%)
Vasant Hegde	236 (10.3%)
Stewart Smith	163 (7.1%)
Vaidyanathan Srinivasan	135 (5.9%)
Vipin K Parashar	8 (0.4%)
Ian Munsie	8 (0.4%)
Daniel Axtens	2 (0.1%)
Michael Neuling	2 (0.1%)
Samuel Mendoza-Jonas	1 (0.0%)

Developers with the most lines removed

Daniel Axtens	2 (0.3%)
Michael Neuling	1 (0.1%)

Developers with the most signoffs (total 44)

Stewart Smith	43 (97.7%)
Neelesh Gupta	1 (2.3%)

Developers with the most reviews (total 8)

Patrick Williams	5 (62.5%)
Samuel Mendoza-Jonas	3 (37.5%)

Developers with the most test credits (total 0)

Developers who gave the most tested-by credits (total 0)

Developers with the most report credits (total 1)

Benjamin Herrenschmidt	1 (100.0%)

Developers who gave the most report credits (total 1)

Samuel Mendoza-Jonas	1 (100.0%)

Top changeset contributors by employer

IBM	67 (100.0%)

Top lines changed by employer

IBM	2281 (100.0%)

Employers with the most signoffs (total 44)

IBM	44 (100.0%)

Employers with the most hackers (total 11)

IBM	11 (100.0%)

4.1.16 skiboot-5.1.20

skiboot-5.1.20 was released on Friday 18th August 2017.

skiboot-5.1.20 is the 21st stable release of 5.1, it follows skiboot-5.1.19 (which was released 16th January 2017).

This release contains a few minor bug fixes backported to the 5.1.x series. All of the fixes have previously appeared in the 5.4.x stable series.

Changes are:

- FSP/CONSOLE: Workaround for unresponsive ipmi daemon

  In some corner cases, where FSP is active but not responding to console MBOX message (due to buggy IPMI) and we have heavy console write happening from kernel, then eventually our console buffer becomes full. At this point OPAL starts sending OPAL_BUSY_EVENT to kernel. Kernel will keep on retrying. This is creating kernel soft lockups. In some extreme case when every CPU is trying to write to console, user will not be able to ssh and thinks system is hang.

  If we reset FSP or restart IPMI daemon on FSP, system recovers and everything becomes normal.

  This patch adds workaround to above issue by returning OPAL_HARDWARE when cosole is full. Side effect of this patch is, we may endup dropping latest console data. But better to drop console data than system hang.

  Alternative approach is to drop old data from console buffer, make space for new data. But in normal condition only FSP can update 'next_out' pointer and if we touch that pointer, it may introduce some other race conditions. Hence we decided to just new console write request.

- FSP: Set status field in response message for timed out message

  For timed out FSP messages, we set message status as "fsp_msg_timeout". But most FSP driver users (like surviellance) are ignoring this field. They always look for FSP returned status value in callback function (second byte in word1). So we endup treating timed out message as success response from FSP.

  Sample output:

```
[69902.432509048,7] SURV: Sending the heartbeat command to FSP
[70023.226860117,4] FSP: Response from FSP timed out, word0 = d66a00d7, word1 = 0␣
↪state: 3
....
[70023.226901445,7] SURV: Received heartbeat acknowledge from FSP
[70023.226903251,3] FSP: fsp_trigger_reset() entry
```

  Here SURV code thought it got valid response from FSP. But actually we didn't receive response from FSP.

- FSP: Improve timeout message

  Presently we print word0 and word1 in error log. word0 contains sequence number and command class. One has to understand word0 format to identify command class.

  Lets explicitly print command class, sub command etc.

- FSP/RTC: Remove local fsp_in_reset variable

  Now that we are using fsp_in_rr() to detect FSP reset/reload, fsp_in_reset become redundant. Lets remove this local variable.

- FSP/RTC: Fix possible FSP R/R issue in rtc write path

  fsp_opal_rtc_write() checks FSP status before queueing message to FSP. But if FSP R/R starts before getting response to queued message then we will continue to return OPAL_BUSY_EVENT to host. In some extreme condition host may experience hang. Once FSP is back we will repost message, get response from FSP and return OPAL_SUCCESS to host.

This patch caches new values and returns OPAL_SUCCESS if FSP R/R is happening. And once FSP is back we will send cached value to FSP.

- hw/fsp/rtc: read/write cached rtc tod on fsp hir.

  Currently fsp-rtc reads/writes the cached RTC TOD on an fsp reset. Use latest fsp_in_rr() function to properly read the cached rtc value when fsp reset initiated by the hir.

  Below is the kernel trace when we set hw clock, when hir process starts.

```
[ 1727.775824] NMI watchdog: BUG: soft lockup - CPU#57 stuck for 23s!␣
↪[hwclock:7688]
[ 1727.775856] Modules linked in: vmx_crypto ibmpowernv ipmi_powernv uio_pdrv_
↪genirq ipmi_devintf powernv_op_panel uio ipmi_msghandler powernv_rng leds_
↪powernv ip_tables x_tables autofs4 ses enclosure scsi_transport_sas crc32c_
↪vpmsum lpfc ipr tg3 scsi_transport_fc
[ 1727.775883] CPU: 57 PID: 7688 Comm: hwclock Not tainted 4.10.0-14-generic #16-
↪Ubuntu
[ 1727.775883] task: c000000fdfdc8400 task.stack: c000000fdfef4000
[ 1727.775884] NIP: c00000000090540c LR: c0000000000846f4 CTR: 000000003006dd70
[ 1727.775885] REGS: c000000fdfef79a0 TRAP: 0901   Not tainted  (4.10.0-14-
↪generic)
[ 1727.775886] MSR: 9000000000009033 <SF,HV,EE,ME,IR,DR,RI,LE>
[ 1727.775889]   CR: 28024442  XER: 20000000
[ 1727.775890] CFAR: c00000000008472c SOFTE: 1
               GPR00: 0000000030005128 c000000fdfef7c20 c00000000144c900␣
↪ffffffffffffff4
               GPR04: 0000000028024442 c00000000090540c 9000000000009033␣
↪0000000000000000
               GPR08: 0000000000000000 0000000031fc4000 c000000000084710␣
↪9000000000001003
               GPR12: c0000000000846e8 c00000000fba0100
[ 1727.775897] NIP [c00000000090540c] opal_set_rtc_time+0x4c/0xb0
[ 1727.775899] LR [c0000000000846f4] opal_return+0xc/0x48
[ 1727.775899] Call Trace:
[ 1727.775900] [c000000fdfef7c20] [c00000000090540c] opal_set_rtc_time+0x4c/0xb0␣
↪(unreliable)
[ 1727.775901] [c000000fdfef7c60] [c000000000900828] rtc_set_time+0xb8/0x1b0
[ 1727.775903] [c000000fdfef7ca0] [c000000000902364] rtc_dev_ioctl+0x454/0x630
[ 1727.775904] [c000000fdfef7d40] [c00000000035b1f4] do_vfs_ioctl+0xd4/0x8c0
[ 1727.775906] [c000000fdfef7de0] [c00000000035bab4] SyS_ioctl+0xd4/0xf0
[ 1727.775907] [c000000fdfef7e30] [c00000000000b184] system_call+0x38/0xe0
[ 1727.775908] Instruction dump:
[ 1727.775909] f821ffc1 39200000 7c832378 91210028 38a10020 39200000 38810028␣
↪f9210020
[ 1727.775911] 4bfffe6d e8810020 80610028 4b77f61d <60000000> 7c7f1b78 3860000a␣
↪2fbffff4
```

  This is found when executing the op-test-framework fspresetReload testcase

  With this fix ran fsp hir torture testcase in the above test which is working fine.

- FSP/CHIPTOD: Return false in error path
- On FSP platforms: notify FSP of Platform Log ID after Host Initiated Reset Reload Trigging a Host Initiated Reset (when the host detects the FSP has gone out to lunch and should be rebooted), would cause "Unknown Command" messages to appear in the OPAL log.

  This patch implements those messages.

  Log showing unknown command:

```
/ # cat /sys/firmware/opal/msglog | grep -i ,3
[  110.232114723,3] FSP: fsp_trigger_reset() entry
[  188.431793837,3] FSP #0: Link down, starting R&R
[  464.109239162,3] FSP #0: Got XUP with no pending message !
[  466.340598554,3] FSP-DPO: Unknown command 0xce0900
[  466.340600126,3] FSP: Unhandled message ce0900
```

- hw/i2c: Fix early lock drop

  When interacting with an I2C master the p8-i2c driver (common to p9) aquires a per-master lock which it holds for the duration of it's interaction with the master. Unfortunately, when p8_i2c_check_initial_status() detects that the master is busy with another transaction it drops the lock and returns OPAL_BUSY. This is contrary to the driver's locking strategy which requires that the caller aquire and drop the lock. This leads to a crash due to the double unlock(), which skiboot treats as fatal.

- head.S: store all of LR and CTR

  When saving the CTR and LR registers the skiboot exception handlers use the 'stw' instruction which only saves the lower 32 bits of the register. Given these are both 64 bit registers this leads to some strange register dumps, for example:

```
***********************************************
Unexpected exception 200 !
SRR0 : 0000000030016968 SRR1 : 9000000000201000
HSRR0: 0000000000000180 HSRR1: 9000000000001000
LR   : 3003438830823f50 CTR  : 3003438800000018
CFAR : 00000000300168fc
CR   : 40004208  XER: 00000000
```

  In this dump the upper 32 bits of LR and CTR are actually stack gunk which obscures the underlying issue.

- hw/fsp: Do not queue SP and SPCN class messages during reset/reload In certain cases of communicating with the FSP (e.g. sensors), the OPAL FSP driver returns a default code (async completion) even though there is no known bound from the time of this error return to the actual data being available. The kernel driver keeps waiting leading to soft-lockup on the host side.

  Mitigate both these (known) cases by returning OPAL_BUSY so the host driver knows to retry later.

4.1.17 skiboot-5.1.21

skiboot-5.1.21 was released on Tuesday 19th September 2017.

skiboot-5.1.21 is the 22nd stable release of 5.1, it follows skiboot-5.1.20 (which was released 18th August 2017).

This release contains one backported bug fix to the 5.1.x series.

Changes are:

- FSP: Add check to detect FSP Reset/Reload inside fsp_sync_msg()

  During FSP Reset/Reload we move outstanding MBOX messages from msgq to rr_queue including inflight message (fsp_reset_cmdclass()). But we are not resetting inflight message state.

  In extreme corner case where we sent message to FSP via fsp_sync_msg() path and FSP Reset/Reload happens before getting respose from FSP, then we will endup waiting in fsp_sync_msg() until everything becomes normal.

  This patch adds fsp_in_rr() check to fsp_sync_msg() and return error to caller if FSP is in R/R.

4.1.18 skiboot-5.1.3

skiboot-5.1.3 was released on September 15th, 2015.

skiboot-5.1.3 is the 4th stable release of 5.1, it follows skiboot-5.1.2 (which was released September 9th, 2015).

Skiboot 5.1.3 contains all fixes from skiboot-5.1.2 and is a minor bugfix release.

Changes

Over skiboot-5.1.2, we have the following changes:

- slot names for firestone platform
- fix display of LPC errors
- SBE based timer support
  - on supported platforms limits reliance on Linux heartbeat
- fix use-after-free in fsp/ipmi
- fix hang on TOD/TB errors (time-of-day/timebase) on OpenPower systems
  - On getting a Hypervizor Maintenance Interrupt to get the timebase back into a running state, we would call prlog which would use the LPC UART console driver on OpenPower systems, which depends on a working timebase, leading to a hang. We now don't depend on a working timebase in this recovery codepath.
- enable prd for garrison platform
- PCI: Clear error bits after changing MPS Chaning MPS on PCI upstream bridge might cause error bits set on downstream endpoints when system boots into Linux as below case shows:

```
host# lspci -vvs 0001:06:00.0
0001:06:00.0 Ethernet controller: Broadcom Corporation \
             NetXtreme II BCM57810 10 Gigabit Ethernet (rev 10)
DevSta:      CorrErr+ UncorrErr- FatalErr- UnsuppReq+ AuxPwr- TransPend-
CESta:       RxErr- BadTLP- BadDLLP- Rollover- Timeout- NonFatalErr+
```

  This clears those error bits in AER and PCIe capability after MPS is changed. With the patch applied, no more error bits are seen.

Contributors

Processed 14 csets from 6 developers 1 employers found A total of 462 lines added, 163 removed (delta 299)

Developers with the most changesets

Benjamin Herrenschmidt	5 (35.7%)
Stewart Smith	4 (28.6%)
Mahesh Salgaonkar	2 (14.3%)
Gavin Shan	1 (7.1%)
Jeremy Kerr	1 (7.1%)
Neelesh Gupta	1 (7.1%)

Developers with the most changed lines

Benjamin Herrenschmidt	407 (80.8%)
Mahesh Salgaonkar	23 (4.6%)
Gavin Shan	19 (3.8%)
Stewart Smith	18 (3.6%)
Jeremy Kerr	5 (1.0%)
Neelesh Gupta	2 (0.4%)

Developers with the most lines removed

Stewart Smith	8 (4.9%)
Jeremy Kerr	3 (1.8%)
Neelesh Gupta	1 (0.6%)

Developers with the most signoffs (total 10)

Stewart Smith	10 (100.0%)

Developers with the most reviews (total 1)

Joel Stanley	1 (100.0%)

Developers with the most test credits (total 0)

Developers who gave the most tested-by credits (total 0)

Developers with the most report credits (total 1)

John Walthour	1 (100.0%)

Developers who gave the most report credits (total 1)

Gavin Shan	1 (100.0%)

Top changeset contributors by employer

IBM	14 (100.0%)

Top lines changed by employer

IBM	504 (100.0%)

Employers with the most signoffs (total 10)

IBM	10 (100.0%)

Employers with the most hackers (total 6)

IBM	6 (100.0%)

4.1.19 skiboot-5.1.4

skiboot-5.1.4 was released on September 26th, 2015.

skiboot-5.1.4 is the 5th stable release of 5.1, it follows skiboot-5.1.3 (which was released September 15th, 2015).

Skiboot 5.1.4 contains all fixes from skiboot-5.1.3 and is an important bug fix release and a strongly recommended update from any prior skiboot-5.1.x release.

Changes

Over skiboot-5.1.3, we have the following changes:

- Rate limit OPAL_MSG_OCC to only one outstanding message to host

  In the event of a lot of OCC events (or many CPU cores), we could send many OCC messages to the host, which if it wasn't calling opal_get_msg really often, would cause skiboot to malloc() additional messages until we ran out of skiboot heap and things didn't end up being much fun.

  When running certain hardware exercisers, they seem to steal all time from Linux being able to call opal_get_msg, causing these to queue up and get "opalmsg: No available node in the free list, allocating" warnings followed by tonnes of backtraces of failing memory allocations.

- Ensure reserved memory ranges are exposed correctly to host (fix corrupted SLW image)

  We seem to have not hit this on ASTBMC based OpenPower machines, but was certainly hit on FSP based machines

4.1.20 skiboot-5.1.5

skiboot-5.1.5 was released on October 1st, 2015.

skiboot-5.1.5 is the 6th stable release of 5.1, it follows skiboot-5.1.4 (which was released September 26th, 2015).

Skiboot 5.1.5 contains all fixes from skiboot-5.1.4 and is a minor bug fix release.

Changes

Over skiboot-5.1.4, we have the following changes:

Generic

- centaur: Add indirect XSCOM support Fixes a bug where opal-prd would not be able to recover from a bunch of errors as the indirect XSCOMs to centaurs would fail.
- xscom: Fix logging of indirect XSCOM errors Better logging of error messages.

- PHB3: Fix wrong PE number in error injection
- Improvement in boot_test.sh utility to support copying a pflash binary to BMCs.

AST BMC machines

- **ipmi-sel: Run power action immediately if host not up** Our normal sequence for a soft power action (IPMI 'power soft' or 'power cycle') involve receiving a SEL from the BMC, sending a message to Linux's opal platform support which instructs the host OS to shut down, and finally the host will request OPAL to cut power.

  When the host is not yet up we will send the message to /dev/null, and no action will be taken. This patches changes that behaviour to perform the action immediately if we know how.

OpenPower machines:

- opal-prd: Increase IPMI timeout to a slightly better value Proactively bump the timeout to 5seconds to match current value in petitboot Observed in the wild that this fixes bugs for petitboot.

4.1.21 skiboot-5.1.6

skiboot-5.1.6 was released on October 8th, 2015.

skiboot-5.1.6 is the 7th stable release of 5.1, it follows skiboot-5.1.5 (which was released October 1st, 2015).

Skiboot 5.1.6 contains all fixes from skiboot-5.1.5 and is a minor bug fix release.

Changes

Over skiboot-5.1.5, we have the following changes:

Generic:

- Ensure we run pollers in cpu_wait_job()

  In root causing a bug on AST BMC Alistair found that pollers weren't being run for around 3800ms.

  This could show as not resetting the boot count sensor on successful boot.

AST BMC Machines

- hw/bt.c: Check for timeout after checking for message response

  When deciding if a BT message has timed out we should first check for a message response. This will ensure that messages will not time out if there was a delay calling the pollers.

  This could show as not resetting the boot count sensor on successful boot.

4.1.22 skiboot-5.1.7

skiboot-5.1.7 was released on October 13th, 2015.

skiboot-5.1.7 is the 8th stable release of 5.1, it follows skiboot-5.1.6 (which was released October 8th, 2015).

Skiboot 5.1.7 contains all fixes from skiboot-5.1.6 and is a minor bug fix release with one important bug fix for FSP systems.

Over skiboot-5.1.6, we have the following changes:

Generic:

- **PHB3: Retry fundamental reset** This introduces another PHB3 state (PHB3_STATE_FRESET_START) allowing to redo fundamental reset if the link doesn't come up in time at the first attempt, to improve the robustness of PHB's fundamental reset. If the link comes up after the first reset, the 2nd reset won't be issued at all.

FSP based systems:

- hw/fsp/fsp-leds.c: use allocated buffer for FSP_CMD_GET_LED_LIST response

  This fixes a bug where we would overwrite roughly 4kb of memory belonging to Linux when the FSP would ask firmware for a list of LEDs in the system. This wouldn't happen often (once before Linux was running and possibly only once during runtime, and *early* runtime at that) but it was possible for this corruption to show up and be detected.

4.1.23 skiboot-5.1.8

skiboot-5.1.8 was released on October 19th, 2015.

skiboot-5.1.8 is the 9th stable release of 5.1, it follows skiboot-5.1.7 (which was released October 13th, 2015).

Skiboot 5.1.8 contains all fixes from skiboot-5.1.7 and is a minor bug fix release, with a single fix for recovery from a (rare) error.

Over skiboot-5.1.7, we have the following change:

- opal/hmi: Fix a soft lockup issue on Hypervisor Maintenance Interrupt for certain timebase errors.

  We also introduce a timeout to handle the worst situation where all other threads are badly stuck without setting a cleanup done bit. Under such situation timeout will help to avoid soft lockups and report failure to kernel.

4.1.24 skiboot-5.1.9

skiboot-5.1.9 was released on October 30th, 2015.

skiboot-5.1.9 is the 10th stable release of 5.1, it follows skiboot-5.1.8 (which was released October 19th, 2015).

Skiboot 5.1.9 contains all fixes from skiboot-5.1.8 and is a minor bug fix release, with a single fix to help diagnosis after a rare error condition.

Over skiboot-5.1.8, we have the following change:

- opal/hmi: Signal PRD about NX unit checkstop. We now signal Processor Recovery & Diagnostics (PRD) correctly following an NX unit checkstop
- minor fix to the boot_test.sh test script

4.1.25 skiboot-5.2.0

skiboot-5.2.0 was released on Wednesday March 16th, 2016.

skiboot-5.2.0 is the first stable release of skiboot 5.2, the new stable release of skiboot, which will take over from the 5.1.x series which was first released August 17th, 2015.

skiboot-5.2.0 contains all bug fixes as of skiboot-5.1.15.

This is the second release that will follow the (now documented) Skiboot stable rules - see *Skiboot stable tree rules and releases*.

Changes since rc2

Over skiboot-5.2.0-rc2, the following fixes are included:

- Include 'extract-gcov' in make clean.
- ipmi-sel: Fix esel event logger to handle early boot PANIC events
- IPMI: Enable synchronous eSEL logging option (for PANIC events)
- libflash/libffs: Reporting seeing all 0xFF bytes during init.
- ipmi-sel: Fix memory leak in error path

Changes since rc1

Over skiboot-5.2.0-rc1, we have the following changes:

- Add Barreleye platform

Generic

- hw/p8-i2c: Speed up SMBUS_WRITE
- Fix early backtraces

FSP Platforms

- fsp-sensor: rework device tree for sensors
- platforms/firenze: Fix I2C clock source frequency

Simics simulator

- Enable Simics UART console

Mambo simulator

- platforms/mambo: Add terminate callback
  - fix hang in multi-threaded mambo
  - add multithreaded mambo tests

IPMI

- hw/ipmi: fix event data 1 for System Firmware Progress sensor
- ipmi: Log exact NetFn value in OPAL logs

AST BMC based platforms

- hw/bt: allow BT driver to use different buffer size

opal-prd utility

- **opal-prd: Add debug output for firmware-driven OCC events** We indicate when we have a user-driven event, so add corresponding outputs for firmware-driven ones too.

getscom utility

- Add Naples chip support

New Features

Over skiboot-5.1, the following features have been added:

- Naples (P8', i.e. P8 with NVLINK) processor support, including NVLINK.
- Improvements in gard, libflash/pflash and opal-prd utilities
  - increased testing
  - increased usability
  - systemd scripts for opal-prd
  - pflash can now use the /dev/mtd device to access BMC flash rather than accessing it directly. It is *important* that you use –mtd if your BMC may otherwise know how to interact with its own flash.
- support for Micron N25Q256Ax and N25Qx256Ax NOR flash.
- support for Winbond W25Q256BV NOR flash
- support for an emulated ("fake") RTC clock, useful in simulators and during bringup
- Explicit 1:1 mapping in ranges properties have been added to PCI bridges. This allows a neat trick with offb and VGA ports that should probably not be told to young children.
- Added support to read the V2 format of the OCC-OPAL memory region, which supports Workload Optimized Frequency (WOF)

Changes in behavior

- Assigning OPAL IDs to PHBs is now fixed and based on the chip id and PHB index on that chip. On POWER7, we continue to use allocated numbers.
- We now query the BMC for BT capabilities rather than making assumptions

Removed support

- p5ioc2 is no longer supported. This affects a grand total of two POWER7 systems in the world.

NOTE: It is planned that skiboot-5.2 will be the last release supporting POWER7 machines.

Bugs fixed

- PHB3: Fix unexpected ER (all) on errinjct by PCI config
- hw/bt: timeout messages when BT interface isn't functional
- On Habanero, Slot3 should have been "Slot 3".
- We now completely flush the console buffer before power down and reboot
- For chips with ibm,occ-functional-state set to false, we don't wait for the OCC to start. This caused needless delay in booting on simulators which did not simulate OCCs.
- Change OCC reset order to always reset slave OCCs first.
- slw: Remove overwrites for EX_PM_CORE_ECO_VRET and EX_PM_CORE_PFET_VRET (these were already initialized in hostboot)
- p8-i2c: send stop bit on timeouts. Some devices can otherwise leave the bus in a held state.

Other improvements

- many fixes of compiler and static analysis warnings
- increased unit test coverage
- Unit test of "boot debian jessie installer"
- ability to plug in other simulators to run existing tests (e.g. simulator for non pegasus p8)
- Support using (patched) Qemu with PowerNV platform support for running unit tests.
- increased support for running with sparse
- We now build with -fstack-protector-strong if supported by the compiler
- We now build with -Werror for -Wformat
- pflash is now built as part of travis-ci and for Coverity Scan.
- There is now a RPM SPEC file that can be used as the basis for packaging skiboot and associated utilities.

Contributors

We have had a number of improvements in workflow over skiboot-5.1.0. Looking back, we have roughly the same number of changesets (372 for 5.1.0, 334 for 5.2.0-rc1 - even closer for 5.1.0-beta1) which indicates a relatively stable rate of development.

Complete statistics are included below (generated by gitdm), but I'd like to draw attention to a couple of stats:

Release	csets	Ack	Reviews	Tested	Reported
5.0	329	15	20	1	0
5.1	372	13	38	1	4
5.2-rc1	334	20	34	6	11

Overall, it looks like we're on the right trajectory for increasing the number of eyeballs looking at code before it heads in tree, especially around testing. Largely, this increase in Tested-by can be attributed to encouraging the existing test teams to start commenting on the patches themselves.

Anyway, here's the full stats from skiboot 5.1.0 to 5.2.0-rc1:

Processed 334 csets from 27 developers 2 employers found A total of 46172 lines added, 23274 removed (delta 22898)

Developers with the most changesets

Stewart Smith	146 (43.7%)
Cyril Bur	52 (15.6%)
Benjamin Herrenschmidt	15 (4.5%)
Joel Stanley	12 (3.6%)
Gavin Shan	12 (3.6%)
Alistair Popple	10 (3.0%)
Vasant Hegde	10 (3.0%)
Michael Neuling	10 (3.0%)
Russell Currey	9 (2.7%)
Cédric Le Goater	8 (2.4%)
Jeremy Kerr	8 (2.4%)
Samuel Mendoza-Jonas	6 (1.8%)
Neelesh Gupta	6 (1.8%)
Shilpasri G Bhat	4 (1.2%)
Oliver O'Halloran	4 (1.2%)
Mahesh Salgaonkar	4 (1.2%)
Vipin K Parashar	3 (0.9%)
Daniel Axtens	3 (0.9%)
Andrew Donnellan	2 (0.6%)
Philippe Bergheaud	2 (0.6%)
Ananth N Mavinakayanahalli	2 (0.6%)
Vaibhav Jain	1 (0.3%)
Sam Mendoza-Jonas	1 (0.3%)
Adriana Kobylak	1 (0.3%)
Shreyas B. Prabhu	1 (0.3%)
Vaidyanathan Srinivasan	1 (0.3%)
Ian Munsie	1 (0.3%)

Developers with the most changed lines

Stewart Smith	19533 (39.4%)
Oliver O'Halloran	17920 (36.1%)
Alistair Popple	3285 (6.6%)
Daniel Axtens	2154 (4.3%)
Cyril Bur	2028 (4.1%)
Benjamin Herrenschmidt	941 (1.9%)
Neelesh Gupta	434 (0.9%)
Gavin Shan	294 (0.6%)
Russell Currey	261 (0.5%)
Vasant Hegde	245 (0.5%)
Cédric Le Goater	209 (0.4%)
Vipin K Parashar	155 (0.3%)
Shilpasri G Bhat	153 (0.3%)
Joel Stanley	140 (0.3%)
Vaidyanathan Srinivasan	135 (0.3%)
Michael Neuling	111 (0.2%)
Samuel Mendoza-Jonas	81 (0.2%)
Jeremy Kerr	60 (0.1%)
Mahesh Salgaonkar	58 (0.1%)
Vaibhav Jain	50 (0.1%)
Ananth N Mavinakayanahalli	43 (0.1%)
Shreyas B. Prabhu	17 (0.0%)
Sam Mendoza-Jonas	12 (0.0%)
Andrew Donnellan	10 (0.0%)
Ian Munsie	8 (0.0%)
Philippe Bergheaud	6 (0.0%)
Adriana Kobylak	6 (0.0%)

Developers with the most lines removed

Daniel Axtens	2149 (9.2%)
Shreyas B. Prabhu	17 (0.1%)
Andrew Donnellan	9 (0.0%)
Vipin K Parashar	2 (0.0%)

Developers with the most signoffs (total 190)

Stewart Smith	188 (98.9%)
Gavin Shan	1 (0.5%)
Neelesh Gupta	1 (0.5%)

Developers with the most reviews (total 34)

Patrick Williams	5 (14.7%)
Joel Stanley	5 (14.7%)
Cédric Le Goater	5 (14.7%)
Vasant Hegde	4 (11.8%)
Alistair Popple	4 (11.8%)
Sam Mendoza-Jonas	3 (8.8%)
Samuel Mendoza-Jonas	3 (8.8%)
Andrew Donnellan	2 (5.9%)
Cyril Bur	2 (5.9%)
Vaibhav Jain	1 (2.9%)

Developers with the most test credits (total 6)

Vipin K Parashar	3 (50.0%)
Vaibhav Jain	2 (33.3%)
Gajendra B Bandhu1	1 (16.7%)

Developers who gave the most tested-by credits (total 6)

Gavin Shan	2 (33.3%)
Ananth N Mavinakayanahalli	2 (33.3%)
Alistair Popple	1 (16.7%)
Stewart Smith	1 (16.7%)

Developers with the most report credits (total 11)

Vaibhav Jain	2 (18.2%)
Paul Nguyen	2 (18.2%)
Alistair Popple	1 (9.1%)
Cédric Le Goater	1 (9.1%)
Aneesh Kumar K.V	1 (9.1%)
Dionysius d. Bell	1 (9.1%)
Pradeep Ramanna	1 (9.1%)
John Walthour	1 (9.1%)
Benjamin Herrenschmidt	1 (9.1%)

Developers who gave the most report credits (total 11)

Gavin Shan	6 (54.5%)
Stewart Smith	3 (27.3%)
Samuel Mendoza-Jonas	1 (9.1%)
Shilpasri G Bhat	1 (9.1%)

4.1.26 skiboot-5.2.0-rc1

skiboot-5.2.0-rc1 was released on Friday Feb 26th, 2016.

skiboot-5.2.0-rc1 is the first release candidate of skiboot 5.2, which will become the new stable release of skiboot following the 5.1 release, first released August 17th, 2015.

skiboot-5.2.0-rc1 contains all bug fixes as of skiboot-5.1.13.

This is the second release that will follow the (now documented) Skiboot stable rules - see *Skiboot stable tree rules and releases*.

The current plan is to release skiboot-5.2.0 mid-March 2016, with a focus on bug fixing for future 5.2.0-rc releases.

New Features

Over skiboot-5.1, the following features have been added:

- Naples (P8', i.e. P8 with NVLINK) processor support, including NVLINK.
- Improvements in gard, libflash/pflash and opal-prd utilities
  - increased testing
  - increased usability
  - systemd scripts for opal-prd
  - pflash can now use the /dev/mtd device to access BMC flash rather than accessing it directly. It is *important* that you use –mtd if your BMC may otherwise know how to interact with its own flash.
- support for Micron N25Q256Ax and N25Qx256Ax NOR flash.
- support for Winbond W25Q256BV NOR flash
- support for an emulated ("fake") RTC clock, useful in simulators and during bringup
- Explicit 1:1 mapping in ranges properties have been added to PCI bridges. This allows a neat trick with offb and VGA ports that should probably not be told to young children.
- Added support to read the V2 format of the OCC-OPAL memory region, which supports Workload Optimized Frequency (WOF)

Changes in behavior

- Assigning OPAL IDs to PHBs is now fixed and based on the chip id and PHB index on that chip. On POWER7, we continue to use allocated numbers.
- We now query the BMC for BT capabilities rather than making assumptions

Removed support

- p5ioc2 is no longer supported. This affects a grand total of two POWER7 systems in the world.

NOTE: It is planned that skiboot-5.2 will be the last release supporting POWER7 machines.

Bugs fixed

- PHB3: Fix unexpected ER (all) on errinjct by PCI config
- hw/bt: timeout messages when BT interface isn't functional
- On Habanero, Slot3 should have been "Slot 3".
- We now completely flush the console buffer before power down and reboot
- For chips with ibm,occ-functional-state set to false, we don't wait for the OCC to start. This caused needless delay in booting on simulators which did not simulate OCCs.
- Change OCC reset order to always reset slave OCCs first.
- slw: Remove overwrites for EX_PM_CORE_ECO_VRET and EX_PM_CORE_PFET_VRET (these were already initialized in hostboot)
- p8-i2c: send stop bit on timeouts. Some devices can otherwise leave the bus in a held state.

Other improvements

- many fixes of compiler and static analysis warnings
- increased unit test coverage
- Unit test of "boot debian jessie installer"
- ability to plug in other simulators to run existing tests (e.g. simulator for non pegasus p8)
- Support using (patched) Qemu with PowerNV platform support for running unit tests.
- increased support for running with sparse
- We now build with -fstack-protector-strong if supported by the compiler
- We now build with -Werror for -Wformat
- pflash is now built as part of travis-ci and for Coverity Scan.
- There is now a RPM SPEC file that can be used as the basis for packaging skiboot and associated utilities.

Contributors

We have had a number of improvements in workflow over skiboot-5.1.0. Looking back, we have roughly the same number of changesets (372 for 5.1.0, 334 for 5.2.0-rc1 - even closer for 5.1.0-beta1) which indicates a relatively stable rate of development.

Complete statistics are included below (generated by gitdm), but I'd like to draw attention to a couple of stats:

Release	csets	Ack	Reviews	Tested	Reported
5.0	329	15	20	1	0
5.1	372	13	38	1	4
5.2-rc1	334	20	34	6	11

Overall, it looks like we're on the right trajectory for increasing the number of eyeballs looking at code before it heads in tree, especially around testing. Largely, this increase in Tested-by can be attributed to encouraging the existing test teams to start commenting on the patches themselves.

Anyway, here's the full stats from skiboot 5.1.0 to 5.2.0-rc1:

Processed 334 csets from 27 developers 2 employers found A total of 46172 lines added, 23274 removed (delta 22898)

Developers with the most changesets

Stewart Smith	146 (43.7%)
Cyril Bur	52 (15.6%)
Benjamin Herrenschmidt	15 (4.5%)
Joel Stanley	12 (3.6%)
Gavin Shan	12 (3.6%)
Alistair Popple	10 (3.0%)
Vasant Hegde	10 (3.0%)
Michael Neuling	10 (3.0%)
Russell Currey	9 (2.7%)
Cédric Le Goater	8 (2.4%)
Jeremy Kerr	8 (2.4%)
Samuel Mendoza-Jonas	6 (1.8%)
Neelesh Gupta	6 (1.8%)
Shilpasri G Bhat	4 (1.2%)
Oliver O'Halloran	4 (1.2%)
Mahesh Salgaonkar	4 (1.2%)
Vipin K Parashar	3 (0.9%)
Daniel Axtens	3 (0.9%)
Andrew Donnellan	2 (0.6%)
Philippe Bergheaud	2 (0.6%)
Ananth N Mavinakayanahalli	2 (0.6%)
Vaibhav Jain	1 (0.3%)
Sam Mendoza-Jonas	1 (0.3%)
Adriana Kobylak	1 (0.3%)
Shreyas B. Prabhu	1 (0.3%)
Vaidyanathan Srinivasan	1 (0.3%)
Ian Munsie	1 (0.3%)

Developers with the most changed lines

Stewart Smith	19533 (39.4%)
Oliver O'Halloran	17920 (36.1%)
Alistair Popple	3285 (6.6%)
Daniel Axtens	2154 (4.3%)
Cyril Bur	2028 (4.1%)
Benjamin Herrenschmidt	941 (1.9%)
Neelesh Gupta	434 (0.9%)
Gavin Shan	294 (0.6%)
Russell Currey	261 (0.5%)
Vasant Hegde	245 (0.5%)
Cédric Le Goater	209 (0.4%)
Vipin K Parashar	155 (0.3%)
Shilpasri G Bhat	153 (0.3%)
Joel Stanley	140 (0.3%)
Vaidyanathan Srinivasan	135 (0.3%)
Michael Neuling	111 (0.2%)
Samuel Mendoza-Jonas	81 (0.2%)
Jeremy Kerr	60 (0.1%)
Mahesh Salgaonkar	58 (0.1%)
Vaibhav Jain	50 (0.1%)
Ananth N Mavinakayanahalli	43 (0.1%)
Shreyas B. Prabhu	17 (0.0%)
Sam Mendoza-Jonas	12 (0.0%)
Andrew Donnellan	10 (0.0%)
Ian Munsie	8 (0.0%)
Philippe Bergheaud	6 (0.0%)
Adriana Kobylak	6 (0.0%)

Developers with the most lines removed

Daniel Axtens	2149 (9.2%)
Shreyas B. Prabhu	17 (0.1%)
Andrew Donnellan	9 (0.0%)
Vipin K Parashar	2 (0.0%)

Developers with the most signoffs (total 190)

Stewart Smith	188 (98.9%)
Gavin Shan	1 (0.5%)
Neelesh Gupta	1 (0.5%)

Developers with the most reviews (total 34)

Patrick Williams	5 (14.7%)
Joel Stanley	5 (14.7%)
Cédric Le Goater	5 (14.7%)
Vasant Hegde	4 (11.8%)
Alistair Popple	4 (11.8%)
Sam Mendoza-Jonas	3 (8.8%)
Samuel Mendoza-Jonas	3 (8.8%)
Andrew Donnellan	2 (5.9%)
Cyril Bur	2 (5.9%)
Vaibhav Jain	1 (2.9%)

Developers with the most test credits (total 6)

Vipin K Parashar	3 (50.0%)
Vaibhav Jain	2 (33.3%)
Gajendra B Bandhu1	1 (16.7%)

Developers who gave the most tested-by credits (total 6)

Gavin Shan	2 (33.3%)
Ananth N Mavinakayanahalli	2 (33.3%)
Alistair Popple	1 (16.7%)
Stewart Smith	1 (16.7%)

Developers with the most report credits (total 11)

Vaibhav Jain	2 (18.2%)
Paul Nguyen	2 (18.2%)
Alistair Popple	1 (9.1%)
Cédric Le Goater	1 (9.1%)
Aneesh Kumar K.V	1 (9.1%)
Dionysius d. Bell	1 (9.1%)
Pradeep Ramanna	1 (9.1%)
John Walthour	1 (9.1%)
Benjamin Herrenschmidt	1 (9.1%)

Developers who gave the most report credits (total 11)

Gavin Shan	6 (54.5%)
Stewart Smith	3 (27.3%)
Samuel Mendoza-Jonas	1 (9.1%)
Shilpasri G Bhat	1 (9.1%)

4.1.27 skiboot-5.2.0-rc2

skiboot-5.2.0-rc2 was released on Wednesday March 9th, 2016.

skiboot-5.2.0-rc2 is the second release candidate of skiboot 5.2, which will become the new stable release of skiboot following the 5.1 release, first released August 17th, 2015.

skiboot-5.2.0-rc2 contains all bug fixes as of skiboot-5.1.14.

This is the second release that will follow the (now documented) Skiboot stable rules - see *Skiboot stable tree rules and releases*.

The current plan is to release skiboot-5.2.0 mid-March 2016, with a focus on bug fixing for future 5.2.0-rc releases (if any - I hope this will be the last)

Over skiboot-5.2.0-rc1, we have the following changes:

New platform!

- Add Barreleye platform

Generic

- hw/p8-i2c: Speed up SMBUS_WRITE
- Fix early backtraces

FSP Platforms

- fsp-sensor: rework device tree for sensors
- platforms/firenze: Fix I2C clock source frequency

Simics simulator

- Enable Simics UART console

Mambo simulator

- platforms/mambo: Add terminate callback
  - fix hang in multi-threaded mambo
  - add multithreaded mambo tests

IPMI

- hw/ipmi: fix event data 1 for System Firmware Progress sensor
- ipmi: Log exact NetFn value in OPAL logs

AST BMC based platforms

- hw/bt: allow BT driver to use different buffer size

opal-prd utility

- **opal-prd: Add debug output for firmware-driven OCC events** We indicate when we have a user-driven event, so add corresponding outputs for firmware-driven ones too.

getscom utility

- Add Naples chip support

4.1.28 skiboot-5.2.1

skiboot-5.2.1 was released on Wednesday April 27th, 2016.

skiboot-5.2.1 is the second stable release of skiboot 5.2, the new stable release of skiboot, which will take over from the 5.1.x series which was first released August 17th, 2015.

skiboot-5.2.1 contains all bug fixes as of skiboot-5.1.15.

This is the second release that will follow the (now documented) Skiboot stable rules - see *Skiboot stable tree rules and releases*.

Changes

Over skiboot-5.2.0, the following fixes are included:

pflash

- Allow building under yocto. Makefile fixes to enable building as part of an OpenBMC build.

Garrison platform

- Add PCIe and NPU slot location names
- hw/npu.c: Add ibm, npu-index property to npu device tree
- hmi: Add handling for NPU checkstops

PHB3 (all POWER8 platforms)

- hw/phb3: Ensure PQ bits are cleared in the IVC when masking IRQ When we mask an interrupt, we may race with another interrupt coming in from the hardware. If this occurs, the P and/or Q bit may end up being set but we never EOI/clear them. This could result in a lost interrupt or the next interrupt that comes in after re-enabling never being presented.

  This fixes a bug seen with some CAPI workloads which have lots of interrupt masking at the same time as high interrupt load. The fix is not specific to CAPI though.

- hw/phb3: Fix potential race in EOI When we EOI we need to clear the present (P) bit in the Interrupt Vector Cache (IVC). We must clear P ensuring that any additional interrupts that come in aren't lost while also maintaining coherency with the Interrupt Vector Table (IVT).

  To do this, the hardware provides a conditional update bit in the IVC. This bit ensures that generation counts between the IVT and the IVC updates are synchronised.

Unfortunately we never set this the bit to conditionally update the P bit in the IVC based on the generation count. Also, we didn't set what we wanted the new generation count to be if the update was successful.

FSP platforms

- OPAL:Handle mbox response with bad status:0x24 during FSP termination OPAL committed a predictive log with SRC BB822411 in some situations.

Generic

- hmi: Fix a bug where partial hmi event was reported to host. This bug fix ensures the CPU PIR is reported correctly:

```
  [  305.628283] Fatal Hypervisor Maintenance interrupt [Not recovered]
  [  305.628341]  Error detail: Malfunction Alert
  [  305.628388]    HMER: 8040000000000000
- [  305.628423]    CPU PIR: 00000000
+ [  200.123021]    CPU PIR: 000008e8
  [  305.628458]    [Unit: VSU] Logic core check stop
```

- xscom: Return OPAL_WRONG_STATE on XSCOM ops if CPU is asleep

Contributors

Processed 15 csets from 7 developers A total of 436 lines added, 59 removed (delta 377)

Developers with the most changesets

Russell Currey	7 (46.7%)
Alistair Popple	2 (13.3%)
Michael Neuling	2 (13.3%)
Patrick Williams	1 (6.7%)
Stewart Smith	1 (6.7%)
Mamatha	1 (6.7%)
Mahesh Salgaonkar	1 (6.7%)

Developers with the most changed lines

Alistair Popple	215 (48.3%)
Russell Currey	140 (31.5%)
Michael Neuling	55 (12.4%)
Mamatha	15 (3.4%)
Patrick Williams	9 (2.0%)
Mahesh Salgaonkar	8 (1.8%)
Stewart Smith	3 (0.7%)

Developers with the most lines removed

Patrick Williams	5 (8.5%)

Developers with the most signoffs (total 30)

Stewart Smith	15 (50.0%)
Russell Currey	7 (23.3%)
Michael Neuling	2 (6.7%)
Alistair Popple	2 (6.7%)
Patrick Williams	1 (3.3%)
Oliver O'Halloran	1 (3.3%)
Mahesh Salgaonkar	1 (3.3%)
Mamatha	1 (3.3%)

Developers with the most reviews (total 11)

Alistair Popple	5 (45.5%)
Andrew Donnellan	3 (27.3%)
Mahesh Salgaonkar	2 (18.2%)
Joel Stanley	1 (9.1%)

Developers with the most Acked-by (total 1)

Alistair Popple	1 (100.0%)

Developers with the most test credits (total 3)

Andrew Donnellan	2 (66.7%)
Vaibhav Jain	1 (33.3%)

Developers who received the most tested-by credits (total 3)

Michael Neuling	3 (100.0%)

4.1.29 skiboot-5.2.2

skiboot-5.2.2 was released on Thursday May 5th, 2016.

skiboot-5.2.2 is the third stable release of skiboot 5.2, the new stable release of skiboot, which will take over from the 5.1.x series which was first released August 17th, 2015.

Skiboot 5.2.2 replaces skiboot-5.2.1 as the current stable version, which was released on April 27th, 2016. Over skiboot-5.2.1, skiboot 5.2.2 contains one bug fix targeted at P8NVL systems, notably the Garrison platform.

skiboot-5.2.2 contains all bug fixes as of skiboot-5.1.16.

This is the second release that will follow the (now documented) Skiboot stable rules - see *Skiboot stable tree rules and releases*.

Over skiboot-5.2.1, the following fixes are included:

P8NVL/Garrison

- **PHB3: Fix corruption of pref window register** On P8+ Garrison platform, the root port's pref window register might be not writable and we have to emulate the window because of hardware defect. In order to detect that, we read the register content, write inversed value and read the register content again. The register is regarded as read-only if the values from the two continuous read are same. However, the original register content isn't written back and it causes corruption on pref window register if it's writable.

  This fixes the above issue by writing the original content back to the register at the end.

4.1.30 skiboot-5.2.3

skiboot-5.2.3 was released on Thursday June 30th, 2016.

skiboot-5.2.3 is the 4th stable release of skiboot 5.2, the new stable release of skiboot, which takes over from the 5.1.x series which was first released August 17th, 2015.

Skiboot 5.2.3 replaces skiboot-5.2.2 as the current stable version, which was released on May 5th, 2016. Over skiboot-5.2.2, skiboot 5.2.3 contains one important bug fix regarding parsing data from the OCC regarding CPU frequency tables, which could lead to no CPU frequency scaling.

skiboot-5.2.3 contains all bug fixes as of skiboot-5.1.16.

This is the second release that will follow the (now documented) Skiboot stable rules - see *Skiboot stable tree rules and releases*.

Over skiboot-5.2.2, the following fixes are included:

OpenPOWER platforms

- occ: Filter out entries from Pmin to Pmax in pstate table (cherry picked from commit eca02ee2e62cee115d921a01cea061782ce47cc7) Without this fix, with newer OCC firmware on some OpenPOWER machines, we would fail to parse the table from the OCC, which meant the host OS would not get a table of supported CPU frequencies.

General

- pci: Do a dummy config write to devices to establish bus number (cherry picked from commit f46c1e506d199332b0f9741278c8ec35b3e39135)

  On PCI Express, devices need to know their own bus number in order to provide the correct source identification (aka RID) in upstream packets they might send, such as error messages or DMAs.

  However while devices know (and hard wire) their own device and function number, they know nothing about bus numbers by default, those are decoded by bridges for routing. All they know is that if their parent bridge sends a "type 0" configuration access, they should decode it provided the device and function numbers match.

  The PCIe spec thus defines that when a device receive such a configuration access and it's a write, it should "capture" the bus number in the source field of the packet, and re-use as the originator bus number of all subsequent outgoing requests.

In order to ensure that a device has this bus number firmly established before it's likely to send error packets upstream, we should thus do a dummy configuration write to it as soon as possible after probing.

- Fix GCC 6 warning in backtrace code (cherry picked from commit 793f6f5b32c96f2774bd955b6062c74a672317ca)
- Backport of user visible typo fixes partial cherry picked from 4c95b5e04e3c4f72e4005574f67cd6e365d3276f

Utilities

- Fix ARM build failure with parallel make

4.1.31 skiboot-5.2.4

skiboot-5.2.4 was released on Tuesday July 12th, 2016.

This is the 5th stable release of skiboot 5.2, the new stable release of skiboot (first release with 5.2.0 on March 16th 2016).

Skiboot 5.2.4 replaces skiboot-5.2.3 as the current stable version, which was released on June 30th 2016. Over skiboot-5.2.3, skiboot 5.2.4 contains bug fixes to make skiboot more resilient to errors in the XSCOM engine and some build improvements for the pflash utility.

skiboot-5.2.4 contains all bug fixes as of skiboot-5.1.16.

This is the second release that will follow the (now documented) Skiboot stable rules - see *Skiboot stable tree rules and releases*.

Over skiboot-5.2.3, the following fixes are included:

All platforms

- Make the XSCOM engine code more resilient to errors:
  - hw/xscom: Reset XSCOM engine after querying sleeping core FIR
  - hw/xscom: Reset XSCOM engine after finite number of retries when busy

Userspace utilities

- pflash build improvements

4.1.32 skiboot-5.2.5

skiboot-5.2.5 was released on Thursday July 28th, 2016.

skiboot-5.2.5 contains all bug fixes as of skiboot-5.1.17.

This is the second release that will follow the (now documented) Skiboot stable rules - see *Skiboot stable tree rules and releases*.

Over skiboot-5.2.4, the following fixes are included:

- pflash: Fix the makefile (cherry picked from commit fd599965f723330da5ec55519c20cdb6aa2b3a2d)

- pflash: Clean up makefiles and resolve build race (cherry picked from commit c327eddd9b291a0e6e54001fa3b1e547bad3fca2)
- FSP/ELOG: Fix OPAL generated elog resend logic (cherry picked from commit a6d4a7884e95cb9c918b8a217c11e46b01218358)
- FSP/ELOG: Fix possible event notifier hangs (cherry picked from commit e7c8cba4ad773055f390632c2996d3242b633bf4)
- FSP/ELOG: Disable event notification if list is not consistent (cherry picked from commit 1fb10de164d3ca034193df81c1f5d007aec37781)
- FSP/ELOG: Improve elog event states (cherry picked from commit cec5750a4a86ff3f69e1d8817eda023f4d40c492)
- FSP/ELOG: Fix OPAL generated elog event notification (cherry picked from commit ec366ad4e2e871096fa4c614ad7e89f5bb6f884f)
- FSP/ELOG: Disable event notification during kexec (cherry picked from commit d2ae07fd97bb9408456279cec799f72cb78680a6)
- hw/xscom: Reset XSCOM engine after querying sleeping core FIR (cherry picked from commit 15cec493804ff14e6246eb1b65e9d0c7cb469a81)
- hw/xscom: Reset XSCOM engine after finite number of retries when busy (cherry picked from commit e761222593a1ae932cddbc81239b6a7cd98ddb70)
- xscom: Return OPAL_WRONG_STATE on XSCOM ops if CPU is asleep (cherry picked from commit 9c2d82394fd2303847cac4a665dee62556ca528a)
- fsp/console: Ignore data on unresponsive consoles (cherry picked from commit fd6b71fcc6912611ce81f455b4805f0531699d5e)
- SEL: Fix eSEL ID while logging eSEL event

4.1.33 skiboot-5.3.0

skiboot-5.3.0 was released on Tuesday August 2nd, 2016.

skiboot-5.3.0 is the first stable release of skiboot 5.3, the new stable release of skiboot, which will take over from the 5.2.x series which was first released Wednesday March 16th, 2016.

skiboot-5.3.0 contains all bug fixes as of skiboot-5.1.17 and skiboot-5.2.5.

Changes over skiboot-5.3.0-rc2: - Adopt libtool rules for soname versioning for libflash

See skiboot-5.3.0-rc2 and skiboot-5.3.0-rc1 release notes for a complete list of changes from skiboot-5.2.0.

4.1.34 skiboot-5.3.0-rc1

skiboot-5.3.0-rc1 was released on Monday July 25th, 2016

skiboot-5.3.0-rc1 is the first release candidate of skiboot 5.3, which will become the new stable release of skiboot following the 5.2 release, first released March 16th 2016.

skiboot-5.3.0-rc1 contains all bug fixes as of skiboot-5.1.16 and skiboot-5.2.4 (the existing stable releases).

For how the skiboot stable releases work, see *Skiboot stable tree rules and releases*.

The current plan is to release skiboot-5.3.0 August 1st 2016.

Over skiboot-5.2, we have the following changes:

OPAL API/Device Tree

- **Reserve OPAL API numbers for XICS emulation for XIVE** Additionally, we put in some skeleton docs for what's coming, key points being that this is for P9 and above, relies on a device being present in the device tree and is modelled on the PAPR calls.
- interrupts: Remove #interrupt-cells from ICP nodes
- Stop adding legacy linux, phandle to device tree, just add phandle No Linux kernel has ever existed for powernv that only knows linux,phandle.

POWER9

- Add base POWER9 support In *NO WAY* is this geared towards real POWER9 hardware. Suitable for use in simulators *only*, and even then, only if you intensely know what you're doing.
- Document changes in OPAL API for POWER9 Some things are going to change, we start documenting them.
- cpu: supply ibm,dec-bits via devicetree
- power9: Add example device tree for phb4
- device-tree: Only advertise ibm, opal-v3 (not v2) on POWER9 and above

CAPI

- phb3: Test CAPI mode on both CAPP units on Naples
- hmi: Recover both CAPP units on Naples after malfunction alert
- chiptod: Sync timebase in both CAPP units on Naples
- phb3: Set CAPI mode for both CAPP units on Naples
- phb3: Load CAPP ucode to both CAPP units on Naples
- **phb3: Add support for CAPP DMA mode** The XSL used in the Mellanox CX4 card uses a DMA mode of CAPI, which requires a few registers configured specially. This adds a new mode to the OPAL_PCI_SET_PHB_CAPI_MODE API to enable CAPI in DMA mode.

PCI

- pci: Do a dummy config write to devices to establish bus number
- phb: Work around XSL bug sending PTE updates with wrong scope
- Support for PCI hotplug (if a platform supports it)

Garrison

- NVLink/NPU support
- Full garrison platform support.

BMC based platforms

- bt: use the maximum retry count returned by the BMC
- **SEL: Fix eSEL ID while logging eSEL event** Commit 127a7dac added eSEL ID to SEL event in reverse order (0700 instead of 0007). This code fixes this issue by adding ID in proper order.

Tests/Simulation

- test/hello_world: always use shutdown type zero
- make check: make test runs less noisy
- boot-tests: force booting from primary (non-golden) side
- mambo: Enable multicore configurations
- mambo: Flatten device tree at the end
- mambo: Increase memory to 4GB and change memory map
- Timebase quirk for slow simulators like AWAN and SIMICS
- chip: Add simics specific quirks
- mambo: Flash driver using bogus disk
- platform/mambo: Add a heartbeat time, making console more responsive
- mambo: Fix bt command and add little endian support

FSP platforms

- beginnings of support for SPIRA-S structure
- Handle mbox response with bad status:0x24 during FSP termination
- FSP: Validate fsp_msg response memory allocation
- FSP/ELOG: Fix OPAL generated elog event notification
- FSP/ELOG: Disable event notification during kexec Possible crash if error log timing around kexec is unfortunate
- fsp/console: Ignore data on unresponsive consoles

  Linux kernels from v4.1 onwards will try to request an irq for each hvc console using OPAL_EVENT_CONSOLE_INPUT, however because the IRQF_SHARED flag is not set any console after the first will fail. If there is data on one of these failed consoles OPAL will set OPAL_EVENT_CONSOLE_INPUT every time fsp_console_read is called, leading to RCU stalls in the kernel.

  As a workaround for unpatched kernels, cease setting OPAL_EVENT_CONSOLE_INPUT for consoles that we have noticed are not being read.

HMI

- hmi: Fix a bug where partial hmi event was reported to host.
- hmi: Add handling for NPU checkstops
- hmi: Only raise a catchall HMI if no other components have

- hmi: Rework HMI event handling of FIR read failure

Tools

- **external: Add a getsram command** The getsram command reads the OCC SRAM. This is useful for debug.
- bug fixes in flash utilities (pflash/gard)
- pflash: Allow building under yocto.
- external/opal-prd: Ensure that struct host_interfaces matches the thunk
- external/pflash: Handle incorrect cmd-line options better
- libflash: fix bug on reading truncated flash file
- pflash: add support for manipulating file rather than flash
- gard: fix compile error on ARM
- libflash: Add sanity checks to ffs init code.
- external: Add dynamically linked pflash

Mambo

- **Test device tree for kernel location** This can reduce the boot time since the kernel no longer needs to relocate itself when loaded directly at 0.

Generic

- hw/lpc: Log LPC SYNC errors as OPAL_PLATFORM_ERR_EVT errors
- Explicitly disable the attn instruction on all CPUs on boot.
- hw/xscom: Reset XSCOM engine after finite number of retries when busy
- hw/xscom: Reset XSCOM engine after querying sleeping core FIR
- core/timer: Add support for platform specific heartbeat
- Fix GCOV_COUNTERS ifdef logic for GCC 6.0
- core: Fix backtrace for gcc 6 fixes a compiler warning on GCC 6 and above
- **cpu: Don't call time_wait with lock held** Also make the locking around re-init safer, properly block the OS from restarting a thread that was caught for re-init.
- flash: Increase the maximum number of flash devices

Contributors

Extending the analysis done for the last few releases, we can see our trends in code review across versions:

Release	csets	Ack	Reviews	Tested	Reported
5.0	329	15	20	1	0
5.1	372	13	38	1	4
5.2-rc1	334	20	34	6	11
5.3-rc1	302	36	53	4	5

An increase in reviews this cycle is great!

Detailed statistics for 5.3.0-rc1 are below:

Processed 302 csets from 31 developers A total of 20887 lines added, 4540 removed (delta 16347)

Developers with the most changesets

Stewart Smith	82 (27.2%)
Gavin Shan	36 (11.9%)
Benjamin Herrenschmidt	28 (9.3%)
Michael Neuling	25 (8.3%)
Vasant Hegde	24 (7.9%)
Russell Currey	14 (4.6%)
Brad Bishop	12 (4.0%)
Vipin K Parashar	10 (3.3%)
Cédric Le Goater	9 (3.0%)
Shreyas B. Prabhu	8 (2.6%)
Jeremy Kerr	7 (2.3%)
Philippe Bergheaud	6 (2.0%)
Cyril Bur	5 (1.7%)
Mukesh Ojha	4 (1.3%)
Alistair Popple	4 (1.3%)
Ian Munsie	4 (1.3%)
Oliver O'Halloran	3 (1.0%)
Chris Smart	3 (1.0%)
Sam Mendoza-Jonas	2 (0.7%)
Joel Stanley	2 (0.7%)
Dinar Valeev	2 (0.7%)
Shilpasri G Bhat	2 (0.7%)
Patrick Williams	2 (0.7%)
Deb McLemore	1 (0.3%)
Balbir Singh	1 (0.3%)
Andrew Donnellan	1 (0.3%)
Suraj Jitindar Singh	1 (0.3%)
Frederic Bonnard	1 (0.3%)
Kamalesh Babulal	1 (0.3%)
Mamatha	1 (0.3%)
Mahesh Salgaonkar	1 (0.3%)

Developers with the most changed lines

Benjamin Herrenschmidt	7491 (34.4%)
Gavin Shan	4821 (22.1%)
Vasant Hegde	4740 (21.7%)
Stewart Smith	1294 (5.9%)
Michael Neuling	620 (2.8%)
Cédric Le Goater	470 (2.2%)
Jeremy Kerr	338 (1.6%)
Shreyas B. Prabhu	330 (1.5%)

Continued on next page

Table 4.2 – continued from previous page

Vipin K Parashar	305 (1.4%)
Russell Currey	295 (1.4%)
Alistair Popple	229 (1.1%)
Philippe Bergheaud	170 (0.8%)
Ian Munsie	133 (0.6%)
Dinar Valeev	126 (0.6%)
Brad Bishop	80 (0.4%)
Oliver O'Halloran	80 (0.4%)
Cyril Bur	62 (0.3%)
Frederic Bonnard	61 (0.3%)
Sam Mendoza-Jonas	32 (0.1%)
Chris Smart	27 (0.1%)
Shilpasri G Bhat	20 (0.1%)
Patrick Williams	18 (0.1%)
Suraj Jitindar Singh	17 (0.1%)
Mamatha	15 (0.1%)
Mukesh Ojha	8 (0.0%)
Mahesh Salgaonkar	8 (0.0%)
Joel Stanley	4 (0.0%)
Balbir Singh	4 (0.0%)
Kamalesh Babulal	2 (0.0%)
Deb McLemore	1 (0.0%)
Andrew Donnellan	1 (0.0%)

Developers with the most lines removed

Dinar Valeev	68 (1.5%)
Patrick Williams	10 (0.2%)
Mukesh Ojha	4 (0.1%)
Kamalesh Babulal	1 (0.0%)

Developers with the most signoffs (total 249)

Stewart Smith	236 (94.8%)
Vaidyanathan Srinivasan	6 (2.4%)
Benjamin Herrenschmidt	3 (1.2%)
Michael Neuling	2 (0.8%)
Oliver O'Halloran	1 (0.4%)
Vipin K Parashar	1 (0.4%)

Developers with the most reviews (total 53)

Andrew Donnellan	11 (20.8%)
Russell Currey	9 (17.0%)
Joel Stanley	7 (13.2%)
Alistair Popple	7 (13.2%)
Mukesh Ojha	5 (9.4%)
Cyril Bur	3 (5.7%)
Mahesh Salgaonkar	2 (3.8%)
Gavin Shan	2 (3.8%)
Vasant Hegde	2 (3.8%)
Stewart Smith	1 (1.9%)
Vaidyanathan Srinivasan	1 (1.9%)
Vipin K Parashar	1 (1.9%)
Frederic Barrat	1 (1.9%)
Cédric Le Goater	1 (1.9%)

Developers with the most test credits (total 4)

Andrew Donnellan	2 (50.0%)
Russell Currey	1 (25.0%)
Vaibhav Jain	1 (25.0%)

Developers who gave the most tested-by credits (total 4)

Michael Neuling	3 (75.0%)
Gavin Shan	1 (25.0%)

Developers with the most report credits (total 5)

Mukesh Ojha	2 (40.0%)
Russell Currey	1 (20.0%)
Pridhiviraj Paidipeddi	1 (20.0%)
Balbir Singh	1 (20.0%)

Developers who gave the most report credits (total 5)

Gavin Shan	2 (40.0%)
Stewart Smith	2 (40.0%)
Vasant Hegde	1 (20.0%)

4.1.35 skiboot-5.3.0-rc2

skiboot-5.3.0-rc2 was released on Thursday July 28th, 2016.

The current plan is to release skiboot-5.3.0 August 1st 2016.

Over skiboot-5.3.0-rc1, we have the following changes:

pflash

- pflash: Clean up makefiles and resolve build race
- pflash: use atexit for musl compatibility

General

- core/flash: Fix passing pointer instead of value

POWER9

- **mambo: Update Radix Tree Size as per ISA 3.0** In Linux we recently changed to this encoding, so we no longer boot. The associated Linux commit is b23d9c5b9c83c05e013aa52460f12a8365062cf4

FSP Platforms

- platforms/ibm-fsp: Fix incorrect struct member access and comparison
- FSP/MDST: Fix TCE alignment issue In some corner cases (like source memory size = 4097) we may endup doing wrong mapping and corrupting part of SYSDUMP.
- hdat/vpd: Add chip-id property to processor chip node under vpd

CAPI

- hw/phb3: Increase AIB TX command credit for DMA read in CAPP DMA mode

4.1.36 skiboot-5.3.1

skiboot-5.3.1 was released on Wednesday August 10th, 2016.

This is the 2nd stable release of skiboot 5.3, the new stable release of skiboot (first released with 5.3.0 on August 2nd, 2016).

Skiboot 5.3.1 replaces skiboot-5.3.0 as the current stable version. It contains a few minor bug fixes.

This release follows the Skiboot stable rules, see *Skiboot stable tree rules and releases*.

Over skiboot-5.3.0, the following fixes are included:

FSP systems:

- **FSP/ELOG: elog_enable flag should be false by default** This issue is one of the corner case, which is related to recent change went upstream and only observed in the petitboot prompt, where we see only one error log instead of getting all error log in /sys/firmware/opal/elog.

NVLink systems (i.e. Garrison):

- **npu: reword "error" to indicate it's actually a warning** Without this patch, you get spurious FirmWare Test Suite (FWTS) warnings about NVLink not working on machines that aren't fully populated with GPUs.
- **hmi: Clean up NPU FIR debug messages** With the skiboot log set to debug, the FIR (and related registers) were logged all in the same message. It was too much for one line, didn't clarify if the numbers were in hex, and didn't show leading zeroes.

General:

- asm: Fix backtrace for unexpected exception
- correct the log level from PR_ERROR down to PR_INFO for some skiboot log messages.

4.1.37 skiboot-5.3.2

skiboot-5.3.2 was released on Friday August 26th, 2016.

This is the 3rd stable release of skiboot 5.3, the new stable release of skiboot (first released with 5.3.0 on August 2nd, 2016).

Skiboot 5.3.2 replaces skiboot-5.3.1 as the current stable version. It contains a few minor bug fixes.

Over skiboot-5.3.1, the following fixes are included:

- opal/hmi: Fix a TOD HMI failure during a race condition. Rare race condition which meant we wouldn't recover from TOD error
- lpc: Log LPC SYNC errors as unrecoverable ones for manufacturing Only affects systems in manufacturing mode. No behaviour change when not in manufacturing mode.
- hw/phb3: Update capi initialization sequence The capi initialization sequence was revised in a circumvention document when a 'link down' error was converted from fatal to Endpoint Recoverable. Other, non-capi, register setup was corrected even before the initial open-source release of skiboot, but a few capi-related registers were not updated then, so this patch fixes it. The point is that a link-down error detected by the UTL logic will lead to an AIB fence, so that the CAPP unit can detect the error.

4.1.38 skiboot-5.3.3

skiboot-5.3.3 was released on Friday September 2nd, 2016.

This is the 4th stable release of skiboot 5.3, the new stable release of skiboot (first released with 5.3.0 on August 2nd, 2016).

Skiboot 5.3.3 replaces skiboot-5.3.2 as the current stable version. It contains two bug fixes for machines utilizing the NPU (i.e. Garrison)

Over skiboot-5.3.2, the following fixes are included:

- hw/npu: assert the NPU irq min is aligned.
- hw/npu: program NPU BUID reg properly The NPU BUID register was incorrectly programmed resulting in npu interrupt level 0 causing a PB_CENT_CRESP_ADDR_ERROR checkstop, and irqs from npus in odd chips being aliased to and processed as the interrupts from the corresponding npu on the even chips.

4.1.39 skiboot-5.3.4

skiboot-5.3.4 was released on Tuesday September 13th, 2016.

This is the 5th stable release of skiboot 5.3, the new stable release of skiboot (first released with 5.3.0 on August 2nd, 2016).

Skiboot 5.3.4 replaces skiboot-5.3.3 as the current stable version. It contains a couple of bug fixes, specifically around failing XSCOMs.

Over skiboot-5.3.3, the following fixes are included:

- xscom: Initialize the data to a known value in xscom_read In case of error, don't leave the data random. It helps debugging when the user fails to check the error code. This happens due to a bug in the PRD wrapper app.

- xscom: Map all HMER status codes to OPAL errors
- centaur: Mark centaur offline after 10 consecutive access errors This avoids spamming the logs when the centaur is dead and PRD constantly tries to access it
- nvlink: Fix bad PE number check in error inject code path (<= rather than <)

4.1.40 skiboot-5.3.5

skiboot-5.3.5 was released on Wednesday September 14th, 2016.

This is the 6th stable release of skiboot 5.3, the new stable release of skiboot (first released with 5.3.0 on August 2nd, 2016).

Skiboot 5.3.5 replaces skiboot-5.3.4 as the current stable version. It contains a couple of minor bug fixes: simply clarifying two error messages.

Over skiboot-5.3.4, the following fixes are included:

- centaur: print message on disabling xscoms to centaur due to many errors
- slw: improve error message for SLW timer stuck We still register dump, but only to in memory console buffer by default.

4.1.41 skiboot-5.3.6

skiboot-5.3.6 was released on Saturday September 17th, 2016.

This is the 7th stable release of skiboot 5.3, the new stable release of skiboot (first released with 5.3.0 on August 2nd, 2016).

Skiboot 5.3.6 replaces skiboot-5.3.5 as the current stable version. It contains one minor bug fix.

Over skiboot-5.3.5, the following fixes are included:

- SLW: Actually print the register dump only to memory A fix in 5.3.5 was only partially correct, we still had the log priority incorrect for dumping of the SLW registers.

4.1.42 skiboot-5.3.7

skiboot-5.3.7 was released on Wednesday October 12th, 2016.

This is the 8th stable release of skiboot 5.3, the new stable release of skiboot (first released with 5.3.0 on August 2nd, 2016).

Skiboot 5.3.7 replaces skiboot-5.3.6 as the current stable version. It contains a few bugfixes, including an important PCI bug fix that could cause some adapters to not be detected.

Over skiboot-5.3.6, the following fixes are included:

PCI:

- **pci: Avoid hot resets at boot time** In the PCI post-fundamental reset code, a hot reset is performed at the end. This is causing issues at boot time as a reset signal is being sent downstream before the links are up, which is causing issues on adapters behind switches. No errors result in skiboot, but the adapters are not usable in Linux as a result.

  This patch fixes some adapters not being configurable in Linux on some systems. The issue was not present in skiboot 5.2.x.

- **core/pci: Fix the power-off timeout in pci_slot_power_off()** The timeout should be 1000ms instead of 1000 ticks while powering off PCI slot in pci_slot_power_off(). Otherwise, it's likely to hit timeout powering off the PCI slot as below skiboot logs reveal:

  [47912590456,5] SkiBoot skiboot-5.3.6 starting... (snip) [5399532365,7] PHB#0005:02:11.0 Bus 0f..ff scanning... [5399540804,7] PHB#0005:02:11.0 No card in slot [5399576870,5] PHB#0005:02:11.0 Timeout powering off slot [5401431782,3] FIRENZE-PCI: Wrong state 00000000 on slot 8000000002880005

PRD:

- **occ/prd/opal-prd: Queue OCC_RESET event message to host in OpenPOWER** During an OCC reset cycle the system is forced to Psafe pstate. When OCC becomes active, the system has to be restored to its last pstate as requested by host. So host needs to be notified of OCC_RESET event or else system will continue to remian in Psafe state until host requests a new pstate after the OCC reset cycle.
- **opal-prd: Fix error code from scom_read & scom_write** Currently, we always return a zero value from scom_read & scom_write, so the HBRT implementation has no way of detecting errors during scom operations. This change uses the actual return value from the scom operation from the kernel instead.
- **opal-prd: Add get_interface_capabilities to host interfaces** We need a way to indicate behaviour changes & fixes in the prd interface, without requiring a major version bump.

  This change introduces the get_interface_capabilities callback, returning a bitmask of capability flags, pertaining to 'sets' of capabilities. We currently return 0 for all.

IBM FSP Platforms:

- platforms/firenze: Fix clock frequency dt property
- platforms/firence: HDAT: Fix typo in nest-frequency property

NVLink:

- **hw/npu.c: Fix reserved PE#** Currently the reserved PE is set to NPU_NUM_OF_PES, which is one greater than the maximum PE resulting in the following kernel errors at boot:

  [0.000000] pnv_ioda_reserve_pe: Invalid PE 4 on PHB#4 [0.000000] pnv_ioda_reserve_pe: Invalid PE 4 on PHB#5

  Due to a HW errata PE#0 is already reserved in the kernel, so update the opal-reserved-pe device-tree property to match this.

4.1.43 skiboot-5.4.0

skiboot-5.4.0 was released on Friday November 11th 2016. It is the new stable skiboot release, taking over from the 5.3.x series (first released August 2nd, 2016). It comes after four release candidates, which have helped to shake out a few issues.

skiboot-5.4.0 contains all bug fixes as of *skiboot-5.3.7* and *skiboot-5.1.18* (the currently maintained stable releases).

Skiboot 5.4.x becomes the new stable release. For how the skiboot stable releases work, see *Skiboot stable tree rules and releases* for details.

Over *skiboot-5.4.0-rc4*, we have a few changes:

- libstb: bump up the byte timeout for tpm i2c requests

  This bumps up the byte timeout for tpm i2c requests from 10ms to 30ms. Some p8dtu systems are getting i2c request timeout.

- external/pflash: Perform the correct cleanup when -F is used to operate on a file.

- Add SuperMicro p8dtu1u and p8dtu2u platforms
- Revert "core/ipmi: Set interrupt-parent property". This reverts commit d997e482705d9fdff8e25fcbe07fb56008f96ae1 (introduced in 5.4.0-rc1)

  A problem was found with pre 4.2 linux kernels where a spurious WARNING would be emitted. This change doesn't matter enough to scare users so we can just revert it.

```
Warning was:
[    0.947741] irq: irq-62==>hwirq-0x3e mapping failed: -22
[    0.947793] ------------[ cut here ]------------
[    0.947838] WARNING: at kernel/irq/irqdomain.c:485
```

- libflash/libffs: Fix possible NULL dereference

Previous Release Candidates

There were four release candidates for skiboot 5.4.0:

- *skiboot-5.4.0-rc4*
- *skiboot-5.4.0-rc3*
- *skiboot-5.4.0-rc2*
- *skiboot-5.4.0-rc1*

Changes since skiboot 5.3

Over skiboot-5.3, we have the following changes:

New Features

- Add SuperMicro p8dtu1u and p8dtu2u platforms
- Initial Trusted Boot support (see *Secure and Trusted Boot Overview*). There are several limitations with this initial release:
  - Only Nuvoton TPM 2.0 is supported
  - Requires hardware rework on late revision Habanero or Firestone boards in order to install TPM.
  - Add i2c Nuvoton TPM 2.0 Driver
  - romcode driver for POWER8 secure ROM
  - See Device tree docs: *Trusted Platform Module (TPM)* and *ibm,secureboot*
  - See *Secure and Trusted Boot Overview*
- Support `ibm,skiboot` NVRAM partition with skiboot configuration options.
  - These should generally only be used if you either completely know what you are doing or need to work around a skiboot bug. They are **not** intended for end users and are *explicitly* **NOT ABI**.
  - Add support for supplying the kernel boot arguments from the `bootargs` configuration string in the `ibm,skiboot` NVRAM partition.
  - Enabling the experimental fast reset feature is done via this method.
- Add support for nap mode on P8 while in skiboot

  - While nap has been exposed to the Operating System since day 1, we have not utilized low power states when in skiboot itself, leading to higher power consumption during boot. We only enable the functionality after the 0x100 vector has been patched, and we disable it before transferring control to Linux.

- libflash: add 128MB MX66L1G45G part
- Pointer validation of OPAL API call arguments.
  - If the kernel called an OPAL API with vmalloc'd address or any other address range in real mode, we would hit a problem with aliasing. Since the top 4 bits are ignored in real mode, pointers from 0xc.. and 0xd.. (and other ranges) could collide and lead to hard to solve bugs. This patch adds the infrastructure for pointer validation and a simple test case for testing the API
  - The checks validate pointers sent in using `opal_addr_valid()`
- Fast reboot for P8

  This makes reboot take an *awful* lot less time, somewhere between four and ten times faster than a full IPL. It is currently experimental and not enabled by default. You can enable the experimental support via nvram option:

```
# nvram -p ibm,skiboot --update-config experimental-fast-reset=feeling-lucky
```

  WARNING: While we *think* we've managed to work out or around most of the kinks with fast-reset, we are *not* enabling it by default in 5.4.

  Notably, fast reset will *not* happen in the following scenarios:

  - platform error

    Most of the time, if we're rebooting due to a platform error, we should trigger a checkstop. However, if we haven't been told what we should do to trigger a checkstop (e.g. on an FSP machine), then we should still fail to fast-reboot.

    So, fast-reboot is disabled in the OPAL_CEC_REBOOT2 code path for the OPAL_REBOOT_PLATFORM_ERROR reboot type.
  - FSP code update
  - Unrecoverable HMI
  - A PHB is in CAPI mode

    If a PHB is in CAPI mode, we cannot safely fast reboot - the PHB will be fenced during the reboot resulting in major problems when we load the new kernel.

    In order to handle this safely, we need to disable CAPI mode before resetting PHBs during the fast reboot. However, we don't currently support this.

    In the meantime, when fast rebooting, check if there are any PHBs with a CAPP attached, and if so, abort the fast reboot and revert to a normal reboot instead.

Documentation

There have been a number of documentation fixes this release. Most prominent is the switch to Sphinx (from the Python project) and ReStructured Text (RST) as the documentation format. RST and Sphinx enable both production of pretty documentation in HTML and PDF formats while remaining readable in their raw form to those with no knowledge of RST.

You can build a HTML site by doing the following:

```
cd doc/
make html
```

As always, documentation patches are very, *very* welcome as we attempt to document the OPAL API, the device tree bindings and important parts of OPAL internals.

We would like the Device Tree documentation to follow the style that can be included in the Device Tree Specification.

General

- Make console-log time more readable: seconds rather than timebase Log format is now `[SECONDS.(tb%512000000),LEVEL]`
- Flash (PNOR) code improvements
  - flash: Make size 64 bit safe This makes the size of flash 64 bit safe so that we can have flash devices greater than 4GB. This is especially useful for mambo disks passed through to Linux.
  - core/flash.c: load actual partition size We are downloading 0x20000 bytes from PNOR for CAPP, but currently the CAPP lid is only 40K.
  - flash: Rework error paths and messages for multiple flash controllers Now that we have mambo bogusdisk flash, we can have many flash chips. This is resulting in some confusing output messages.
- core/init: Fix "failure of getting node in the free list" warning on boot.
- slw: improve error message for SLW timer stuck
- Centaur / XSCOM error handling
  - print message on disabling xscoms to centaur due to many errors
  - Mark centaur offline after 10 consecutive access errors
- XSCOM improvements
  - xscom: Map all HMER status codes to OPAL errors
  - xscom: Initialize the data to a known value in `xscom_read` In case of error, don't leave the data random. It helps debugging when the user fails to check the error code. This happens due to a bug in the PRD wrapper app.
  - chip: Add a quirk for when core direct control XSCOMs are missing
- p8-i2c: Don't crash if a centaur errored out
- cpu: Make endian switch message more informative
- cpu: Display number of started CPUs during boot
- core/init: ensure that HRMOR is zero at boot
- asm: Fix backtrace for unexpected exception
- cpu: Remove pollers calling heuristics from `cpu_wait_job` This will be handled by `time_wait_ms()`. Also remove a useless `smt_medium()`. Note that this introduce a difference in behaviour: time_wait will only call the pollers on the boot CPU while `cpu_wait_job()` could call them on any. However, I can't think of a case where this is a problem.
- cpu: Remove global job queue Instead, target a specific CPU for a global job at queuing time. This will allow us to wake up the target using an interrupt when implementing nap mode. The algorithm used is to look for idle primary threads first, then idle secondaries, and finally the less loaded thread. If nothing can be found, we fallback to a synchronous call.
- lpc: Log LPC SYNC errors as unrecoverable ones for manufacturing
- lpc: Optimize SerIRQ dispatch based on which PSI IRQ fired

- **interrupts: Add new source `->attributes()` callback** This allows a given source to provide per-interrupt attributes such as whether it targets OPAL or Linux and it's estimated frequency.

  The former allows to get rid of the double set of ops used to decide which interrupts go where on some modules like the PHBs and the latter will be eventually used to implement smart caching of the source lookups.

- opal/hmi: Fix a TOD HMI failure during a race condition.
- platform: Add BT to Generic platform

NVRAM

- Support `ibm,skiboot` partition for skiboot specific configuration options
- **flash: Size NVRAM based on ECC for OpenPOWER platforms** If NVRAM has ECC (as per the ffs header) then the actual size of the partition is less than reported by the ffs header in the PNOR then the actual size of the partition is less than reported by the ffs header.

NVLink/NPU

- Fix reserved PE#
- NPU bdfn allocation bugfix
- **Fix bad PE number check** NPUs have 4 PEs which are zero indexed, so {0, 1, 2, 3}. A bad PE number check in npu_err_inject checks if the PE number is greater than 4 as a fail case, so it would wrongly perform operations on a non-existant PE 4.
- Use PCI virtual device
- assert the NPU irq min is aligned.
- program NPU BUID reg properly
- **npu: reword "error" to indicate it's actually a warning** Incorrect FWTS annotation. Without this patch, you get spurious FirmWare Test Suite (FWTS) warnings about NVLink not working on machines that aren't fully populated with GPUs.
- **external: NPU hardware procedure script** Performing NPU hardware procedures requires some config space magic. Put all that magic into a script, so you can just specify the target device and the procedure number.

PCI

- Generic fixes
  - Claim surprise hotplug capability
  - Reserve PCI buses for RC's slot
  - Update PCI topology after power change
  - Return slot cached power state
  - Cache power state on slot without power control
  - Avoid hot resets at boot time
  - Fix initial PCIe slot power state

  - Print CRS retry times It's useful to know the CRS retry times before the PCI device is detected successfully. In PCI hot add case, it usually indicates time consumed for the adapter's firmware to be partially ready (responsive PCI config space).
  - core/pci: Fix the power-off timeout in `pci_slot_power_off()` The timeout should be 1000ms instead of 1000 ticks while powering off PCI slot in `pci_slot_power_off()`. Otherwise, it's likely to hit timeout powering off the PCI slot as below skiboot logs reveal:

```
[5399576870,5] PHB#0005:02:11.0 Timeout powering off slot
```

  - pci: Check power state before powering off slot. Prevents the erroneous "Error -1 powering off slot" error message.

- PHB3
  - Override root slot's `prepare_link_change()` with PHB's
  - Disable surprise link down event on PCI slots
  - Disable ECRC on Broadcom adapter behind PMC switch
- astbmc platforms
  - Support dynamic PCI slot. We might insert a PCIe switch to PHB direct slot and the downstream ports of the PCIe switch supports PCI hotplug.

CAPI

- **hw/phb3: Update capi initialization sequence** The capi initialization sequence was revised in a circumvention document when a 'link down' error was converted from fatal to Endpoint Recoverable. Other, non-capi, register setup was corrected even before the initial open-source release of skiboot, but a few capi-related registers were not updated then, so this patch fixes it.

Mambo Simulator

- Helpers for POWER9 Mambo.
- mambo: Advertise available RADIX page sizes
- mambo: Add section for kernel command line boot args Users can set kernel command line boot arguments for Mambo in a tcl script.
- mambo: add exception and qtrace helpers
- external/mambo: Update skiboot.tcl to add page-sizes nodes to device tree

Simics Simulator

- chiptod: Enable ChipTOD in SIMICS

Utilities

- pflash
  - fix harmless buffer overflow: `fl_total_size` was `uint32_t` not `uint64_t`.
  - Don't try to write protect when writing to flash file

  - Misc small improvements to code and code style
  - makefile bug fixes
  - external/pflash: Make MTD accesses the default

    Now that BMC and host kernel mtd drivers exist and have matured we should use them by default.

    This is especially important since we seem to be telling everyone to use pflash (pflash world domination plans are continuing on schedule).
  - external/pflash: Catch incompatible combination of flags
  - external/common: arm: Don't error trying to wrprotect with MTD access
  - libflash/libffs: Use blocklevel_smart_write() when updating partitions
- external/boot_tests
  - remove lid from the BMC after flashing
  - add the nobooting option -N
  - add arbitrary lid option -F
- **`getscom` / `getsram` / `putscom`: Parse chip-id as hex** We print the chip-id in hex (without a leading 0x), but we fail to parse that same value correctly in `getscom` / `getsram` / `putscom`

```
# getscom -l
...
80000000 | DD2.0 | Centaur memory buffer
# getscom -c 80000000 201140a
Error -19 reading XSCOM
```

  Fix this by assuming base 16 when parsing chip-id.

PRD

- opal-prd: Fix error code from `scom_read` and `scom_write`
- opal-prd: Add get_interface_capabilities to host interfaces
- opal-prd: fix for 64-bit pnor sizes
- **occ/prd/opal-prd: Queue OCC_RESET event message to host in OpenPOWER** During an OCC reset cycle the system is forced to Psafe pstate. When OCC becomes active, the system has to be restored to its last pstate as requested by host. So host needs to be notified of OCC_RESET event or else system will continue to remian in Psafe state until host requests a new pstate after the OCC reset cycle.

IBM FSP Based Platforms

- **fsp/console: Allocate irq for each hvc console** Allocate an irq number for each hvc console and set its interrupt-parent property so that Linux can use the opal irqchip instead of the OPAL_EVENT_CONSOLE_INPUT interface.
- platforms/firenze: Fix clock frequency dt property:

```
[ 1.212366090,3] DT: Unexpected property length /xscom@3fc0000000000/i2cm@a0020/
↪clock-frequency
```

- **HDAT: Fix typo in nest-frequency property** nest-frquency -> nest-frequency

- **platforms/ibm-fsp: Use power_ctl bit when determining slot reset method** The power_ctl bit is used to represent if power management is available. If power_ctl is set to true, then the I2C based external power management functionality will be populated on the PCI slot. Otherwise we will try to use the inband PERST as the fundamental reset, as before.
- **FSP/ELOG: Fix elog timeout issue** Presently we set timeout value as soon as we add elog to queue. If we have multiple elogs to write, it doesn't consider queue wait time. Instead set timeout value when we are actually sending elog to FSP.
- **FSP/ELOG: elog_enable flag should be false by default** This issue is one of the corner case, which is related to recent change went upstream and only observed in the petitboot prompt, where we see only one error log instead of getting all error log in `/sys/firmware/opal/elog`.

POWER9

Skiboot 5.4 contains only *preliminary* support for POWER9. It's suitable only for use in simulators. If working on hardware, use more recent skiboot or development branches. We will not be backporting POWER9 fixes to 5.4.x.

- mambo: Make POWER9 look like DD2
- core/cpu.c: Add OPAL call to setup Nest MMU
- psi: On p9, create an interrupt-map for routing PSI interrupts
- lpc: Add P9 LPC interrupts support
- chiptod: Basic P9 support
- psi: Add P9 support

Testing and Debugging

- test/qemu: bump qemu version used in CI, adds IPMI support
- platform/qemu: add BT and IPMI support Enables testing BT and IPMI functionality in the Qemu simulator
- init: In debug builds, enable debug output to console
- **mem_region: Be a bit smarter about poisoning** Don't poison chunks that are already free and poison regions on first allocation. This speeds things up dramatically.
- **libc: Use 8-bytes stores for non-0 memset too** Memory poisoning hammers this, so let's be a bit smart about it and avoid falling back to byte stores when the data is not 0
- fwts: add annotation for manufacturing mode
- check: Fix bugs in mem region tests
- **Don't set -fstack-protector-all unconditionally** We set it already in DEBUG builds and we use -fstack-protector-strong in release builds which provides most of the benefits and is more efficient.
- **Build host programs (and checks) with debug enabled** This enables memory poisoning in allocations and list checking among other things.
- Add global DEBUG make flag

Command line arguments to BOOTKERNEL

- core/init.c: Fix bootargs parsing

  Currently the bootargs are unconditionally deleted, which causes a bug where the bootargs passed in by the device tree are lost.

  This patch deletes bootargs only if it needs to be replaced by the NVRAM entry.

  This patch also removes KERNEL_COMMAND_LINE config option in favour of using the NVRAM or a device tree.

Other changes

- extract-gcov: build with -m64 if compiler supports it.

  Fixes build break on 32bit ppc64 (e.g. PowerMac G5, where user space is mostly 32bit).

Flash on OpenPOWER platforms

- flash: rework flash_load_resource to correctly read FFS/STB

  This fixes the previous reverts of loading the CAPP partition with STB headers (which broke CAPP partitions without STB headers).

  The new logic fixes both CAPP partition loading with STB headers *and* addresses a long standing bug due to differing interpretations of FFS.

  The f_part utility that *constructs* PNOR files just sets actualSize=totalSize no matter on what the size of the partition is. Prior to this patch, skiboot would always load actualSize, leading to longer than needed IPL.

  The pflash utility updates actualSize, so no developer has really ever noticed this, apart from maybe an inkling that it's odd that a freshly baked PNOR from op-build takes ever so slightly longer to boot than one that has had individual partitions pflashed in.

  With this patch, we now compute actualSize. For partitions with a STB header, we take the payload size from the STB header. For partitions that don't have a STB header, we compute the size either by parsing the ELF header or by looking at the subpartition header and computing it.

  We now need to read the entire partition for partitions with subpartitions so that we pass consistent values to be measured as part of Trusted Boot.

  As of this patch, the actualSize field in FFS is *not* relied on for partition size, we determine it from the content of the partition.

  However, this patch *will* break loading of partitions that are not ELF and do not contain subpartitions. Luckily, nothing in-tree makes use of that.

Contributors

Extending the analysis done for the last few releases, we can see our trends in code review across versions:

Release	csets	Ack	Reviews	Tested	Reported
5.0	329	15	20	1	0
5.1	372	13	38	1	4
5.2-rc1	334	20	34	6	11
5.3-rc1	302	36	53	4	5
5.4-rc1	278	8	19	0	4
5.4.0	361	16	28	1	9

Interesting is the stats of 5.4.0-rc1 versus the final 5.4.0, there's been a doubling of Acks, an increase in reviewed-by and reported-by. There's nothing like an impending release to get people to look closer.

Processed 361 csets from 34 developers A total of 20206 lines added, 5843 removed (delta 14363)

Developers with the most changesets:

Developer	#	%
Stewart Smith	105	(29.1%)
Benjamin Herrenschmidt	50	(13.9%)
Claudio Carvalho	47	(13.0%)
Gavin Shan	24	(6.6%)
Cyril Bur	20	(5.5%)
Oliver O'Halloran	18	(5.0%)
Michael Neuling	12	(3.3%)
Mukesh Ojha	12	(3.3%)
Pridhiviraj Paidipeddi	7	(1.9%)
Vasant Hegde	7	(1.9%)
Russell Currey	7	(1.9%)
Joel Stanley	4	(1.1%)
Alistair Popple	4	(1.1%)
Mahesh Salgaonkar	4	(1.1%)
Nageswara R Sastry	4	(1.1%)
Chris Smart	3	(0.8%)
Sam Mendoza-Jonas	3	(0.8%)
Vipin K Parashar	3	(0.8%)
Balbir Singh	3	(0.8%)
Frederic Barrat	3	(0.8%)
leoluo	2	(0.6%)
Rafael Fonseca	2	(0.6%)
Jack Miller	2	(0.6%)
Patrick Williams	2	(0.6%)
Jeremy Kerr	2	(0.6%)
Suraj Jitindar Singh	2	(0.6%)
Milton Miller	2	(0.6%)
Andrew Donnellan	1	(0.3%)
Shilpasri G Bhat	1	(0.3%)
Frederic Bonnard	1	(0.3%)
Breno Leitao	1	(0.3%)
Anton Blanchard	1	(0.3%)
Nicholas Piggin	1	(0.3%)
Cédric Le Goater	1	(0.3%)

Developers with the most changed lines:

Developer	#	%
Claudio Carvalho	6947	(32.9%)
Stewart Smith	6667	(31.6%)
Benjamin Herrenschmidt	2586	(12.3%)
Gavin Shan	1185	(5.6%)
Cyril Bur	692	(3.3%)
Mukesh Ojha	565	(2.7%)
Oliver O'Halloran	343	(1.6%)
Russell Currey	343	(1.6%)
leoluo	269	(1.3%)
Pridhiviraj Paidipeddi	236	(1.1%)
Balbir Singh	227	(1.1%)
Michael Neuling	211	(1.0%)
Nageswara R Sastry	132	(0.6%)
Cédric Le Goater	115	(0.5%)
Vipin K Parashar	68	(0.3%)
Alistair Popple	66	(0.3%)
Vasant Hegde	65	(0.3%)
Mahesh Salgaonkar	50	(0.2%)
Shilpasri G Bhat	45	(0.2%)
Suraj Jitindar Singh	41	(0.2%)
Nicholas Piggin	34	(0.2%)
Sam Mendoza-Jonas	33	(0.2%)
Jack Miller	32	(0.2%)
Chris Smart	28	(0.1%)
Jeremy Kerr	23	(0.1%)
Milton Miller	19	(0.1%)
Joel Stanley	13	(0.1%)
Andrew Donnellan	13	(0.1%)
Rafael Fonseca	12	(0.1%)
Patrick Williams	11	(0.1%)
Frederic Barrat	6	(0.0%)
Anton Blanchard	3	(0.0%)
Frederic Bonnard	2	(0.0%)
Breno Leitao	2	(0.0%)

Developers with the most lines removed:

Developer	#	%
Cyril Bur	206	(3.5%)
Rafael Fonseca	8	(0.1%)

Developers with the most signoffs (total 278):

Developer	#	%
Stewart Smith	268	(96.4%)
Alistair Popple	4	(1.4%)
Jim Yuan	2	(0.7%)
Cyril Bur	1	(0.4%)
Michael Neuling	1	(0.4%)
Jeremy Kerr	1	(0.4%)
Benjamin Herrenschmidt	1	(0.4%)

Developers with the most reviews (total 28):

Developer	#	%
Andrew Donnellan	6	(21.4%)
Vasant Hegde	5	(17.9%)
Mukesh Ojha	5	(17.9%)
Joel Stanley	3	(10.7%)
Russell Currey	3	(10.7%)
Cyril Bur	2	(7.1%)
Balbir Singh	2	(7.1%)
Alistair Popple	1	(3.6%)
Vaidyanathan Srinivasan	1	(3.6%)

Developers with the most test credits (total 1):

Developer	#	%
Pridhiviraj Paidipeddi	1	(100.0%)

Developers who gave the most tested-by credits (total 1):

Developer	#	%
Gavin Shan	1	(100.0%)

Developers with the most report credits (total 9):

Developer	#	%
Pridhiviraj Paidipeddi	3	(33.3%)
Gavin Shan	1	(11.1%)
Vasant Hegde	1	(11.1%)
Michael Neuling	1	(11.1%)
Benjamin Herrenschmidt	1	(11.1%)
Andrei Warkenti	1	(11.1%)
Li Meng	1	(11.1%)

4.1.44 skiboot-5.4.0-rc1

skiboot-5.4.0-rc1 was released on Monday October 17th 2016. It is the first release candidate of skiboot 5.4, which will become the new stable release of skiboot following the 5.3 release, first released August 2nd 2016.

skiboot-5.4.0-rc1 contains all bug fixes as of *skiboot-5.3.7* and *skiboot-5.1.18* (the currently maintained stable releases).

For how the skiboot stable releases work, see *Skiboot stable tree rules and releases* for details.

The current plan is to release a new release candidate every week until we feel good about it. The aim is for skiboot-5.4.x to be in op-build v1.13, which is due by November 23rd 2016.

Over skiboot-5.3, we have the following changes:

New Features

- Initial Trusted Boot support (see *Secure and Trusted Boot Overview*). There are several limitations with this initial release:
  - CAPP partition is not measured correctly
  - Only Nuvoton TPM 2.0 is supported
  - Requires hardware rework on late revision Habanero or Firestone boards in order to install TPM.
  - Add i2c Nuvoton TPM 2.0 Driver
  - romcode driver for POWER8 secure ROM
  - See Device tree docs for tpm and ibm,secureboot nodes
  - See main secure and trusted boot documentation.
- Fast reboot for P8

  This makes reboot take an *awful* lot less time, somewhere between four and ten times faster than a full IPL. It is currently experimental and not enabled by default. You can enable the experimental support via nvram option:

  ```
  # nvram -p ibm,skiboot --update-config experimental-fast-reset=feeling-lucky
  ```

  WARNING: This has *known* bugs. For example, if you have used a device in CAPI mode, we will currently *NOT* reset it back to plain PCI. There are also some known issues in most simulators.
- Support `ibm,skiboot` NVRAM partition with skiboot configuration options.
  - These should generally only be used if you either completely know what you are doing or need to work around a skiboot bug. They are **not** intended for end users.
  - Add support for supplying the kernel boot arguments from the `bootargs` configuration string in the `ibm,skiboot` NVRAM partition.
  - Enabling the experimental fast reset feature is done via this method.
- Add support for nap mode on P8 while in skiboot
  - While nap has been exposed to the Operating System since day 1, we have not utilized low power states when in skiboot itself, leading to higher power consumption during boot. We only enable the functionality after the 0x100 vector has been patched, and we disable it before transferring control to Linux.
- libflash: add 128MB MX66L1G45G part
- Pointer validation of OPAL API call arguments.
  - If the kernel called an OPAL API with vmalloc'd address or any other address range in real mode, we would hit a problem with aliasing. Since the top 4 bits are ignored in real mode, pointers from 0xc.. and 0xd.. (and other ranges) could collide and lead to hard to solve bugs. This patch adds the infrastructure for pointer validation and a simple test case for testing the API
  - The checks validate pointers sent in using `opal_addr_valid()`

Documentation

There have been a number of documentation fixes this release. Most prominent is the switch to Sphinx (from the Python project) and ReStructured Text (RST) as the documentation format. RST and Sphinx enable both production of pretty documentation in HTML and PDF formats while remaining readable in their raw form to those with no knowledge of RST.

You can build a HTML site by doing the following:

```
cd doc/
make html
```

As always, documentation patches are very, *very* welcome as we attempt to document the OPAL API, the device tree bindings and important parts of OPAL internals.

We would like the Device Tree documentation to follow the style that can be included in the Device Tree Specification.

General

- Make console-log time more readable: seconds rather than timebase Log format is now `[SECONDS.(tb%512000000),LEVEL]`
- Flash (PNOR) code improvements
  - flash: Make size 64 bit safe This makes the size of flash 64 bit safe so that we can have flash devices greater than 4GB. This is especially useful for mambo disks passed through to Linux.
  - core/flash.c: load actual partition size We are downloading 0x20000 bytes from PNOR for CAPP, but currently the CAPP lid is only 40K.
  - flash: Rework error paths and messages for multiple flash controllers Now that we have mambo bogusdisk flash, we can have many flash chips. This is resulting in some confusing output messages.
- core/init: Fix "failure of getting node in the free list" warning on boot.
- slw: improve error message for SLW timer stuck
- Centaur / XSCOM error handling
  - print message on disabling xscoms to centaur due to many errors
  - Mark centaur offline after 10 consecutive access errors
- XSCOM improvements
  - xscom: Map all HMER status codes to OPAL errors
  - xscom: Initialize the data to a known value in `xscom_read` In case of error, don't leave the data random. It helps debugging when the user fails to check the error code. This happens due to a bug in the PRD wrapper app.
  - chip: Add a quirk for when core direct control XSCOMs are missing
- p8-i2c: Don't crash if a centaur errored out
- cpu: Make endian switch message more informative
- cpu: Display number of started CPUs during boot
- core/init: ensure that HRMOR is zero at boot
- asm: Fix backtrace for unexpected exception

- cpu: Remove pollers calling heuristics from `cpu_wait_job` This will be handled by `time_wait_ms()`. Also remove a useless `smt_medium()`. Note that this introduce a difference in behaviour: time_wait will only call the pollers on the boot CPU while `cpu_wait_job()` could call them on any. However, I can't think of a case where this is a problem.
- cpu: Remove global job queue Instead, target a specific CPU for a global job at queuing time. This will allow us to wake up the target using an interrupt when implementing nap mode. The algorithm used is to look for idle primary threads first, then idle secondaries, and finally the less loaded thread. If nothing can be found, we fallback to a synchronous call.
- lpc: Log LPC SYNC errors as unrecoverable ones for manufacturing
- lpc: Optimize SerIRQ dispatch based on which PSI IRQ fired
- **interrupts: Add new source `->attributes()` callback** This allows a given source to provide per-interrupt attributes such as whether it targets OPAL or Linux and it's estimated frequency.

  The former allows to get rid of the double set of ops used to decide which interrupts go where on some modules like the PHBs and the latter will be eventually used to implement smart caching of the source lookups.
- opal/hmi: Fix a TOD HMI failure during a race condition.
- platform: Add BT to Generic platform

NVRAM

- Support `ibm,skiboot` partition for skiboot specific configuration options
- **flash: Size NVRAM based on ECC for OpenPOWER platforms** If NVRAM has ECC (as per the ffs header) then the actual size of the partition is less than reported by the ffs header in the PNOR then the actual size of the partition is less than reported by the ffs header.

NVLink/NPU

- Fix reserved PE#
- NPU bdfn allocation bugfix
- **Fix bad PE number check** NPUs have 4 PEs which are zero indexed, so {0, 1, 2, 3}. A bad PE number check in npu_err_inject checks if the PE number is greater than 4 as a fail case, so it would wrongly perform operations on a non-existant PE 4.
- Use PCI virtual device
- assert the NPU irq min is aligned.
- program NPU BUID reg properly
- **npu: reword "error" to indicate it's actually a warning** Incorrect FWTS annotation. Without this patch, you get spurious FirmWare Test Suite (FWTS) warnings about NVLink not working on machines that aren't fully populated with GPUs.
- **external: NPU hardware procedure script** Performing NPU hardware procedures requires some config space magic. Put all that magic into a script, so you can just specify the target device and the procedure number.

PCI

- Generic fixes
  - Claim surprise hotplug capability
  - Reserve PCI buses for RC's slot
  - Update PCI topology after power change
  - Return slot cached power state
  - Cache power state on slot without power control
  - Avoid hot resets at boot time
  - Fix initial PCIe slot power state
  - Print CRS retry times It's useful to know the CRS retry times before the PCI device is detected successfully. In PCI hot add case, it usually indicates time consumed for the adapter's firmware to be partially ready (responsive PCI config space).
  - core/pci: Fix the power-off timeout in `pci_slot_power_off()` The timeout should be 1000ms instead of 1000 ticks while powering off PCI slot in `pci_slot_power_off()`. Otherwise, it's likely to hit timeout powering off the PCI slot as below skiboot logs reveal:

    ```
    [5399576870,5] PHB#0005:02:11.0 Timeout powering off slot
    ```

- PHB3
  - Override root slot's `prepare_link_change()` with PHB's
  - Disable surprise link down event on PCI slots
  - Disable ECRC on Broadcom adapter behind PMC switch
- astbmc platforms
  - Support dynamic PCI slot. We might insert a PCIe switch to PHB direct slot and the downstream ports of the PCIe switch supports PCI hotplug.

CAPI

- **hw/phb3: Update capi initialization sequence** The capi initialization sequence was revised in a circumvention document when a 'link down' error was converted from fatal to Endpoint Recoverable. Other, non-capi, register setup was corrected even before the initial open-source release of skiboot, but a few capi-related registers were not updated then, so this patch fixes it.

IPMI

- **core/ipmi: Set interrupt-parent property** This allows ipmi-opal to properly use the OPAL irqchip rather than falling back to the event interface in Linux.

Mambo Simulator

- Helpers for POWER9 Mambo.
- mambo: Advertise available RADIX page sizes

- mambo: Add section for kernel command line boot args Users can set kernel command line boot arguments for Mambo in a tcl script.
- mambo: add exception and qtrace helpers
- external/mambo: Update skiboot.tcl to add page-sizes nodes to device tree

Simics Simulator

- chiptod: Enable ChipTOD in SIMICS

Utilities

- pflash
  - fix harmless buffer overflow: `fl_total_size` was `uint32_t` not `uint64_t`.
  - Don't try to write protect when writing to flash file
  - Misc small improvements to code and code style
  - makefile bug fixes
- external/boot_tests
  - remove lid from the BMC after flashing
  - add the nobooting option -N
  - add arbitrary lid option -F
- **`getscom` / `getsram` / `putscom`: Parse chip-id as hex** We print the chip-id in hex (without a leading 0x), but we fail to parse that same value correctly in `getscom` / `getsram` / `putscom`

  ```
  # getscom -l
  ...
  80000000 | DD2.0 | Centaur memory buffer
  # getscom -c 80000000 201140a
  Error -19 reading XSCOM
  ```

  Fix this by assuming base 16 when parsing chip-id.

PRD

- opal-prd: Fix error code from `scom_read` and `scom_write`
- opal-prd: Add get_interface_capabilities to host interfaces
- opal-prd: fix for 64-bit pnor sizes
- **occ/prd/opal-prd: Queue OCC_RESET event message to host in OpenPOWER** During an OCC reset cycle the system is forced to Psafe pstate. When OCC becomes active, the system has to be restored to its last pstate as requested by host. So host needs to be notified of OCC_RESET event or else system will continue to remian in Psafe state until host requests a new pstate after the OCC reset cycle.

IBM FSP Based Platforms

- **fsp/console: Allocate irq for each hvc console** Allocate an irq number for each hvc console and set its interrupt-parent property so that Linux can use the opal irqchip instead of the OPAL_EVENT_CONSOLE_INPUT interface.
- platforms/firenze: Fix clock frequency dt property:

```
[ 1.212366090,3] DT: Unexpected property length /xscom@3fc0000000000/i2cm@a0020/
↪clock-frequency
```

- **HDAT: Fix typo in nest-frequency property** nest-frquency -> nest-frequency
- **platforms/ibm-fsp: Use power_ctl bit when determining slot reset method** The power_ctl bit is used to represent if power management is available. If power_ctl is set to true, then the I2C based external power management functionality will be populated on the PCI slot. Otherwise we will try to use the inband PERST as the fundamental reset, as before.
- **FSP/ELOG: Fix elog timeout issue** Presently we set timeout value as soon as we add elog to queue. If we have multiple elogs to write, it doesn't consider queue wait time. Instead set timeout value when we are actually sending elog to FSP.
- **FSP/ELOG: elog_enable flag should be false by default** This issue is one of the corner case, which is related to recent change went upstream and only observed in the petitboot prompt, where we see only one error log instead of getting all error log in `/sys/firmware/opal/elog`.

POWER9

- mambo: Make POWER9 look like DD2
- **flash: Move flash node under `ibm,opal/flash/`** This changes the boot ABI, so it's only active for P9 and later systems, even though it's unrelated to hardware changes. There is an associated Linux change to properly search for this node as well.
- core/cpu.c: Add OPAL call to setup Nest MMU
- psi: On p9, create an interrupt-map for routing PSI interrupts
- lpc: Add P9 LPC interrupts support
- chiptod: Basic P9 support
- psi: Add P9 support

Testing and Debugging

- test/qemu: bump qemu version used in CI, adds IPMI support
- platform/qemu: add BT and IPMI support Enables testing BT and IPMI functionality in the Qemu simulator
- init: In debug builds, enable debug output to console
- **mem_region: Be a bit smarter about poisoning** Don't poison chunks that are already free and poison regions on first allocation. This speeds things up dramatically.
- **libc: Use 8-bytes stores for non-0 memset too** Memory poisoning hammers this, so let's be a bit smart about it and avoid falling back to byte stores when the data is not 0
- fwts: add annotation for manufacturing mode
- check: Fix bugs in mem region tests

- **Don't set -fstack-protector-all unconditionally** We set it already in DEBUG builds and we use -fstack-protector-strong in release builds which provides most of the benefits and is more efficient.
- **Build host programs (and checks) with debug enabled** This enables memory poisoning in allocations and list checking among other things.
- Add global DEBUG make flag

Contributors

Extending the analysis done for the last few releases, we can see our trends in code review across versions:

Release	csets	Ack	Reviews	Tested	Reported
5.0	329	15	20	1	0
5.1	372	13	38	1	4
5.2-rc1	334	20	34	6	11
5.3-rc1	302	36	53	4	5
5.4-rc1	278	8	19	0	4

This release has fewer changesets over previous 5.x first release candidates, but that is not indicative of the size or complexity of these changes.

Processed 278 csets from 31 developers A total of 17052 lines added, 4745 removed (delta 12307)

Developers with the most changesets

Stewart Smith	71	(25.5%)
Benjamin Herrenschmidt	50	(18.0%)
Claudio Carvalho	38	(13.7%)
Gavin Shan	20	(7.2%)
Oliver O'Halloran	18	(6.5%)
Mukesh Ojha	9	(3.2%)
Cyril Bur	7	(2.5%)
Russell Currey	7	(2.5%)
Vasant Hegde	7	(2.5%)
Pridhiviraj Paidipeddi	6	(2.2%)
Michael Neuling	6	(2.2%)
Alistair Popple	4	(1.4%)
Sam Mendoza-Jonas	3	(1.1%)
Vipin K Parashar	3	(1.1%)
Balbir Singh	3	(1.1%)
Mahesh Salgaonkar	3	(1.1%)
Frederic Barrat	3	(1.1%)
Chris Smart	2	(0.7%)
Jack Miller	2	(0.7%)
Patrick Williams	2	(0.7%)
Jeremy Kerr	2	(0.7%)
Suraj Jitindar Singh	2	(0.7%)
Milton Miller	2	(0.7%)
Shilpasri G Bhat	1	(0.4%)
Frederic Bonnard	1	(0.4%)

Continued on next page

Table 4.5 – continued from previous page

Joel Stanley	1	(0.4%)
Breno Leitao	1	(0.4%)
Anton Blanchard	1	(0.4%)
Nicholas Piggin	1	(0.4%)
Nageswara R Sastry	1	(0.4%)
Cédric Le Goater	1	(0.4%)

Developers with the most changed lines

Claudio Carvalho	6817	(38.2%)
Stewart Smith	4677	(26.2%)
Benjamin Herrenschmidt	2586	(14.5%)
Gavin Shan	1005	(5.6%)
Cyril Bur	509	(2.9%)
Mukesh Ojha	361	(2.0%)
Oliver O'Halloran	343	(1.9%)
Russell Currey	343	(1.9%)
Balbir Singh	227	(1.3%)
Pridhiviraj Paidipeddi	194	(1.1%)
Michael Neuling	121	(0.7%)
Cédric Le Goater	115	(0.6%)
Vipin K Parashar	68	(0.4%)
Alistair Popple	66	(0.4%)
Vasant Hegde	65	(0.4%)
Shilpasri G Bhat	45	(0.3%)
Suraj Jitindar Singh	41	(0.2%)
Nicholas Piggin	34	(0.2%)
Sam Mendoza-Jonas	33	(0.2%)
Jack Miller	32	(0.2%)
Nageswara R Sastry	32	(0.2%)
Jeremy Kerr	23	(0.1%)
Mahesh Salgaonkar	21	(0.1%)
Chris Smart	20	(0.1%)
Milton Miller	19	(0.1%)
Patrick Williams	11	(0.1%)
Frederic Barrat	6	(0.0%)
Anton Blanchard	3	(0.0%)
Frederic Bonnard	2	(0.0%)
Joel Stanley	2	(0.0%)
Breno Leitao	2	(0.0%)

Developers with the most lines removed

Cyril Bur	299	(6.3%)

Developers with the most signoffs (total 226)

Stewart Smith	219	(96.9%)
Alistair Popple	4	(1.8%)
Cyril Bur	1	(0.4%)
Jeremy Kerr	1	(0.4%)
Benjamin Herrenschmidt	1	(0.4%)

Developers with the most reviews (total 19)

Mukesh Ojha	5	(26.3%)
Andrew Donnellan	4	(21.1%)
Vasant Hegde	3	(15.8%)
Russell Currey	3	(15.8%)
Balbir Singh	2	(10.5%)
Cyril Bur	1	(5.3%)
Vaidyanathan Srinivasan	1	(5.3%)

Developers with the most test credits (total 0)

Developers who gave the most tested-by credits (total 0)

Developers with the most report credits (total 4)

Benjamin Herrenschmidt	1	(25.0%)
Li Meng	1	(25.0%)
Pridhiviraj Paidipeddi	1	(25.0%)
Gavin Shan	1	(25.0%)

Developers who gave the most report credits (total 4)

Gavin Shan	1	(25.0%)
Vasant Hegde	1	(25.0%)
Russell Currey	1	(25.0%)
Stewart Smith	1	(25.0%)

4.1.45 skiboot-5.4.0-rc2

skiboot-5.4.0-rc2 was released on Wednesday October 26th 2016. It is the second release candidate of skiboot 5.4, which will become the new stable release of skiboot following the 5.3 release, first released August 2nd 2016.

skiboot-5.4.0-rc2 contains all bug fixes as of *skiboot-5.3.7* and *skiboot-5.1.18* (the currently maintained stable releases).

For how the skiboot stable releases work, see *Skiboot stable tree rules and releases* for details.

Since this is a release candidate, it should *NOT* be put into production.

The current plan is to release a new release candidate every week until we feel good about it. The aim is for skiboot-5.4.x to be in op-build v1.13, which is due by November 23rd 2016.

Over *skiboot-5.4.0-rc1*, we have a few changes:

Secure and Trusted Boot

skiboot 5.4.0-rc2 improves upon the progress towards Secure and Trusted Boot in rc1. It is important to note that this is *not* a complete, end-to-end secure/trusted boot implementation.

With the current code, it is now possible to verify and measure resources loaded from PNOR by skiboot (namely the CAPP and BOOTKERNEL partitions).

Note that this functionality is currently *only* available on systems that use the libflash backend. It is *NOT* enabled on IBM FSP based systems. There is some support for some simulators though.

- libstb/stb.c: ignore the secure mode flag unless forced in NVRAM

  For this stage in Trusted Boot development, we are wishing to not force Secure Mode through the whole firmware boot process, but we are wanting to be able to test it (classic chicken and egg problem with build infrastructure).

  We disabled secure mode if the secure-enabled devtree property is read from the device tree *IF* we aren't overriding it through NVRAM. Seeing as we can only increase (not decrease) what we're checking through the NVRAM variable, it is safe.

  The NVRAM setting is force-secure-mode=true in the ibm,skiboot partition.

  However, if you want to force secure mode even if Hostboot has *not* set the secure-enabled proprety in the device tree, set force-secure-mode to "always".

  There is also a force-trusted-mode NVRAM setting to force trusted mode even if Hostboot has not enabled it int the device tree.

  To indicate to Linux that we haven't gone through the whole firmware process in secure mode, we replace the 'secure-enabled' property with 'partial-secure-enabled', to indicate that only part of the firmware boot process has gone through secure mode.

Command line arguments to BOOTKERNEL

- core/init.c: Fix bootargs parsing

  Currently the bootargs are unconditionally deleted, which causes a bug where the bootargs passed in by the device tree are lost.

  This patch deletes bootargs only if it needs to be replaced by the NVRAM entry.

  This patch also removes KERNEL_COMMAND_LINE config option in favour of using the NVRAM or a device tree.

pflash utility

- external/pflash: Make MTD accesses the default

  Now that BMC and host kernel mtd drivers exist and have matured we should use them by default.

  This is especially important since we seem to be telling everyone to use pflash (pflash world domination plans are continuing on schedule).

- external/pflash: Catch incompatible combination of flags
- external/common: arm: Don't error trying to wrprotect with MTD access
- libflash/libffs: Use blocklevel_smart_write() when updating partitions

Other changes

- extract-gcov: build with -m64 if compiler supports it.

  Fixes build break on 32bit ppc64 (e.g. PowerMac G5, where user space is mostly 32bit).

Fast Reset

- fast-reset: disable fast reboot in event of platform error

  Most of the time, if we're rebooting due to a platform error, we should trigger a checkstop. However, if we haven't been told what we should do to trigger a checkstop (e.g. on an FSP machine), then we should still fail to fast-reboot.

  So, disable fast-reboot in the OPAL_CEC_REBOOT2 code path for OPAL_REBOOT_PLATFORM_ERROR reboot type.

- fast-reboot: disable on FSP code update or unrecoverable HMI
- fast-reboot: abort fast reboot if CAPP attached

  If a PHB is in CAPI mode, we cannot safely fast reboot - the PHB will be fenced during the reboot resulting in major problems when we load the new kernel.

  In order to handle this safely, we need to disable CAPI mode before resetting PHBs during the fast reboot. However, we don't currently support this.

  In the meantime, when fast rebooting, check if there are any PHBs with a CAPP attached, and if so, abort the fast reboot and revert to a normal reboot instead.

OpenPOWER Platforms

For all hardware platforms that aren't IBM FSP machines:

- Revert "flash: Move flash node under ibm,opal/flash/"

  This reverts commit e1e6d009860d0ef60f9daf7a0fbe15f869516bd0.

  Breaks DT enough that it makes people cranky, reverting for now. This could break access to flash with existing kernels in POWER9 simulators

- flash: rework flash_load_resource to correctly read FFS/STB

  This fixes the previous reverts of loading the CAPP partition with STB headers (which broke CAPP partitions without STB headers).

  The new logic fixes both CAPP partition loading with STB headers *and* addresses a long standing bug due to differing interpretations of FFS.

  The f_part utility that *constructs* PNOR files just sets actualSize=totalSize no matter on what the size of the partition is. Prior to this patch, skiboot would always load actualSize, leading to longer than needed IPL.

  The pflash utility updates actualSize, so no developer has really ever noticed this, apart from maybe an inkling that it's odd that a freshly baked PNOR from op-build takes ever so slightly longer to boot than one that has had individual partitions pflashed in.

  With this patch, we now compute actualSize. For partitions with a STB header, we take the payload size from the STB header. For partitions that don't have a STB header, we compute the size either by parsing the ELF header or by looking at the subpartition header and computing it.

We now need to read the entire partition for partitions with subpartitions so that we pass consistent values to be measured as part of Trusted Boot.

As of this patch, the actualSize field in FFS is *not* relied on for partition size, we determine it from the content of the partition.

However, this patch *will* break loading of partitions that are not ELF and do not contain subpartitions. Luckily, nothing in-tree makes use of that.

PCI

- pci: Check power state before powering off slot

  Prevents the erroneous "Error -1 powering off slot" error message.

Contributors

Since *skiboot-5.4.0-rc1*, we have 23 csets from 8 developers.

A total of 876 lines added, 621 removed (delta 255)

Developers with the most changesets

Developer	#	%
Stewart Smith	7	(30.4%)
Cyril Bur	5	(21.7%)
Mukesh Ojha	3	(13.0%)
Gavin Shan	3	(13.0%)
Claudio Carvalho	2	(8.7%)
Chris Smart	1	(4.3%)
Andrew Donnellan	1	(4.3%)
Nageswara R Sastry	1	(4.3%)

Developers with the most changed lines

Developer	#	%
Stewart Smith	424	(45.7%)
Mukesh Ojha	204	(22.0%)
Gavin Shan	173	(18.6%)
Cyril Bur	69	(7.4%)
Claudio Carvalho	35	(3.8%)
Andrew Donnellan	13	(1.4%)
Chris Smart	8	(0.9%)
Nageswara R Sastry	2	(0.2%)

Developers with the most lines removed

Developer	#	%
Gavin Shan	9	(1.4%)
Chris Smart	4	(0.6%)

Developers with the most signoffs (total 16)

Developer	#	%
Stewart Smith	16	(100.0%)

Developers with the most reviews (total 4)

Developer	#	%
Vasant Hegde	2	(50.0%)
Andrew Donnellan	2	(50.0%)

Developers with the most test credits (total 1)

Developer	#	%
Pridhiviraj Paidipeddi	1	(100.0%)

Developers who gave the most tested-by credits (total 1)

Developer	#	%
Gavin Shan	1	(100.0%)

Developers with the most report credits (total 3)

Developer	#	%
Pridhiviraj Paidipeddi	1	(33.3%)
Andrei Warkenti	1	(33.3%)
Michael Neuling	1	(33.3%)

Developers who gave the most report credits (total 3)

Developer	#	%
Stewart Smith	2	(66.7%)
Gavin Shan	1	(33.3%)

4.1.46 skiboot-5.4.0-rc3

skiboot-5.4.0-rc3 was released on Wednesday November 2nd 2016. It is the third release candidate of skiboot 5.4, which will become the new stable release of skiboot following the 5.3 release, first released August 2nd 2016.

skiboot-5.4.0-rc3 contains all bug fixes as of *skiboot-5.3.7* and *skiboot-5.1.18* (the currently maintained stable releases).

For how the skiboot stable releases work, see *Skiboot stable tree rules and releases* for details.

Since this is a release candidate, it should *NOT* be put into production.

The current plan is to release a new release candidate every week until we feel good about it. The aim is for skiboot-5.4.x to be in op-build v1.13, which is due by November 23rd 2016.

Over *skiboot-5.4.0-rc2*, we have a few changes:

- pflash: Fail when file is larger than partition You can still shoot yourself in the foot by passing –force.
- core/flash: Don't do anything clever for OPAL_FLASH_{READ, WRITE, ERASE} This fixes a bug where opal-prd and opal-gard could fail. Fixes: https://github.com/open-power/skiboot/issues/44

- boot-tests: force BMC to boot from non-golden side
- fast-reset: Send special reset sequence to operational CPUs only. Fixes fast-reset for cases where there are garded CPUs
- Secure/Trusted boot: be much clearer about what is being measured where.
- Secure/Trusted boot: be more resilient to disabled TPM(s).
- Secure/Trusted boot: The `force-secure-mode` NVRAM setting introduced temporarily in *skiboot-5.4.0-rc2* has changed behaviour. Now, by default, the `secure-mode` flag in the device tree is obeyed. As always, any skiboot NVRAM options are in no way ABI, API or supported and may cause unfinished verbose analogies to appear in release notes relating to the dangers of using developer only options.
- gard: Fix compiler warning on modern GCC targetting ARM 32-bit
- opal-prd: systemd scripts improvements, only run on supported systems

4.1.47 skiboot-5.4.0-rc4

skiboot-5.4.0-rc4 was released on Tuesday November 8th 2016. It is the fourth (and hopefully final) release candidate of skiboot 5.4, which will become the new stable release of skiboot following the 5.3 release, first released August 2nd 2016.

skiboot-5.4.0-rc4 contains all bug fixes as of *skiboot-5.3.7* and *skiboot-5.1.18* (the currently maintained stable releases).

For how the skiboot stable releases work, see *Skiboot stable tree rules and releases* for details.

Since this is a release candidate, it should *NOT* be put into production.

With this release candidate, I'm hoping that it's the last one, and that within the week we're able to tag a final 5.4.0 release. There is one bit of code I'm hoping to merge in before the final 5.4.0, and that's the p8dtu platform definition. The aim is for skiboot-5.4.x to be in op-build v1.13, which is due by November 23rd 2016.

Over *skiboot-5.4.0-rc3*, we have a few changes:

- Add BMC platform to enable correct OEM IPMI commands

  An out of tree platform (p8dtu) uses a different IPMI OEM command for IPMI_PARTIAL_ADD_ESEL. This exposed some assumptions about the BMC implementation in our core code.

  Now, with platform.bmc, each platform can dictate (or detect) the BMC that is present. We allow it to be set at runtime rather than purely statically in struct platform as it's possible to have differing BMC implementations on the one machine (e.g. AMI BMC or OpenBMC).

- hw/ipmi-sensor: Fix setting of firmware progress sensor properly.

  On FSP systems, OPAL was incorrectly setting firmware status on a sensor id "00" which doesn't exist.

- pflash: remove stray d in from info message
- libflash/pflash: support whole chip erase on mtd access
- boot_test: fix typo in console message
- core/pci: Fix criteria in pci_cfg_reg_filter(), i.e. NVLink didn't work.
- Remove KERNEL_COMMAND_LINE mention from config.h

  We removed the functionality but not the define.

4.1.48 skiboot-5.4.1

skiboot-5.4.1 was released on Tuesday November 29th 2016. It replaces *skiboot-5.4.0* as the current stable release.

Over *skiboot-5.4.0*, we have a few changes:

- Nuvoton i2c TPM driver: bug fixes and improvements, especially around timeouts and error handling.
- Limit number of "Poller recursion detected" errors to display. In some error conditions, we could spiral out of control on this and spend all of our time printing the exact same backtrace.
- slw: do SLW timer testing while holding xscom lock. In some situations without this, it could take long enough to get the xscom lock that the 1ms timeout would expire and we'd falsely think the SLW timer didn't work when in fact it did.
- p8i2c: Use calculated poll_interval when booting OPAL. Otherwise we'd default to 2seconds (TIMER_POLL) during boot on chips with a functional i2c interrupt, leading to slow i2c during boot (or hitting timeouts instead).
- i2c: More efficiently run TPM I2C operations during boot, avoiding hitting timeouts
- fsp: Don't recurse pollers in ibm_fsp_terminate

4.1.49 skiboot-5.4.2

skiboot-5.4.2 was released on Friday December 2nd 2016. It replaces *skiboot-5.4.1* as the current stable release.

Over *skiboot-5.4.1*, we have two bug fixes exclusively aimed at machines with TPMs:

- i2c: Add nuvoton TPM quirk, disallowing i2cdetect as it can hard lock the TPM
- p8-i2c improve I2C reset code path, solves getting stuck resetting i2c engine

4.1.50 skiboot-5.4.3

skiboot-5.4.3 was released on Monday January 16th, 2017. It replaces *skiboot-5.4.2* as the current stable release.

Over *skiboot-5.4.2*, we have a small number of bug fixes:

- Makefile: Disable stack protector due to gcc problems
- Makefile: Use -ffixed-r13. We use r13 for our own stuff, make sure it's properly fixed
- phb3: Lock the PHB on set_xive callbacks
- arch_flash_arm: Don't assume mtd labels are short
- Stop using 3-operand cmp[l][i] for latest binutils
- hw/phb3: fix error handling in complete reset

4.1.51 skiboot-5.4.4

skiboot-5.4.4 was released on Wednesday May 3rd, 2017. It replaces *skiboot-5.4.3* as the current stable release in the 5.4.x series.

Over *skiboot-5.4.3*, we have a small number of bug fixes:

- hw/fsp: Do not queue SP and SPCN class messages during reset/reload In certain cases of communicating with the FSP (e.g. sensors), the OPAL FSP driver returns a default code (async completion) even though there is no known bound from the time of this error return to the actual data being available. The kernel driver keeps waiting leading to soft-lockup on the host side.

Mitigate both these (known) cases by returning OPAL_BUSY so the host driver knows to retry later.

- core/pci: Fix PCIe slot's presence According to PCIe spec, the presence bit is hardcoded to 1 if PCIe switch downstream port doesn't support slot capability. The register used for the check in pcie_slot_get_presence_state() is wrong. It should be PCIe capability register instead of PCIe slot capability register. Otherwise, we always have present bit on the PCI topology.

  The issue is found on Supermicro's p8dtu2u machine:

```
 # lspci -t
 -+-[0022:00]---00.0-[01-08]----00.0-[02-08]--+-01.0-[03]----00.0
  |                                           \-02.0-[04-08]--
 # cat /sys/bus/pci/slots/S002204/adapter
 1
 # lspci -vvs 0022:02:02.0
 # lspci -vvs 0022:02:02.0
 0022:02:02.0 PCI bridge: PLX Technology, Inc. PEX 8718 16-Lane, \
 5-Port PCI Express Gen 3 (8.0 GT/s) Switch (rev ab) (prog-if 00 [Normal decode])
    :
 Capabilities: [68] Express (v2) Downstream Port (Slot+), MSI 00
    :
    SltSta:    Status: AttnBtn- PowerFlt- MRL- CmdCplt- PresDet- Interlock-
               Changed: MRL- PresDet- LinkState-

This fixes the issue by checking the correct register (PCIe capability).
Also, the register's value is cached in advance as we did for slot and
link capability.
```

- core/pci: More reliable way to update PCI slot power state

  The power control bit (SLOT_CTL, offset: PCIe cap + 0x18) isn't reliable enough to reflect the PCI slot's power state. Instead, the power indication bits are more reliable comparatively. This leads to mismatch between the cached power state and PCI slot's presence state, resulting in the hotplug driver in kernel refuses to unplug the devices properly on the request. The issue was found on below NVMe card on "supermicro,p8dtu2u" machine. We don't have this issue on the integrated PLX 8718 switch.

```
# lspci
0022:01:00.0 PCI bridge: PLX Technology, Inc. PEX 9733 33-lane, \
             9-port PCI Express Gen 3 (8.0 GT/s) Switch (rev aa)
0022:02:01.0 PCI bridge: PLX Technology, Inc. PEX 9733 33-lane, \
             9-port PCI Express Gen 3 (8.0 GT/s) Switch (rev aa)
0022:02:04.0 PCI bridge: PLX Technology, Inc. PEX 9733 33-lane, \
             9-port PCI Express Gen 3 (8.0 GT/s) Switch (rev aa)
0022:02:05.0 PCI bridge: PLX Technology, Inc. PEX 9733 33-lane, \
             9-port PCI Express Gen 3 (8.0 GT/s) Switch (rev aa)
0022:02:06.0 PCI bridge: PLX Technology, Inc. PEX 9733 33-lane, \
             9-port PCI Express Gen 3 (8.0 GT/s) Switch (rev aa)
0022:02:07.0 PCI bridge: PLX Technology, Inc. PEX 9733 33-lane, \
             9-port PCI Express Gen 3 (8.0 GT/s) Switch (rev aa)
0022:17:00.0 Non-Volatile memory controller: Device 19e5:0123 (rev 45)
```

  This updates the cached PCI slot's power state using the power indication bits instead of power control bit, to fix above issue.

- core/pci: Avoid hreset after freset

4.1.52 skiboot-5.4.5

skiboot-5.4.5 was released on Friday June 9th, 2017. It replaces *skiboot-5.4.4* as the current stable release in the 5.4.x series.

Over *skiboot-5.4.4*, we have a small number of bug fixes:

- On FSP platforms: notify FSP of Platform Log ID after Host Initiated Reset Reload Trigging a Host Initiated Reset (when the host detects the FSP has gone out to lunch and should be rebooted), would cause "Unknown Command" messages to appear in the OPAL log.

  This patch implements those messages.

  Log showing unknown command:

```
/ # cat /sys/firmware/opal/msglog | grep -i ,3
[  110.232114723,3] FSP: fsp_trigger_reset() entry
[  188.431793837,3] FSP #0: Link down, starting R&R
[  464.109239162,3] FSP #0: Got XUP with no pending message !
[  466.340598554,3] FSP-DPO: Unknown command 0xce0900
[  466.340600126,3] FSP: Unhandled message ce0900
```

- hw/i2c: Fix early lock drop

  When interacting with an I2C master the p8-i2c driver (common to p9) aquires a per-master lock which it holds for the duration of it's interaction with the master. Unfortunately, when p8_i2c_check_initial_status() detects that the master is busy with another transaction it drops the lock and returns OPAL_BUSY. This is contrary to the driver's locking strategy which requires that the caller aquire and drop the lock. This leads to a crash due to the double unlock(), which skiboot treats as fatal.

- head.S: store all of LR and CTR

  When saving the CTR and LR registers the skiboot exception handlers use the 'stw' instruction which only saves the lower 32 bits of the register. Given these are both 64 bit registers this leads to some strange register dumps, for example:

```
*********************************************
Unexpected exception 200 !
SRR0 : 0000000030016968 SRR1 : 9000000000201000
HSRR0: 0000000000000180 HSRR1: 9000000000001000
LR   : 3003438830823f50 CTR  : 3003438800000018
CFAR : 00000000300168fc
CR   : 40004208  XER: 00000000
```

  In this dump the upper 32 bits of LR and CTR are actually stack gunk which obscures the underlying issue.

4.1.53 skiboot-5.4.6

skiboot-5.4.6 was released on Wednesday June 14th, 2017. It replaces *skiboot-5.4.5* as the current stable release in the 5.4.x series.

Over *skiboot-5.4.5*, we have a small number of bug fixes for FSP based platforms:

- FSP/CONSOLE: Workaround for unresponsive ipmi daemon

  In some corner cases, where FSP is active but not responding to console MBOX message (due to buggy IPMI) and we have heavy console write happening from kernel, then eventually our console buffer becomes full. At this point OPAL starts sending OPAL_BUSY_EVENT to kernel. Kernel will keep on retrying. This is creating kernel soft lockups. In some extreme case when every CPU is trying to write to console, user will not be able to ssh and thinks system is hang.

If we reset FSP or restart IPMI daemon on FSP, system recovers and everything becomes normal.

This patch adds workaround to above issue by returning OPAL_HARDWARE when cosole is full. Side effect of this patch is, we may endup dropping latest console data. But better to drop console data than system hang.

Alternative approach is to drop old data from console buffer, make space for new data. But in normal condition only FSP can update 'next_out' pointer and if we touch that pointer, it may introduce some other race conditions. Hence we decided to just new console write request.

- FSP: Set status field in response message for timed out message

  For timed out FSP messages, we set message status as "fsp_msg_timeout". But most FSP driver users (like surviellance) are ignoring this field. They always look for FSP returned status value in callback function (second byte in word1). So we endup treating timed out message as success response from FSP.

  Sample output:

```
[69902.432509048,7] SURV: Sending the heartbeat command to FSP
[70023.226860117,4] FSP: Response from FSP timed out, word0 = d66a00d7, word1 = 0␣
↪state: 3
....
[70023.226901445,7] SURV: Received heartbeat acknowledge from FSP
[70023.226903251,3] FSP: fsp_trigger_reset() entry
```

  Here SURV code thought it got valid response from FSP. But actually we didn't receive response from FSP.

- FSP: Improve timeout message

  Presently we print word0 and word1 in error log. word0 contains sequence number and command class. One has to understand word0 format to identify command class.

  Lets explicitly print command class, sub command etc.

- FSP/RTC: Remove local fsp_in_reset variable

  Now that we are using fsp_in_rr() to detect FSP reset/reload, fsp_in_reset become redundant. Lets remove this local variable.

- FSP/RTC: Fix possible FSP R/R issue in rtc write path

  fsp_opal_rtc_write() checks FSP status before queueing message to FSP. But if FSP R/R starts before getting response to queued message then we will continue to return OPAL_BUSY_EVENT to host. In some extreme condition host may experience hang. Once FSP is back we will repost message, get response from FSP and return OPAL_SUCCESS to host.

  This patch caches new values and returns OPAL_SUCCESS if FSP R/R is happening. And once FSP is back we will send cached value to FSP.

- hw/fsp/rtc: read/write cached rtc tod on fsp hir.

  Currently fsp-rtc reads/writes the cached RTC TOD on an fsp reset. Use latest fsp_in_rr() function to properly read the cached rtc value when fsp reset initiated by the hir.

  Below is the kernel trace when we set hw clock, when hir process starts.

```
[ 1727.775824] NMI watchdog: BUG: soft lockup - CPU#57 stuck for 23s!␣
↪[hwclock:7688]
[ 1727.775856] Modules linked in: vmx_crypto ibmpowernv ipmi_powernv uio_pdrv_
↪genirq ipmi_devintf powernv_op_panel uio ipmi_msghandler powernv_rng leds_
↪powernv ip_tables x_tables autofs4 ses enclosure scsi_transport_sas crc32c_
↪vpmsum lpfc ipr tg3 scsi_transport_fc
[ 1727.775883] CPU: 57 PID: 7688 Comm: hwclock Not tainted 4.10.0-14-generic #16-
↪Ubuntu
```

```
[ 1727.775883] task: c000000fdfdc8400 task.stack: c000000fdfef4000
[ 1727.775884] NIP: c00000000090540c LR: c0000000000846f4 CTR: 000000003006dd70
[ 1727.775885] REGS: c000000fdfef79a0 TRAP: 0901   Not tainted  (4.10.0-14-
↪generic)
[ 1727.775886] MSR: 9000000000009033 <SF,HV,EE,ME,IR,DR,RI,LE>
[ 1727.775889]   CR: 28024442  XER: 20000000
[ 1727.775890] CFAR: c00000000008472c SOFTE: 1
               GPR00: 0000000030005128 c000000fdfef7c20 c00000000144c900␣
↪ffffffffffffffff4
               GPR04: 0000000028024442 c00000000090540c 9000000000009033␣
↪0000000000000000
               GPR08: 0000000000000000 0000000031fc4000 c000000000084710␣
↪9000000000001003
               GPR12: c0000000000846e8 c00000000fba0100
[ 1727.775897] NIP [c00000000090540c] opal_set_rtc_time+0x4c/0xb0
[ 1727.775899] LR [c0000000000846f4] opal_return+0xc/0x48
[ 1727.775899] Call Trace:
[ 1727.775900] [c000000fdfef7c20] [c00000000090540c] opal_set_rtc_time+0x4c/0xb0␣
↪(unreliable)
[ 1727.775901] [c000000fdfef7c60] [c000000000900828] rtc_set_time+0xb8/0x1b0
[ 1727.775903] [c000000fdfef7ca0] [c000000000902364] rtc_dev_ioctl+0x454/0x630
[ 1727.775904] [c000000fdfef7d40] [c00000000035b1f4] do_vfs_ioctl+0xd4/0x8c0
[ 1727.775906] [c000000fdfef7de0] [c00000000035bab4] SyS_ioctl+0xd4/0xf0
[ 1727.775907] [c000000fdfef7e30] [c00000000000b184] system_call+0x38/0xe0
[ 1727.775908] Instruction dump:
[ 1727.775909] f821ffc1 39200000 7c832378 91210028 38a10020 39200000 38810028␣
↪f9210020
[ 1727.775911] 4bfffe6d e8810020 80610028 4b77f61d <60000000> 7c7f1b78 3860000a␣
↪2fbffff4
```

This is found when executing the op-test-framework fspresetReload testcase

With this fix ran fsp hir torture testcase in the above test which is working fine.

- FSP/CHIPTOD: Return false in error path

4.1.54 skiboot-5.4.7

skiboot-5.4.7 was released on Tuesday September 19th, 2017. It replaces *skiboot-5.4.6* as the current stable release in the 5.4.x series.

Over *skiboot-5.4.6*, we have two backported bug fixes for FSP platforms:

- FSP: Add check to detect FSP Reset/Reload inside fsp_sync_msg()

  During FSP Reset/Reload we move outstanding MBOX messages from msgq to rr_queue including inflight message (fsp_reset_cmdclass()). But we are not resetting inflight message state.

  In extreme corner case where we sent message to FSP via fsp_sync_msg() path and FSP Reset/Reload happens before getting respose from FSP, then we will endup waiting in fsp_sync_msg() until everything becomes normal.

  This patch adds fsp_in_rr() check to fsp_sync_msg() and return error to caller if FSP is in R/R.

- platforms/ibm-fsp/firenze: Fix PCI slot power-off pattern

  When powering off the PCI slot, the corresponding bits should be set to 0bxx00xx00 instead of 0bxx11xx11. Otherwise, the specified PCI slot can't be put into power-off state. Fortunately, it didn't introduce any side-effects so far.

4.1.55 skiboot-5.4.8

skiboot-5.4.8 was released on Wednesday October 11th, 2017. It replaces *skiboot-5.4.7* as the current stable release in the 5.4.x series.

Over *skiboot-5.4.7*, we have a few bug fixes for FSP platforms:

- libflash/file: Handle short read()s and write()s correctly

  Currently we don't move the buffer along for a short read() or write() and nor do we request only the remaining amount.

- FSP/NVRAM: Handle "get vNVRAM statistics" command

  FSP sends MBOX command (cmd : 0xEB, subcmd : 0x05, mod : 0x00) to get vNVRAM statistics. OPAL doesn't maintain any such statistics. Hence return FSP_STATUS_INVALID_SUBCMD.

  Sample OPAL log:

```
[16944.384670488,3] FSP: Unhandled message eb0500
[16944.474110465,3] FSP: Unhandled message eb0500
[16945.111280784,3] FSP: Unhandled message eb0500
[16945.293393485,3] FSP: Unhandled message eb0500
```

- FSP/CONSOLE: Limit number of error logging

  Commit c8a7535f (FSP/CONSOLE: Workaround for unresponsive ipmi daemon, added in skiboot 5.4.6 and 5.7-rc1) added error logging when buffer is full. In some corner cases kernel may call this function multiple time and we may endup logging error again and again.

  This patch fixes it by generating error log only once.

- FSP/CONSOLE: Fix fsp_console_write_buffer_space() call

  Kernel calls fsp_console_write_buffer_space() to check console buffer space availability. If there is enough buffer space to write data, then kernel will call fsp_console_write() to write actual data.

  In some extreme corner cases (like one explained in commit c8a7535f) console becomes full and this function returns 0 to kernel (or space available in console buffer < next incoming data size). Kernel will continue retrying until it gets enough space. So we will start seeing RCU stalls.

  This patch keeps track of previous available space. If previous space is same as current means not enough space in console buffer to write incoming data. It may be due to very high console write operation and slow response from FSP -OR- FSP has stopped processing data (ex: because of ipmi daemon died). At this point we will start timer with timeout of SER_BUFFER_OUT_TIMEOUT (10 secs). If situation is not improved within 10 seconds means something went bad. Lets return OPAL_RESOURCE so that kernel can drop console write and continue.

- FSP/CONSOLE: Close SOL session during R/R

  Presently we are not closing SOL and FW console sessions during R/R. Host will continue to write to SOL buffer during FSP R/R. If there is heavy console write operation happening during FSP R/R (like running *top* command inside console), then at some point console buffer becomes full. fsp_console_write_buffer_space() returns 0 (or less than required space to write data) to host. While one thread is busy writing to console, if some other threads tries to write data to console we may see RCU stalls (like below) in kernel.

  kernel call trace:

```
[ 2082.828363] INFO: rcu_sched detected stalls on CPUs/tasks: { 32} (detected by␣
↪16, t=6002 jiffies, g=23154, c=23153, q=254769)
[ 2082.828365] Task dump for CPU 32:
[ 2082.828368] kworker/32:3    R  running task        0  4637      2 0x00000884
```

```
[ 2082.828375] Workqueue: events dump_work_fn
[ 2082.828376] Call Trace:
[ 2082.828382] [c000000f1633fa00] [c00000000013b6b0] console_unlock+0x570/0x600
↪(unreliable)
[ 2082.828384] [c000000f1633fae0] [c00000000013ba34] vprintk_emit+0x2f4/0x5c0
[ 2082.828389] [c000000f1633fb60] [c00000000099e644] printk+0x84/0x98
[ 2082.828391] [c000000f1633fb90] [c0000000000851a8] dump_work_fn+0x238/0x250
[ 2082.828394] [c000000f1633fc60] [c0000000000ecb98] process_one_work+0x198/0x4b0
[ 2082.828396] [c000000f1633fcf0] [c0000000000ed3dc] worker_thread+0x18c/0x5a0
[ 2082.828399] [c000000f1633fd80] [c0000000000f4650] kthread+0x110/0x130
[ 2082.828403] [c000000f1633fe30] [c000000000009674] ret_from_kernel_thread+0x5c/
↪0x68
```

Hence lets close SOL (and FW console) during FSP R/R.

- FSP/CONSOLE: Do not associate unavailable console

  Presently OPAL sends associate/unassociate MBOX command for all FSP serial console (like below OPAL message). We have to check console is available or not before sending this message.

  OPAL log:

```
[ 5013.227994012,7] FSP: Reassociating HVSI console 1
[ 5013.227997540,7] FSP: Reassociating HVSI console 2
```

- FSP: Disable PSI link whenever FSP tells OPAL about impending Reset/Reload

  Commit 42d5d047 fixed scenario where DPO has been initiated, but FSP went into reset before the CEC power down came in. But this is generic issue that can happen in normal shutdown path as well.

  Hence disable PSI link as soon as we detect FSP impending R/R.

- fsp: return OPAL_BUSY_EVENT on failure sending FSP_CMD_POWERDOWN_NORM Also, return OPAL_BUSY_EVENT on failure sending FSP_CMD_REBOOT / DEEP_REBOOT.

  We had a race condition between FSP Reset/Reload and powering down the system from the host:

  Roughly:

#	FSP	Host
1	Power on	
2		Power on
3	(inject EPOW)	
4	(trigger FSP R/R)	
5		Processes EPOW event, starts shutting down
6		calls OPAL_CEC_POWER_DOWN
7	(is still in R/R)	
8		gets OPAL_INTERNAL_ERROR, spins in opal_poll_events
9	(FSP comes back)	
10		spinning in opal_poll_events
11	(thinks host is running)	

  The call to OPAL_CEC_POWER_DOWN is only made once as the reset/reload error path for fsp_sync_msg() is to return -1, which means we give the OS OPAL_INTERNAL_ERROR, which is fine, except that our own API docs give us the opportunity to return OPAL_BUSY when trying again later may be successful, and we're ambiguous as to if you should retry on OPAL_INTERNAL_ERROR.

  For reference, the linux code looks like this:

```
static void __noreturn pnv_power_off(void)
{
        long rc = OPAL_BUSY;

        pnv_prepare_going_down();

        while (rc == OPAL_BUSY || rc == OPAL_BUSY_EVENT) {
                rc = opal_cec_power_down(0);
                if (rc == OPAL_BUSY_EVENT)
                        opal_poll_events(NULL);
                else
                        mdelay(10);
        }
        for (;;)
                opal_poll_events(NULL);
}
```

Which means that *practically* our only option is to return OPAL_BUSY or OPAL_BUSY_EVENT.

We choose OPAL_BUSY_EVENT for FSP systems as we do want to ensure we're running pollers to communicate with the FSP and do the final bits of Reset/Reload handling before we power off the system.

4.1.56 skiboot-5.5.0

skiboot-5.5.0 was released on Friday April 7th 2017. It is the new stable release of skiboot, taking over from the 5.4 release, first released on November 11th 2016.

skiboot-5.5.0 contains all bug fixes as of *skiboot-5.4.3* and *skiboot-5.1.19* (the currently maintained stable releases).

For how the skiboot stable releases work, see *Skiboot stable tree rules and releases* for details.

This release is a good level set of POWER9 support for bringup activities. If you are doing bringup, it is strongly suggested you continue to follow skiboot master.

After skiboot 5.5.0, we move to a regular six week release cycle, similar to op-build, but slightly offset to allow for a short stabilisation period. Expected release dates and contents are tracked using GitHub milestone and issues: https://github.com/open-power/skiboot/milestones

Changes in skiboot-5.5.0

See changes in the release candidates:

- *skiboot-5.5.0-rc1*
- *skiboot-5.5.0-rc2*
- *skiboot-5.5.0-rc3*

Changes since skiboot-5.5.0-rc3

- hdat: parse processor attached i2c devices

  Adds basic parsing for i2c devices that are attached to the processor I2C interfaces. This is mainly VPD SEEPROMs.

- libflash/blocklevel: Add blocklevel_smart_erase()

With recent changes to flash drivers in linux not all erase blocks are 4K anymore. While most level of the pflash/gard tool stacks were written to not mind, it turns out there are bugs which means not 4K erase block backing stores aren't handled all that well. Part of the problem is the FFS layout that is 4K aligned and with larger block sizes pflash and the gard tool don't check if their erase commands are erase block aligned - which they are usually not with 64K erase blocks.

This patch aims to add common functionality to blocklevel so that (at least) pflash and the gard tool don't need to worry about the problem anymore.

- external/pflash: Use blocklevel_smart_erase()
- external/gard: Use blocklevel_smart_erase()
- libstb/create-container: Add full container build and sign with imprint keys

  This adds support for writing all the public key and signature fields to the container header, and for dumping the prefix and software headers so they may may be signed, and for signing those headers with the imprint keys.
- asm: do not set SDR1 on POWER9. This register does not exist in ISAv3.

Testing:

- mambo: Allow setting the Linux command line from the environment

  For automated testing it's helpful to be able to set the Linux command line via an environment variable.
- mambo: Add util function for breaking on console output

Contributors

Processed 408 csets from 31 developers

3 employers found

A total of 24073 lines added, 16759 removed (delta 7314)

Extending the analysis done for the last few releases, we can see our trends in code review across versions:

Release	csets	Ack	Reviews	Tested	Reported
5.0	329	15	20	1	0
5.1	372	13	38	1	4
5.2-rc1	334	20	34	6	11
5.3-rc1	302	36	53	4	5
5.4.0	361	16	28	1	9
5.5.0	408	11	48	14	10

I am absolutely *thrilled* as to the uptick of reviews and tested-by occuring over our 5.4.0 release. Although we are not yet back up to 5.3 era levels for review, we're much closer. For tested-by, we've set a new record, which is excellent!

Developers with the most changesets

Developer	#	%
Benjamin Herrenschmidt	139	(34.1%)
Stewart Smith	60	(14.7%)
Oliver O'Halloran	54	(13.2%)
Gavin Shan	23	(5.6%)

Continued on next page

Table 4.7 – continued from previous page

Developer	#	%
Michael Neuling	20	(4.9%)
Vasant Hegde	15	(3.7%)
Cyril Bur	15	(3.7%)
Claudio Carvalho	14	(3.4%)
Andrew Donnellan	11	(2.7%)
Ananth N Mavinakayanahalli	9	(2.2%)
Alistair Popple	6	(1.5%)
Nicholas Piggin	5	(1.2%)
Cédric Le Goater	5	(1.2%)
Pridhiviraj Paidipeddi	5	(1.2%)
Michael Ellerman	4	(1.0%)
Shilpasri G Bhat	4	(1.0%)
Russell Currey	3	(0.7%)
Jack Miller	2	(0.5%)
Chris Smart	2	(0.5%)
Dave Heller	1	(0.2%)
Akshay Adiga	1	(0.2%)
Reza Arbab	1	(0.2%)
Matt Brown	1	(0.2%)
Frederic Barrat	1	(0.2%)
Hank Chang	1	(0.2%)
Willie Liauw	1	(0.2%)
Werner Fischer	1	(0.2%)
Jeremy Kerr	1	(0.2%)
Patrick Williams	1	(0.2%)
Joel Stanley	1	(0.2%)
Alexey Kardashevskiy	1	(0.2%)

Developers with the most changed lines

Developer	#	%
Oliver O'Halloran	18278	(48.5%)
Benjamin Herrenschmidt	5512	(14.6%)
Cyril Bur	3184	(8.4%)
Alistair Popple	3102	(8.2%)
Stewart Smith	2757	(7.3%)
Gavin Shan	802	(2.1%)
Ananth N Mavinakayanahalli	544	(1.4%)
Claudio Carvalho	489	(1.3%)
Dave Heller	425	(1.1%)
Willie Liauw	361	(1.0%)
Andrew Donnellan	315	(0.8%)
Michael Neuling	290	(0.8%)
Vasant Hegde	253	(0.7%)
Shilpasri G Bhat	228	(0.6%)
Nicholas Piggin	222	(0.6%)
Reza Arbab	198	(0.5%)
Russell Currey	158	(0.4%)

Continued on next page

Table 4.8 – continued from previous page

Developer	#	%
Jack Miller	127	(0.3%)
Cédric Le Goater	126	(0.3%)
Chris Smart	95	(0.3%)
Akshay Adiga	57	(0.2%)
Hank Chang	56	(0.1%)
Pridhiviraj Paidipeddi	47	(0.1%)
Michael Ellerman	29	(0.1%)
Matt Brown	29	(0.1%)
Alexey Kardashevskiy	2	(0.0%)
Frederic Barrat	1	(0.0%)
Werner Fischer	1	(0.0%)
Jeremy Kerr	1	(0.0%)
Patrick Williams	1	(0.0%)
Joel Stanley	1	(0.0%)

Developers with the most lines removed

Developer	#	%
Oliver O'Halloran	8516	(50.8%)
Werner Fischer	1	(0.0%)

Developers with the most signoffs

Total: 364

Developer	#	%
Stewart Smith	348	(95.6%)
Michael Neuling	6	(1.6%)
Oliver O'Halloran	3	(0.8%)
Benjamin Herrenschmidt	2	(0.5%)
Vaidyanathan Srinivasan	1	(0.3%)
Hank Chang	1	(0.3%)
Jack Miller	1	(0.3%)
Gavin Shan	1	(0.3%)
Alistair Popple	1	(0.3%)

Developers with the most reviews

Total 50

Developer	#	%
Vasant Hegde	14	(28.0%)
Andrew Donnellan	9	(18.0%)
Russell Currey	6	(12.0%)
Cédric Le Goater	5	(10.0%)
Oliver O'Halloran	4	(8.0%)
Vaidyanathan Srinivasan	3	(6.0%)
Gavin Shan	3	(6.0%)
Alistair Popple	2	(4.0%)
Frederic Barrat	2	(4.0%)
Mahesh Salgaonkar	1	(2.0%)
Cyril Bur	1	(2.0%)

Developers with the most test credits

Total 14

Developer	#	%
Willie Liauw	4	(28.6%)
Mark E Schreiter	3	(21.4%)
Claudio Carvalho	3	(21.4%)
Gavin Shan	1	(7.1%)
Michael Neuling	1	(7.1%)
Pridhiviraj Paidipeddi	1	(7.1%)
Chris Smart	1	(7.1%)

Developers who gave the most tested-by credits

Total 14

Developer	#	%
Gavin Shan	7	(50.0%)
Stewart Smith	4	(28.6%)
Chris Smart	1	(7.1%)
Oliver O'Halloran	1	(7.1%)
Ananth N Mavinakayanahalli	1	(7.1%)

Developers with the most report credits

Total 10

Developer	#	%
Hank Chang	4	(40.0%)
Mark E Schreiter	3	(30.0%)
Guilherme G. Piccoli	1	(10.0%)
Colin Ian King	1	(10.0%)
Pradipta Ghosh	1	(10.0%)

Developers who gave the most report credits

Total 10

Developer	#	%
Gavin Shan	8	(80.0%)
Andrew Donnellan	1	(10.0%)
Jeremy Kerr	1	(10.0%)

Top changeset contributors by employer

Employer	#	%
IBM	406	(99.5%)
SuperMicro	1	(0.2%)
Thomas-Krenn AG	1	(0.2%)

Top lines changed by employer

Employer	#	%
IBM	37329	(99.0%)
SuperMicro	361	(1.0%)
Thomas-Krenn AG	1	(0.0%)

Employers with the most signoffs

Total 364

Employer	#	%
IBM	363	(99.7%)
(Unknown)	1	(0.3%)

Employers with the most hackers

Total 31

Employer	#	%
IBM	29	(93.5%)
Thomas-Krenn AG	1	(3.2%)
SuperMicro	1	(3.2%)

4.1.57 skiboot-5.5.0-rc1

skiboot-5.5.0-rc1 was released on Tuesday March 28th 2017. It is the first release candidate of skiboot 5.5, which will become the new stable release of skiboot following the 5.4 release, first released November 11th 2016.

skiboot-5.5.0-rc1 contains all bug fixes as of *skiboot-5.4.3* and *skiboot-5.1.19* (the currently maintained stable releases).

For how the skiboot stable releases work, see *Skiboot stable tree rules and releases* for details.

The current plan is to cut the final 5.5.0 by April 8th, with skiboot 5.5.0 being for all POWER8 and POWER9 platforms in op-build v1.16 (Due April 12th). This is a short cycle as this release is mainly targetted towards POWER9 bringup efforts.

Following skiboot-5.5.0, we will move to a regular six week release cycle, similar to op-build, but slightly offset to allow for a short stabilisation period. Expected release dates and contents are tracked using GitHub milestone and issues: https://github.com/open-power/skiboot/milestones

Over skiboot-5.4, we have the following changes:

New Platforms

- SuperMicro's (SMC) P8DNU: An astbmc based POWER8 platform
- Add a generic platform to help with bringup of new systems.
- Four POWER9 based systems (NOTE: All POWER9 systems should be considered for bringup use only at this point):
  - Romulus
  - Witherspoon (a POWER9 system with NVLink2 attached GPUs)
  - Zaius (OpenCompute platform, also known as "Barreleye 2")
  - ZZ (FSP based system)

New features

- System reset IPI facility and Mambo implementation Add an opal call *OPAL_SIGNAL_SYSTEM_RESET* which allows system reset exceptions to be raised on other CPUs and act as an NMI IPI. There is an initial simple Mambo implementation, but allowances are made for a more complex hardware implementation.

  The Mambo implementation is based on the RFC implementation for POWER8 hardware (see https://patchwork.ozlabs.org/patch/694794/) which we hope makes it into a future release.

  This implements an in-band NMI equivalent.
- add CONTRIBUTING.md, ensuring that people new to the project have a one-stop place to find out how to get started.
- interrupts: Add optional name for OPAL interrupts

  This adds the infrastructure for an interrupt source to provide a name for an interrupt directed toward OPAL. Those names will be put into an "opal-interrupts-names" property which is a standard DT string list corresponding 1:1 with the "opal-interrupts" property. PSI interrupts get names, and this is visible in Linux through /proc/interrupts
- platform: add OPAL_REBOOT_FULL_IPL reboot type

  There may be circumstances in which a user wants to force a full IPL reboot rather than using fast reboot. Add a new reboot type, OPAL_REBOOT_FULL_IPL, that disables fast reboot. On platforms which don't support fast reboot, this will be equivalent to a normal reboot.

- phb3: Trick to allow control of the PCIe link width and speed

  This implements a hook inside OPAL that catches 16 and 32 bit writes to the link status register of the PHB.

  It allows you to write a new speed or a new width, and OPAL will then cause the PHB to renegociate.

  Example:

  First read the link status on PHB4:

```
setpci -s 0004:00:00.0 0x5a.w
a103
```

  It's at x16 Gen3 speed (8GT/s)

  bits 0x0ff0 are the width and 0x000f the speed. The width can be 1 to 16 and the speed 1 to 3 (2.5, 5 and 8GT/s)

  Then try to bring it down to 1x Gen1 :

```
setpci -s 0004:00:00.0 0x5a.w=0xa011
```

  Observe the result in the PHB:

```
/ # lspci -s 0004:00:00.0 -vv
0004:00:00.0 PCI bridge: IBM Device 03dc (prog-if 00 [Normal decode])
.../...
LnkSta: Speed 2.5GT/s, Width x1, TrErr- Train- SlotClk- DLActive+ BWMgmt-␣
↪ABWMgmt+
```

  And in the device:

```
/ # lspci -s 0004:01:00.0 -vv
.../...
LnkSta: Speed 2.5GT/s, Width x1, TrErr- Train- SlotClk+ DLActive- BWMgmt-␣
↪ABWMgmt-
```

- core/init: Add hdat-map property to OPAL node.

  Exports the HDAT heap to the OS. This allows the OS to view the HDAT heap directly. This allows us to view the HDAT area without having to use getmemproc.

- Add a generic platform: If /bmc in device tree, attempt to init one For the most part, this gets us somewhere on some OpenPOWER systems before there's a platform file for that machine.

  Useful in bringup only, and marked as such with scary looking log messages.

Core

- asm: Don't try to set LPCR:LPES1 on P8 and P9, the bit doesn't exist.
- pci: Add a framework for quirks

  In future we may want to be able to do fixups for specific PCI devices in skiboot, so add a small framework for doing this.

  This is not intended for the same purposes as quirks in the Linux kernel, as the PCI devices that quirks can match for in skiboot are not properly configured. This is intended to enable having a custom path to make changes that don't directly interact with the PCI device, for example adding device tree entries.

- hw/slw: fix possible NULL dereference

- slw: Print enabled stop states on boot
- uart: Fix Linux pass-through policy, provide NVRAM override option
- libc/stdio/vsnprintf.c: add explicit fallthrough, this silences a recent (GCC 7.x) warning
- init: print the FDT blob size in decimal
- init: Print some more info before booting linux

  The kernel command line from nvram and the stdout-path are useful to know when debugging console related problems.
- Makefile: Disable stack protector due to gcc problems

  Depending on how it was built, gcc will use the canary from a global (works for us) or from the TLS (doesn't work for us and accesses random stuff instead).

  Fixing that would be tricky. There are talks of adding a gcc option to force use of globals, but in the meantime, disable the stack protector.
- Stop using 3-operand cmp[l][i] for latest binutils Since a5721ba270, binutils does not support 3-operand cmp[l][i]. This adds (previously optional) parameter L.
- buddy: Add a simple generic buddy allocator
- stack: Don't recurse into __stack_chk_fail
- Makefile: Use -ffixed-r13 We use r13 for our own stuff, make sure it's properly fixed
- Always set ibm,occ-functional-state correctly
- psi: fix the xive registers initialization on P8, which seems to be fine for real HW but causes a lof of pain under qemu
- slw: Set PSSCR value for idle states
- Limit number of "Poller recursion detected" errors to display

  In some error conditions, we could spiral out of control on this and spend all of our time printing the exact same backtrace.

  Limit it to 16 times, because 16 is a nice number.
- slw: do SLW timer testing while holding xscom lock

  We add some routines that let a caller get the xscom lock once and then do a bunch of xscoms while holding it. In some situations without this, it could take long enough to get the xscom lock that the 1ms timeout would expire and we'd falsely think the SLW timer didn't work when in fact it did.
- wait_for_resource_loaded: don't needlessly sleep for 5ms
- run pollers in cpu_process_local_jobs() if running job synchonously
- fsp: Don't recurse pollers in ibm_fsp_terminate
- chiptod: More hardening against -1 chip ID
- interrupts: Rewrite/correct doc for opal_set/get_xive
- cpu: Don't enable nap mode/PM mode on non-P8
- platform: Call generic platform probe and init UART there
- psi: Don't register more interrupts than the HW supports
- psi: Add DT option to disable LPC interrupts

I2C and TPM

- p8i2c: Use calculated poll_interval when booting OPAL Otherwise we'd default to 2seconds (TIMER_POLL) during boot on chips with a functional i2c interrupt, leading to slow i2c during boot (or hitting timeouts instead).
- i2c: Add i2c_run_req() to crank the state machine for a request
- tpm_i2c_nuvoton: work out the polling time using mftb()
- tpm_i2c_nuvoton: handle errors after reading the tpm fifo
- tpm_i2c_nuvoton: cleanup variables in tpm_read_fifo()
- tpm_i2c_nuvoton: handle errors after writting the tpm fifo
- tpm_i2c_nuvoton: cleanup variables in tpm_write_fifo()
- tpm_i2c_nuvoton: handle errors after writing sts.commandReady in step 5
- tpm_i2c_nuvoton: handle errors after writing sts.go
- tpm_i2c_nuvoton: handle errors after checking the tpm fifo status
- tpm_i2c_nuvoton: return burst_count in tpm_read_burst_count()
- tpm_i2c_nuvoton: isolate the code that handles the TPM_TIMEOUT_D timeout
- tpm_i2c_nuvoton: handle errors after reading sts.commandReady
- tpm_i2c_nuvoton: add tpm_status_read_byte()
- tpm_i2c_nuvoton: add tpm_check_status()
- tpm_i2c_nuvoton: rename defines to shorter names
- tpm_i2c_interface: decouple rc from being done with i2c request
- tpm_i2c_interface: set timeout before each request
- i2c: Add nuvoton quirk, disallowing i2cdetect as it locks TPM

  p8-i2c reset things manually in some error conditions
- stb: create-container and wrap skiboot in Secure/Trusted Boot container

  We produce **UNSIGNED** skiboot.lid.stb and skiboot.lid.xz.stb as build artifacts.

  These are suitable blobs for flashing onto Trusted Boot enabled op-build builds *WITH* the secure boot jumpers *ON* (i.e. *NOT* in secure mode). It's just enough of the Secure and Trusted Boot container format to make Hostboot behave.

PCI

- core/pci: Support SRIOV VFs

  Currently, skiboot can't see SRIOV VFs. It introduces some troubles as I can see: The device initialization logic (phb->ops->device_init()) isn't applied to VFs, meaning we have to maintain same and duplicated mechanism in kernel for VFs only. It introduces difficulty to code maintaining and prone to lose sychronization.

  This was motivated by bug reported by Carol: The VF's Max Payload Size (MPS) isn't matched with PF's on Mellanox's adapter even kernel tried to make them same. It's caused by readonly PCIECAP_EXP_DEVCTL register on VFs. The skiboot would be best place to emulate this bits to eliminate the gap as I can see.

  This supports SRIOV VFs. When the PF's SRIOV capability is populated, the number of maximal VFs (struct pci_device) are instanciated, but but not usable yet. In the mean while, PCI config register filter is registered

against PCIECAP_SRIOV_CTRL_VFE to capture the event of enabling or disabling VFs. The VFs are initialized, put into the PF's children list (pd->children), populate its PCI capabilities, and register PCI config register filter against PCICAP_EXP_DEVCTL. The filter's handler caches what is written to MPS field and returns the cached value on read, to eliminate the gap mentioned as above.

- core/pci: Avoid hreset after freset

  Commit 5ac71c9 ("pci: Avoid hot resets at boot time") missed to avoid hot reset after fundamental reset for PCIe common slots.

  This fixes it.

- core/pci: Enforce polling PCIe link in hot-add path

  In surprise hot-add path, the power state isn't changed on hardware. Instead, we set the cached power state (@slot->power_state) and return OPAL_SUCCESS. The upper layer starts the PCI probing immediately when receiving OPAL_SUCCESS. However, the PCIe link behind the PCI slot is likely down. Nothing will be probed from the PCI slot even we do have PCI adpater connected to the slot.

  This fixes the issue by returning OPAL_ASYNC_COMPLETION to force upper layer to poll the PCIe link before probing the PCI devices behind the slot in surprise and managed hot-add paths.

- **hw/phb3: fix error handling in complete reset** During a complete reset, when we get a timeout waiting for pending transaction in state PHB3_STATE_CRESET_WAIT_CQ, we mark the PHB as permanently broken.

  Set the state to PHB3_STATE_FENCED so that the kernel can retry the complete reset.

- phb3: Lock the PHB on set_xive callbacks

p8dnu platform

- astbmc/p8dnu: Enable PCI slot's power supply on PEX9733 in hot-add path
- astbmc/p8dnu: Enable PCI slot's power supply on PEX8718 in hot-add path
- core/pci: Mark broken PDC on slots without surprise hotplug capability

  We has to support surprise hotplug on PCI slots that don't support it on hardware. So we're fully utilizing the PCIe link state change event to detect the events (hot-remove and hot-add). The PDC (Presence Detection Change) event isn't reliable for the purpose. For example, PEX8718 on superMicro's machines.

  This adds another PCI slot property "ibm,slot-broken-pdc" in the device-tree, to indicate the PDC isn't reliable on those (software claimed) surprise pluggable slots.

- core/pci: Fix PCIe slot's presence

  According to PCIe spec, the presence bit is hardcoded to 1 if PCIe switch downstream port doesn't support slot capability. The register used for the check in pcie_slot_get_presence_state() is wrong. It should be PCIe capability register instead of PCIe slot capability register. Otherwise, we always have present bit on the PCI topology. The issue is found on Supermicro's p8dtu2u machine:

```
# lspci -t
-+-[0022:00]---00.0-[01-08]----00.0-[02-08]--+-01.0-[03]----00.0
 |                                            \-02.0-[04-08]--
# cat /sys/bus/pci/slots/S002204/adapter
1
# lspci -vvs 0022:02:02.0
# lspci -vvs 0022:02:02.0
0022:02:02.0 PCI bridge: PLX Technology, Inc. PEX 8718 16-Lane, \
5-Port PCI Express Gen 3 (8.0 GT/s) Switch (rev ab) (prog-if 00 [Normal decode])
```

```
    :
 Capabilities: [68] Express (v2) Downstream Port (Slot+), MSI 00
    :
    SltSta:    Status: AttnBtn- PowerFlt- MRL- CmdCplt- PresDet- Interlock-
               Changed: MRL- PresDet- LinkState-

This fixes the issue by checking the correct register (PCIe capability).
Also, the register's value is cached in advance as we did for slot and
link capability.
```

- core/pci: More reliable way to update PCI slot power state

  The power control bit (SLOT_CTL, offset: PCIe cap + 0x18) isn't reliable enough to reflect the PCI slot's power state. Instead, the power indication bits are more reliable comparatively. This leads to mismatch between the cached power state and PCI slot's presence state, resulting in the hotplug driver in kernel refuses to unplug the devices properly on the request. The issue was found on below NVMe card on "supermicro,p8dtu2u" machine. We don't have this issue on the integrated PLX 8718 switch.

```
 # lspci
 0022:01:00.0 PCI bridge: PLX Technology, Inc. PEX 9733 33-lane, \
              9-port PCI Express Gen 3 (8.0 GT/s) Switch (rev aa)
 0022:02:01.0 PCI bridge: PLX Technology, Inc. PEX 9733 33-lane, \
              9-port PCI Express Gen 3 (8.0 GT/s) Switch (rev aa)
 0022:02:04.0 PCI bridge: PLX Technology, Inc. PEX 9733 33-lane, \
              9-port PCI Express Gen 3 (8.0 GT/s) Switch (rev aa)
 0022:02:05.0 PCI bridge: PLX Technology, Inc. PEX 9733 33-lane, \
              9-port PCI Express Gen 3 (8.0 GT/s) Switch (rev aa)
 0022:02:06.0 PCI bridge: PLX Technology, Inc. PEX 9733 33-lane, \
              9-port PCI Express Gen 3 (8.0 GT/s) Switch (rev aa)
 0022:02:07.0 PCI bridge: PLX Technology, Inc. PEX 9733 33-lane, \
              9-port PCI Express Gen 3 (8.0 GT/s) Switch (rev aa)
 0022:17:00.0 Non-Volatile memory controller: Device 19e5:0123 (rev 45)

This updates the cached PCI slot's power state using the power
indication bits instead of power control bit, to fix above issue.
```

Utilities

- opal-prd: Direct systemd to always restart opal-prd Always restart the opal-prd daemon, irrespective of why it stopped.
- external/ffspart: Simple C program to be able to make an FFS partition
- getscom: Add chip info for P9.
- gard: Fix make dist target
- pflash/libflash: arch_flash_arm: Don't assume mtd labels are short

libffs

- libffs: Understand how to create FFS partition TOCs and entries.

BMC Based systems

- platforms/astbmc: Support PCI slots for palmetto

- habanero/slottable: Remove Network Mezz(2, 0) from PHB1.
- BMC/PCI: Check slot tables against detected devices On BMC machines, we have slot tables of built in PHBs, slots and devices that are physically present in the system (such as the BMC itself). We can use these tables to check what we *detected* against what *should* be in the system and throw an error if they differ.

  We have seen this occur a couple of times while still booting, giving the user just an empty petitboot screen and not much else to go on. This patch helps in that we get a skiboot error message, and at some point in the future when we pump them up to the OS we could get a big friendly error message telling you you're having a bad day.
- pci/quirk: Populate device tree for AST2400 VGA

  Adding these properties enables the kernel to function in the same way that it would if it could no longer access BMC configuration registers through a backdoor, which may become the default in future.

  The comments describe how isolating the host from the BMC could be achieved in skiboot, assuming all kernels that the system boots support this. Isolating the BMC and the host from each other is important if they are owned by different parties; for example, a cloud provider renting machines "bare metal".
- astbmc/pnor: Use mbox-flash for flash accesses

  If the BMC is MBOX protocol aware, request flash reads/writes over the MBOX regs. This inits the blocklevel for pnor access with mbox-flash.
- ast: Account for differences between 2400 vs 2500
- platform: set default bmc_platform The bmc_platform pointer is set to NULL by default and on non-AMI BMC platforms. As a result a few places in hw/ipmi/ipmi-sel.c will blindly dereference a NULL pointer.

POWER9

- external: Update xscom utils for type 1 indirect accesses
- xscom: Harden indirect writes
- xscom: Add POWER9 scom reset
- homer : Enable HOMER region reservation for POWER9
- slw: Define stop idle states for P9 DD1
- slw: Fix parsing of supported STOP states
- slw: only enable supported STOP states
- dts: add support for p9 cores
- asm: Add POWER9 case to init_shared_sprs

  For now, setup the HID and HMEER. We'll add more as we get good default values from HW.
- xive/psi/lpc: Handle proper clearing of LPC SerIRQ latch on POWER9 DD1
- lpc: Mark the power9 LPC bus as compatible with power8
- Fix typo in PIR mask for POWER9. Fixes booting multi-chip.
- vpd: add vpd_valid() to check keyword VPD blobs

  Adds a function to check whether a blob is a valid IBM ASCII keyword VPD blob. This allows us to recognise when we do and do not have a VPD blob and act accordingly.
- core/cpu.c: Use a device-tree node to detect nest mmu presence The nest mmu address scom was hardcoded which could lead to boot failure on POWER9 systems without a nest mmu. For example Mambo doesn't model the nest mmu which results in failure when calling opal_nmmu_set_ptcr() during kernel load.

- psi: Fix P9 BAR setup on multi-chips

PHB4:

- phb4: Fix TVE encoding for start address
- phb4: Always assign powerbus BARs

  HostBoot configure them with weird values that confuse us, instead let's just own the assignment. This is temporary, I will centralize memory map management next but this gets us going.
- phb4: Fix endian issue with link control2/status2 registers Fixes training at larger than PCIe Gen1 speeds.
- phb4: Add ability to log config space access Useful for debugging
- phb4: Change debug prints Currently we print "PHB4" and mean either "PHB version 4" or "PHB number 4" which can be quite confusing.
- phb4: Fix config space enable bits on DD1
- phb4: Fix location of EEH enable bits
- phb4: Fix setting of max link speed
- phb4: Updated inits as of PHB4 spec 0.52

HDAT fixes:

- hdat: Parse BMC nodes much earlier

  This moves the parsing of the BMC and LPC details to the start of the HDAT parsing. This allows us to enable the Skiboot log console earlier so we can get debug output while parsing the rest of the HDAT.
- astbmc: Don't do P8 PSI or DT fixups on P9

  Previously the HDAT format was only ever used with IBM hardware so it would store vital product data (VPD) blobs in the IBM ASCII Keyword VPD format. With P9 HDAT is used on OpenPower machines which use Industry Standard DIMMs that provide their product data through a "Serial Present Detect" EEPROM mounted on the DIMM.

  The SPD blob has a different format and is exported in the device-tree under the "spd" property rather than the "ibm,vpd" property. This patch adds support for recognising these blobs and placing them in the appropriate DT property.
- hdat: Add __packed to all HDAT structures and workaround HB reserve

  Some HDAT structures aren't properly aligned. We were using __packed on some but not others and got at least one wrong (HB reserve). This adds it everywhere to avoid such problems.

  However this then triggers another problem where HB gives us a crazy range (0.256M) to reserve with no label, which triggers an assertion failure later on in mem_regions.c.

  So also add a test to skip any region starting at 0 until we can undertand that better and have it fixed one way or another.
- hdat: Ignore broken memory reserves

  Ignore HDAT memory reserves > 512MB. These are considered bogus and workaround known HDAT bugs.
- hdat: Add BMC device-tree node for P9 OpenPOWER systems
- hdat: Fix interrupt & device_type of UART node

  The interrupt should use a standard "interrupts" property. The UART node also need a device_type="serial" property for historical reasons otherwise Linux won't pick it up.
- parse and export STOP levels

- add new sppcrd_chip_info fields
- add radix-AP-encodings
- stop using proc_int_line in favor of pir
- rename add_icp() to add_xics_icp()
- Add support for PHB4
- create XIVE nodes under each xscom node
- Add P9 compatible property
- Parse hostboot memory reservations from HDAT
- Add new fields to IPL params structure and update sys family for p9.
- Fix ibm,pa-features for all CPU types
- Fix XSCOM nodes for P9
- Remove deprecated 'ibm, mem-interleave-scope' from DT on POWER9
- Grab system model name from HDAT when available
- Grab vendor information from HDAT when available
- SPIRA-H/S changes for P9
- Add BMC and LPC IOPATH support
- handle ISDIMM SPD blobs
- make HDIF_child() print more useful errors
- Add PSI HB xscom details
- Add new fields to proc_init_data structure
- Add processor version check for hs service ntuple
- add_iplparams_serial - Validate HDIF_get_iarray_size() return value

XIVE:

The list of XIVE fixes and updates is extensive. Below is only a portion of the changes that have gone into skiboot 5.5.0-rc1 for the new XIVE hardware that is present in POWER9:

- xive: Enable backlog on queues
- xive: Use for_each_present_cpu() for setting up XIVE
- xive: Fix logic in opal_xive_get_xirr()
- xive: Properly initialize new VP and EQ structures
- xive: Improve/fix EOI of LSIs
- xive: Add FIXME comments about mask/umask races
- xive: Fix memory barrier in opal_xive_get_xirr()
- xive: Don't try to find a target EQ for prio 0xff
- xive: Bump table sizes in direct mode
- xive: Properly register escalation interrupts
- xive: Split the OPAL irq flags from the internal ones

- xive: Don't touch ESB masks unless masking/unmasking
- xive: Fix xive_get_ir_targetting()
- xive: Cleanup escalation PQ on queue change
- xive: Add *any chip* for allocating interrupts
- xive: Add chip_id to get_vp_info
- xive: Add opal_xive_get/set_vp_info
- xive: Add VP alloc/free OPAL functions
- xive: Workaround for bad DD1 checker
- xive: Add more checks for exploitation mode
- xive: Add support for EOIs via OPAL
- xive/phb4: Work around broken LSI control on P9 DD1
- xive: Forward interrupt names callback
- xive: Export opal_xive_reset() arguments in OPAL API
- xive: Add interrupt allocator
- xive: Implement xive_reset
- xive: Don't assert if xive_get_vp() fails
- xive: Expose exploitation mode DT properties
- xive: Use a constant for max# of chips
- xive: Keep track of which interrupts were ever enabled In order to speed up xive reset
- xive: Implement internal VP allocator
- xive: Add xive_get/set_queue_info
- xive: Add helpers to encode and decode VP numbers
- xive: Add API to donate pages in indirect mode
- xive: Add asynchronous cache updates and update irq targetting
- xive: Split xive_provision_cpu() and use cache watch for VP
- xive: Add cache scrub to push watch updates to memory
- xive: Mark XIVE owned EQs with a specific flag
- xive: Use an allocator for EQDs
- xive: Break assumption that block ID == chip ID
- xive/phb4: Handle bad ESB offsets in PHB4 DD1
- xive: Implement get/set_irq_config APIs
- xive: Rework xive_set_eq_info() to store all info even when masking
- xive: Implement cache watch and use it for EQs
- xive: Add locking to some API calls
- xive: Add opal_xive_get_irq_info()
- xive: Add CPU node "interrupts" properties representing the IPIs

- xive: Add basic opal_xive_reset() call and exploitation mode
- xive: Add support for escalation interrupts
- xive: OPAL API update
- xive: Add some dump facility for debugging
- xive: Document exploitation mode (Pretty much work in progress)
- xive: Indirect table entries must have top bits "type" set
- xive: Remove unused field and clarify comment
- xive: Provide a way to override some IPI sources
- xive: Add helper to retrieve an IPI trigger port
- xive: Fix IPI EOI logic in opal_xive_eoi()
- xive: Don't try to EOI a masked source
- xive: Fix comments in xive_source_set_xive()
- xive: Fix comments in xive_get_ive()
- xive: Configure forwarding ports
- xive: Fix mangling of interrupt server# in opal_get/set_xive()
- xive: Fix interrupt number mangling

Fast-reboot

- fast-reboot: creset PHBs on fast reboot On fast reboot, perform a creset of all PHBs. This ensures that any PHBs that are fenced will be working after the reboot.
- fast-reboot: Enable fast reboot with CAPI adapters in CAPI mode CAPI mode is disabled as part of OPAL_SYNC_HOST_REBOOT.
- opal/fast-reboot: set fw_progress sensor status with IPMI_FW_PCI_INIT.

CAPI

- hmi: Print CAPP FIR information when handling CAPP malfunction alerts

FSP based systems

- hw/fsp: Do not queue SP and SPCN class messages during reset/reload This could cause soft lockups if FSP reset reload was done while in OPAL During FSP R/R, the FSP is inaccessible and will lose state. Messages to the FSP are generally queued for sending later.

Tests

- **core/test/run-trace: Reduce number of samples when running under valgrind** This reduces 'make check' run time by ~10 seconds on my laptop, and just the run-trace test itself takes 15 seconds less (under valgrind).
- test/sreset_world: Kind of like Hello World, but from the SRESET vector. A regression test for the mambo implementation of OPAL_SIGNAL_SYSTEM_RESET.

- **nvram-format: Fix endian issues** NVRAM formats are always BE, so let's use the sparse annotation to catch any issues (and correct said issues).

  On LE platforms, the test was erroneously passing as with building the nvram-format code on LE we were produces an incorrect NVRAM image.

- test/hello_world: use P9MAMBO to differentiate from P8
- hdata_to_dt: Specify PVR on command line
- hdata/test: Add DTS output for the test cases
- hdata/test: strip blobs from the DT output
- mambo: add mprintf()

  mprintf() is printf(), but it goes straight to the mambo console. This allows it to be independent of Skiboot's actual console infrastructure so it can be used for debugging the console drivers and for debugging code that runs before the console is setup.

- generate-fwts-olog: add support for parsing prerror()
- **Add bitmap test** The worst test suite ever
- mambo_utils: add ascii output to hexdump
- mambo_utils: add p_str <addr> [limit]
- mambo_utils: make p return a value
- hello_world: print out full path of missing MAMBO_BINARY
- print-stb-container: Fix build on centos7
- Travis-ci improvements: - install expect on ubuntu 12.04, disable qemu on 16.04/latest - build and test more on centos7 - hello_world: run p9 mambo tests - install systemsim-p8 on centos7 - install systemsim-p8 on centos6 - install systemsim-p9 - enable fedora25 - always pull new docker image - add fedora rawhide
- Add fwts annotation for duplicate DT node entries.

  Reference bug: https://github.com/open-power/op-build/issues/751

- external/fwts: Add 'last-tag' to FWTS olog output This isn't so useful at the moment, but this will make cleaning out crufty old error definitions much easier.
- external/fwts: Add FWTS olog merge script A script to merge olog error definitions from multiple skiboot versions into a single olog JSON file. Will prompt when conflicting patterns are found to update the pattern, or add both.
- mambo: fake NVRAM support
- mambo: Add Fake NVRAM driver
- external/mambo: add shortcut to print all GPRs

Contributors

Processed 363 csets from 28 developers. A total of 18105 lines added, 16499 removed (delta 1606)

Developers with the most changesets

Developer	#	%
Benjamin Herrenschmidt	138	(38.0%)
Stewart Smith	56	(15.4%)
Oliver O'Halloran	47	(12.9%)
Michael Neuling	18	(5.0%)
Gavin Shan	15	(4.1%)
Claudio Carvalho	14	(3.9%)
Vasant Hegde	11	(3.0%)
Cyril Bur	11	(3.0%)
Andrew Donnellan	11	(3.0%)
Ananth N Mavinakayanahalli	5	(1.4%)
Cédric Le Goater	5	(1.4%)
Pridhiviraj Paidipeddi	5	(1.4%)
Shilpasri G Bhat	4	(1.1%)
Nicholas Piggin	4	(1.1%)
Russell Currey	3	(0.8%)
Alistair Popple	2	(0.6%)
Jack Miller	2	(0.6%)
Chris Smart	2	(0.6%)
Matt Brown	1	(0.3%)
Michael Ellerman	1	(0.3%)
Frederic Barrat	1	(0.3%)
Hank Chang	1	(0.3%)
Willie Liauw	1	(0.3%)
Werner Fischer	1	(0.3%)
Jeremy Kerr	1	(0.3%)
Patrick Williams	1	(0.3%)
Joel Stanley	1	(0.3%)
Alexey Kardashevskiy	1	(0.3%)

Developers with the most changed lines

Developer	#	%
Oliver O'Halloran	17961	(56.7%)
Benjamin Herrenschmidt	5509	(17.4%)
Cyril Bur	2801	(8.8%)
Stewart Smith	1649	(5.2%)
Gavin Shan	653	(2.1%)
Claudio Carvalho	489	(1.5%)
Willie Liauw	361	(1.1%)
Ananth N Mavinakayanahalli	340	(1.1%)
Andrew Donnellan	315	(1.0%)
Michael Neuling	240	(0.8%)
Shilpasri G Bhat	228	(0.7%)
Nicholas Piggin	219	(0.7%)
Vasant Hegde	207	(0.7%)
Russell Currey	158	(0.5%)
Jack Miller	127	(0.4%)
Cédric Le Goater	126	(0.4%)
Chris Smart	95	(0.3%)
Hank Chang	56	(0.2%)
Pridhiviraj Paidipeddi	47	(0.1%)
Alistair Popple	39	(0.1%)
Matt Brown	29	(0.1%)
Michael Ellerman	3	(0.0%)
Alexey Kardashevskiy	2	(0.0%)
Frederic Barrat	1	(0.0%)
Werner Fischer	1	(0.0%)
Jeremy Kerr	1	(0.0%)
Patrick Williams	1	(0.0%)
Joel Stanley	1	(0.0%)

Developers with the most lines removed

Developer	#	%
Oliver O'Halloran	8810	(53.4%)
Ananth N Mavinakayanahalli	98	(0.6%)
Alistair Popple	9	(0.1%)
Michael Ellerman	3	(0.0%)
Werner Fischer	1	(0.0%)

Developers with the most signoffs

Total 322

Developer	#	%
Stewart Smith	307	(95.3%)
Michael Neuling	6	(1.9%)
Oliver O'Halloran	3	(0.9%)
Benjamin Herrenschmidt	2	(0.6%)
Vaidyanathan Srinivasan	1	(0.3%)
Hank Chang	1	(0.3%)
Jack Miller	1	(0.3%)
Gavin Shan	1	(0.3%)

Developers with the most reviews

Total: 45

Developer	#	%
Vasant Hegde	10	(22.2%)
Andrew Donnellan	9	(20.0%)
Russell Currey	6	(13.3%)
Cédric Le Goater	5	(11.1%)
Oliver O'Halloran	4	(8.9%)
Gavin Shan	3	(6.7%)
Vaidyanathan Srinivasan	2	(4.4%)
Alistair Popple	2	(4.4%)
Frederic Barrat	2	(4.4%)
Mahesh Salgaonkar	1	(2.2%)
Cyril Bur	1	(2.2%)

Developers with the most test credits

Total 11

Developer	#	%
Willie Liauw	4	(36.4%)
Claudio Carvalho	3	(27.3%)
Gavin Shan	1	(9.1%)
Michael Neuling	1	(9.1%)
Pridhiviraj Paidipeddi	1	(9.1%)
Chris Smart	1	(9.1%)

Developers who gave the most tested-by credits

Total 11

Developer	#	%
Gavin Shan	4	(36.4%)
Stewart Smith	4	(36.4%)
Chris Smart	1	(9.1%)
Oliver O'Halloran	1	(9.1%)
Ananth N Mavinakayanahalli	1	(9.1%)

Developers with the most report credits

Total 7

Developer	#	%
Hank Chang	4	(57.1%)
Guilherme G. Piccoli	1	(14.3%)
Colin Ian King	1	(14.3%)
Pradipta Ghosh	1	(14.3%)

Developers who gave the most report credits

Total 7

Developer	#	%
Gavin Shan	5	(71.4%)
Andrew Donnellan	1	(14.3%)
Jeremy Kerr	1	(14.3%)

4.1.58 skiboot-5.5.0-rc2

skiboot-5.5.0-rc2 was released on Monday April 3rd 2017. It is the second release candidate of skiboot 5.5, which will become the new stable release of skiboot following the 5.4 release, first released November 11th 2016.

skiboot-5.5.0-rc2 contains all bug fixes as of *skiboot-5.4.3* and *skiboot-5.1.19* (the currently maintained stable releases).

For how the skiboot stable releases work, see *Skiboot stable tree rules and releases* for details.

The current plan is to cut the final 5.5.0 by April 8th, with skiboot 5.5.0 being for all POWER8 and POWER9 platforms in op-build v1.16 (Due April 12th). This is a short cycle as this release is mainly targetted towards POWER9 bringup efforts.

Following skiboot-5.5.0, we will move to a regular six week release cycle, similar to op-build, but slightly offset to allow for a short stabilisation period. Expected release dates and contents are tracked using GitHub milestone and issues: https://github.com/open-power/skiboot/milestones

Over *skiboot-5.5.0-rc1*, we have the following changes:

NVLINK2

- Introduce NPU2 support

NVLink2 is a new feature introduced on POWER9 systems. It is an evolution of of the NVLink1 feature included in POWER8+ systems but adds several new features including support for GPU address translation using the Nest MMU and cache coherence.

Similar to NVLink1 the functionality is exposed to the OS as a series of virtual PCIe devices. However the actual hardware interfaces are significantly different which limits the amount of common code that can be shared between implementations in the firmware.

This patch adds basic hardware initialisation and exposure of the virtual NVLink2 PCIe devices to the running OS.

- npu2: Add OPAL calls for nvlink2 address translation services (see *OPAL NPU2 calls*)

  Adds three OPAL calls for interacting with NPU2 devices: *OPAL_NPU_INIT_CONTEXT*, *OPAL_NPU_DESTROY_CONTEXT* and *OPAL_NPU_MAP_LPAR*.

  These are used to setup and configure address translation services (ATS) for a process/partition on a given NVLink2 device.

POWER9

- hdata/memory: ignore homer and occ reserved ranges

  We populate these from the HOMER BARs in the PBA directly. There's no need to take the hostboot supplied values so just ignore the corresponding reserved ranges.

- hdata/vpd: Parse the OpenPOWER OPFR record

  Parse the OpenPOWER FRU VPD (OPFR) record on OpenPOWER instead of the VINI records.

- hdata/vpd: Parse additional VINI records

  These records provide hardware version details, CCIN extension information, card type details and hardware characteristics of the FRU

- hdata/cpu: account for p9 shared caches

  On P9 the L2 and L3 caches are shared between pairs of SMT=4 cores. Currently this is not accounted for when creating caches nodes in the device tree. This patch adds additional checking so that a cache node is only created for the first core in the pair and the second core will reference the cache correctly.

- hdata: print backtraces on HDAT errors

- hdat: ignore zero length reserves

  Hostboot can export reserved regions with a length of zero and these should be ignored rather than being turned into reserved range. While we're here fix a memory leak by moving the "too large" region check to before we allocate space for the label.

- SLW: Add init for power9 power management

  This patch adds new function to init core for power9 power management. SPECIAL_WKUP_* SCOM registers, if set, can hold the cores from going into idle states. Hence, clear PPM_SPECIAL_WKUP_HYP_REG scom register for each core during init. (This init are not required for MAMBO)

PCI

- hw/phb3: Adjust ECRC on root port dynamically

  The Samsung NVMe adapter is lost when it's connected to PMC 8546 PCIe switch, until ECRC is disabled on the root port. We found similar issue prevously when Broadcom adapter is connected to same part of PCIe

switch and it was fixed by commit 60ce59ccd0e9 ("hw/phb3: Disable ECRC on Broadcom adapter behind PMC switch"). Unfortunately, the commit doesn't fix the Samsung NVMe adapter lost issue.

This fixes the issues by disable ECRC generation/check on root port when PMC 8546 PCIe switch ports are found. This can be extended for other PCIe switches or endpoints in future: Each PHB maintains the count of PCI devices (PMC 8546 PCIe switch ports currently) which require to disable ECRC on root port. The ECRC functionality is enabled when first PMC 8546 switch port is probed and disabled when last PMC 8546 switch port is destroyed (in PCI hot remove scenario). Except PHB's reinitialization after complete reset, the ECRC on root port is untouched.

- core/pci: Fix lost NVMe adapter behind PMC 8546 switch

  The NVMe adapter in below PCI topology is lost. The root cause is the presence bit on its PCI slot is missed, but the PCIe link has been up. The PCI core doesn't probe the adapter behind the slot, leading to lost NVMe adapter in the particular case.

  - PHB3 root port
  - PLX switch 8748 (10b5:8748)
  - PLX swich 9733 (10b5:9733)
  - PMC 8546 swtich (11f8:8546)
  - NVMe adapter (1c58:0023)

  This fixes the issue by overriding the PCI slot presence bit with PCIe link state bit.

- hw/phb4: Locate AER capability position if necessary
- core/pci: Disable surprise hotplug on root port
- core/pci: Ignore PCI slot capability on root port

  We are creating PCI slot on root port, where the PCI slot isn't supported from hardware. For this case, we shouldn't read the PCI slot capability from hardware. When bogus data returned from the hardware, we will attempt to the PCI slot's power state or enable surprise hotplug functionality. All of them can't be accomplished without hardware support.

  This leaves the PCI slot's capability list 0 if PCICAP_EXP_CAP_SLOT isn't set in hardware (pcie_cap + 0x2). Otherwise, the PCI slot's capability list is retrieved from hardware (pcie_cap + 0x14).

- phb4: Default to PCIe GEN2 on DD1

  Default to PCIe GEN2 link speeds on DD1 for stability.

  Can be overridden using nvram pcie-max-link-speed=4 parameter.

- phb3/4: Set max link speed via nvram

  This adds an nvram parameter pcie-max-link-speed to configure the max speed of the pcie link. This can be set from the petitboot prompt using:

```
nvram -p ibm,skiboot --update-config pcie-max-link-speed=4
```

  This takes preference over anything set in the device tree and is global to all PHBs.

Tests

- Mambo/Qemu boot tests: expect (and fail) on checkstop

  This allows us to fail a lot faster if we checkstop

4.1.59 skiboot-5.5.0-rc3

skiboot-5.5.0-rc3 was released on Wednesday April 5th 2017. It is the third release candidate of skiboot 5.5, which will become the new stable release of skiboot following the 5.4 release, first released November 11th 2016.

skiboot-5.5.0-rc3 contains all bug fixes as of *skiboot-5.4.3* and *skiboot-5.1.19* (the currently maintained stable releases).

For how the skiboot stable releases work, see *Skiboot stable tree rules and releases* for details.

The current plan is to cut the final 5.5.0 by April 8th, with skiboot 5.5.0 being for all POWER8 and POWER9 platforms in op-build v1.16 (Due April 12th). This is a short cycle as this release is mainly targetted towards POWER9 bringup efforts.

Following skiboot-5.5.0, we will move to a regular six week release cycle, similar to op-build, but slightly offset to allow for a short stabilisation period. Expected release dates and contents are tracked using GitHub milestone and issues: https://github.com/open-power/skiboot/milestones

Over *skiboot-5.5.0-rc2*, we have the following changes:

- xive: Fix setting of remote NVT VSD

  This fixes a checkstop when using my XIVE exploitation mode on some multi-chip machines.

- core/init: Use '_' as separator in names of "exports" properties

  The names of the properties under /ibm,opal/firmware/exports are used directly by Linux to create files in sysfs. To remain consistent with the existing naming of OPAL sysfs files, use '_' as the separator.

  In particular for the symbol map which is already exported separately, it's cleaner for the two files to have the same name, eg:

  ```
  /sys/firmware/opal/exports/symbol_map
  /sys/firmware/opal/symbol_map
  ```

- hdata: fix reservation size

  The hostboot reserved ranges are [start, end] pairs rather than [start, end) so we need to stick a +1 in there to calculate the size properly.

- hdat: Add model-name property for OpenPower system
- hdat: Read description from ibm, vpd binary blob
- hdat: Populate model property with 'Unknown' in error path

4.1.60 skiboot-5.6.0

skiboot-5.6.0 was released on Wednesday 24th May 2017. It is the new stable release of skiboot, taking over from the 5.5 release, first released on April 7th 2017. It is the first release done in a regular six week release cycle, mirroring that of op-build.

skiboot-5.6.0 contains all bug fixes as of *skiboot-5.4.4* and *skiboot-5.1.19* (the currently maintained stable releases). We do not currently expect to do any 5.5.x stable releases.

For how the skiboot stable releases work, see *Skiboot stable tree rules and releases* for details.

This release is a good level set of POWER9 support for bringup activities. If you are doing bringup, it is strongly suggested you continue to follow skiboot master.

Changes in skiboot-5.6.0

See changes in the release candidates:

- *skiboot-5.6.0-rc1*
- *skiboot-5.6.0-rc2*

The final 5.6.0 release has no functional changes over the 5.6.0-rc2.

4.1.61 skiboot-5.6.0-rc1

skiboot-5.6.0-rc1 was released on Tuesday May 16th 2017. It is the first release candidate of skiboot 5.6, which will become the new stable release of skiboot following the 5.5 release, first released April 7th 2017.

skiboot-5.6.0-rc1 contains all bug fixes as of *skiboot-5.4.4* and *skiboot-5.1.19* (the currently maintained stable releases). We do not currently expect to do any 5.5.x stable releases.

For how the skiboot stable releases work, see *Skiboot stable tree rules and releases* for details.

The current plan is to cut the final 5.6.0 by May 22nd, with skiboot 5.6.0 being for all POWER8 and POWER9 platforms in op-build v1.17 (Due May 24th). This is a short cycle as this release is mainly targetted towards POWER9 bringup efforts.

This is the first release using the new regular six week release cycle, similar to op-build, but slightly offset to allow for a short stabilisation period. Expected release dates and contents are tracked using GitHub milestone and issues: https://github.com/open-power/skiboot/milestones

Over skiboot-5.5, we have the following changes:

New Platforms

Thanks to SuperMicro for submitting support for the p9dsu platform, AKA Boston.

POWER9

XIVE:

- xive: Clear emulation mode queue on reset
- xive: Fixes/improvements to xive reset for multi-chip systems
- xive: Synchronize after disable IRQs in opal_xive_reset()
- xive: Workaround a problem with indirect TM access
- hdata: Make FSPv1 work again One less thing to work around for those crazy enough to try.
- xive: Log more information in opal_xive_dump() for emulation state

  Add a counter of total interrupts taken by a CPU, dump the queue buffer both before and after the current pointer, and also display the HW state of the queue descriptor and the PQ state of the IPI.
- xive: Add a per-cpu logging mechanism to XICS emulation

  This is a small 32-entries rolling buffer that logs a few operations. It's useful to debug odd problems. The output is printed when opal_xive_dump() is called.

- xive: Check queues for duplicates in DEBUG builds.

  There should never be duplicate interrupts in a queue. This adds code to check that when looking at the queue content. Since it can be a performance loss, this is only done for debug builds.

- xive+phb4: Fix exposing trigger page to Linux

HDAT Parsing:

- hdata/spira.c: Add device-tree bindings for nest mmu
- hdata/i2c: Workaround broken i2c devices
- hdata: indicate when booted with elevated risk level

  When the system is IPLed with an elevated risk level Hostboot will set a flag in the IPL parameters structure. Parse and export this in the device tree at: /ipl-params/sys-params/elevated-risk-level

- hdata: Respect OCC and HOMER resevations

  In the past we've ignored these since Hostboot insisted in exporting broken reservations and the OCC was not being used yet. This situation seems to have resolved itself so we should respect the reservations that hostboot provides.

I2C:

- i2c: Add interrupts support on P9

  Some older revisions of hostboot populate the host i2c device fields with all zero entires. Detect and ignore these so we don't crash on boot.

  Without this we get:

```
[  151.251240444,3] DT: dt_attach_root failed, duplicate unknown@0
[  151.251300274,3] ********************************************
[  151.251339330,3] Unexpected exception 200 !
[  151.251363654,3] SRR0 : 0000000030090c28 SRR1 : 9000000000201000
[  151.251409207,3] HSRR0: 0000000000000010 HSRR1: 9000000000001000
[  151.251444114,3] LR   : 30034018300c5ab0 CTR  : 30034018300a343c
[  151.251478314,3] CFAR : 0000000030024804
[  151.251500346,3] CR   : 40004208  XER: 00000000
    <snip GPRS>
[  151.252083372,0] Aborting!
CPU 0034 Backtrace:
 S: 0000000031cd36a0 R: 000000003001364c   .backtrace+0x2c
 S: 0000000031cd3730 R: 0000000030018db8   ._abort+0x4c
 S: 0000000031cd37b0 R: 0000000030025c6c   .exception_entry+0x114
 S: 0000000031cd3840 R: 0000000000001f00 * +0x1f00
 S: 0000000031cd3a10 R: 0000000031cd3ab0 *
 S: 0000000031cd3aa0 R: 00000000300248b8   .new_property+0x90
 S: 0000000031cd3b30 R: 0000000030024b50   .__dt_add_property_cells+0x30
 S: 0000000031cd3bd0 R: 000000003009abec   .parse_i2c_devs+0x350
 S: 0000000031cd3cf0 R: 0000000030093ffc   .parse_hdat+0x11e4
 S: 0000000031cd3e30 R: 00000000300144c8   .main_cpu_entry+0x138
 S: 0000000031cd3f00 R: 0000000030002648   boot_entry+0x198
```

PHB4:

- phb4: Enforce root complex config space size of 2048

  The root complex config space size on PHB4 is 2048. This patch sets that size and enforces it when trying to read/write the config space in the root complex.

  Without this someone reading the config space via /sysfs in linux will cause an EEH on the PHB.

If too high, reads returns 1s and writes are silently dropped.

- phb4: Add an option for disabling EEH MMIO in nvram

  Having the option to disable EEH for MMIO without rebuilding skiboot could be useful for testing, so check for pci-eeh-mmio=disabled in nvram.

  This is not designed to be a supported option or configuration, just an option that's useful in bringup and development of POWER9 systems.

- phb4: Fix slot presence detect

  This has the nice side effect of improving boot times since we no longer waste time tring to train links that don't have anything present.

- phb4: Enable EEH for MMIO
- phb4: Implement fence check
- phb4: Implement diag data

OCC:

- occ/irq: Fix SCOM address and irq reasons for P9 OCC

  This patch fixes the SCOM address for OCC_MISC register which is used for OCC interupts. In P9, OCC sends an interrupt to notify change in the shared memory like throttle status. This patch handles this interrupt reason.

PRD:

- prd: Fix PRD scoms for P9

NX/DARN:

- nx: Add POWER9 DARN support

NPU2:

- npu2: Do not attempt to initialise non DD1 hardware

  There are significant changes to hardware register addresses and meanings on newer chip revisions making them unlikely to work correctly with the existing code. Better to fail clearly and early.

- npu, npu2: Describe diag data size in device tree

Memory Reservation:

- mem_region: Add reserved regions after memory init

  When a new memory region is added (e.g for memory reserved by firmware) the list of existing memory regions is iterated through and a cut-out is made in any existing region that overlaps with the new one. Prior to the HDAT reservations being made the region init process was always:

  1. Create regions from the memory@<addr> DT nodes. (mostly large)
  2. Create reserved regions from the device-tree. (mostly small)

  When adding new regions we have assumed that the new region will only every intersect with at most one existing region, which it will split. Adding reservations inside the HDAT parser breaks this because when adding the memory@<addr> node regions we can potentially overlap with multiple reserved regions. This patch fixes this by maintaining a seperate list of memory reservations and delaying merging them until after the normal memory init has finished, similar to how DT reservations are handled.

PCI

- pci: Describe PHB diag data size in device tree

  Linux hardcodes the PHB diag data buffer at (as of this commit) 8192 bytes. This has been enough for P7IOC and PHB3, but the 512 PEs of PHB4 pushes the diag data blob over this size. Rather than just increasing the hardcoded size in Linux, provide the size of the diag data blob in the device tree so that the OS can dynamically allocate as much as it needs. This both enables more space for PHB4 and less wasted memory for P7IOC and PHB3.

  P7IOC communicates both hub and PHB data using this buffer, so when setting the size, use whichever struct is largest.

- hdata/i2c: Fix bus and clock frequencies
- ibm-fsp: use opal-prd on p9 and above

  Previously the PRD tooling ran on the FSP, but it was moved into userspace on the host for OpenPower systems. For P9 this system was adopted for FSP systems too.

I2C

- i2c: Remove old hack for bad clock frequency

  This hack dates back to ancient P8 hostboots. The value it would use if it detected the "bad" value was incorrect anyway.

- i2c: Log the engine clock frequency at boot

FSP Systems

These include the Apollo, Firenze and ZZ platforms.

- Remove multiple logging for un-handled fsp sub commands.

  If any new or unknown command need to be handled, just log un-hnadled message from only fsp, not required from fsp-dpo.

```
cat /sys/firmware/opal/msglog | grep -i ,3
[  110.232114723,3] FSP: fsp_trigger_reset() entry
[  188.431793837,3] FSP #0: Link down, starting R&R
[  464.109239162,3] FSP #0: Got XUP with no pending message !
[  466.340598554,3] FSP-DPO: Unknown command 0xce0900
[  466.340600126,3] FSP: Unhandled message ce0900
```

- FSP: Notify FSP of Platform Log ID after Host Initiated Reset Reload

  Trigging a Host Initiated Reset (when the host detects the FSP has gone out to lunch and should be rebooted), would cause "Unknown Command" messages to appear in the OPAL log.

  This patch implements those messages

  How to trigger FSP RR(HIR):

```
$ putmemproc 300000f8 0x00000000deadbeef
s1      k0:n0:s0:p00
ecmd_ppc putmemproc 300000f8 0x00000000deadbeef

Log showing unknown command:
/ # cat /sys/firmware/opal/msglog | grep -i ,3
```

```
[  110.232114723,3] FSP: fsp_trigger_reset() entry
[  188.431793837,3] FSP #0: Link down, starting R&R
[  464.109239162,3] FSP #0: Got XUP with no pending message !
[  466.340598554,3] FSP-DPO: Unknown command 0xce0900
[  466.340600126,3] FSP: Unhandled message ce0900
```

The message we need to handle is "Get PLID after host initiated FipS reset/reload". When the FSP comes back from HIR, it asks "hey, so, which error log explains why you rebooted me?". So, we tell it.

Misc

- hdata_to_dt: Misc improvements in the utility and unit test
- GCC7: fixes for -Wimplicit-fallthrough expected regexes

  It turns out GCC7 adds a useful warning and does fancy things like parsing your comments to work out that you intended to do the fallthrough. There's a few places where we don't match the regex. Fix them, as it's harmless to do so.

  Found by building on Fedora Rawhide in Travis.

  While we do not have everything needed to start building successfully with GCC7 (well, at least doing so warning clean), it's a start.
- hdata/i2c: avoid possible int32_t overflow

  We're safe up until engine number 524288. Found by static analysis (of course)
- tpm_i2c_nuvoton: fix use-after-free in tpm_register_chip failure path
- mambo: Fix reserved-ranges node
- external/mambo: add helper for machine checks
- console: Set log level from nvram

  This adds two new nvram options to set the console log level for the driver/uart and in memory. These are called log-level-memory and log-level-driver.

  These are only set once we have nvram inited.

  To set them you do:

```
nvram -p ibm,skiboot --update-config log-level-memory=9
nvram -p ibm,skiboot --update-config log-level-driver=9
```

  You can also use the named versions of emerg, alert, crit, err, warning, notice, printf, info, debug, trace or insane. ie.

```
nvram -p ibm,skiboot --update-config log-level-driver=insane
```

- npu: Implement Function Level Reset (FLR)
- mbox: Sanitize interrupts registers
- xive: Fix potential for lost IPIs when manipulating CPPR
- xive: Don't double EOI interrupts that have an EOI override
- libflash/file: Only use 64bit MTD erase ioctl() when needed

  We recently made MTD 64 bit safe in e5720d3fe94 which now requires the 64 bit MTD erase ioctl. Unfortunately this ioctl is not present in older kernels used by some BMC vendors that use pflash.

This patch addresses this by only using the 64bit version of the erase ioctl() if the parameters exceed 32bit in size.

If an erase requires the 64bit ioctl() on a kernel which does not support it, the code will still attempt it. There is no way of knowing beforehand if the kernel supports it. The ioctl() will fail and an error will be returned from from the function.

Contributors

This release contains 81 csets from 15 developers, working at 2 employers. A total of 2496 lines added, 641 removed (delta 1855)

Developers with the most changesets

Developer	#	%
Oliver O'Halloran	17	(21.0%)
Benjamin Herrenschmidt	17	(21.0%)
Michael Neuling	16	(19.8%)
Stewart Smith	9	(11.1%)
Russell Currey	8	(9.9%)
Alistair Popple	5	(6.2%)
ppaidipe@linux.vnet.ibm.com	1	(1.2%)
Dave Heller	1	(1.2%)
Jeff Scheel	1	(1.2%)
Nicholas Piggin	1	(1.2%)
Ananth N Mavinakayanahalli	1	(1.2%)
Cyril Bur	1	(1.2%)
Alexey Kardashevskiy	1	(1.2%)
Jim Yuan	1	(1.2%)
Shilpasri G Bhat	1	(1.2%)

Developers with the most changed lines

Developer	#	%
Michael Neuling	748	(28.4%)
Benjamin Herrenschmidt	405	(15.4%)
Russell Currey	360	(13.7%)
Oliver O'Halloran	297	(11.3%)
Nicholas Piggin	187	(7.1%)
Alistair Popple	183	(7.0%)
Stewart Smith	175	(6.6%)
Shilpasri G Bhat	79	(3.0%)
Jim Yuan	56	(2.1%)
Ananth N Mavinakayanahalli	45	(1.7%)
Cyril Bur	38	(1.4%)
Alexey Kardashevskiy	37	(1.4%)
Jeff Scheel	19	(0.7%)
Dave Heller	2	(0.1%)
Pridhiviraj Paidipeddi	1	(0.0%)

Developers with the most lines removed

Developer	#	%
Pridhiviraj Paidipeddi	1	(0.2%)

Developers with the most signoffs

Total of 73.

Developer	#	%
Stewart Smith	56	(76.7%)
Michael Neuling	16	(21.9%)
Oliver O'Halloran	1	(1.4%)

Developers with the most reviews

Total of 6.

Developer	#	%
Oliver O'Halloran	3	(50.0%)
Andrew Donnellan	1	(16.7%)
Gavin Shan	1	(16.7%)
Cyril Bur	1	(16.7%)

Developers with the most test credits

Total of 5.

Developer	#	%
Oliver O'Halloran	2	(40.0%)
Vaidyanathan Srinivasan	1	(20.0%)
Vasant Hegde	1	(20.0%)
Michael Ellerman	1	(20.0%)

Developers who gave the most tested-by credits

Total of 5.

Developer	#	%
Oliver O'Halloran	2	(40.0%)
Benjamin Herrenschmidt	2	(40.0%)
Nicholas Piggin	1	(20.0%)

Developers with the most report credits

Total of 2.

Developer	#	%
Benjamin Herrenschmidt	1	(50.0%)
Pridhiviraj Paidipeddi	1	(50.0%)

Developers who gave the most report credits

Total of 2.

Developer	#	%
Stewart Smith	2	(100.0%)

Top changeset contributors by employer

Total of 2.

Employer	#	%
IBM	80	(98.8%)
SuperMicro	1	(1.2%)

Top lines changed by employer

Employer	#	%
IBM	2576	(97.9%)
SuperMicro	56	(2.1%)

Employers with the most signoffs

Total 73.

Employer	#	%
IBM	73	(100.0%)

Employers with the most hackers

Total 15.

Employer	#	%
IBM	14	(93.3%)
SuperMicro	1	(6.7%)

4.1.62 skiboot-5.6.0-rc2

skiboot-5.6.0-rc2 was released on Friday May 19th 2017. It is the second release candidate of skiboot 5.6, which will become the new stable release of skiboot following the 5.5 release, first released April 7th 2017.

skiboot-5.6.0-rc2 contains all bug fixes as of *skiboot-5.4.4* and *skiboot-5.1.19* (the currently maintained stable releases). We do not currently expect to do any 5.5.x stable releases.

For how the skiboot stable releases work, see *Skiboot stable tree rules and releases* for details.

The current plan is to cut the final 5.6.0 by May 22nd, with skiboot 5.6.0 being for all POWER8 and POWER9 platforms in op-build v1.17 (Due May 24th). This is a short cycle as this release is mainly targetted towards POWER9 bringup efforts.

With skiboot 5.6.0, we are moving to a regular six week release cycle, similar to op-build, but slightly offset to allow for a short stabilisation period. Expected release dates and contents are tracked using GitHub milestone and issues: https://github.com/open-power/skiboot/milestones

Over *skiboot-5.6.0-rc1*, we have the following changes:

- hw/i2c: Fix early lock drop

  When interacting with an I2C master the p8-i2c driver (common to p9) aquires a per-master lock which it holds for the duration of it's interaction with the master. Unfortunately, when p8_i2c_check_initial_status() detects that the master is busy with another transaction it drops the lock and returns OPAL_BUSY. This is contrary to the driver's locking strategy which requires that the caller aquire and drop the lock. This leads to a crash due to the double unlock(), which skiboot treats as fatal.

- mambo: Add skiboot/linux symbol lookup

  Adds the skisym and linsym commands which can be used to find the address of a Linux or Skiboot symbol. To function this requires the user to provide the SKIBOOT_MAP and VMLINUX_MAP environmental variables which indicate which skiboot.map and System.map files should be used.

  Examples:

  – Look up a symbol address:

  ```
  systemsim % skisym .load_and_boot_kernel
  0x0000000030013a08
  ```

  – Set a breakpoint there:

  ```
  systemsim % b [skisym .load_and_boot_kernel]
  breakpoint set at [0:0]: 0x0000000030013a08 (0x0000000030013A08)␣
  ↪Enc:0x7D800026 : mfcr    r12
  ```

- libstb: Fix build in OpenSSL 1.1

  The build failure was as follows:

  ```
  [ HOSTCC ] libstb/create-container.c
  In file included from /usr/include/openssl/asn1.h:24:0,
                   from /usr/include/openssl/ec.h:30,
                   from libstb/create-container.c:36:
  libstb/create-container.c: In function 'getSigRaw':
  libstb/create-container.c:104:31: error: dereferencing pointer to incomplete
                                    type 'ECDSA_SIG {aka struct ECDSA_SIG_st}'
    rlen = BN_num_bytes(signature->r);
                                ^
  ```

4.1.63 skiboot-5.7

skiboot v5.7 was released on Tuesday July 25th 2017. It follows two release candidates of skiboot 5.7, and is now the new stable release of skiboot following the 5.6 release, first released 24th May 2017.

skiboot v5.7 contains all bug fixes as of *skiboot-5.4.6* and *skiboot-5.1.19* (the currently maintained stable releases). We do not currently expect to do any 5.6.x stable releases.

For how the skiboot stable releases work, see *Skiboot stable tree rules and releases* for details.

POWER9 is still in development, and thus all POWER9 users must upgrade to skiboot v5.7.

This is the second release using the new regular six week release cycle, similar to op-build, but slightly offset to allow for a short stabilisation period. Expected release dates and contents are tracked using GitHub milestone and issues: https://github.com/open-power/skiboot/milestones

New Features

Since *skiboot-5.6.0*, we have a few new features:

New features in this release for POWER9 systems:

- In Memory Counters (IMC) (See *OPAL/Skiboot In-Memory Collection (IMC) interface Documentation* for details)
- phb4: Activate shared PCI slot on witherspoon (see *Shared Slot*)
- phb4 capi (i.e. CAPI2): Enable capi mode for PHB4 (see *CAPI on PHB4*)

New feature for IBM FSP based systems:

- fsp/tpo: Provide support for disabling TPO alarm

  This patch adds support for disabling a preconfigured Timed-Power-On(TPO) alarm on FSP based systems. Presently once a TPO alarm is configured from the kernel it will be triggered even if its subsequently disabled.

  With this patch a TPO alarm can be disabled by passing y_m_d==hr_min==0 to fsp_opal_tpo_write(). A branch is added to the function to handle this case by sending FSP_CMD_TPO_DISABLE message to the FSP instead of usual FSP_CMD_TPO_WRITE message. The kernel is expected to call opal_tpo_write() with y_m_d==hr_min==0 to request opal to disable TPO alarm.

POWER9

There are many important changes for POWER9 DD1 and DD2 systems. POWER9 support should be considered in development and skiboot 5.7 is certainly **NOT** suitable for POWER9 production environments.

Since *skiboot-5.7-rc2*:

- platform/witherspoon: Enable eSEL logging

  OpenBMC stack added IPMI OEM extension to log eSEL events. Lets enable eSEL logging from OPAL side.

  See: https://github.com/openbmc/openpower-host-ipmi-oem/blob/d9296050bcece5c2eca5ede0932d944b0ced66c9/oemhandler.cpp#L142 (yes, that is the documentation)

- hdat/i2c: Fix array version check
- mem_region: Check for no-map in reserved nodes

  Regions with the no-map property should be handled seperately to "normal" firmware reservations. When creating mem_region regions from a reserved-memory DT node use the no-map property to select the right reservation type.

- hdata/memory: Add memory reservations to the DT

  Currently we just add these to a list of pre-boot reserved regions which is then converted into a the contents of the /reserved-memory/ node just before Skiboot jumps into the firmware kernel.

  This approach is insufficent because we need to add the ibm,prd-instance labels to the various hostboot reserved regions. To do this we want to create these resevation nodes inside the HDAT parser rather than having the mem_region flattening code handle it. On P8 systems Hostboot placed its memory reservations under the /ibm,hostboot/ node and this patch makes the HDAT parser do the same.

Since Since *skiboot-5.7-rc1*:

- HDAT: Add IPMI sensor data under /bmc node
- numa/associativity: Add a new level of NUMA for GPU's

  Today we have an issue where the NUMA nodes corresponding to GPU's have the same affinity/distance as normal memory nodes. Our reference-points today supports two levels [0x4, 0x4] for normal systems and [0x4, 0x3] for Power8E systems. This patch adds a new level [0x4, X, 0x2] and uses node-id as at all levels for the GPU.

- xive: Enable memory backing of queues

  This dedicates 6x64k pages of memory permanently for the XIVE to use for internal queue overflow. This allows the XIVE to deal with some corner cases where the internal queues might prove insufficient.

- xive: Properly get rid of donated indirect pages during reset

  Otherwise they keep being used accross kexec causing memory corruption in subsequent kernels once KVM has been used.

- cpu: Better handle unknown flags in opal_reinit_cpus()

  At the moment, if we get passed flags we don't know about, we return OPAL_UNSUPPORTED but we still perform whatever actions was requied by the flags we do support. Additionally, on P8, we attempt a SLW re-init which hasn't been supported since Murano DD2.0 and will crash your system.

  It's too late to fix on existing systems so Linux will have to be careful at least on P8, but to avoid future issues let's clean that up, make sure we only use slw_reinit() when HILE isn't supported.

- cpu: Unconditionally cleanup TLBs on P9 in opal_reinit_cpus()

  This can work around problems where Linux fails to properly cleanup part or all of the TLB on kexec.

- Fix scom addresses for power9 nx checkstop hmi handling.

  Scom addresses for NX status, DMA & ENGINE FIR and PBI FIR has changed for Power9. Fixup thoes while handling nx checkstop for Power9.

- Fix scom addresses for power9 core checkstop hmi handling.

  Scom addresses for CORE FIR (Fault Isolation Register) and Malfunction Alert Register has changed for Power9. Fixup those while handling core checkstop for Power9.

  Without this change HMI handler fails to check for correct reason for core checkstop on Power9.

- core/mem_region: check return value of add_region

  The only sensible thing to do if this fails is to abort() as we've likely just failed reserving reserved memory regions, and nothing good comes from that.

Since Since *skiboot-5.6.0*:

- hdata: Reserve Trace Areas

When hostboot is configured to setup in memory tracing it will reserve some memory for use by the hardware tracing facility. We need to mark these areas as off limits to the operating system and firmware.

- hdata: Make out-of-range idata print at PR_DEBUG

  Some fields just aren't populated on some systems.

- hdata: Ignore unnamed memory reservations.

  Hostboot should name any and all memory reservations that it provides. Currently some hostboots export a broken reservation covering the first 256MB of memory and this causes the system to crash at boot due to an invalid free because this overlaps with the static "ibm,os-reserve" region (which covers the first 768MB of memory).

  According to the hostboot team unnamed reservations are invalid and can be ignored.

- hdata: Check the Host I2C devices array version

  Currently this is not populated on FSP machines which causes some obnoxious errors to appear in the boot log. We also only want to parse version 1 of this structure since future versions will completely change the array item format.

- Ensure P9 DD1 workarounds apply only to Nimbus

  The workarounds for P9 DD1 are only needed for Nimbus. P9 Cumulus will be DD1 but don't need these same workarounds.

  This patch ensures the P9 DD1 workarounds only apply to Nimbus. It also renames some things to make clear what's what.

- cpu: Cleanup AMR and IAMR when re-initializing CPUs

  There's a bug in current Linux kernels leaving crap in those registers accross kexec and not sanitizing them on boot. This breaks kexec under some circumstances (such as booting a hash kernel from a radix one on P9 DD2.0).

  The long term fix is in Linux, but this workaround is a reasonable way of "sanitizing" those SPRs when Linux calls opal_reinit_cpus() and shouldn't have adverse effects.

  We could also use that same mechanism to cleanup other things as well such as restoring some other SPRs to their default value in the future.

- Set POWER9 RPR SPR to 0x00000103070F1F3F. Same value as P8.

  Without this, thread priorities inside a core don't work.

- cpu: Support setting HID[RADIX] and set it by default on P9

  This adds new opal_reinit_cpus() flags to setup radix or hash mode in HID[8] on POWER9.

  By default HID[8] will be set. On P9 DD1.0, Linux will change it as needed. On P9 DD2.0 hash works in radix mode (radix is really "dual" mode) so KVM won't break and existing kernels will work.

  Newer kernels built for hash will call this to clear the HID bit and thus get the full size of the TLB as an optimization.

- Add "cleanup_global_tlb" for P9 and later

  Uses broadcast TLBIE's to cleanup the TLB on all cores and on the nest MMU

- xive: DD2.0 updates

  Add support for StoreEOI, fix StoreEOI MMIO offset in ESB page, and other cleanups

- Update default TSCR value for P9 as recommended by HW folk.

- xive: Fix initialisation of xive_cpu_state struct

  When using XIVE emulation with DEBUG=1, we run into crashes in log_add() due to the xive_cpu_state->log_pos being uninitialised (and thus, with DEBUG enabled, initialised to the poison value of 0x99999999).

PHB4

Since *skiboot-5.7-rc2*:

- phb4: Add link training trace mode

  Add a mode to PHB4 to trace training process closely. This activates as soon as PERST is deasserted and produces human readable output of the process.

  This may increase training times since it duplicates some of the training code. This code has it's own simple checks for fence and timeout but will fall through to the default training code once done.

  Output produced, looks like the "TRACE:" lines below:

```
[    3.410799664,7] PHB#0001[0:1]: FRESET: Starts
[    3.410802000,7] PHB#0001[0:1]: FRESET: Prepare for link down
[    3.410806624,7] PHB#0001[0:1]: FRESET: Assert skipped
[    3.410808848,7] PHB#0001[0:1]: FRESET: Deassert
[    3.410812176,3] PHB#0001[0:1]: TRACE: 0x0000000101000000  0ms
[    3.417170176,3] PHB#0001[0:1]: TRACE: 0x0000100101000000 12ms presence
[    3.436289104,3] PHB#0001[0:1]: TRACE: 0x0000180101000000 49ms training
[    3.436373312,3] PHB#0001[0:1]: TRACE: 0x00001d0811000000 49ms trained
[    3.436420752,3] PHB#0001[0:1]: TRACE: Link trained.
[    3.436967856,7] PHB#0001[0:1]: LINK: Start polling
[    3.437482240,7] PHB#0001[0:1]: LINK: Electrical link detected
[    3.437996864,7] PHB#0001[0:1]: LINK: Link is up
[    4.438000048,7] PHB#0001[0:1]: LINK: Link is stable
```

  Enabled via nvram using:

```
nvram -p ibm,skiboot --update-config pci-tracing=true
```

- phb4: Improve reset and link training timing

  This improves PHB reset and link training timing.

- phb4: Add phb4_check_reg() to sanity check failures

  This adds a function phb4_check_reg() to sanity check when we do MMIO reads from the PHB to make sure it's not fenced.

- phb4: Remove retry on electrical link timeout

  Currently we retry if we don't detect an electrical link. This is pointless as all devices should respond in the given time.

  This patches removes this retry and just returns OPAL_HARDWARE if we don't detect an electrical link.

  This has the additional benefit of improving boot times on machines that have badly wired presence detect (ie. says a device is present when there isn't).

- phb4: Read PERST signal rather than assuming it's asserted

  Currently we assume on boot that PERST is asserted so that we can skip having to assert it ourselves.

  This instead reads the PERST status and determines if we need to assert it based on that.

- phb4: Fix endian of TLP headers print

  Byte swap TLP headers so they are the same as the PCIe spec.

- phb4: Change timeouts prints to error level

  If the link doesn't have a electrical link or the link doesn't train we should make that more obvious to the user.

- phb4: Better logs why the slot didn't work

  Better logs why the slot didn't work and make it a PR_ERR so users see it by default.

- phb4: Force verbose EEH logging

  Force verbose EEH. This is a heavy handed and we should turn if off later as things stabilise, but is useful for now.

- phb4: Initialization sequence updates

  Mostly errata workarounds, some DD1 specific.

  The step Init_5 was moved to Init_16, so the numbering was updated to reflect this.

Since *skiboot-5.7-rc1*:

- phb4: Do more retries on link training failures Currently we only retry once when we have a link training failure. This changes this to be 3 retries as 1 retry is not giving us enough reliablity.

  This will increase the boot time, especially on systems where we incorrectly detect a link presence when there really is nothing present. I'll post a followup patch to optimise our timings to help mitigate this later.

- phb4: Workaround phy lockup by doing full PHB reset on retry

  For PHB4 it's possible that the phy may end up in a bad state where it can no longer recieve data. This can manifest as the link not retraining. A simple PERST will not clear this. The PHB must be completely reset.

  This changes the retry state to CRESET to do this.

  This issue may also manifest itself as the link training in a degraded state (lower speed or narrower width). This patch doesn't attempt to fix that (will come later).

- pci: Add ability to trace timing

  PCI link training is responsible for a huge chunk of the skiboot boot time, so add the ability to trace it waiting in the main state machine.

- pci: Print resetting PHB notice at higher log level

  Currently during boot there a long delay while we wait for the PHBs to be reset and train. During this time, there is no output from skiboot and the last message doesn't give an indication of what's happening.

  This boosts the PHB reset message from info to notice so users can see what's happening during this long period of waiting.

- phb4: Only set one bit in nfir

  The MPIPL procedure says to only set bit 26 when forcing the PEC into freeze mode. Currently we set bits 24-27.

  This changes the code to follow spec and only set bit 26.

- phb4: Fix order of pfir/nfir clearing in CRESET

  According to the workbook, pfir must be cleared before the nfir. The way we have it now causes the nfir to not clear properly in some error circumstances.

  This swaps the order to match the workbook.

- phb4: Remove incorrect state transition

  When waiting in PHB4_SLOT_CRESET_WAIT_CQ for transations to end, we incorrectly move onto the next state. Generally we don't hit this as the transactions have ended already anyway.

  This removes the incorrect state transition.

- phb4: Set default lane equalisation

  Set default lane equalisation if there is nothing in the device-tree.

  Default value taken from hdat and confirmed by hardware team. Neatens the code up a bit too.

- hdata: Fix phb4 lane-eq property generation

  The lane-eq data we get from hdat is all 7s but what we end up in the device tree is:

```
xscom@603fc00000000/pbcq@4010c00/stack@0/ibm,lane-eq
                  00000000 31c339e0 00000000 0000000c
                  00000000 00000000 00000000 00000000
                  00000000 31c30000 77777777 77777777
                  77777777 77777777 77777777 77777777
```

  This fixes grabbing the properties from hdat and fixes the call to put them in the device tree.

- phb4: Fix PHB4 fence recovery.

  We had a few problems:

  - We used the wrong register to trigger the reset (spec bug)
  - We should clear the PFIR and NFIR while the reset is asserted
  - ... and in the right order !
  - We should only apply the DD1 workaround after the reset has been lifted.
  - We should ensure we use ASB whenever we are fenced or doing a CRESET
  - Make config ops write with ASB

- phb4: Verbose EEH options

  Enabled via nvram pci-eeh-verbose=true. ie.

```
nvram -p ibm,skiboot --update-config pci-eeh-verbose=true
```

- phb4: Print more info when PHB fences

  For now at PHBERR level. We don't have room in the diags data passed to Linux for these unfortunately.

Since *skiboot-5.6.0*:

- phb4: Fix number of index bits in IODA tables

  On PHB4 the number of index bits in the IODA table address register was bumped to 10 bits to accomodate for 1024 MSIs and 1024 TVEs (DD2).

  However our macro only defined the field to be 9 bits, thus causing "interesting" behaviours on some systems.

- phb4: Harden init with bad PHBs

  Currently if we read all 1's from the EEH or IRQ capabilities, we end up train wrecking on some other random code (eg. an assert() in xive).

  This hardens the PHB4 code to look for these bad reads and more gracefully fails the init for that PHB alone. This allows the rest of the system to boot and ignore those bad PHBs.

- phb4 capi (i.e. CAPI2): Handle HMI events

  Find the CAPP on the chip associated with the HMI event for PHB4. The recovery mode (re-initialization of the capp, resume of functional operations) is only available with P9 DD2. A new patch will be provided to support this feature.

- phb4 capi (i.e. CAPI2): Enable capi mode for PHB4

  Enable the Coherently attached processor interface. The PHB is used as a CAPI interface. CAPI Adapters can be connected to either PEC0 or PEC2. Single port CAPI adapter can be connected to either PEC0 or PEC2, but Dual-Port Adapter can be only connected to PEC2 * CAPP0 attached to PHB0(PEC0 - single port) * CAPP1 attached to PHB3(PEC2 - single or dual port)

- hw/phb4: Rework phb4_get_presence_state()

  There are two issues in current implementation: It should return errcode visibile to Linux, which has prefix OPAL_*. The code isn't very obvious.

  This returns OPAL_HARDWARE when the PHB is broken. Otherwise, OPAL_SUCCESS is always returned. In the mean while, It refactors the code to make it obvious: OPAL_PCI_SLOT_PRESENT is returned when the presence signal (low active) or PCIe link is active. Otherwise, OPAL_PCI_SLOT_EMPTY is returned.

- phb4: Error injection for config space

  Implement CFG (config space) error injection.

  This works the same as PHB3. MMIO and DMA error injection require a rewrite, so they're unsupported for now.

  While it's not feature complete, this at least provides an easy way to inject an error that will trigger EEH.

- phb4: Error clear implementation

- phb4: Mask link down errors during reset

  During a hot reset the PCI link will drop, so we need to mask link down events to prevent unnecessary errors.

- phb4: Implement root port initialization

  phb4_root_port_init() was a NOP before, so fix that.

- phb4: Complete reset implementation

  This implements complete reset (creset) functionality for POWER9 DD1.

  Only partially tested and contends with some DD1 errata, but it's a start.

- phb4: Activate shared PCI slot on witherspoon

  Witherspoon systems come with a 'shared' PCI slot: physically, it looks like a x16 slot, but it's actually two x8 slots connected to two PHBs of two different chips. Taking advantage of it requires some logic on the PCI adapter. Only the Mellanox CX5 adapter is known to support it at the time of this writing.

  This patch enables support for the shared slot on witherspoon if a x16 adapter is detected. Each x8 slot has a presence bit, so both bits need to be set for the activation to take place. Slot sharing is activated through a gpio.

  Note that there's no easy way to be sure that the card is indeed a shared-slot compatible PCI adapter and not a normal x16 card. Plugging a normal x16 adapter on the shared slot should be avoided on witherspoon, as the link won't train on the second slot, resulting in a timeout and a longer boot time. Only the first slot is usable and the x16 adapter will end up using only half the lines.

  If the PCI card plugged on the physical slot is only x8 (or less), then the presence bit of the second slot is not set, so this patch does nothing. The x8 (or less) adapter should work like on any other physical slot.

- phb4: Block D-state power management on direct slots

  As current revisions of PHB4 don't properly handle the resulting L1 link transition.

- phb4: Call pci config filters
- phb4: Mask out write-1-to-clear registers in RC cfg

  The root complex config space only supports 4-byte accesses. Thus, when the client requests a smaller size write, we do a read-modify-write to the register.

  However, some register have bits defined as "write 1 to clear".

  If we do a RMW cycles on such a register and such bits are 1 in the part that the client doesn't intend to modify, we will accidentally write back those 1's and clear the corresponding bit.

  This avoids it by masking out those magic bits from the "old" value read from the register.

- phb4: Properly mask out link down errors during reset
- phb3/4: Silence a useless warning

  PHB's don't have base location codes on non-FSP systems and it's normal.

- phb4: Workaround bug in spec 053

  Wait for DLP PGRESET to clear *after* lifting the PCIe core reset

- phb4: DD2.0 updates

  Support StoreEOI, full complements of PEs (twice as big TVT) and other updates.

  Also renumber init steps to match spec 063

NPU2

Note that currently NPU2 support is limited to POWER9 DD1 hardware.

Since *skiboot-5.6.0*:

- platforms/astbmc/witherspoon.c: Add NPU2 slot mappings

  For NVLink2 to function PCIe devices need to be associated with the right NVLinks. This association is supposed to be passed down to Skiboot via HDAT but those fields are still not correctly filled out. To work around this we add slot tables for the NVLinks similar to what we have for P8+.

- hw/npu2.c: Fix device aperture calculation

  The POWER9 NPU2 implements an address compression scheme to compress 56-bit P9 physical addresses to 47-bit GPU addresses. System software needs to know both addresses, unfortunately the calculation of the compressed address was incorrect. Fix it here.

- hw/npu2.c: Change MCD BAR allocation order

  MCD BARs need to be correctly aligned to the size of the region. As GPU memory is allocated from the top of memory down we should start allocating from the highest GPU memory address to the lowest to ensure correct alignment.

- NPU2: Add flag to nvlink config space indicating DL reset state

  Device drivers need to be able to determine if the DL is out of reset or not so they can safely probe to see if links have already been trained. This patch adds a flag to the vendor specific config space indicating if the DL is out of reset.

- hw/npu2.c: Hardcode MSR_SF when setting up npu XTS contexts

  We don't support anything other than 64-bit mode for address translations so we can safely hardcode it.

- hw/npu2-hw-procedures.c: Add nvram option to override zcal calculations

  In some rare cases the zcal state machine may fail and flag an error. According to hardware designers it is sometimes ok to ignore this failure and use nominal values for the calculations. In this case we add a nvram variable (nv_zcal_override) which will cause skiboot to ignore the failure and use the nominal value specified in nvram.

- npu2: Fix npu2_{read,write}_4b()

  When writing or reading 4-byte values, we need to use the upper half of the 64-bit SCOM register.

  Fix npu2_{read,write}_4b() and their callers to use uint32_t, and appropriately shift the value being written or returned.

- hw/npu2.c: Fix opal_npu_map_lpar to search for existing BDF

- hw/npu2-hw-procedures.c: Fix running of zcal procedure

  The zcal procedure should only be run once per obus (ie. once per group of 3 links). Clean up the code and fix the potential buffer overflow due to a typo. Also updates the zcal settings to their proper values.

- hw/npu2.c: Add memory coherence directory programming

  The memory coherence directory (MCD) needs to know which system memory addresses belong to the GPU. This amounts to setting a BAR and a size in the MCD to cover the addresses assigned to each of the GPUs. To ease assignment we assume GPUs are assigned memory in a contiguous block per chip.

OCC/Power Management

With this release, it's possible to boot POWER9 systems with the OCC enabled and change CPU frequencies. Doing so does require other firmware components to also support this (otherwise the frequency will not be set).

Since *skiboot-5.6.0*:

- occ: Skip setting cores to nominal frequency in P9

  In P9, once OCC is up, it is supposed to setup the cores to nominal frequency. So skip this step in OPAL.

- occ: Fix Pstate ordering for P9

  In P9 the pstate values are positive. They are continuous set of unsigned integers [0 to +N] where Pmax is 0 and Pmin is N. The linear ordering of pstates for P9 has changed compared to P8. P8 has neagtive pstate values advertised as [0 to -N] where Pmax is 0 and Pmin is -N. This patch adds helper routines to abstract pstate comparison with pmax and adds sanity pstate limit checks. This patch also fixes pstate arithmetic by using labs().

- p8-i2c: occ: Add support for OCC to use I2C engines

  This patch adds support to share the I2C engines with host and OCC. OCC uses I2C engines to read DIMM temperatures and to communicate with GPU. OCC Flag register is used for locking between host and OCC. Host requests for the bus by setting a bit in OCC Flag register. OCC sends an interrupt to indicate the change in ownership.

opal-prd/PRD

Since *skiboot-5.6.0*:

- opal-prd: Handle SBE passthrough message passing

  This patch adds support to send SBE pass through command to HBRT.

- SBE: Add passthrough command support

  SBE sends passthrough command. We have to capture this interrupt and send event to HBRT via opal-prd (user space daemon).

- opal-prd: hook up reset_pm_complex

  This change provides the facility to invoke HBRT's reset_pm_complex, in the same manner is done with process_occ_reset previously.

  We add a control command for *opal-prd pm-complex reset*, which is just an alias for occ_reset at this stage.

- prd: Implement firmware side of opaque PRD channel

  This change introduces the firmware side of the opaque HBRT <–> OPAL message channel. We define a base message format to be shared with HBRT (in include/prd-fw-msg.h), and allow firmware requests and responses to be sent over this channel.

  We don't currently have any notifications defined, so have nothing to do for firmware_notify() at this stage.

- opal-prd: Add firmware_request & firmware_notify implementations

  This change adds the implementation of firmware_request() and firmware_notify(). To do this, we need to add a message queue, so that we can properly handle out-of-order messages coming from firmware.

- opal-prd: Add support for variable-sized messages

  With the introductuion of the opaque firmware channel, we want to support variable-sized messages. Rather than expecting to read an entire 'struct opal_prd_msg' in one read() call, we can split this over mutiple reads, potentially expanding our message buffer.

- opal-prd: Sync hostboot interfaces with HBRT

  This change adds new callbacks defined for p9, and the base thunks for the added calls.

- opal-prd: interpret log level prefixes from HBRT

  Interpret the (optional) *_MRK log prefixes on HBRT messages, and set the syslog log priority to suit.

- opal-prd: Add occ reset to usage text

- opal-prd: allow different chips for occ control actions

  The *occ reset* and *occ error* actions can both take a chip id argument, but we're currently just using zero. This change changes the control message format to pass the chip ID from the control process to the opal-prd daemon.

IBM FSP based platforms

Since *skiboot-5.7-rc2*:

- FSP/CONSOLE: Do not enable input irq in write path

  We use irq for reading input from console, but not in output path. Hence do not enable input irq in write path.

  Fixes : 583c8203 (fsp/console: Allocate irq for each hvc console)

Since *skiboot-5.6.0*:

- FSP/CONSOLE: Fix possible NULL dereference

- platforms/ibm-fsp/firenze: Fix PCI slot power-off pattern

  When powering off the PCI slot, the corresponding bits should be set to 0bxx00xx00 instead of 0bxx11xx11. Otherwise, the specified PCI slot can't be put into power-off state. Fortunately, it didn't introduce any side-effects so far.

- FSP/CONSOLE: Workaround for unresponsive ipmi daemon

  We use TCE mapped area to write data to console. Console header (fsp_serbuf_hdr) is modified by both FSP and OPAL (OPAL updates next_in pointer in fsp_serbuf_hdr and FSP updates next_out pointer).

  Kernel makes opal_console_write() OPAL call to write data to console. OPAL write data to TCE mapped area and sends MBOX command to FSP. If our console becomes full and we have data to write to console, we keep on waiting until FSP reads data.

  In some corner cases, where FSP is active but not responding to console MBOX message (due to buggy IPMI) and we have heavy console write happening from kernel, then eventually our console buffer becomes full. At this point OPAL starts sending OPAL_BUSY_EVENT to kernel. Kernel will keep on retrying. This is creating kernel soft lockups. In some extreme case when every CPU is trying to write to console, user will not be able to ssh and thinks system is hang.

  If we reset FSP or restart IPMI daemon on FSP, system recovers and everything becomes normal.

  This patch adds workaround to above issue by returning OPAL_HARDWARE when cosole is full. Side effect of this patch is, we may endup dropping latest console data. But better to drop console data than system hang.

- FSP: Set status field in response message for timed out message

  For timed out FSP messages, we set message status as "fsp_msg_timeout". But most FSP driver users (like surviellance) are ignoring this field. They always look for FSP returned status value in callback function (second byte in word1). So we endup treating timed out message as success response from FSP.

  Sample output:

```
[69902.432509048,7] SURV: Sending the heartbeat command to FSP
[70023.226860117,4] FSP: Response from FSP timed out, word0 = d66a00d7, word1 = 0␣
↪state: 3
....
[70023.226901445,7] SURV: Received heartbeat acknowledge from FSP
[70023.226903251,3] FSP: fsp_trigger_reset() entry
```

  Here SURV code thought it got valid response from FSP. But actually we didn't receive response from FSP.

  This patch fixes above issue by updating status field in response structure.

- FSP: Improve timeout message
- FSP/RTC: Fix possible FSP R/R issue in rtc write path
- hw/fsp/rtc: read/write cached rtc tod on fsp hir.

  Currently fsp-rtc reads/writes the cached RTC TOD on an fsp reset. Use latest fsp_in_rr() function to properly read the cached rtc value when fsp reset initiated by the hir.

  Below is the kernel trace when we set hw clock, when hir process starts.

```
[ 1727.775824] NMI watchdog: BUG: soft lockup - CPU#57 stuck for 23s!␣
↪[hwclock:7688]
[ 1727.775856] Modules linked in: vmx_crypto ibmpowernv ipmi_powernv uio_pdrv_
↪genirq ipmi_devintf powernv_op_panel uio ipmi_msghandler powernv_rng leds_
↪powernv ip_tables x_tables autofs4 ses enclosure scsi_transport_sas crc32c_
↪vpmsum lpfc ipr tg3 scsi_transport_fc
[ 1727.775883] CPU: 57 PID: 7688 Comm: hwclock Not tainted 4.10.0-14-generic #16-
↪Ubuntu
```

```
[ 1727.775883] task: c000000fdfdc8400 task.stack: c000000fdfef4000
[ 1727.775884] NIP: c00000000090540c LR: c0000000000846f4 CTR: 000000003006dd70
[ 1727.775885] REGS: c000000fdfef79a0 TRAP: 0901   Not tainted  (4.10.0-14-
↪generic)
[ 1727.775886] MSR: 9000000000009033 <SF,HV,EE,ME,IR,DR,RI,LE>
[ 1727.775889]   CR: 28024442  XER: 20000000
[ 1727.775890] CFAR: c00000000008472c SOFTE: 1
               GPR00: 0000000030005128 c000000fdfef7c20 c00000000144c900␣
↪ffffffffffffff4
               GPR04: 0000000028024442 c00000000090540c 9000000000009033␣
↪0000000000000000
               GPR08: 0000000000000000 0000000031fc4000 c000000000084710␣
↪9000000000001003
               GPR12: c0000000000846e8 c00000000fba0100
[ 1727.775897] NIP [c00000000090540c] opal_set_rtc_time+0x4c/0xb0
[ 1727.775899] LR [c0000000000846f4] opal_return+0xc/0x48
[ 1727.775899] Call Trace:
[ 1727.775900] [c000000fdfef7c20] [c00000000090540c] opal_set_rtc_time+0x4c/0xb0␣
↪(unreliable)
[ 1727.775901] [c000000fdfef7c60] [c000000000900828] rtc_set_time+0xb8/0x1b0
[ 1727.775903] [c000000fdfef7ca0] [c000000000902364] rtc_dev_ioctl+0x454/0x630
[ 1727.775904] [c000000fdfef7d40] [c00000000035b1f4] do_vfs_ioctl+0xd4/0x8c0
[ 1727.775906] [c000000fdfef7de0] [c00000000035bab4] SyS_ioctl+0xd4/0xf0
[ 1727.775907] [c000000fdfef7e30] [c00000000000b184] system_call+0x38/0xe0
[ 1727.775908] Instruction dump:
[ 1727.775909] f821ffc1 39200000 7c832378 91210028 38a10020 39200000 38810028␣
↪f9210020
[ 1727.775911] 4bfffe6d e8810020 80610028 4b77f61d <60000000> 7c7f1b78 3860000a␣
↪2fbffff4
```

This is found when executing the testcase https://github.com/open-power/op-test-framework/blob/master/testcases/fspresetReload.py

With this fix ran fsp hir torture testcase in the above test which is working fine.

- occ: Set return variable to correct value

  When entering this section of code rc will be zero. If fsp_mkmsg() fails the code responsible for printing an error message won't be set. Resetting rc should allow for the error case to trigger if fsp_mkmsg fails.

- capp: Fix hang when CAPP microcode LID is missing on FSP machine

  When the LID is absent, we fail early with an error from start_preload_resource. In that case, capp_ucode_info.load_result isn't set properly causing a subsequent capp_lid_download() to call wait_for_resource_loaded() on something that isn't being loaded, thus hanging.

- FSP: Add check to detect FSP R/R inside fsp_sync_msg()

  OPAL sends MBOX message to FSP and updates message state from fsp_msg_queued -> fsp_msg_sent. fsp_sync_msg() queues message and waits until we get response from FSP. During FSP R/R we move outstanding MBOX messages from msgq to rr_queue including inflight message (fsp_reset_cmdclass()). But we are not resetting inflight message state.

  In extreme croner case where we sent message to FSP via fsp_sync_msg() path and FSP R/R happens before getting respose from FSP, then we will endup waiting in fsp_sync_msg() until everything becomes normal.

  This patch adds fsp_in_rr() check to fsp_sync_msg() and return error to caller if FSP is in R/R.

- FSP: Add check to detect FSP R/R inside fsp_sync_msg()

  OPAL sends MBOX message to FSP and updates message state from fsp_msg_queued -> fsp_msg_sent.

fsp_sync_msg() queues message and waits until we get response from FSP. During FSP R/R we move outstanding MBOX messages from msgq to rr_queue including inflight message (fsp_reset_cmdclass()). But we are not resetting inflight message state.

In extreme croner case where we sent message to FSP via fsp_sync_msg() path and FSP R/R happens before getting respose from FSP, then we will endup waiting in fsp_sync_msg() until everything becomes normal.

This patch adds fsp_in_rr() check to fsp_sync_msg() and return error to caller if FSP is in R/R.

- capp: Fix hang when CAPP microcode LID is missing on FSP machine

  When the LID is absent, we fail early with an error from start_preload_resource. In that case, capp_ucode_info.load_result isn't set properly causing a subsequent capp_lid_download() to call wait_for_resource_loaded() on something that isn't being loaded, thus hanging.

- FSP/CONSOLE: Do not free fsp_msg in error path

  as we reuse same msg to send next output message.

- platform/zz: Acknowledge OCC_LOAD mbox message in ZZ

  In P9 FSP box, OCC image is pre-loaded. So do not handle the load command and send SUCCESS to FSP on recieving OCC_LOAD mbox message.

- FSP/RTC: Improve error log

astbmc systems

Since *skiboot-5.6.0*:

- platforms/astbmc: Don't validate model on palmetto

  The platform isn't compatible with palmetto until the root device-tree node's "model" property is NULL or "palmetto". However, we could have "TN71-BP012" for the property on palmetto.

  ```
  linux# cat /proc/device-tree/model
  TN71-BP012
  ```

  This skips the validation on root device-tree node's "model" property on palmetto, meaning we check the "compatible" property only.

General

Since *skiboot-5.7-rc2*:

- core/pci: Fix mem-leak on fast-reboot

  Fast-reboot has a memory leak which causes the system to crash after about 250 fast-reboots. The patch fixes the memory leak. The cause of the leak was the pci_device's being freed, without freeing the pci_slot within it.

- gcov: properly handle gard and pflash code coverage

Since *skiboot-5.6.0*:

- Reduce log level on non-error log messages

  90% of what we print isn't useful to a normal user. This dramatically reduces the amount of messages printed by OPAL in normal circumstances.

- init: Silence messages and call ourselves "OPAL"

- psi: Switch to ESB mode later

  There's an errata, if we switch to ESB mode before setting up the various ESB mode related registers, a pending interrupts can go wrong.

- lpc: Enable "new" SerIRQ mode
- hw/ipmi/ipmi-sel: missing newline in prlog warning
- p8-i2c OCC lock: fix locking in p9_i2c_bus_owner_change
- Convert important polling loops to spin at lowest SMT priority

  The pattern of calling cpu_relax() inside a polling loop does not suit the powerpc SMT priority instructions. Prefrred is to set a low priority then spin until break condition is reached, then restore priority.

- Improve cpu_idle when PM is disabled

  Split cpu_idle() into cpu_idle_delay() and cpu_idle_job() rather than requesting the idle type as a function argument. Have those functions provide a default polling (non-PM) implentation which spin at the lowest SMT priority.

- core/fdt: Always add a reserve map

  Currently we skip adding the reserved ranges block to the generated FDT blob if we are excluding the root node. This can result in a DTB that dtc will barf on because the reserved memory ranges overlap with the start of the dt_struct block. As an example:

```
$ fdtdump broken.dtb -d
/dts-v1/;
// magic:               0xd00dfeed
// totalsize:           0x7f3 (2035)
// off_dt_struct:       0x30  <----\
// off_dt_strings:      0x7b8       | this is bad!
// off_mem_rsvmap:      0x30  <----/
// version:             17
// last_comp_version:   16
// boot_cpuid_phys:     0x0
// size_dt_strings:     0x3b
// size_dt_struct:      0x788

/memreserve/ 0x100000000 0x300000004;
/memreserve/ 0x3300000001 0x169626d2c;
/memreserve/ 0x706369652d736c6f 0x7473000000000003;
        *continues*
```

  With this patch:

```
$ fdtdump working.dtb -d
/dts-v1/;
// magic:               0xd00dfeed
// totalsize:           0x803 (2051)
// off_dt_struct:       0x40
// off_dt_strings:      0x7c8
// off_mem_rsvmap:      0x30
// version:             17
// last_comp_version:   16
// boot_cpuid_phys:     0x0
// size_dt_strings:     0x3b
// size_dt_struct:      0x788

// 0040: tag: 0x00000001 (FDT_BEGIN_NODE)
```

```
/ {
// 0048: tag: 0x00000003 (FDT_PROP)
// 07fb: string: phandle
// 0054: value
    phandle = <0x00000001>;
        *continues*
```

- hw/lpc-mbox: Use message registers for interrupts

  Currently the BMC raises the interrupt using the BMC control register. It does so on all accesses to the 16 'data' registers meaning that when the BMC only wants to set the ATTN (on which we have interrupts enabled) bit we will also get a control register based interrupt.

  The solution here is to mask that interrupt permanantly and enable interrupts on the protocol defined 'response' data byte.

PCI

Since *skiboot-5.6.0*:

- pci: Wait 20ms before checking presence detect on PCIe

  As the PHB presence logic has a debounce timer that can take a while to settle.

- phb3+iov: Fixup support for config space filters

  The filter should be called before the HW access and its return value control whether to perform the access or not

- core/pci: Use PCI slot's power facality in pci_enable_bridge()

  The current implmentation has incorrect assumptions: there is always a PCI slot associated with root port and PCIe switch downstream port and all of them are capable to change its power state by register PCICAP_EXP_SLOTCTL. Firstly, there might not a PCI slot associated with the root port or PCIe switch downstream port. Secondly, the power isn't controlled by standard config register (PCICAP_EXP_SLOTCTL). There are I2C slave devices used to control the power states on Tuleta.

  In order to use the PCI slot's methods to manage the power states, this does:

  - Introduce PCI_SLOT_FLAG_ENFORCE, indicates the request operation is enforced to be applied.
  - pci_enable_bridge() is split into 3 functions: pci_bridge_power_on() to power it on; pci_enable_bridge() as a place holder and pci_bridge_wait_link() to wait the downstream link to come up.
  - In pci_bridge_power_on(), the PCI slot's specific power management methods are used if there is a PCI slot associated with the PCIe switch downstream port or root port.

- platforms/astbmc/slots.c: Allow comparison of bus numbers when matching slots

  When matching devices on multiple down stream PLX busses we need to compare more than just the device-id of the PCIe BDFN, so increase the mask to do so.

Debugging, Tests and simulators

Since *skiboot-5.7-rc2*:

- boot_tests: add PFLASH_TO_COPY for OpenBMC

- travis: Add debian stretch and unstable

  At the moment, we mark them both as being able to fail, as we're hitting an assert in one of the unit tests on debian stretch, and that hasn't yet been chased down.

- core/backtrace: Serialise printing backtraces

  Add a lock so that only one thread can print a backtrace at a time. This should prevent multiple threads from garbaling each other's backtraces.

Since *skiboot-5.7-rc1*:

- lpc: remove double LPC prefix from messages
- opal-ci/fetch-debian-jessie-installer: follow redirects Fixes some CI failures
- test/qemu-jessie: bail out fast on kernel panic
- test/qemu-jessie: dump boot log on failure
- travis: add fedora26
- xz: add fallthrough annotations to silence GCC7 warning

Since *skiboot-5.6.0*:

- boot-tests: add OpenBMC support
- boot_test.sh: Add SMC BMC support

  Your BMC needs a special debug image flashed to use this, the exact image and methods aren't something I can publish here, but if you work for IBM or SMC you can find out from the right sources.

  A few things are needed to move around to be able to flash to a SMC BMC.

  For a start, the SSH daemon will only accept connections after a special incantation (which I also can't share), but you should put that in the ~/.skiboot_boot_tests file along with some other default login information we don't publicise too broadly (because Security Through Obscurity is *obviously* a good idea. . . .)

  We also can't just directly "ssh /bin/true", we need an expect script, and we can't scp, but we can anonymous rsync!

  You also need a pflash binary to copy over.

- hdata_to_dt: Add PVR overrides to the usage text
- mambo: Add a reservation for the initramfs

  On most systems the initramfs is loaded inside the part of memory reserved for the OS [0x0-0x30000000] and skiboot will never touch it. On mambo it's loaded at 0x80000000 and if you're unlucky skiboot can allocate over the top of it and corrupt the initramfs blob.

  There might be the downside that the kernel cannot re-use the initramfs memory since it's marked as reserved, but the kernel might also free it anyway.

- mambo: Update P9 PVR to reflect Scale out 24 core chips

  The P9 PVR bits 48:51 don't indicate a revision but instead different configurations. From BookIV we have:

Bits	Configuration
0	Scale out 12 cores
1	Scale out 24 cores
2	Scale up 12 cores
3	Scale up 24 cores

Skiboot will mostly the use "Scale out 24 core" configuration (ie. SMT4 not SMT8) so reflect this in mambo.

- core: Move enable_mambo_console() into chip initialisation

  Rather than having a wart in main_cpu_entry() that initialises the mambo console, we can move it into init_chips() which is where we discover that we're on mambo.

- mambo: Create multiple chips when we have multiple CPUs

  Currently when we boot mambo with multiple CPUs, we create multiple CPU nodes in the device tree, and each claims to be on a separate chip.

  However we don't create multiple xscom nodes, which means skiboot only knows about a single chip, and all CPUs end up on it. At the moment mambo is not able to create multiple xscom controllers. We can create fake ones, just by faking the device tree up, but that seems uglier than this solution.

  So create a mambo-chip for each CPU other than 0, to tell skiboot we want a separate chip created. This then enables Linux to see multiple chips:

```
smp: Brought up 2 nodes, 2 CPUs
numa: Node 0 CPUs: 0
numa: Node 1 CPUs: 1
```

- chip: Add support for discovering chips on mambo

  Currently the only way for skiboot to discover chips is by looking for xscom nodes. But on mambo it's currently not possible to create multiple xscom nodes, which means we can only simulate a single chip system.

  However it seems we can fairly cleanly add support for a special mambo chip node, and use that to instantiate multiple chips.

  Add a check in init_chip() that we're not clobbering an already initialised chip, now that we have two places that initialise chips.

- mambo: Make xscom claim to be DD 2.0

  In the mambo tcl we set the CPU version to DD 2.0, because mambo is not bug compatible with DD 1.

  But in xscom_read_cfam_chipid() we have a hard coded value, to work around the lack of the f000f register, which claims to be P9 DD 1.0.

  This doesn't seem to cause crashes or anything, but at boot we do see:

```
[    0.003893084,5] XSCOM: chip 0x0 at 0x1a0000000000 [P9N DD1.0]
```

  So fix it to claim that the xscom is also DD 2.0 to match the CPU.

- mambo: Match whole string when looking up symbols with linsym/skisym

  linsym/skisym use a regex to match the symbol name, and accepts a partial match against the entry in the symbol map, which can lead to somewhat confusing results, eg:

```
systemsim % linsym early_setup
0xc000000000027890
systemsim % linsym early_setup$
0xc000000000aa8054
systemsim % linsym early_setup_secondary
0xc000000000027890
```

  I don't think that's the behaviour we want, so append a $ to the name so that the symbol has to match against the whole entry, eg:

```
systemsim % linsym early_setup
0xc000000000aa8054
```

- Disable nap on P8 Mambo, public release has bugs
- mambo: Allow loading multiple CPIOs

  Currently we have support for loading a single CPIO and telling Linux to use it as the initrd. But the Linux code actually supports having multiple CPIOs contiguously in memory, between initrd-start and end, and will unpack them all in order. That is a really nice feature as it means you can have a base CPIO with your root filesystem, and then tack on others as you need for various tests etc.

  So expand the logic to handle SKIBOOT_INITRD, and treat it as a comma separated list of CPIOs to load. I chose comma as it's fairly rare in filenames, but we could make it space, colon, whatever. Or we could add a new environment variable entirely. The code also supports trimming whitespace from the values, so you can have "cpio1, cpio2".
- hdata/test: Add memory reservations to hdata_to_dt

  Currently memory reservations are parsed, but since they are not processed until mem_region_init() they don't appear in the output device tree blob. Several bugs have been found with memory reservations so we want them to be part of the test output.

  Add them and clean up several usages of printf() since we want only the dtb to appear in standard out.

pflash/libffs

Since *skiboot-5.7-rc2*:

- pflash option to retrieve PNOR partition flags

  This commit extends pflash with an option to retrieve and print information for a particular partition, including the content from "pflash -i" and a verbose list of set miscellaneous flags. -i option is also updated to print a short list of flags in addition to the ECC flag, with one character per flag. A test of the new option is included in libflash/test.

Since *skiboot-5.6.0*:

- libflash/libffs: Zero checksum words

  On writing ffs entries to flash libffs doesn't zero checksum words before calculating the checksum across the entire structure. This causes an inaccurate calculation of the checksum as it may calculate a checksum on non-zero checksum bytes.
- libffs: Fix ffs_lookup_part() return value

  It would return success when the part wasn't found
- libflash/libffs: Correctly update the actual size of the partition

  libffs has been updating FFS partition information in the wrong place which leads to incomplete erases and corruption.
- libflash: Initialise entries list earlier

  In the bail-out path we call ffs_close() to tear down the partially initialised ffs_handle. ffs_close() expects the entries list to be initialised so we need to do that earlier to prevent a null pointer dereference.

mbox-flash

mbox-flash is the emerging standard way of talking to host PNOR flash on POWER9 systems.

- libflash/mbox-flash: Implement MARK_WRITE_ERASED mbox call

  Version two of the mbox-flash protocol defines a new command: MARK_WRITE_ERASED.

  This command provides a simple way to mark a region of flash as all 0xff without the need to go and write all 0xff. This is an optimisation as there is no need for an erase before a write, it is the responsibility of the BMC to deal with the flash correctly, however in v1 it was ambiguous what a client should do if the flash should be erased but not actually written to. This allows of a optimal path to resolve this problem.

- libflash/mbox-flash: Update to V2 of the protocol

  Updated version 2 of the protocol can be found at: https://github.com/openbmc/mboxbridge/blob/master/Documentation/mbox_protocol.md

  This commit changes mbox-flash such that it will preferentially talk version 2 to any capable daemon but still remain capable of talking to v1 daemons.

  Version two changes some of the command definitions for increased consistency and usability. Version two includes more attention bits - these are now dealt with at a simple level.

- libflash/mbox-flash: Implement MARK_WRITE_ERASED mbox call

  Version two of the mbox-flash protocol defines a new command: MARK_WRITE_ERASED.

  This command provides a simple way to mark a region of flash as all 0xff without the need to go and write all 0xff. This is an optimisation as there is no need for an erase before a write, it is the responsibility of the BMC to deal with the flash correctly, however in v1 it was ambiguous what a client should do if the flash should be erased but not actually written to. This allows of a optimal path to resolve this problem.

- libflash/mbox-flash: Update to V2 of the protocol

  Updated version 2 of the protocol can be found at: https://github.com/openbmc/mboxbridge/blob/master/Documentation/mbox_protocol.md

  This commit changes mbox-flash such that it will preferentially talk version 2 to any capable daemon but still remain capable of talking to v1 daemons.

  Version two changes some of the command definitions for increased consistency and usability. Version two includes more attention bits - these are now dealt with at a simple level.

- hw/lpc-mbox: Use message registers for interrupts

  Currently the BMC raises the interrupt using the BMC control register. It does so on all accesses to the 16 'data' registers meaning that when the BMC only wants to set the ATTN (on which we have interrupts enabled) bit we will also get a control register based interrupt.

  The solution here is to mask that interrupt permanantly and enable interrupts on the protocol defined 'response' data byte.

Contributors

- Processed 232 csets from 29 developers.
- 1 employer found
- A total of 13043 lines added, 2517 removed (delta 10526)

Extending the analysis done for some previous releases, we can see our trends in code review across versions:

Release	csets	Ack %	Reviews %	Tested %	Reported %
5.0	329	15 (5%)	20 (6%)	1 (0%)	0 (0%)
5.1	372	13 (3%)	38 (10%)	1 (0%)	4 (1%)
5.2-rc1	334	20 (6%)	34 (10%)	6 (2%)	11 (3%)
5.3-rc1	302	36 (12%)	53 (18%)	4 (1%)	5 (2%)
5.4	361	16 (4%)	28 (8%)	1 (0%)	9 (2%)
5.5	408	11 (3%)	48 (12%)	14 (3%)	10 (2%)
5.6	87	12 (14%)	6 (7%)	5 (6%)	2 (2%)
5.7	232	30 (13%)	32 (14%)	5 (2%)	2 (1%)

This cycle has been good for reviews/acks, scoring second highest percentage ever on both, as well as being right up there on absolute numbers.

Developers with the most changesets

Developer	#	%
Benjamin Herrenschmidt	41	(17.7%)
Stewart Smith	31	(13.4%)
Michael Neuling	28	(12.1%)
Oliver O'Halloran	18	(7.8%)
Vasant Hegde	18	(7.8%)
Jeremy Kerr	12	(5.2%)
Alistair Popple	11	(4.7%)
Gavin Shan	10	(4.3%)
Russell Currey	9	(3.9%)
Michael Ellerman	9	(3.9%)
Madhavan Srinivasan	7	(3.0%)
Cyril Bur	6	(2.6%)
Christophe Lombard	5	(2.2%)
Shilpasri G Bhat	5	(2.2%)
Andrew Donnellan	3	(1.3%)
Nicholas Piggin	3	(1.3%)
Mahesh Salgaonkar	2	(0.9%)
Anju T Sudhakar	2	(0.9%)
Hemant Kumar	2	(0.9%)
Matt Brown	1	(0.4%)
Michael Tritz	1	(0.4%)
Joel Stanley	1	(0.4%)
Balbir Singh	1	(0.4%)
Frederic Barrat	1	(0.4%)
Andrew Jeffery	1	(0.4%)
Pridhiviraj Paidipeddi	1	(0.4%)
Reza Arbab	1	(0.4%)
Suraj Jitindar Singh	1	(0.4%)
Vaibhav Jain	1	(0.4%)

Developers with the most changed lines

Developer	#	%
Hemant Kumar	3056	(23.0%)
Stewart Smith	1826	(13.7%)
Benjamin Herrenschmidt	1348	(10.1%)
Christophe Lombard	937	(7.0%)
Shilpasri G Bhat	770	(5.8%)
Madhavan Srinivasan	755	(5.7%)
Jeremy Kerr	731	(5.5%)
Cyril Bur	674	(5.1%)
Alistair Popple	477	(3.6%)
Gavin Shan	414	(3.1%)
Russell Currey	396	(3.0%)
Michael Neuling	336	(2.5%)
Vasant Hegde	308	(2.3%)
Oliver O'Halloran	300	(2.3%)
Anju T Sudhakar	300	(2.3%)
Michael Tritz	167	(1.3%)
Frederic Barrat	113	(0.8%)
Nicholas Piggin	93	(0.7%)
Mahesh Salgaonkar	76	(0.6%)
Michael Ellerman	66	(0.5%)
Suraj Jitindar Singh	59	(0.4%)
Andrew Donnellan	53	(0.4%)
Joel Stanley	20	(0.2%)
Balbir Singh	12	(0.1%)
Reza Arbab	10	(0.1%)
Vaibhav Jain	9	(0.1%)
Pridhiviraj Paidipeddi	2	(0.0%)
Matt Brown	1	(0.0%)
Andrew Jeffery	1	(0.0%)

Developers with the most signoffs

(total 242)

Developer	#	%
Stewart Smith	201	(83.1%)
Michael Neuling	29	(12.0%)
Madhavan Srinivasan	4	(1.7%)
Suraj Jitindar Singh	3	(1.2%)
Anju T Sudhakar	2	(0.8%)
Hemant Kumar	2	(0.8%)
Cyril Bur	1	(0.4%)

Developers with the most reviews

(total 32)

Developer	#	%
Vasant Hegde	8	(25.0%)
Cyril Bur	7	(21.9%)
Andrew Donnellan	5	(15.6%)
Frederic Barrat	5	(15.6%)
Andrew Jeffery	2	(6.2%)
Gavin Shan	2	(6.2%)
Joel Stanley	1	(3.1%)
Oliver O'Halloran	1	(3.1%)
Alistair Popple	1	(3.1%)

Developers with the most test credits

(total 5)

Developer	#	%
Vasant Hegde	2	(40.0%)
Oliver O'Halloran	1	(20.0%)
Ananth N Mavinakayanahalli	1	(20.0%)
Michael Ellerman	1	(20.0%)

Developers who gave the most tested-by credits

(total 5)

Developer	#	%
Jeremy Kerr	2	(40.0%)
Vasant Hegde	1	(20.0%)
Oliver O'Halloran	1	(20.0%)
Michael Ellerman	1	(20.0%)

Developers with the most report credits

(total 2)

Developer	#	%
Oliver O'Halloran	1	(50.0%)
Alastair D'Silva	1	(50.0%)

Developers who gave the most report credits

(total 2)

Developer	#	%
Andrew Donnellan	1	(50.0%)
Stewart Smith	1	(50.0%)

4.1.64 skiboot-5.7-rc1

skiboot v5.7-rc1 was released on Monday July 3rd 2017. It is the first release candidate of skiboot 5.7, which will become the new stable release of skiboot following the 5.6 release, first released 24th May 2017.

skiboot v5.7-rc1 contains all bug fixes as of *skiboot-5.4.6* and *skiboot-5.1.19* (the currently maintained stable releases). We do not currently expect to do any 5.6.x stable releases.

For how the skiboot stable releases work, see *Skiboot stable tree rules and releases* for details.

The current plan is to cut the final 5.7 by July 12th, with skiboot 5.7 being for all POWER8 and POWER9 platforms in op-build v1.18 (Due July 12th). This is a short cycle as this release is mainly targetted towards POWER9 bringup efforts.

This is the second release using the new regular six week release cycle, similar to op-build, but slightly offset to allow for a short stabilisation period. Expected release dates and contents are tracked using GitHub milestone and issues: https://github.com/open-power/skiboot/milestones

Over skiboot-5.6, we have the following changes:

New Features

New features in this release for POWER9 systems:

- In Memory Counters (IMC) (See *OPAL/Skiboot In-Memory Collection (IMC) interface Documentation* for details)
- phb4: Activate shared PCI slot on witherspoon (see *Shared Slot*)
- phb4 capi (i.e. CAPI2): Enable capi mode for PHB4 (see *CAPI on PHB4*)

New feature for IBM FSP based systems:

- fsp/tpo: Provide support for disabling TPO alarm

  This patch adds support for disabling a preconfigured Timed-Power-On(TPO) alarm on FSP based systems. Presently once a TPO alarm is configured from the kernel it will be triggered even if its subsequently disabled.

  With this patch a TPO alarm can be disabled by passing y_m_d==hr_min==0 to fsp_opal_tpo_write(). A branch is added to the function to handle this case by sending FSP_CMD_TPO_DISABLE message to the FSP instead of usual FSP_CMD_TPO_WRITE message. The kernel is expected to call opal_tpo_write() with y_m_d==hr_min==0 to request opal to disable TPO alarm.

POWER9

Development on POWER9 systems continues in earnest.

This release includes the first support for POWER9 DD2 chips. Future releases will likely contain more bug fixes, this release has booted on real hardware.

- hdata: Reserve Trace Areas

  When hostboot is configured to setup in memory tracing it will reserve some memory for use by the hardware tracing facility. We need to mark these areas as off limits to the operating system and firmware.

- hdata: Make out-of-range idata print at PR_DEBUG

  Some fields just aren't populated on some systems.

- hdata: Ignore unnamed memory reservations.

  Hostboot should name any and all memory reservations that it provides. Currently some hostboots export a broken reservation covering the first 256MB of memory and this causes the system to crash at boot due to an invalid free because this overlaps with the static "ibm,os-reserve" region (which covers the first 768MB of memory).

  According to the hostboot team unnamed reservations are invalid and can be ignored.

- hdata: Check the Host I2C devices array version

  Currently this is not populated on FSP machines which causes some obnoxious errors to appear in the boot log. We also only want to parse version 1 of this structure since future versions will completely change the array item format.

- Ensure P9 DD1 workarounds apply only to Nimbus

  The workarounds for P9 DD1 are only needed for Nimbus. P9 Cumulus will be DD1 but don't need these same workarounds.

  This patch ensures the P9 DD1 workarounds only apply to Nimbus. It also renames some things to make clear what's what.

- cpu: Cleanup AMR and IAMR when re-initializing CPUs

  There's a bug in current Linux kernels leaving crap in those registers accross kexec and not sanitizing them on boot. This breaks kexec under some circumstances (such as booting a hash kernel from a radix one on P9 DD2.0).

  The long term fix is in Linux, but this workaround is a reasonable way of "sanitizing" those SPRs when Linux calls opal_reinit_cpus() and shouldn't have adverse effects.

  We could also use that same mechanism to cleanup other things as well such as restoring some other SPRs to their default value in the future.

- Set POWER9 RPR SPR to 0x00000103070F1F3F. Same value as P8.

  Without this, thread priorities inside a core don't work.

- cpu: Support setting HID[RADIX] and set it by default on P9

  This adds new opal_reinit_cpus() flags to setup radix or hash mode in HID[8] on POWER9.

  By default HID[8] will be set. On P9 DD1.0, Linux will change it as needed. On P9 DD2.0 hash works in radix mode (radix is really "dual" mode) so KVM won't break and existing kernels will work.

  Newer kernels built for hash will call this to clear the HID bit and thus get the full size of the TLB as an optimization.

- Add "cleanup_global_tlb" for P9 and later

  Uses broadcast TLBIE's to cleanup the TLB on all cores and on the nest MMU

- xive: DD2.0 updates

  Add support for StoreEOI, fix StoreEOI MMIO offset in ESB page, and other cleanups

- Update default TSCR value for P9 as recommended by HW folk.

- xive: Fix initialisation of xive_cpu_state struct

  When using XIVE emulation with DEBUG=1, we run into crashes in log_add() due to the xive_cpu_state->log_pos being uninitialised (and thus, with DEBUG enabled, initialised to the poison value of 0x99999999).

OCC/Power Management

With this release, it's possible to boot POWER9 systems with the OCC enabled and change CPU frequencies. Doing so does require other firmware components to also support this (otherwise the frequency will not be set).

- occ: Skip setting cores to nominal frequency in P9

  In P9, once OCC is up, it is supposed to setup the cores to nominal frequency. So skip this step in OPAL.

- occ: Fix Pstate ordering for P9

  In P9 the pstate values are positive. They are continuous set of unsigned integers [0 to +N] where Pmax is 0 and Pmin is N. The linear ordering of pstates for P9 has changed compared to P8. P8 has neagtive pstate values advertised as [0 to -N] where Pmax is 0 and Pmin is -N. This patch adds helper routines to abstract pstate comparison with pmax and adds sanity pstate limit checks. This patch also fixes pstate arithmetic by using labs().

- p8-i2c: occ: Add support for OCC to use I2C engines

  This patch adds support to share the I2C engines with host and OCC. OCC uses I2C engines to read DIMM temperatures and to communicate with GPU. OCC Flag register is used for locking between host and OCC. Host requests for the bus by setting a bit in OCC Flag register. OCC sends an interrupt to indicate the change in ownership.

opal-prd/PRD

- opal-prd: Handle SBE passthrough message passing

  This patch adds support to send SBE pass through command to HBRT.

- SBE: Add passthrough command support

  SBE sends passthrough command. We have to capture this interrupt and send event to HBRT via opal-prd (user space daemon).

- opal-prd: hook up reset_pm_complex

  This change provides the facility to invoke HBRT's reset_pm_complex, in the same manner is done with process_occ_reset previously.

  We add a control command for *opal-prd pm-complex reset*, which is just an alias for occ_reset at this stage.

- prd: Implement firmware side of opaque PRD channel

  This change introduces the firmware side of the opaque HBRT <–> OPAL message channel. We define a base message format to be shared with HBRT (in include/prd-fw-msg.h), and allow firmware requests and responses to be sent over this channel.

  We don't currently have any notifications defined, so have nothing to do for firmware_notify() at this stage.

- opal-prd: Add firmware_request & firmware_notify implementations

  This change adds the implementation of firmware_request() and firmware_notify(). To do this, we need to add a message queue, so that we can properly handle out-of-order messages coming from firmware.

- opal-prd: Add support for variable-sized messages

  With the introductuion of the opaque firmware channel, we want to support variable-sized messages. Rather than expecting to read an entire 'struct opal_prd_msg' in one read() call, we can split this over mutiple reads, potentially expanding our message buffer.

- opal-prd: Sync hostboot interfaces with HBRT

  This change adds new callbacks defined for p9, and the base thunks for the added calls.

- opal-prd: interpret log level prefixes from HBRT

  Interpret the (optional) *_MRK log prefixes on HBRT messages, and set the syslog log priority to suit.

- opal-prd: Add occ reset to usage text

- opal-prd: allow different chips for occ control actions

  The *occ reset* and *occ error* actions can both take a chip id argument, but we're currently just using zero. This change changes the control message format to pass the chip ID from the control process to the opal-prd daemon.

PCI/PHB4

- phb4: Fix number of index bits in IODA tables

  On PHB4 the number of index bits in the IODA table address register was bumped to 10 bits to accomodate for 1024 MSIs and 1024 TVEs (DD2).

  However our macro only defined the field to be 9 bits, thus causing "interesting" behaviours on some systems.

- phb4: Harden init with bad PHBs

  Currently if we read all 1's from the EEH or IRQ capabilities, we end up train wrecking on some other random code (eg. an assert() in xive).

  This hardens the PHB4 code to look for these bad reads and more gracefully fails the init for that PHB alone. This allows the rest of the system to boot and ignore those bad PHBs.

- phb4 capi (i.e. CAPI2): Handle HMI events

  Find the CAPP on the chip associated with the HMI event for PHB4. The recovery mode (re-initialization of the capp, resume of functional operations) is only available with P9 DD2. A new patch will be provided to support this feature.

- phb4 capi (i.e. CAPI2): Enable capi mode for PHB4

  Enable the Coherently attached processor interface. The PHB is used as a CAPI interface. CAPI Adapters can be connected to either PEC0 or PEC2. Single port CAPI adapter can be connected to either PEC0 or PEC2, but Dual-Port Adapter can be only connected to PEC2 * CAPP0 attached to PHB0(PEC0 - single port) * CAPP1 attached to PHB3(PEC2 - single or dual port)

- hw/phb4: Rework phb4_get_presence_state()

  There are two issues in current implementation: It should return errcode visibile to Linux, which has prefix OPAL_*. The code isn't very obvious.

  This returns OPAL_HARDWARE when the PHB is broken. Otherwise, OPAL_SUCCESS is always returned. In the mean while, It refactors the code to make it obvious: OPAL_PCI_SLOT_PRESENT is returned when the presence signal (low active) or PCIe link is active. Otherwise, OPAL_PCI_SLOT_EMPTY is returned.

- phb4: Error injection for config space

  Implement CFG (config space) error injection.

  This works the same as PHB3. MMIO and DMA error injection require a rewrite, so they're unsupported for now.

  While it's not feature complete, this at least provides an easy way to inject an error that will trigger EEH.

- phb4: Error clear implementation

- phb4: Mask link down errors during reset

  During a hot reset the PCI link will drop, so we need to mask link down events to prevent unnecessary errors.

- phb4: Implement root port initialization

  phb4_root_port_init() was a NOP before, so fix that.

- phb4: Complete reset implementation

  This implements complete reset (creset) functionality for POWER9 DD1.

  Only partially tested and contends with some DD1 errata, but it's a start.

- phb4: Activate shared PCI slot on witherspoon

  Witherspoon systems come with a 'shared' PCI slot: physically, it looks like a x16 slot, but it's actually two x8 slots connected to two PHBs of two different chips. Taking advantage of it requires some logic on the PCI adapter. Only the Mellanox CX5 adapter is known to support it at the time of this writing.

  This patch enables support for the shared slot on witherspoon if a x16 adapter is detected. Each x8 slot has a presence bit, so both bits need to be set for the activation to take place. Slot sharing is activated through a gpio.

  Note that there's no easy way to be sure that the card is indeed a shared-slot compatible PCI adapter and not a normal x16 card. Plugging a normal x16 adapter on the shared slot should be avoided on witherspoon, as the link won't train on the second slot, resulting in a timeout and a longer boot time. Only the first slot is usable and the x16 adapter will end up using only half the lines.

  If the PCI card plugged on the physical slot is only x8 (or less), then the presence bit of the second slot is not set, so this patch does nothing. The x8 (or less) adapter should work like on any other physical slot.

- phb4: Block D-state power management on direct slots

  As current revisions of PHB4 don't properly handle the resulting L1 link transition.

- phb4: Call pci config filters

- phb4: Mask out write-1-to-clear registers in RC cfg

  The root complex config space only supports 4-byte accesses. Thus, when the client requests a smaller size write, we do a read-modify-write to the register.

  However, some register have bits defined as "write 1 to clear".

  If we do a RMW cycles on such a register and such bits are 1 in the part that the client doesn't intend to modify, we will accidentally write back those 1's and clear the corresponding bit.

  This avoids it by masking out those magic bits from the "old" value read from the register.

- phb4: Properly mask out link down errors during reset

- phb3/4: Silence a useless warning

  PHB's don't have base location codes on non-FSP systems and it's normal.

- phb4: Workaround bug in spec 053

  Wait for DLP PGRESET to clear *after* lifting the PCIe core reset

- phb4: DD2.0 updates

  Support StoreEOI, full complements of PEs (twice as big TVT) and other updates.

  Also renumber init steps to match spec 063

NPU2

Note that currently NPU2 support is limited to POWER9 DD1 hardware.

- platforms/astbmc/witherspoon.c: Add NPU2 slot mappings

  For NVLink2 to function PCIe devices need to be associated with the right NVLinks. This association is supposed to be passed down to Skiboot via HDAT but those fields are still not correctly filled out. To work around this we add slot tables for the NVLinks similar to what we have for P8+.

- hw/npu2.c: Fix device aperture calculation

  The POWER9 NPU2 implements an address compression scheme to compress 56-bit P9 physical addresses to 47-bit GPU addresses. System software needs to know both addresses, unfortunately the calculation of the compressed address was incorrect. Fix it here.

- hw/npu2.c: Change MCD BAR allocation order

  MCD BARs need to be correctly aligned to the size of the region. As GPU memory is allocated from the top of memory down we should start allocating from the highest GPU memory address to the lowest to ensure correct alignment.

- NPU2: Add flag to nvlink config space indicating DL reset state

  Device drivers need to be able to determine if the DL is out of reset or not so they can safely probe to see if links have already been trained. This patch adds a flag to the vendor specific config space indicating if the DL is out of reset.

- hw/npu2.c: Hardcode MSR_SF when setting up npu XTS contexts

  We don't support anything other than 64-bit mode for address translations so we can safely hardcode it.

- hw/npu2-hw-procedures.c: Add nvram option to override zcal calculations

  In some rare cases the zcal state machine may fail and flag an error. According to hardware designers it is sometimes ok to ignore this failure and use nominal values for the calculations. In this case we add a nvram variable (nv_zcal_override) which will cause skiboot to ignore the failure and use the nominal value specified in nvram.

- npu2: Fix npu2_{read,write}_4b()

  When writing or reading 4-byte values, we need to use the upper half of the 64-bit SCOM register.

  Fix npu2_{read,write}_4b() and their callers to use uint32_t, and appropriately shift the value being written or returned.

- hw/npu2.c: Fix opal_npu_map_lpar to search for existing BDF

- hw/npu2-hw-procedures.c: Fix running of zcal procedure

  > The zcal procedure should only be run once per obus (ie. once per group of 3 links). Clean up the code and fix the potential buffer overflow due to a typo. Also updates the zcal settings to their proper values.

- hw/npu2.c: Add memory coherence directory programming

  The memory coherence directory (MCD) needs to know which system memory addresses belong to the GPU. This amounts to setting a BAR and a size in the MCD to cover the addresses assigned to each of the GPUs. To ease assignment we assume GPUs are assigned memory in a contiguous block per chip.

pflash/libflash

- libflash/libffs: Zero checksum words

  On writing ffs entries to flash libffs doesn't zero checksum words before calculating the checksum across the entire structure. This causes an inaccurate calculation of the checksum as it may calculate a checksum on non-zero checksum bytes.

- libffs: Fix ffs_lookup_part() return value

  It would return success when the part wasn't found

- libflash/libffs: Correctly update the actual size of the partition

  libffs has been updating FFS partition information in the wrong place which leads to incomplete erases and corruption.

- libflash: Initialise entries list earlier

  In the bail-out path we call ffs_close() to tear down the partially initialised ffs_handle. ffs_close() expects the entries list to be initialised so we need to do that earlier to prevent a null pointer dereference.

mbox-flash

mbox-flash is the emerging standard way of talking to host PNOR flash on POWER9 systems.

- libflash/mbox-flash: Implement MARK_WRITE_ERASED mbox call

  Version two of the mbox-flash protocol defines a new command: MARK_WRITE_ERASED.

  This command provides a simple way to mark a region of flash as all 0xff without the need to go and write all 0xff. This is an optimisation as there is no need for an erase before a write, it is the responsibility of the BMC to deal with the flash correctly, however in v1 it was ambiguous what a client should do if the flash should be erased but not actually written to. This allows of a optimal path to resolve this problem.

- libflash/mbox-flash: Update to V2 of the protocol

  Updated version 2 of the protocol can be found at: https://github.com/openbmc/mboxbridge/blob/master/Documentation/mbox_protocol.md

  This commit changes mbox-flash such that it will preferentially talk version 2 to any capable daemon but still remain capable of talking to v1 daemons.

  Version two changes some of the command definitions for increased consistency and usability. Version two includes more attention bits - these are now dealt with at a simple level.

- libflash/mbox-flash: Implement MARK_WRITE_ERASED mbox call

  Version two of the mbox-flash protocol defines a new command: MARK_WRITE_ERASED.

  This command provides a simple way to mark a region of flash as all 0xff without the need to go and write all 0xff. This is an optimisation as there is no need for an erase before a write, it is the responsibility of the BMC to deal with the flash correctly, however in v1 it was ambiguous what a client should do if the flash should be erased but not actually written to. This allows of a optimal path to resolve this problem.

- libflash/mbox-flash: Update to V2 of the protocol

  Updated version 2 of the protocol can be found at: https://github.com/openbmc/mboxbridge/blob/master/Documentation/mbox_protocol.md

  This commit changes mbox-flash such that it will preferentially talk version 2 to any capable daemon but still remain capable of talking to v1 daemons.

Version two changes some of the command definitions for increased consistency and usability. Version two includes more attention bits - these are now dealt with at a simple level.

- hw/lpc-mbox: Use message registers for interrupts

  Currently the BMC raises the interrupt using the BMC control register. It does so on all accesses to the 16 'data' registers meaning that when the BMC only wants to set the ATTN (on which we have interrupts enabled) bit we will also get a control register based interrupt.

  The solution here is to mask that interrupt permanantly and enable interrupts on the protocol defined 'response' data byte.

General fixes

- Reduce log level on non-error log messages

  90% of what we print isn't useful to a normal user. This dramatically reduces the amount of messages printed by OPAL in normal circumstances.

- init: Silence messages and call ourselves "OPAL"
- psi: Switch to ESB mode later

  There's an errata, if we switch to ESB mode before setting up the various ESB mode related registers, a pending interrupts can go wrong.

- lpc: Enable "new" SerIRQ mode
- hw/ipmi/ipmi-sel: missing newline in prlog warning
- p8-i2c OCC lock: fix locking in p9_i2c_bus_owner_change
- Convert important polling loops to spin at lowest SMT priority

  The pattern of calling cpu_relax() inside a polling loop does not suit the powerpc SMT priority instructions. Prefrred is to set a low priority then spin until break condition is reached, then restore priority.

- Improve cpu_idle when PM is disabled

  Split cpu_idle() into cpu_idle_delay() and cpu_idle_job() rather than requesting the idle type as a function argument. Have those functions provide a default polling (non-PM) implentation which spin at the lowest SMT priority.

- core/fdt: Always add a reserve map

  Currently we skip adding the reserved ranges block to the generated FDT blob if we are excluding the root node. This can result in a DTB that dtc will barf on because the reserved memory ranges overlap with the start of the dt_struct block. As an example:

```
$ fdtdump broken.dtb -d
/dts-v1/;
// magic:               0xd00dfeed
// totalsize:           0x7f3 (2035)
// off_dt_struct:       0x30  <----\
// off_dt_strings:      0x7b8       | this is bad!
// off_mem_rsvmap:      0x30  <----/
// version:             17
// last_comp_version:   16
// boot_cpuid_phys:     0x0
// size_dt_strings:     0x3b
// size_dt_struct:      0x788
```

```
/memreserve/ 0x100000000 0x300000004;
/memreserve/ 0x3300000001 0x169626d2c;
/memreserve/ 0x706369652d736c6f 0x7473000000000003;
        *continues*
```

With this patch:

```
$ fdtdump working.dtb -d
/dts-v1/;
// magic:               0xd00dfeed
// totalsize:           0x803 (2051)
// off_dt_struct:       0x40
// off_dt_strings:      0x7c8
// off_mem_rsvmap:      0x30
// version:             17
// last_comp_version:   16
// boot_cpuid_phys:     0x0
// size_dt_strings:     0x3b
// size_dt_struct:      0x788

// 0040: tag: 0x00000001 (FDT_BEGIN_NODE)
/ {
// 0048: tag: 0x00000003 (FDT_PROP)
// 07fb: string: phandle
// 0054: value
    phandle = <0x00000001>;
        *continues*
```

- hw/lpc-mbox: Use message registers for interrupts

  Currently the BMC raises the interrupt using the BMC control register. It does so on all accesses to the 16 'data' registers meaning that when the BMC only wants to set the ATTN (on which we have interrupts enabled) bit we will also get a control register based interrupt.

  The solution here is to mask that interrupt permanantly and enable interrupts on the protocol defined 'response' data byte.

PCI

- pci: Wait 20ms before checking presence detect on PCIe

  As the PHB presence logic has a debounce timer that can take a while to settle.

- phb3+iov: Fixup support for config space filters

  The filter should be called before the HW access and its return value control whether to perform the access or not

- core/pci: Use PCI slot's power facality in pci_enable_bridge()

  The current implmentation has incorrect assumptions: there is always a PCI slot associated with root port and PCIe switch downstream port and all of them are capable to change its power state by register PCICAP_EXP_SLOTCTL. Firstly, there might not a PCI slot associated with the root port or PCIe switch downstream port. Secondly, the power isn't controlled by standard config register (PCICAP_EXP_SLOTCTL). There are I2C slave devices used to control the power states on Tuleta.

  In order to use the PCI slot's methods to manage the power states, this does:

  – Introduce PCI_SLOT_FLAG_ENFORCE, indicates the request operation is enforced to be applied.

  - pci_enable_bridge() is split into 3 functions: pci_bridge_power_on() to power it on; pci_enable_bridge() as a place holder and pci_bridge_wait_link() to wait the downstream link to come up.
  - In pci_bridge_power_on(), the PCI slot's specific power management methods are used if there is a PCI slot associated with the PCIe switch downstream port or root port.

- platforms/astbmc/slots.c: Allow comparison of bus numbers when matching slots

  When matching devices on multiple down stream PLX busses we need to compare more than just the device-id of the PCIe BDFN, so increase the mask to do so.

Tests and simulators

- boot-tests: add OpenBMC support
- boot_test.sh: Add SMC BMC support

  Your BMC needs a special debug image flashed to use this, the exact image and methods aren't something I can publish here, but if you work for IBM or SMC you can find out from the right sources.

  A few things are needed to move around to be able to flash to a SMC BMC.

  For a start, the SSH daemon will only accept connections after a special incantation (which I also can't share), but you should put that in the ~/.skiboot_boot_tests file along with some other default login information we don't publicise too broadly (because Security Through Obscurity is *obviously* a good idea. . . .)

  We also can't just directly "ssh /bin/true", we need an expect script, and we can't scp, but we can anonymous rsync!

  You also need a pflash binary to copy over.

- hdata_to_dt: Add PVR overrides to the usage text
- mambo: Add a reservation for the initramfs

  On most systems the initramfs is loaded inside the part of memory reserved for the OS [0x0-0x30000000] and skiboot will never touch it. On mambo it's loaded at 0x80000000 and if you're unlucky skiboot can allocate over the top of it and corrupt the initramfs blob.

  There might be the downside that the kernel cannot re-use the initramfs memory since it's marked as reserved, but the kernel might also free it anyway.

- mambo: Update P9 PVR to reflect Scale out 24 core chips

  The P9 PVR bits 48:51 don't indicate a revision but instead different configurations. From BookIV we have:

Bits	Configuration
0	Scale out 12 cores
1	Scale out 24 cores
2	Scale up 12 cores
3	Scale up 24 cores

  Skiboot will mostly the use "Scale out 24 core" configuration (ie. SMT4 not SMT8) so reflect this in mambo.

- core: Move enable_mambo_console() into chip initialisation

  Rather than having a wart in main_cpu_entry() that initialises the mambo console, we can move it into init_chips() which is where we discover that we're on mambo.

- mambo: Create multiple chips when we have multiple CPUs

Currently when we boot mambo with multiple CPUs, we create multiple CPU nodes in the device tree, and each claims to be on a separate chip.

However we don't create multiple xscom nodes, which means skiboot only knows about a single chip, and all CPUs end up on it. At the moment mambo is not able to create multiple xscom controllers. We can create fake ones, just by faking the device tree up, but that seems uglier than this solution.

So create a mambo-chip for each CPU other than 0, to tell skiboot we want a separate chip created. This then enables Linux to see multiple chips:

```
smp: Brought up 2 nodes, 2 CPUs
numa: Node 0 CPUs: 0
numa: Node 1 CPUs: 1
```

- chip: Add support for discovering chips on mambo

  Currently the only way for skiboot to discover chips is by looking for xscom nodes. But on mambo it's currently not possible to create multiple xscom nodes, which means we can only simulate a single chip system.

  However it seems we can fairly cleanly add support for a special mambo chip node, and use that to instantiate multiple chips.

  Add a check in init_chip() that we're not clobbering an already initialised chip, now that we have two places that initialise chips.

- mambo: Make xscom claim to be DD 2.0

  In the mambo tcl we set the CPU version to DD 2.0, because mambo is not bug compatible with DD 1.

  But in xscom_read_cfam_chipid() we have a hard coded value, to work around the lack of the f000f register, which claims to be P9 DD 1.0.

  This doesn't seem to cause crashes or anything, but at boot we do see:

```
[    0.003893084,5] XSCOM: chip 0x0 at 0x1a0000000000 [P9N DD1.0]
```

  So fix it to claim that the xscom is also DD 2.0 to match the CPU.

- mambo: Match whole string when looking up symbols with linsym/skisym

  linsym/skisym use a regex to match the symbol name, and accepts a partial match against the entry in the symbol map, which can lead to somewhat confusing results, eg:

```
systemsim % linsym early_setup
0xc000000000027890
systemsim % linsym early_setup$
0xc00000000aa8054
systemsim % linsym early_setup_secondary
0xc000000000027890
```

  I don't think that's the behaviour we want, so append a $ to the name so that the symbol has to match against the whole entry, eg:

```
systemsim % linsym early_setup
0xc00000000aa8054
```

- Disable nap on P8 Mambo, public release has bugs

- mambo: Allow loading multiple CPIOs

  Currently we have support for loading a single CPIO and telling Linux to use it as the initrd. But the Linux code actually supports having multiple CPIOs contiguously in memory, between initrd-start and end, and will unpack

them all in order. That is a really nice feature as it means you can have a base CPIO with your root filesystem, and then tack on others as you need for various tests etc.

So expand the logic to handle SKIBOOT_INITRD, and treat it as a comma separated list of CPIOs to load. I chose comma as it's fairly rare in filenames, but we could make it space, colon, whatever. Or we could add a new environment variable entirely. The code also supports trimming whitespace from the values, so you can have "cpio1, cpio2".

- hdata/test: Add memory reservations to hdata_to_dt

  Currently memory reservations are parsed, but since they are not processed until mem_region_init() they don't appear in the output device tree blob. Several bugs have been found with memory reservations so we want them to be part of the test output.

  Add them and clean up several usages of printf() since we want only the dtb to appear in standard out.

IBM FSP systems

- FSP/CONSOLE: Fix possible NULL dereference
- platforms/ibm-fsp/firenze: Fix PCI slot power-off pattern

  When powering off the PCI slot, the corresponding bits should be set to 0bxx00xx00 instead of 0bxx11xx11. Otherwise, the specified PCI slot can't be put into power-off state. Fortunately, it didn't introduce any side-effects so far.

- FSP/CONSOLE: Workaround for unresponsive ipmi daemon

  We use TCE mapped area to write data to console. Console header (fsp_serbuf_hdr) is modified by both FSP and OPAL (OPAL updates next_in pointer in fsp_serbuf_hdr and FSP updates next_out pointer).

  Kernel makes opal_console_write() OPAL call to write data to console. OPAL write data to TCE mapped area and sends MBOX command to FSP. If our console becomes full and we have data to write to console, we keep on waiting until FSP reads data.

  In some corner cases, where FSP is active but not responding to console MBOX message (due to buggy IPMI) and we have heavy console write happening from kernel, then eventually our console buffer becomes full. At this point OPAL starts sending OPAL_BUSY_EVENT to kernel. Kernel will keep on retrying. This is creating kernel soft lockups. In some extreme case when every CPU is trying to write to console, user will not be able to ssh and thinks system is hang.

  If we reset FSP or restart IPMI daemon on FSP, system recovers and everything becomes normal.

  This patch adds workaround to above issue by returning OPAL_HARDWARE when cosole is full. Side effect of this patch is, we may endup dropping latest console data. But better to drop console data than system hang.

- FSP: Set status field in response message for timed out message

  For timed out FSP messages, we set message status as "fsp_msg_timeout". But most FSP driver users (like surviellance) are ignoring this field. They always look for FSP returned status value in callback function (second byte in word1). So we endup treating timed out message as success response from FSP.

  Sample output:

```
[69902.432509048,7] SURV: Sending the heartbeat command to FSP
[70023.226860117,4] FSP: Response from FSP timed out, word0 = d66a00d7, word1 = 0␣
↪state: 3
....
[70023.226901445,7] SURV: Received heartbeat acknowledge from FSP
[70023.226903251,3] FSP: fsp_trigger_reset() entry
```

Here SURV code thought it got valid response from FSP. But actually we didn't receive response from FSP.

This patch fixes above issue by updating status field in response structure.

- FSP: Improve timeout message
- FSP/RTC: Fix possible FSP R/R issue in rtc write path
- hw/fsp/rtc: read/write cached rtc tod on fsp hir.

  Currently fsp-rtc reads/writes the cached RTC TOD on an fsp reset. Use latest fsp_in_rr() function to properly read the cached rtc value when fsp reset initiated by the hir.

  Below is the kernel trace when we set hw clock, when hir process starts.

```
[ 1727.775824] NMI watchdog: BUG: soft lockup - CPU#57 stuck for 23s!␣
↪[hwclock:7688]
[ 1727.775856] Modules linked in: vmx_crypto ibmpowernv ipmi_powernv uio_pdrv_
↪genirq ipmi_devintf powernv_op_panel uio ipmi_msghandler powernv_rng leds_
↪powernv ip_tables x_tables autofs4 ses enclosure scsi_transport_sas crc32c_
↪vpmsum lpfc ipr tg3 scsi_transport_fc
[ 1727.775883] CPU: 57 PID: 7688 Comm: hwclock Not tainted 4.10.0-14-generic #16-
↪Ubuntu
[ 1727.775883] task: c000000fdfdc8400 task.stack: c000000fdfef4000
[ 1727.775884] NIP: c00000000090540c LR: c00000000000846f4 CTR: 000000003006dd70
[ 1727.775885] REGS: c000000fdfef79a0 TRAP: 0901   Not tainted  (4.10.0-14-
↪generic)
[ 1727.775886] MSR: 9000000000009033 <SF,HV,EE,ME,IR,DR,RI,LE>
[ 1727.775889]   CR: 28024442  XER: 20000000
[ 1727.775890] CFAR: c00000000008472c SOFTE: 1
               GPR00: 0000000030005128 c000000fdfef7c20 c00000000144c900␣
↪fffffffffffffff4
               GPR04: 0000000028024442 c00000000090540c 9000000000009033␣
↪0000000000000000
               GPR08: 0000000000000000 0000000031fc4000 c000000000084710␣
↪9000000000001003
               GPR12: c00000000000846e8 c00000000fba0100
[ 1727.775897] NIP [c00000000090540c] opal_set_rtc_time+0x4c/0xb0
[ 1727.775899] LR [c00000000000846f4] opal_return+0xc/0x48
[ 1727.775899] Call Trace:
[ 1727.775900] [c000000fdfef7c20] [c00000000090540c] opal_set_rtc_time+0x4c/0xb0␣
↪(unreliable)
[ 1727.775901] [c000000fdfef7c60] [c000000000900828] rtc_set_time+0xb8/0x1b0
[ 1727.775903] [c000000fdfef7ca0] [c000000000902364] rtc_dev_ioctl+0x454/0x630
[ 1727.775904] [c000000fdfef7d40] [c00000000035b1f4] do_vfs_ioctl+0xd4/0x8c0
[ 1727.775906] [c000000fdfef7de0] [c00000000035bab4] SyS_ioctl+0xd4/0xf0
[ 1727.775907] [c000000fdfef7e30] [c00000000000b184] system_call+0x38/0xe0
[ 1727.775908] Instruction dump:
[ 1727.775909] f821ffc1 39200000 7c832378 91210028 38a10020 39200000 38810028␣
↪f9210020
[ 1727.775911] 4bfffe6d e8810020 80610028 4b77f61d <60000000> 7c7f1b78 3860000a␣
↪2fbffff4
```

This is found when executing the testcase https://github.com/open-power/op-test-framework/blob/master/testcases/fspresetReload.py

With this fix ran fsp hir torture testcase in the above test which is working fine.

- occ: Set return variable to correct value

  When entering this section of code rc will be zero. If fsp_mkmsg() fails the code responsible for printing an error message won't be set. Resetting rc should allow for the error case to trigger if fsp_mkmsg fails.

- capp: Fix hang when CAPP microcode LID is missing on FSP machine

  When the LID is absent, we fail early with an error from start_preload_resource. In that case, capp_ucode_info.load_result isn't set properly causing a subsequent capp_lid_download() to call wait_for_resource_loaded() on something that isn't being loaded, thus hanging.

- FSP: Add check to detect FSP R/R inside fsp_sync_msg()

  OPAL sends MBOX message to FSP and updates message state from fsp_msg_queued -> fsp_msg_sent. fsp_sync_msg() queues message and waits until we get response from FSP. During FSP R/R we move outstanding MBOX messages from msgq to rr_queue including inflight message (fsp_reset_cmdclass()). But we are not resetting inflight message state.

  In extreme croner case where we sent message to FSP via fsp_sync_msg() path and FSP R/R happens before getting respose from FSP, then we will endup waiting in fsp_sync_msg() until everything becomes normal.

  This patch adds fsp_in_rr() check to fsp_sync_msg() and return error to caller if FSP is in R/R.

- FSP: Add check to detect FSP R/R inside fsp_sync_msg()

  OPAL sends MBOX message to FSP and updates message state from fsp_msg_queued -> fsp_msg_sent. fsp_sync_msg() queues message and waits until we get response from FSP. During FSP R/R we move outstanding MBOX messages from msgq to rr_queue including inflight message (fsp_reset_cmdclass()). But we are not resetting inflight message state.

  In extreme croner case where we sent message to FSP via fsp_sync_msg() path and FSP R/R happens before getting respose from FSP, then we will endup waiting in fsp_sync_msg() until everything becomes normal.

  This patch adds fsp_in_rr() check to fsp_sync_msg() and return error to caller if FSP is in R/R.

- capp: Fix hang when CAPP microcode LID is missing on FSP machine

  When the LID is absent, we fail early with an error from start_preload_resource. In that case, capp_ucode_info.load_result isn't set properly causing a subsequent capp_lid_download() to call wait_for_resource_loaded() on something that isn't being loaded, thus hanging.

- FSP/CONSOLE: Do not free fsp_msg in error path

  as we reuse same msg to send next output message.

- platform/zz: Acknowledge OCC_LOAD mbox message in ZZ

  In P9 FSP box, OCC image is pre-loaded. So do not handle the load command and send SUCCESS to FSP on recieving OCC_LOAD mbox message.

- FSP/RTC: Improve error log

astbmc systems

- platforms/astbmc: Don't validate model on palmetto

  The platform isn't compatible with palmetto until the root device-tree node's "model" property is NULL or "palmetto". However, we could have "TN71-BP012" for the property on palmetto.

```
linux# cat /proc/device-tree/model
TN71-BP012
```

  This skips the validation on root device-tree node's "model" property on palmetto, meaning we check the "compatible" property only.

4.1.65 skiboot-5.7-rc2

skiboot v5.7-rc2 was released on Thursday July 13th 2017. It is the second release candidate of skiboot 5.7, which will become the new stable release of skiboot following the 5.6 release, first released 24th May 2017.

skiboot v5.7-rc2 contains all bug fixes as of *skiboot-5.4.6* and *skiboot-5.1.19* (the currently maintained stable releases). We do not currently expect to do any 5.6.x stable releases.

For how the skiboot stable releases work, see *Skiboot stable tree rules and releases* for details.

The current plan is to cut the final 5.7 in the next week or so, with skiboot 5.7 being for all POWER8 and POWER9 platforms in op-build v1.18 (due July 12th, but will come *after* skiboot 5.7).

This is the second release using the new regular six week release cycle, similar to op-build, but slightly offset to allow for a short stabilisation period. Expected release dates and contents are tracked using GitHub milestone and issues: https://github.com/open-power/skiboot/milestones

Over *skiboot-5.7-rc1*, we have the following changes:

POWER9

There are many important changes for POWER9 DD1 and DD2 systems. POWER9 support should be considered in development and skiboot 5.7 is certainly **NOT** suitable for POWER9 production environments.

- HDAT: Add IPMI sensor data under /bmc node
- numa/associativity: Add a new level of NUMA for GPU's

  Today we have an issue where the NUMA nodes corresponding to GPU's have the same affinity/distance as normal memory nodes. Our reference-points today supports two levels [0x4, 0x4] for normal systems and [0x4, 0x3] for Power8E systems. This patch adds a new level [0x4, X, 0x2] and uses node-id as at all levels for the GPU.
- xive: Enable memory backing of queues

  This dedicates 6x64k pages of memory permanently for the XIVE to use for internal queue overflow. This allows the XIVE to deal with some corner cases where the internal queues might prove insufficient.
- xive: Properly get rid of donated indirect pages during reset

  Otherwise they keep being used accross kexec causing memory corruption in subsequent kernels once KVM has been used.
- cpu: Better handle unknown flags in opal_reinit_cpus()

  At the moment, if we get passed flags we don't know about, we return OPAL_UNSUPPORTED but we still perform whatever actions was requied by the flags we do support. Additionally, on P8, we attempt a SLW re-init which hasn't been supported since Murano DD2.0 and will crash your system.

  It's too late to fix on existing systems so Linux will have to be careful at least on P8, but to avoid future issues let's clean that up, make sure we only use slw_reinit() when HILE isn't supported.
- cpu: Unconditionally cleanup TLBs on P9 in opal_reinit_cpus()

  This can work around problems where Linux fails to properly cleanup part or all of the TLB on kexec.
- Fix scom addresses for power9 nx checkstop hmi handling.

  Scom addresses for NX status, DMA & ENGINE FIR and PBI FIR has changed for Power9. Fixup thoes while handling nx checkstop for Power9.

- Fix scom addresses for power9 core checkstop hmi handling.

  Scom addresses for CORE FIR (Fault Isolation Register) and Malfunction Alert Register has changed for Power9. Fixup those while handling core checkstop for Power9.

  Without this change HMI handler fails to check for correct reason for core checkstop on Power9.

- core/mem_region: check return value of add_region

  The only sensible thing to do if this fails is to abort() as we've likely just failed reserving reserved memory regions, and nothing good comes from that.

PHB4

- phb4: Do more retries on link training failures Currently we only retry once when we have a link training failure. This changes this to be 3 retries as 1 retry is not giving us enough reliablity.

  This will increase the boot time, especially on systems where we incorrectly detect a link presence when there really is nothing present. I'll post a followup patch to optimise our timings to help mitigate this later.

- phb4: Workaround phy lockup by doing full PHB reset on retry

  For PHB4 it's possible that the phy may end up in a bad state where it can no longer recieve data. This can manifest as the link not retraining. A simple PERST will not clear this. The PHB must be completely reset.

  This changes the retry state to CRESET to do this.

  This issue may also manifest itself as the link training in a degraded state (lower speed or narrower width). This patch doesn't attempt to fix that (will come later).

- pci: Add ability to trace timing

  PCI link training is responsible for a huge chunk of the skiboot boot time, so add the ability to trace it waiting in the main state machine.

- pci: Print resetting PHB notice at higher log level

  Currently during boot there a long delay while we wait for the PHBs to be reset and train. During this time, there is no output from skiboot and the last message doesn't give an indication of what's happening.

  This boosts the PHB reset message from info to notice so users can see what's happening during this long period of waiting.

- phb4: Only set one bit in nfir

  The MPIPL procedure says to only set bit 26 when forcing the PEC into freeze mode. Currently we set bits 24-27.

  This changes the code to follow spec and only set bit 26.

- phb4: Fix order of pfir/nfir clearing in CRESET

  According to the workbook, pfir must be cleared before the nfir. The way we have it now causes the nfir to not clear properly in some error circumstances.

  This swaps the order to match the workbook.

- phb4: Remove incorrect state transition

  When waiting in PHB4_SLOT_CRESET_WAIT_CQ for transations to end, we incorrectly move onto the next state. Generally we don't hit this as the transactions have ended already anyway.

  This removes the incorrect state transition.

- phb4: Set default lane equalisation

  Set default lane equalisation if there is nothing in the device-tree.

  Default value taken from hdat and confirmed by hardware team. Neatens the code up a bit too.

- hdata: Fix phb4 lane-eq property generation

  The lane-eq data we get from hdat is all 7s but what we end up in the device tree is:

```
xscom@603fc00000000/pbcq@4010c00/stack@0/ibm,lane-eq
                  00000000 31c339e0 00000000 0000000c
                  00000000 00000000 00000000 00000000
                  00000000 31c30000 77777777 77777777
                  77777777 77777777 77777777 77777777
```

  This fixes grabbing the properties from hdat and fixes the call to put them in the device tree.

- phb4: Fix PHB4 fence recovery.

  We had a few problems:

  - We used the wrong register to trigger the reset (spec bug)
  - We should clear the PFIR and NFIR while the reset is asserted
  - ... and in the right order !
  - We should only apply the DD1 workaround after the reset has been lifted.
  - We should ensure we use ASB whenever we are fenced or doing a CRESET
  - Make config ops write with ASB

- phb4: Verbose EEH options

  Enabled via nvram pci-eeh-verbose=true. ie.

```
nvram -p ibm,skiboot --update-config pci-eeh-verbose=true
```

- phb4: Print more info when PHB fences

  For now at PHBERR level. We don't have room in the diags data passed to Linux for these unfortunately.

Testing/development

- lpc: remove double LPC prefix from messages
- opal-ci/fetch-debian-jessie-installer: follow redirects Fixes some CI failures
- test/qemu-jessie: bail out fast on kernel panic
- test/qemu-jessie: dump boot log on failure
- travis: add fedora26
- xz: add fallthrough annotations to silence GCC7 warning

4.1.66 skiboot-5.8

skiboot v5.8 was released on Thursday August 31st 2017. It is the first release of skiboot 5.8, which becomes the new stable release. It follows the 5.7 release, first released 25th July 2017.

skiboot v5.8 contains all bug fixes as of *skiboot-5.4.6* and *skiboot-5.1.20* (the currently maintained stable releases). We do not currently expect to do any 5.7.x stable releases.

For how the skiboot stable releases work, see *Skiboot stable tree rules and releases* for details.

Over *skiboot-5.7*, we have the following changes:

New Features

- sensors: occ: Add support to clear sensor groups

  Adds a generic API to clear sensor groups. OCC inband sensor groups such as CSM, Profiler and Job Scheduler can be cleared using this API. It will clear the min/max of all sensors belonging to OCC sensor groups.

- sensors: occ: Add CSM_{min/max} sensors

  HWMON's lowest/highest attribute is used by CSM agent, so map min/max device-tree properties "sensor-data-min" and "sensor-data-max" to the min/max of CSM.

- sensors: occ: Add support for OCC inband sensors

  Add support to parse and export OCC inband sensors which are copied by OCC to main memory in P9. Each OCC writes three buffers which includes one names buffer for sensor meta data and two buffers for sensor readings. While OCC writes to one buffer the sensor values can be read from the other buffer. The sensors are updated every 100ms.

  This patch adds power, temperature, current and voltage sensors to `/ibm,opal/sensors` device-tree node which can be exported by the ibmpowernv-hwmon driver in Linux.

- psr: occ: Add support to change power-shifting-ratio

  Add support to set the CPU-GPU power shifting ratio which is used by the OCC power capping algorithm. PSR value of 100 takes all power away from CPU first and a PSR value of 0 caps GPU first.

- powercap: occ: Add a generic powercap framework

  This patch adds a generic powercap framework and exports OCC powercap sensors using which system powercap can be set inband through OPAL-OCC command-response interface.

- phb4: Enable PCI peer-to-peer

  P9 supports PCI peer-to-peer: a PCI device can write directly to the mmio space of another PCI device. It completely by-passes the CPU.

  It requires some configuration on the PHBs involved:

  1. on the initiating side, the address for the read/write operation is in the mmio space of the target, i.e. well outside the range normally allowed. So we disable range-checking on the TVT entry in bypass mode.
  2. on the target side, we need to explicitly enable p2p by setting a bit in a configuration register. It has the side-effect of reserving an outbound (as seen from the CPU) store queue for p2p. Therefore we only enable p2p on the PHBs using it, as we don't want to waste the resource if we don't have to.

  P9 supports p2p mmio writes. Reads are currently only supported if the two devices are under the same PHB but that is expected to change in the future, and it raises questions about intermediate switches configuration, so we report an error for the time being.

  The patch adds a new OPAL call to allow the OS to declare a p2p (initiator, target) pair.

- NX 842 and GZIP support on POWER9

POWER9 DD2

Further support for POWER9 DD2 revision chips. Notable changes include:

- xscom: Grab P9 DD2 revision level
- vas: Set mmio enable bits in DD2

  POWER9 DD2 added some new "enable" bits that must be set for VAS to work. These bits were unused in DD1.
- hdat: Add POWER9 DD2.0 specific pa_features

  Same as the default but with TM off.

POWER9

Since *skiboot-5.8-rc1*:

- hw/npu2.c: Add ibm,nvlink-speed device-tree property

  NVLink2 links can support multiple different speeds. However the device driver has no way of determining which speed was programmed so pass it down as a device tree property.
- hw/npu2-hw-procedures.c: Update PHY_RESET procedure

  Newer versions of Hostboot will have various clocks powered down by default to save power. Therefore we need to power them up before accessing the OBUS PHY.
- p8-i2c: Fix random data corruption (POWER9 specific) While waiting for the OCC to signal that it has finished using the I2C master we put the master into the, poorly named, occache_dis state. While in this state the transaction hasn't been started, but p8_i2c_check_status() will only skip it's checks when the master is in the idle state. Any action that checks that cranks the I2C state machine (interrupt, poll, etc) will call p8_i2c_check_status() and since the master is not idle, it will check the status register, see the transaction complete flag set and complete the i2c request without actually doing anything.

  If the transaction was a I2C read, the resulting output will be a zeroed data buffer.
- hw/p8-i2c: Fix OCC locking (POWER9 specific)

  There's a few issues with the Host<->OCC I2C bus handshaking. First up, skiboot is currently examining the wrong bit when checking if the OCC is currently using the bus. Secondly, when we need to wait for the OCC to release the bus we are scheduling a recovery timer to run zero timebase ticks after the current moment so the recovery timeout handler will run immediately after the bus was requested, which will in turn re-schedule itself, etc, etc. There's also a race between the OCC interrupt and the recovery handler which can result in an assertion failure in the recovery thread. All of this is bad.

  This patch addresses all these issues and sets the recovery timeout to 10ms.
- vas: export chip-id to vas platform device This is needed so VAS in the kernel can perform cpu to vas id mapping.
- slw: Modify the power9 stop0_lite latency & residency

  Currently skiboot exposes the exit-latency for stop0_lite as 200ns and the target-residency to be 2us.

  However, the kernel cpu-idle infrastructure rounds up the latency to microseconds and lists the stop0_lite latency as 0us, putting it on par with snooze state. As a result, when the predicted latency is small (< 1us), cpuidle will select stop0_lite instead of snooze. The difference between these states is that snooze doesn't require an interrupt to exit from the state, but stop0_lite does. And the value 200ns doesn't include the interrupt latency.

  This shows up in the context_switch2 benchmark (http://ozlabs.org/~anton/junkcode/context_switch2.c) where the number of context switches per second with the stop0_lite disabled is found to be roughly 30% more than

with stop0_lite enabled. This can be correlated with the number of times cpuidle enters stop0_lite compared to snooze.

Hence, bump up the exit latency of stop0_lite to 1us. Since the target residency is chosen to be 10 times the exit latency, set the target residency to 10us.

With these values, we see a 50% improvement in the number of context switches.

Since *skiboot-5.7*:

- Base NPU2 support on POWER9 DD2
- hdata/i2c: Work around broken I2C array version

  Work around a bug in the I2C devices array that shows the array version as being v2 when only the v1 data is populated.

- Recognize the 2s2u zz platform

  OPAL currently doesn't know about the 2s2u zz. It recognizes such a box as a generic BMC machine and fails to boot. Add the 2s2u as a supported platform.

  There will subsequently be a 2s2u-L system which may have a different compatible property, which will need to be handled later.

- hdata/spira: POWER9 NX isn't software compatible with P7/P8 NX, don't claim so
- NX: Add P9 NX support for gzip compression engine

  Power 9 introduces NX gzip compression engine. This patch adds gzip compression support in NX. Virtual Accelerator Switch (VAS) is used to access NX gzip engine and the channel configuration will be done with the receive FIFO. So RxFIFO address, logical partition ID (lpid), process ID (pid) and thread ID (tid) are used to configure RxFIFO. P9 NX supports high and normal priority FIFOS. Skiboot configures User Mode Access Control (UMAC) noitify match register with these values and also enables other registers to enable / disable the engine.

  Creates the following device-tree entries to provide RxFIFO address, RxFIFO size, Fifo priority, lpid, pid and tid values so that kernel can drive P9 NX gzip engine.

  The following nodes are located under an xscom node: ::

  /xscom@<xscom_addr>/nx@<nx_addr>

  /ibm,gzip-high-fifo : High priority gzip RxFIFO /ibm,gzip-normal-fifo : Normal priority gzip RxFIFO

  Each RxFIFO node contain:s

  compatible ibm,p9-nx-gzip

  priority High or Normal

  rx-fifo-address RxFIFO address

  rx-fifo-size RxFIFO size

  lpid 0xfff (1's for 12 bits in UMAC notify match register)

  pid gzip coprocessor type

  tid counter for gzip

- NX: Add P9 NX support for 842 compression engine

  This patch adds changes needed for 842 compression engine on power 9. Virtual Accelerator Switch (VAS) is used to access NX 842 engine on P9 and the channel setup will be done with receive FIFO. So RxFIFO address, logical partition ID (lpid), process ID (pid) and thread ID (tid) are used for this setup. p9 NX supports high

and normal priority FIFOs. skiboot is not involved to process data with 842 engine, but configures User Mode Access Control (UMAC) noitify match register with these values and export them to kernel with device-tree entries.

Also configure registers to setup and enable / disable the engine with the appropriate registers. Creates the following device-tree entries to provide RxFIFO address, RxFIFO size, Fifo priority, lpid, pid and tid values so that kernel can drive P9 NX 842 engine.

The following nodes are located under an xscom node: `/xscom@<xscom_addr>/nx@<nx_addr>`

/ibm,842-high-fifo High priority 842 RxFIFO

/ibm,842-normal-fifo Normal priority 842 RxFIFO

Each RxFIFO node contains:

compatible ibm,p9-nx-842

priority High or Normal

rx-fifo-address RxFIFO address

rx-fifo-size RXFIFO size

lpid 0xfff (1's for 12 bits set in UMAC notify match register)

pid 842 coprocessor type

tid Counter for 842

- vas: Create MMIO device tree node

  Create a device tree node for VAS and add properties that Linux will need to configure/use VAS.

- opal: Extract sw checkstop fir address from HDAT.

  Extract sw checkstop fir address info from HDAT and populate device tree node ibm,sw-checkstop-fir.

  This patch is required for OPAL_CEC_REBOOT2 OPAL call to work as expected on p9.

  With this patch a device property 'ibm,sw-checkstop-fir' is now properly populated:

```
# lsprop ibm,sw-checkstop-fir
ibm,sw-checkstop-fir
                 05012000 0000001f
```

PHB4

- hdat: Fix PCIe GEN4 lane-eq setting for DD2

  For PCIe GEN4, DD2 uses only 1 byte per PCIe lane for the lane-eq settings (DD1 uses 2 bytes)

- pci: Wait for CRS and switch link when restoring bus numbers

  When a complete reset occurs, after the PHB recovers it propagates a reset down the wire to every device. At the same time, skiboot talks to every device in order to restore the state of devices to what they were before the reset.

  In some situations, such as devices that recovered slowly and/or were behind a switch, skiboot attempted to access config space of the device before the link was up and the device could respond.

  Fix this by retrying CRS until the device responds correctly, and for devices behind a switch, making sure the switch has its link up first.

- pci: Track whether a PCI device is a virtual function

  This can be checked from config space, but we will need to know this when restoring the PCI topology, and it is not always safe to access config space during this period.

- phb4: Enhanced PCIe training tracing

  This add more details to the PCI training tracing (aka Rick Mata mode). It enables the PCIe Link Training and Status State Machine (LTSSM) tracing and details on speed and link width.

  Output now looks like this when enabled (via nvram):

```
[    1.096995141,3] PHB#0000[0:0]: TRACE:0x0000001101000000  0ms
↪GEN1:x16:detect
[    1.102849137,3] PHB#0000[0:0]: TRACE:0x0000102101000000 11ms presence␣
↪GEN1:x16:polling
[    1.104341838,3] PHB#0000[0:0]: TRACE:0x0000182101000000 14ms training␣
↪GEN1:x16:polling
[    1.104357444,3] PHB#0000[0:0]: TRACE:0x00001c5101000000 14ms training␣
↪GEN1:x16:recovery
[    1.104580394,3] PHB#0000[0:0]: TRACE:0x00001c5103000000 14ms training␣
↪GEN3:x16:recovery
[    1.123259359,3] PHB#0000[0:0]: TRACE:0x00001c5104000000 51ms training␣
↪GEN4:x16:recovery
[    1.141737656,3] PHB#0000[0:0]: TRACE:0x0000144104000000 87ms presence␣
↪GEN4:x16:L0
[    1.141752318,3] PHB#0000[0:0]: TRACE:0x0000154904000000 87ms trained ␣
↪GEN4:x16:L0
[    1.141757964,3] PHB#0000[0:0]: TRACE: Link trained.
[    1.096834019,3] PHB#0001[0:1]: TRACE:0x0000001101000000  0ms
↪GEN1:x16:detect
[    1.105578525,3] PHB#0001[0:1]: TRACE:0x0000102101000000 17ms presence␣
↪GEN1:x16:polling
[    1.112763075,3] PHB#0001[0:1]: TRACE:0x0000183101000000 31ms training␣
↪GEN1:x16:config
[    1.112778956,3] PHB#0001[0:1]: TRACE:0x00001c5081000000 31ms training␣
↪GEN1:x08:recovery
[    1.113002083,3] PHB#0001[0:1]: TRACE:0x00001c5083000000 31ms training␣
↪GEN3:x08:recovery
[    1.114833873,3] PHB#0001[0:1]: TRACE:0x0000144083000000 35ms presence␣
↪GEN3:x08:L0
[    1.114848832,3] PHB#0001[0:1]: TRACE:0x0000154883000000 35ms trained ␣
↪GEN3:x08:L0
[    1.114854650,3] PHB#0001[0:1]: TRACE: Link trained.
```

- phb4: Fix reading wrong size registers in EEH dump

  These registers are supposed to be 16bit, and it makes part of the register dump misleading.

- phb4: Ignore slot state if performing complete reset

  If a PHB is being completely reset, its state is about to be blown away anyway, so if it's not in an appropriate state, creset it regardless.

- phb4: Prepare for link down when creset called from kernel

  phb4_creset() is typically called by functions that prepare the link to go down. In cases where creset() is called directly by the kernel, this isn't the case and it can cause issues. Prepare for link down in creset, just like we do in freset and hreset.

- phb4: Skip attempting to fix PHBs broken on boot

If a PHB is marked broken it didn't work on boot, and if it didn't work on boot then there's no point trying to recover it later

- phb4: Fix duplicate in EEH register dump
- phb4: Be more conservative on link presence timeout

  In this patch we tuned our link timing to be more agressive: `cf960e2884 phb4: Improve reset and link training timing`

  Cards should take only 32ms but unfortunately we've seen some take up to 440ms. Hence bump our timer up to 1000ms.

  This can hurt boot times on systems where slots indicate a hotplug status but no electrical link is present (which we've seen). Since we have to wait 1 second between PERST and touching config space anyway, it shouldn't hurt too much.
- phb4: Assert PERST before PHB reset

  Currently we don't assert PERST before issuing a PHB reset. This means any link issues while resetting the PHB will be logged as errors.

  This asserts PERST before we start resetting the PHB to avoid this.
- Revert "phb4: Read PERST signal rather than assuming it's asserted"

  This reverts commit b42ff2b904165addf32e77679cebb94a08086966

  The original patch assumes that PERST has been asserted well before (> 250ms) we hit here (ie. during host-boot).

  In a subesquent patch this will no longer be the case as we need to assert PERST during PHB reset, which may only be a few milliseconds before we hit this code.

  Hence revert this patch. Go back to the software mechanism using skip_perst to determine if PERST should be asserted or not. This allows us to keep the speed optimisation on boot.
- phb4: Set REGB error enables based on link state

  Currently we always set these enables when initing the PHB. If the link is already down, we shouldn't set them as it may cause spurious errors.

  This changes the code to only sets them if the link is up.
- phb4: Mark PHB as fenced on creset

  If we have to inject an error to trigger recover, we end up not marking the PHB as fenced in the PHB struct. This fixes that.
- phb4: Clear errors before deasserting reset

  During reset we may have logged some errors (eg. due to the link going down).

  Hence before we deassert PERST or Hot Reset, we need to clear these errors. This ensures that once link training starts, only new errors are logged.
- phb4: Disable device config space access when fenced

  On DD2 you can't access device config space when fenced, so just disable access whenever we are fenced.
- phb4: Dump devctl and devstat registers

  Dump devctl and devstat registers. These would have been useful when debugging the MPS issue.
- phb4: Only clear some PHB config space registers on errors

Currently on error we clear the entire PHB config space. This is a problem as the PCIe Maximum Payload Size (MPS) negotiation may have already occurred. Clearing MPS in the PHB back to a default of 128 bytes will result an error for a device which already has a larger MPS configured.

This will manifest itself as error due to a malformed TLP packet. ie. `phbPblErrorStatus bit 41 = "Malformed TLP error"`

This has been seen after kexec on with some adapters.

This fixes the problem by only clearing a subset of registers on a phb error.

Utilities

- external/xscom-utils: Add `--list-bits`

  When using getscom/putscom it's helpful to know what bits are set in the register. This patch adds an option to print out which bits are set along with the value that was read/written to the register. Note that this output indicates which bits are set using the IBM bit ordering since that's what the XSCOM documentation uses.

opal-prd

- opal-prd: Do not pass pnor file while starting daemon.

  This change to the included systemd init file means opal-prd can start and run on IBM FSP based systems.

  We do not have pnor support on all the system. Also we have logic to autodetect PNOR. Hence do not pass `--pnor` by default.

- opal-prd: Disable pnor access interface on FSP system

  On FSP system host does not have access to PNOR. Hence disable PNOR access interfaces.

OPAL Sensors

- sensor-groups : occ: Add 'ops' DT property

  Add new device-tree property 'ops' to define different operations supported on each sensor-group.

- OCC: Map OCC sensor to a chip-id

  Parse device tree to get chip-id for OCC sensor.

- HDAT: Add chip-id property to ipmi sensors

  Presently we do not have a way to map sensor to chip id. Hence we are always passing chip id 0 for occ_reset request (see occ_sensor_id_to_chip()).

  This patch adds chip-id property to sensors (whenever its available) so that we can map occ sensor to chip-id and pass valid chip-id to occ_reset request.

- xive: Check for valid PIR index when decoding

  This fixes an unlikely but possible assert() fail on kdump.

- sensors: occ: Skip the deconfigured core sensors

  This patch skips the deconfigured cores from the core sensors while parsing the sensor names in the main memory as these sensor values are not updated by OCC.

IBM FSP systems

Since *skiboot-5.8-rc1*:

- mktime: fix off-by-one error calling days_in_month

  From auditing all the mktime() users, there seems to be only a *very* small window around new years day where we could possibly return incorrect data to the OS, and even then, there would have to be FSP reset/reload on FSP machines. I don't *think* there's an opportunity on other machines.

Tests

Since *skiboot-5.8-rc1*:

- travis: Debian Stretch must pass
- test kernels: link with -N
- core/test/run-msg: don't depend on unittest mem layout

Since *skiboot-5.7*:

- hdata_to_dt: use a realistic PVR and chip revision
- nx: PR_INFO that NX RNG and Crypto not yet supported on POWER9
- external/pflash: Add tests
- external/pflash: Reinstate the progress bars

  Recent work did some optimising which unfortunately removed some of the progress bars in pflash.

  It turns out that there's only one thing people prefer to correctly programmed flash chips, it is the ability to watch little equals characters go across their screens for potentially minutes.

- external/pflash: Correct erase alignment checks

  pflash should check the alignment of addresses and sizes when asked to erase. There are two possibilities:

  1. The user has specified sizes manually in which case pflash should be as flexible as possible, block-level_smart_erase() permits this. To prevent possible mistakes pflash will require –force to perform a manual erase of unaligned sizes.
  2. The user used -P to specify a partition, partitions aren't necessarily erase granule aligned anymore, block-level_smart_erase() can handle. In this it doesn't make sense to warn/error about misalignment since the misalignment is inherent to the FFS partition and not really user input.

- external/pflash: Check the result of strtoul

  Also add 0x in front of –info output to avoid a copy and paste mistake.

- libflash/file: Break up MTD erase ioctl() calls

  Unfortunately not all drivers are created equal and several drivers on which pflash relies block in the kernel for quite some time and ignore signals.

  This is really only a problem if pflash is to perform large erases. So don't, perform these ops in small chunks.

  An in kernel fix is possible in most cases but it takes time and systems will be running older drivers for quite some time. Since sector erases aren't significantly slower than whole chip erases there isn't much of a performance penalty to breaking up the erase ioctl()s.

General

Since *skiboot-5.8-rc1*:

- gcov: support GCC 7.1+
- Tests build and pass on Debian A few things related to the Debian toolchain.

Since *skiboot-5.7*:

- opal-msg: Increase the max-async completion count by max chips possible
- occ: Add support for OPAL-OCC command/response interface

  This patch adds support for a shared memory based command/response interface between OCC and OPAL. In HOMER, there is an OPAL command buffer and an OCC response buffer which is used to send inband commands to OCC.
- HDAT/device-tree: only add lid-type on pre-POWER9 systems

  Largely a relic of back when we had multiple entry points into OPAL depending on which mechanism on an FSP we were using to get loaded, this isn't needed on modern P9 as we only have one entry point (we don't do the PHYP LID hack).

Contributors

- Processed 156 csets from 17 developers
- 1 employers found
- A total of 6888 lines added, 1089 removed (delta 5799)

Developers with the most changesets

Developer	#	%
Cyril Bur	35	(22.4%)
Stewart Smith	32	(20.5%)
Michael Neuling	23	(14.7%)
Sukadev Bhattiprolu	11	(7.1%)
Reza Arbab	10	(6.4%)
Russell Currey	9	(5.8%)
Shilpasri G Bhat	9	(5.8%)
Oliver O'Halloran	5	(3.2%)
Haren Myneni	5	(3.2%)
Alistair Popple	4	(2.6%)
Vasant Hegde	4	(2.6%)
Nicholas Piggin	3	(1.9%)
Andrew Donnellan	2	(1.3%)
Gautham R. Shenoy	1	(0.6%)
Mahesh Salgaonkar	1	(0.6%)
Ananth N Mavinakayanahalli	1	(0.6%)
Frederic Barrat	1	(0.6%)

Developers with the most changed lines

Developer	#	%
Shilpasri G Bhat	1935	(27.9%)
Cyril Bur	1868	(26.9%)
Stewart Smith	866	(12.5%)
Sukadev Bhattiprolu	663	(9.5%)
Haren Myneni	584	(8.4%)
Michael Neuling	384	(5.5%)
Frederic Barrat	168	(2.4%)
Reza Arbab	98	(1.4%)
Oliver O'Halloran	98	(1.4%)
Vasant Hegde	93	(1.3%)
Alistair Popple	77	(1.1%)
Russell Currey	60	(0.9%)
Mahesh Salgaonkar	28	(0.4%)
Andrew Donnellan	11	(0.2%)
Gautham R. Shenoy	6	(0.1%)
Nicholas Piggin	4	(0.1%)
Ananth N Mavinakayanahalli	1	(0.0%)

Developers with the most signoffs

Developer	#	%
Stewart Smith	124	(97.6%)
Benjamin Herrenschmidt	2	(1.6%)
Vaidyanathan Srinivasan	1	(0.8%)
Total	127	(100%)

Developers with the most reviews

Developer	#	%
Samuel Mendoza-Jonas	19	(52.8%)
Andrew Donnellan	11	(30.6%)
Vasant Hegde	2	(5.6%)
Cédric Le Goater	1	(2.8%)
Russell Currey	1	(2.8%)
Reza Arbab	1	(2.8%)
Cyril Bur	1	(2.8%)
Total	36	(100%)

Developers with the most test credits

Developer	#	%
Vasant Hegde	1	(50.0%)
Hari Bathini	1	(50.0%)

Developers who gave the most tested-by credits

Developer	#	%
Russell Currey	1	(50.0%)
Mahesh Salgaonkar	1	(50.0%)

Developers with the most report credits

Developer	#	%
Anton Blanchard	1	(16.7%)
Mark Linimon	1	(16.7%)
Pavaman Subramaniyam	1	(16.7%)
Pridhiviraj Paidipeddi	1	(16.7%)
Rob Lippert	1	(16.7%)
Michael Neuling	1	(16.7%)

Developers who gave the most report credits

Developer	#	%
Stewart Smith	2	(33.3%)
Michael Neuling	1	(16.7%)
Andrew Donnellan	1	(16.7%)
Cyril Bur	1	(16.7%)
Gautham R. Shenoy	1	(16.7%)

4.1.67 skiboot-5.8-rc1

skiboot v5.8-rc1 was released on Tuesday August 22nd 2017. It is the first release candidate of skiboot 5.8, which will become the new stable release of skiboot following the 5.7 release, first released 25th July 2017.

skiboot v5.8-rc1 contains all bug fixes as of *skiboot-5.4.6* and *skiboot-5.1.20* (the currently maintained stable releases). We do not currently expect to do any 5.7.x stable releases.

For how the skiboot stable releases work, see *Skiboot stable tree rules and releases* for details.

The current plan is to cut the final 5.8 by August 25th, with skiboot 5.8 being for all POWER8 and POWER9 platforms in op-build v1.19 (Due August 25th). This is a short cycle as this release is mainly targetted towards POWER9 bringup efforts.

Over skiboot-5.7, we have the following changes:

New Features

- sensors: occ: Add support to clear sensor groups

  Adds a generic API to clear sensor groups. OCC inband sensor groups such as CSM, Profiler and Job Scheduler can be cleared using this API. It will clear the min/max of all sensors belonging to OCC sensor groups.

- sensors: occ: Add CSM_{min/max} sensors

  HWMON's lowest/highest attribute is used by CSM agent, so map min/max device-tree properties "sensor-data-min" and "sensor-data-max" to the min/max of CSM.

- sensors: occ: Add support for OCC inband sensors

  Add support to parse and export OCC inband sensors which are copied by OCC to main memory in P9. Each OCC writes three buffers which includes one names buffer for sensor meta data and two buffers for sensor readings. While OCC writes to one buffer the sensor values can be read from the other buffer. The sensors are updated every 100ms.

  This patch adds power, temperature, current and voltage sensors to `/ibm,opal/sensors` device-tree node which can be exported by the ibmpowernv-hwmon driver in Linux.

- psr: occ: Add support to change power-shifting-ratio

  Add support to set the CPU-GPU power shifting ratio which is used by the OCC power capping algorithm. PSR value of 100 takes all power away from CPU first and a PSR value of 0 caps GPU first.

- powercap: occ: Add a generic powercap framework

  This patch adds a generic powercap framework and exports OCC powercap sensors using which system powercap can be set inband through OPAL-OCC command-response interface.

- phb4: Enable PCI peer-to-peer

  P9 supports PCI peer-to-peer: a PCI device can write directly to the mmio space of another PCI device. It completely by-passes the CPU.

  It requires some configuration on the PHBs involved:

  1. on the initiating side, the address for the read/write operation is in the mmio space of the target, i.e. well outside the range normally allowed. So we disable range-checking on the TVT entry in bypass mode.
  2. on the target side, we need to explicitly enable p2p by setting a bit in a configuration register. It has the side-effect of reserving an outbound (as seen from the CPU) store queue for p2p. Therefore we only enable p2p on the PHBs using it, as we don't want to waste the resource if we don't have to.

  P9 supports p2p mmio writes. Reads are currently only supported if the two devices are under the same PHB but that is expected to change in the future, and it raises questions about intermediate switches configuration, so we report an error for the time being.

  The patch adds a new OPAL call to allow the OS to declare a p2p (initiator, target) pair.

- NX 842 and GZIP support on POWER9

POWER9 DD2

Further support for POWER9 DD2 revision chips. Notable changes include:

- xscom: Grab P9 DD2 revision level
- vas: Set mmio enable bits in DD2

  POWER9 DD2 added some new "enable" bits that must be set for VAS to work. These bits were unused in DD1.

- hdat: Add POWER9 DD2.0 specific pa_features

  Same as the default but with TM off.

POWER9

- Base NPU2 support on POWER9 DD2
- hdata/i2c: Work around broken I2C array version

  Work around a bug in the I2C devices array that shows the array version as being v2 when only the v1 data is populated.

- Recognize the 2s2u zz platform

  OPAL currently doesn't know about the 2s2u zz. It recognizes such a box as a generic BMC machine and fails to boot. Add the 2s2u as a supported platform.

  There will subsequently be a 2s2u-L system which may have a different compatible property, which will need to be handled later.

- hdata/spira: POWER9 NX isn't software compatible with P7/P8 NX, don't claim so
- NX: Add P9 NX support for gzip compression engine

  Power 9 introduces NX gzip compression engine. This patch adds gzip compression support in NX. Virtual Accelerator Switch (VAS) is used to access NX gzip engine and the channel configuration will be done with the receive FIFO. So RxFIFO address, logical partition ID (lpid), process ID (pid) and thread ID (tid) are used to configure RxFIFO. P9 NX supports high and normal priority FIFOS. Skiboot configures User Mode Access Control (UMAC) noitify match register with these values and also enables other registers to enable / disable the engine.

  Creates the following device-tree entries to provide RxFIFO address, RxFIFO size, Fifo priority, lpid, pid and tid values so that kernel can drive P9 NX gzip engine.

  The following nodes are located under an xscom node: ::

  > /xscom@<xscom_addr>/nx@<nx_addr>
  >
  > /ibm,gzip-high-fifo : High priority gzip RxFIFO /ibm,gzip-normal-fifo : Normal priority gzip RxFIFO

  Each RxFIFO node contain:s

  `compatible` `ibm,p9-nx-gzip`

  `priority` High or Normal

  `rx-fifo-address` RxFIFO address

  `rx-fifo-size` RxFIFO size

  `lpid` 0xfff (1's for 12 bits in UMAC notify match register)

  `pid` gzip coprocessor type

  `tid` counter for gzip

- NX: Add P9 NX support for 842 compression engine

  This patch adds changes needed for 842 compression engine on power 9. Virtual Accelerator Switch (VAS) is used to access NX 842 engine on P9 and the channel setup will be done with receive FIFO. So RxFIFO address, logical partition ID (lpid), process ID (pid) and thread ID (tid) are used for this setup. p9 NX supports high and normal priority FIFOs. skiboot is not involved to process data with 842 engine, but configures User Mode Access Control (UMAC) noitify match register with these values and export them to kernel with device-tree entries.

Also configure registers to setup and enable / disable the engine with the appropriate registers. Creates the following device-tree entries to provide RxFIFO address, RxFIFO size, Fifo priority, lpid, pid and tid values so that kernel can drive P9 NX 842 engine.

The following nodes are located under an xscom node: `/xscom@<xscom_addr>/nx@<nx_addr>`

/ibm,842-high-fifo High priority 842 RxFIFO

/ibm,842-normal-fifo Normal priority 842 RxFIFO

Each RxFIFO node contains:

compatible ibm,p9-nx-842

priority High or Normal

rx-fifo-address RxFIFO address

rx-fifo-size RXFIFO size

lpid 0xfff (1's for 12 bits set in UMAC notify match register)

pid 842 coprocessor type

tid Counter for 842

- vas: Create MMIO device tree node

  Create a device tree node for VAS and add properties that Linux will need to configure/use VAS.

- opal: Extract sw checkstop fir address from HDAT.

  Extract sw checkstop fir address info from HDAT and populate device tree node ibm,sw-checkstop-fir.

  This patch is required for OPAL_CEC_REBOOT2 OPAL call to work as expected on p9.

  With this patch a device property 'ibm,sw-checkstop-fir' is now properly populated:

```
# lsprop ibm,sw-checkstop-fir
ibm,sw-checkstop-fir
                 05012000 0000001f
```

PHB4

- hdat: Fix PCIe GEN4 lane-eq setting for DD2

  For PCIe GEN4, DD2 uses only 1 byte per PCIe lane for the lane-eq settings (DD1 uses 2 bytes)

- pci: Wait for CRS and switch link when restoring bus numbers

  When a complete reset occurs, after the PHB recovers it propagates a reset down the wire to every device. At the same time, skiboot talks to every device in order to restore the state of devices to what they were before the reset.

  In some situations, such as devices that recovered slowly and/or were behind a switch, skiboot attempted to access config space of the device before the link was up and the device could respond.

  Fix this by retrying CRS until the device responds correctly, and for devices behind a switch, making sure the switch has its link up first.

- pci: Track whether a PCI device is a virtual function

  This can be checked from config space, but we will need to know this when restoring the PCI topology, and it is not always safe to access config space during this period.

- phb4: Enhanced PCIe training tracing

  This add more details to the PCI training tracing (aka Rick Mata mode). It enables the PCIe Link Training and Status State Machine (LTSSM) tracing and details on speed and link width.

  Output now looks like this when enabled (via nvram):

```
[    1.096995141,3] PHB#0000[0:0]: TRACE:0x0000001101000000  0ms
↪GEN1:x16:detect
[    1.102849137,3] PHB#0000[0:0]: TRACE:0x0000102101000000 11ms presence␣
↪GEN1:x16:polling
[    1.104341838,3] PHB#0000[0:0]: TRACE:0x0000182101000000 14ms training␣
↪GEN1:x16:polling
[    1.104357444,3] PHB#0000[0:0]: TRACE:0x00001c5101000000 14ms training␣
↪GEN1:x16:recovery
[    1.104580394,3] PHB#0000[0:0]: TRACE:0x00001c5103000000 14ms training␣
↪GEN3:x16:recovery
[    1.123259359,3] PHB#0000[0:0]: TRACE:0x00001c5104000000 51ms training␣
↪GEN4:x16:recovery
[    1.141737656,3] PHB#0000[0:0]: TRACE:0x0000144104000000 87ms presence␣
↪GEN4:x16:L0
[    1.141752318,3] PHB#0000[0:0]: TRACE:0x0000154904000000 87ms trained ␣
↪GEN4:x16:L0
[    1.141757964,3] PHB#0000[0:0]: TRACE: Link trained.
[    1.096834019,3] PHB#0001[0:1]: TRACE:0x0000001101000000  0ms
↪GEN1:x16:detect
[    1.105578525,3] PHB#0001[0:1]: TRACE:0x0000102101000000 17ms presence␣
↪GEN1:x16:polling
[    1.112763075,3] PHB#0001[0:1]: TRACE:0x0000183101000000 31ms training␣
↪GEN1:x16:config
[    1.112778956,3] PHB#0001[0:1]: TRACE:0x00001c5081000000 31ms training␣
↪GEN1:x08:recovery
[    1.113002083,3] PHB#0001[0:1]: TRACE:0x00001c5083000000 31ms training␣
↪GEN3:x08:recovery
[    1.114833873,3] PHB#0001[0:1]: TRACE:0x0000144083000000 35ms presence␣
↪GEN3:x08:L0
[    1.114848832,3] PHB#0001[0:1]: TRACE:0x0000154883000000 35ms trained ␣
↪GEN3:x08:L0
[    1.114854650,3] PHB#0001[0:1]: TRACE: Link trained.
```

- phb4: Fix reading wrong size registers in EEH dump

  These registers are supposed to be 16bit, and it makes part of the register dump misleading.

- phb4: Ignore slot state if performing complete reset

  If a PHB is being completely reset, its state is about to be blown away anyway, so if it's not in an appropriate state, creset it regardless.

- phb4: Prepare for link down when creset called from kernel

  phb4_creset() is typically called by functions that prepare the link to go down. In cases where creset() is called directly by the kernel, this isn't the case and it can cause issues. Prepare for link down in creset, just like we do in freset and hreset.

- phb4: Skip attempting to fix PHBs broken on boot

  If a PHB is marked broken it didn't work on boot, and if it didn't work on boot then there's no point trying to recover it later

- phb4: Fix duplicate in EEH register dump

- phb4: Be more conservative on link presence timeout

  In this patch we tuned our link timing to be more agressive: `cf960e2884 phb4: Improve reset and link training timing`

  Cards should take only 32ms but unfortunately we've seen some take up to 440ms. Hence bump our timer up to 1000ms.

  This can hurt boot times on systems where slots indicate a hotplug status but no electrical link is present (which we've seen). Since we have to wait 1 second between PERST and touching config space anyway, it shouldn't hurt too much.

- phb4: Assert PERST before PHB reset

  Currently we don't assert PERST before issuing a PHB reset. This means any link issues while resetting the PHB will be logged as errors.

  This asserts PERST before we start resetting the PHB to avoid this.

- Revert "phb4: Read PERST signal rather than assuming it's asserted"

  This reverts commit b42ff2b904165addf32e77679cebb94a08086966

  The original patch assumes that PERST has been asserted well before (> 250ms) we hit here (ie. during host-boot).

  In a subesquent patch this will no longer be the case as we need to assert PERST during PHB reset, which may only be a few milliseconds before we hit this code.

  Hence revert this patch. Go back to the software mechanism using skip_perst to determine if PERST should be asserted or not. This allows us to keep the speed optimisation on boot.

- phb4: Set REGB error enables based on link state

  Currently we always set these enables when initing the PHB. If the link is already down, we shouldn't set them as it may cause spurious errors.

  This changes the code to only sets them if the link is up.

- phb4: Mark PHB as fenced on creset

  If we have to inject an error to trigger recover, we end up not marking the PHB as fenced in the PHB struct. This fixes that.

- phb4: Clear errors before deasserting reset

  During reset we may have logged some errors (eg. due to the link going down).

  Hence before we deassert PERST or Hot Reset, we need to clear these errors. This ensures that once link training starts, only new errors are logged.

- phb4: Disable device config space access when fenced

  On DD2 you can't access device config space when fenced, so just disable access whenever we are fenced.

- phb4: Dump devctl and devstat registers

  Dump devctl and devstat registers. These would have been useful when debugging the MPS issue.

- phb4: Only clear some PHB config space registers on errors

  Currently on error we clear the entire PHB config space. This is a problem as the PCIe Maximum Payload Size (MPS) negotiation may have already occurred. Clearing MPS in the PHB back to a default of 128 bytes will result an error for a device which already has a larger MPS configured.

  This will manifest itself as error due to a malformed TLP packet. ie. `phbPblErrorStatus bit 41 = "Malformed TLP error"`

This has been seen after kexec on with some adapters.

This fixes the problem by only clearing a subset of registers on a phb error.

Utilities

- external/xscom-utils: Add `--list-bits`

  When using getscom/putscom it's helpful to know what bits are set in the register. This patch adds an option to print out which bits are set along with the value that was read/written to the register. Note that this output indicates which bits are set using the IBM bit ordering since that's what the XSCOM documentation uses.

opal-prd

- opal-prd: Do not pass pnor file while starting daemon.

  This change to the included systemd init file means opal-prd can start and run on IBM FSP based systems.

  We do not have pnor support on all the system. Also we have logic to autodetect PNOR. Hence do not pass `--pnor` by default.

- opal-prd: Disable pnor access interface on FSP system

  On FSP system host does not have access to PNOR. Hence disable PNOR access interfaces.

OPAL Sensors

- sensor-groups : occ: Add 'ops' DT property

  Add new device-tree property 'ops' to define different operations supported on each sensor-group.

- OCC: Map OCC sensor to a chip-id

  Parse device tree to get chip-id for OCC sensor.

- HDAT: Add chip-id property to ipmi sensors

  Presently we do not have a way to map sensor to chip id. Hence we are always passing chip id 0 for occ_reset request (see occ_sensor_id_to_chip()).

  This patch adds chip-id property to sensors (whenever its available) so that we can map occ sensor to chip-id and pass valid chip-id to occ_reset request.

- xive: Check for valid PIR index when decoding

  This fixes an unlikely but possible assert() fail on kdump.

- sensors: occ: Skip the deconfigured core sensors

  This patch skips the deconfigured cores from the core sensors while parsing the sensor names in the main memory as these sensor values are not updated by OCC.

Tests

- hdata_to_dt: use a realistic PVR and chip revision
- nx: PR_INFO that NX RNG and Crypto not yet supported on POWER9
- external/pflash: Add tests

- external/pflash: Reinstate the progress bars

  Recent work did some optimising which unfortunately removed some of the progress bars in pflash.

  It turns out that there's only one thing people prefer to correctly programmed flash chips, it is the ability to watch little equals characters go across their screens for potentially minutes.

- external/pflash: Correct erase alignment checks

  pflash should check the alignment of addresses and sizes when asked to erase. There are two possibilities:

  1. The user has specified sizes manually in which case pflash should be as flexible as possible, block-level_smart_erase() permits this. To prevent possible mistakes pflash will require –force to perform a manual erase of unaligned sizes.
  2. The user used -P to specify a partition, partitions aren't necessarily erase granule aligned anymore, block-level_smart_erase() can handle. In this it doesn't make sense to warn/error about misalignment since the misalignment is inherent to the FFS partition and not really user input.

- external/pflash: Check the result of strtoul

  Also add 0x in front of –info output to avoid a copy and paste mistake.

- libflash/file: Break up MTD erase ioctl() calls

  Unfortunately not all drivers are created equal and several drivers on which pflash relies block in the kernel for quite some time and ignore signals.

  This is really only a problem if pflash is to perform large erases. So don't, perform these ops in small chunks.

  An in kernel fix is possible in most cases but it takes time and systems will be running older drivers for quite some time. Since sector erases aren't significantly slower than whole chip erases there isn't much of a performance penalty to breaking up the erase ioctl()s.

General

- opal-msg: Increase the max-async completion count by max chips possible
- occ: Add support for OPAL-OCC command/response interface

  This patch adds support for a shared memory based command/response interface between OCC and OPAL. In HOMER, there is an OPAL command buffer and an OCC response buffer which is used to send inband commands to OCC.

- HDAT/device-tree: only add lid-type on pre-POWER9 systems

  Largely a relic of back when we had multiple entry points into OPAL depending on which mechanism on an FSP we were using to get loaded, this isn't needed on modern P9 as we only have one entry point (we don't do the PHYP LID hack).

4.1.68 skiboot-5.9

skiboot v5.9 was released on Tuesday October 31st 2017. It is the first release of skiboot 5.9 and becomes the new stable release of skiboot following the 5.8 release, first released August 31st 2017. In this cyle we have had five release candidate releases, mostly centered around bug fixing for POWER9 platforms.

This release should be considered suitable for early-access POWER9 systems.

skiboot v5.9 contains all bug fixes as of *skiboot-5.4.8* and *skiboot-5.1.21* (the currently maintained stable releases). There may be some 5.9.x stable releases, depending on what issues are found.

For how the skiboot stable releases work, see *Skiboot stable tree rules and releases* for details.

Over *skiboot-5.8*, we have the following changes:

New Features

POWER8

- fast-reset by default (if possible)

  Currently, this is limited to POWER8 systems.

  A normal reboot will, rather than doing a full IPL, go through a fast reboot procedure. This reduces the "reboot to petitboot" time from minutes to a handful of seconds.

POWER9

Since *skiboot-5.9-rc3*:

- occ-sensors : Add OCC inband sensor region to exports (useful for debugging)

Two SRESET fixes (see below for feature description):

- core: direct-controls: Fix clearing of special wakeup

  'special_wakeup_count' is incremented on successfully asserting special wakeup. So we will never clear the special wakeup if we check 'special_wakeup_count' to be zero. Fix this issue by checking the 'special_wakeup_count' to 1 in dctl_clear_special_wakeup().

- core/direct-controls: increase special wakeup timeout on POWER9

  Some instances have been observed where the special wakeup assert times out. The current timeout is too short for deeper sleep states. Hostboot uses 100ms, so match that.

Since *skiboot-5.9-rc2*: - cpu: Add OPAL_REINIT_CPUS_TM_SUSPEND_DISABLED

  Add a new CPU reinit flag, "TM Suspend Disabled", which requests that CPUs be configured so that TM (Transactional Memory) suspend mode is disabled.

  Currently this always fails, because skiboot has no way to query the state. A future hostboot change will add a mechanism for skiboot to determine the status and return an appropriate error code.

Since *skiboot-5.8*:

- POWER9 power management during boot

  Less power should be consumed during boot.

- OPAL_SIGNAL_SYSTEM_RESET for POWER9

  This implements OPAL_SIGNAL_SYSTEM_RESET, using scom registers to quiesce the target thread and raise a system reset exception on it. It has been tested on DD2 with stop0 ESL=0 and ESL=1 shallow power saving modes.

  DD1 is not implemented because it is sufficiently different as to make support difficult.

- Enable deep idle states for POWER9
  - SLW: Add support for p9_stop_api

    p9_stop_api's are used to set SPR state on a core wakeup form a deeper low power state. p9_stop_api uses low level platform formware and self-restore microcode to restore the sprs to requested values.

Code is taken from : https://github.com/open-power/hostboot/tree/master/src/import/chips/p9/procedures/utils/stopreg

- SLW: Removing timebase related flags for stop4

  When a core enters stop4, it does not loose decrementer and time base. Hence removing flags OPAL_PM_DEC_STOP and OPAL_PM_TIMEBASE_STOP.

- SLW: Allow deep states if homer address is known

  Use a common variable has_wakeup_engine instead of has_slw to tell if the: - SLW image is populated in case of power8 - CME image is populated in case of power9

  Currently we expect CME to be loaded if homer address is known (except for simulators)

- SLW: Configure self-restore for HRMOR

  Make a stop api call using libpore to restore HRMOR register. HRMOR needs to be cleared so that when thread exits stop, they arrives at linux system_reset vector (0x100).

- SLW: Add opal_slw_set_reg support for power9

  This OPAL call is made from Linux to OPAL to configure values in various SPRs after wakeup from a deep idle state.

- PHB4: CAPP recovery

  CAPP recovery is initiated when a CAPP Machine Check is detected. The capp recovery procedure is initiated via a Hypervisor Maintenance interrupt (HMI).

  CAPP Machine Check may arise from either an error that results in a PHB freeze or from an internal CAPP error with CAPP checkstop FIR action. An error that causes a PHB freeze will result in the link down signal being asserted. The system continues running and the CAPP and PSL will be re-initialized.

  This implements CAPP recovery for POWER9 systems

- Add `wafer-location` property for POWER9

  Extract wafer-location from ECID and add property under xscom node. - bits 64:71 are the chip x location (7:0) - bits 72:79 are the chip y location (7:0)

  Sample output:

```
[root@wsp xscom@623fc00000000]# lsprop ecid
ecid              019a00d4 03100718 852c0000 00fd7911
[root@wsp xscom@623fc00000000]# lsprop wafer-location
wafer-location    00000085 0000002c
```

- Add `wafer-id` property for POWER9

  Wafer id is derived from ECID data. - bits 4:63 are the wafer id (ten 6 bit fields each containing a code)

  Sample output:

```
[root@wsp xscom@623fc00000000]# lsprop ecid
ecid              019a00d4 03100718 852c0000 00fd7911
[root@wsp xscom@623fc00000000]# lsprop wafer-id
wafer-id          "6Q0DG340SO"
```

- Add `ecid` property under `xscom` node for POWER9. Sample output:

```
[root@wsp xscom@623fc00000000]# lsprop ecid
ecid              019a00d4 03100718 852c0000 00fd7911
```

- Add ibm,firmware-versions device tree node

  In P8, hostboot provides mini device tree. It contains `/ibm,firmware-versions` node which has various firmware component version details.

  In P9, OPAL is building device tree. This patch adds support to parse VERSION section of PNOR and create `/ibm,firmware-versions` device tree node.

  Sample output:

```
/sys/firmware/devicetree/base/ibm,firmware-versions # lsprop .
occ              "6a00709"
skiboot          "v5.7-rc1-p344fb62"
buildroot        "2017.02.2-7-g23118ce"
capp-ucode       "9c73e9f"
petitboot        "v1.4.3-p98b6d83"
sbe              "02021c6"
open-power       "witherspoon-v1.17-128-gf1b53c7-dirty"
....
....
```

POWER9

Since *skiboot-5.9-rc5*:

- Suppress XSCOM chiplet-offline errors on P9

  Workaround on P9: PRD does operations it *knows* will fail with this error to work around a hardware issue where accesses via the PIB (FSI or OCC) work as expected, accesses via the ADU (what xscom goes through) do not. The chip logic will always return all FFs if there is any error on the scom.

- asm/head: initialize preferred DSCR value

  POWER7/8 use DSCR=0. POWER9 preferred value has "stride-N" enabled.

Since *skiboot-5.9-rc4*: - opal/hmi: Workaround Power9 hw logic bug for couple of TFMR TB errors. - opal/hmi: Fix TB reside and HDEC parity error recovery for power9

Since *skiboot-5.9-rc2*: - hw/imc: Fix IMC Catalog load for DD2.X processors

Since *skiboot-5.9-rc1*: - xive: Fix VP free block group mode false-positive parameter check

  The check to ensure the buddy allocation idx is aligned to its allocation order was not taking into account the allocation split. This would result in opal_xive_free_vp_block failures despite giving the same value as returned by opal_xive_alloc_vp_block.

  E.g., starting then stopping 4 KVM guests gives the following pattern in the host:

```
opal_xive_alloc_vp_block(5)=0x45000020
opal_xive_alloc_vp_block(5)=0x45000040
opal_xive_alloc_vp_block(5)=0x45000060
opal_xive_alloc_vp_block(5)=0x45000080
opal_xive_free_vp_block(0x45000020)=-1
opal_xive_free_vp_block(0x45000040)=0
opal_xive_free_vp_block(0x45000060)=-1
opal_xive_free_vp_block(0x45000080)=0
```

- hw/imc: pause microcode at boot

  IMC nest counters has both in-band (ucode access) and out of band access to it. Since not all nest counter configurations are supported by ucode, out of band tools are used to characterize other configuration.

So it is prefer to pause the nest microcode at boot to aid the nest out of band tools. If the ucode not paused and OS does not have IMC driver support, then out to band tools will race with ucode and end up getting undesirable values. Patch to check and pause the ucode at boot.

OPAL provides APIs to control IMC counters. OPAL_IMC_COUNTERS_INIT is used to initialize these counters at boot. OPAL_IMC_COUNTERS_START and OPAL_IMC_COUNTERS_STOP API calls should be used to start and pause these IMC engines. *doc/opal-api/opal-imc-counters.rst* details the OPAL APIs and their usage.

- hdata/i2c: update the list of known i2c devs

  This updates the list of known i2c devices - as of HDAT spec v10.5e - so that they can be properly identified during the hdat parsing.

- hdata/i2c: log unknown i2c devices

  An i2c device is unknown if either the i2c device list is outdated or the device is marked as unknown (0xFF) in the hdat.

Since *skiboot-5.8*:

- Disable Transactional Memory on Power9 DD 2.1

  Update pa_features_p9[] to disable TM (Transactional Memory). On DD 2.1 TM is not usable by Linux without other workarounds, so skiboot must disable it.

- xscom: Do not print error message for 'chiplet offline' return values

  xscom_read/write operations returns CHIPLET_OFFLINE when chiplet is offline. Some multicast xscom_read/write requests from HBRT results in xscom operation on offline chiplet(s) and printing below warnings in OPAL console:

```
[ 135.036327572,3] XSCOM: Read failed, ret = -14
[ 135.092689829,3] XSCOM: Read failed, ret = -14
```

  Some SCOM users can deal correctly with this error code (notably opal-prd), so the error message is (in practice) erroneous.

- IMC: Fix the core_imc_event_mask

  CORE_IMC_EVENT_MASK is a scom that contains bits to control event sampling for different machine state for core imc. The current event-mask setting sample events only on host kernel (hypervisor) and host userspace.

  Patch to enable the sampling of events in other machine states (like guest kernel and guest userspace).

- IMC: Update the nest_pmus array with occ/gpe microcode uav updates

  OOC/gpe nest microcode maintains the list of individual nest units supported. Sync the recent updates to the UAV with nest_pmus array.

  For reference occ/gpr microcode link for the UAV: https://github.com/open-power/occ/blob/master/src/occ_gpe1/gpe1_24x7.h

- Parse IOSLOT information from HDAT

  Add structure definitions that describe the physical PCIe topology of a system and parse them into the device-tree based PCIe slot description.

- idle: user context state loss flags fix for stop states

  The "lite" stop variants with PSSCR[ESL]=PSSCR[EC]=1 do not lose user context, while the non-lite variants do (ESL: enable state loss).

  Some of the POWER9 idle states had these wrong.

CAPI

- POWER9 DD2 update

  The CAPI initialization sequence has been updated in DD2. This patch adapts to the changes, retaining compatibility with DD1. The patch includes some changes to DD1 fix-ups as well.

- Load CAPP microcode for POWER9 DD2.0 and DD2.1

- capi: Mask Psl Credit timeout error for POWER9

  Mask the PSL credit timeout error in CAPP FIR Mask register bit(46). As per the h/w team this error is now deprecated and shouldn't cause any fir-action for P9.

NVLINK2

A notabale change is that we now generate the device tree description of NVLINK based on the HDAT we get from hostboot. Since Hostboot will generate HDAT based on VPD, you now *MUST* have correct VPD programmed or we will *default* to a Sequoia layout, which will lead to random problems if you are not booting a Sequoia Witherspoon planar. In the case of booting with old VPD and/or Hostboot, we print a **giant scary warning** in order to scare you.

Since *skiboot-5.9-rc2*: - Revert "npu2: Add vendor cap for IRQ testing"

> This reverts commit 9817c9e29b6fe00daa3a0e4420e69a97c90eb373 which seems to break setting the PCI dev flag and the link number in the PCIe vendor specific config space. This leads to the device driver attempting to re-init the DL when it shouldn't which can cause HMI's.

Since *skiboot-5.8*:

- npu2: Read slot label from the HDAT link node

  Binding GPU to emulated NPU PCI devices is done using the slot labels since the NPU devices do not have a patching slot node we need to copy the label in here.

- npu2: Copy link speed from the npu HDAT node

  This needs to be in the PCI device node so the speed of the NVLink can be passed to the GPU driver.

- npu2: hw-procedures: Add settings to PHY_RESET

  Set a few new values in the PHY_RESET procedure, as specified by our updated programming guide documentation.

- Parse NVLink information from HDAT

  Add the per-chip structures that descibe how the A-Bus/NVLink/OpenCAPI phy is configured. This generates the npu@xyz nodes for each chip on systems that support it.

- npu2: Add vendor cap for IRQ testing

  Provide a way to test recoverable data link interrupts via a new vendor capability byte.

- npu2: Enable recoverable data link (no-stall) interrupts

  Allow the NPU2 to trigger "recoverable data link" interrupts.

- npu2: Implement basic FLR (Function Level Reset)

- npu2: hw-procedures: Update PHY DC calibration procedure

- npu2: hw-procedures: Change rx_pr_phase_step value

XIVE

- xive: Fix opal_xive_dump_tm() to access W2 properly. The HW only supported limited access sizes.
- xive: Make opal_xive_allocate_irq() properly try all chips

  When requested via OPAL_XIVE_ANY_CHIP, we need to try all chips. We first try the current one (on which the caller sits) and if that fails, we iterate all chips until the allocation succeeds.
- xive: Fix initialization & cleanup of HW thread contexts

  Instead of trying to "pull" everything and clear VT (which didn't work and caused some FIRs to be set), instead just clear and then set the PTER thread enable bit. This has the side effect of completely resetting the corresponding thread context.

  This fixes the spurrious XIVE FIRs reported by PRD and fircheck
- xive: Add debug option for detecting misrouted IPI in emulation

  This is high overhead so we don't enable it by default even in debug builds, it's also a bit messy, but it allowed me to detect and debug a locking issue earlier so it can be useful.
- xive: Increase the interrupt "gap" on debug builds

  We normally allocate IPIs from 0x10. Make that 0x1000 on debug builds to limit the chances of overlapping with Linux interrupt numbers which makes debugging code that confuses them easier.

  Also add a warning in emulation if we get an interrupt in the queue whose number is below the gap.
- xive: Fix locking around cache scrub & watch

  Thankfully the missing locking only affects debug code and init code that doesn't run concurrently. Also adds a DEBUG option that checks the lock is properly held.
- xive: Workaround HW issue with scrub facility

  Without this, we sometimes don't observe from a CPU the values written to the ENDs or NVTs via the cache watch.
- xive: Add exerciser for cache watch/scrub facility in DEBUG builds
- xive: Make assertion in xive_eq_for_target() more informative
- xive: Add debug code to check initial cache updates
- xive: Ensure pressure relief interrupts are disabled

  We don't use them and we hijack the VP field with their configuration to store the EQ reference, so make sure the kernel or guest can't turn them back on by doing MMIO writes to ACK#
- xive: Don't try setting the reserved ACK# field in VPs

  That doesn't work, the HW doesn't implement it in the cache watch facility anyway.
- xive: Remove useless memory barriers in VP/EQ inits

  We no longer update "live" memory structures, we use a temporary copy on the stack and update the actual memory structure using the cache watch, so those barriers are pointless.

PHB4

Since *skiboot-5.9-rc4*:

- phb4: Escalate freeze to fence to avoid checkstop

  Freeze events such as MMIO loads can cause the PHB to lose it's limited powerbus credits. If all credits are used and a further MMIO will cause a checkstop.

  To work around this, we escalate the troublesome freeze events to a fence. The fence will cause a full PHB reset which resets the powerbus credits and avoids the checkstop.

- phb4: Update some init registers

  New inits based on next PHB4 workbook. Increases some timeouts to avoid some spurious error conditions.

- phb4: Enable PHB MMIO in phb4_root_port_init()

  Linux EEH flow is somewhat broken. It saves the PCIe config space of the PHB on boot, which it then uses to restore on EEH recovery. It does this to restore MMIO bars and some other pieces.

  Unfortunately this save is done before any drivers are bound to devices under the PHB. A number of other things are configured in the PHB after drivers start, hence some configuration space settings aren't saved correctly. These include bus master and MMIO bits in the command register.

  Linux tried to hack around this in this linux commit `bf898ec5cb` powerpc/eeh: Enable PCI_COMMAND_MASTER for PCI bridges This sets the bus master bit but ignores the MMIO bit.

  Hence we lose MMIO after a full PHB reset. This causes the next MMIO access to the device to fail and for us to perform a PE freeze recovery, which still doesn't set the MMIO bit and hence we still fail.

  This works around this by forcing MMIO on during phb4_root_port_init().

  With this we can recovery from a PHB fence event on POWER9.

- phb4: Reduce link degraded message log level to debug

  If we hit this message we'll retry and fix the problem. If we run out of retries and can't fix the problem, we'll still print a log message at error level indicating a problem.

- phb4: Fix GEN3 for DD2.00

  In this fix: `62ac7631ae phb4: Fix PCIe GEN4 on DD2.1 and above` We fixed DD2.1 GEN4 but broke DD2.00 as GEN3.

  This fixes DD2.00 back to GEN3. This time for sure!

Since *skiboot-5.9-rc3*: - phb4: Fix PCIe GEN4 on DD2.1 and above

  In this change: eef0e197ab PHB4: Default to PCIe GEN3 on POWER9 DD2.00

  We clamped DD2.00 parts to GEN3 but unfortunately this change also applies to DD2.1 and above.

  This fixes this to only apply to DD2.00.

Since *skiboot-5.8*:

- phb4: Mask RXE_ARB: DEC Stage Valid Error

  Change the inits to mask out the RXE ARB: DEC Stage Valid Error (bit 370. This has been a fatal error but should be informational only.

  This update will be in the next version of the phb4 workbook.

- phb4: Add additional adapter to retrain whitelist

  The single port version of the ConnectX-5 has a different device ID 0x1017. Updated descriptions to match pciutils database.

- PHB4: Default to PCIe GEN3 on POWER9 DD2.00

  You can use the NVRAM override for DD2.00 screened parts.

- phb4: Retrain link if degraded

  On P9 Scale Out (Nimbus) DD2.0 and Scale in (Cumulus) DD1.0 (and below) the PCIe PHY can lockup causing training issues. This can cause a degradation in speed or width in ~5% of training cases (depending on the card). This is fixed in later chip revisions. This issue can also cause PCIe links to not train at all, but this case is already handled.

  This patch checks if the PCIe link has trained optimally and if not, does a full PHB reset (to fix the PHY lockup) and retrain.

  One complication is some devices are known to train degraded unless device specific configuration is performed. Because of this, we only retrain when the device is in a whitelist. All devices in the current whitelist have been testing on a P9DSU/Boston, ZZ and Witherspoon.

  We always gather information on the link and print it in the logs even if the card is not in the whitelist.

  For testing purposes, there's an nvram to retry all PCIe cards and all P9 chips when a degraded link is detected. The new option is 'pci-retry-all=true' which can be set using: *nvram -p ibm,skiboot –update-config pci-retry-all=true*. This option may increase the boot time if used on a badly behaving card.

IBM FSP platforms

Since *skiboot-5.9-rc5*: - FSP/CONSOLE: Disable notification on unresponsive consoles

Commit fd6b71fc fixed the situation where ipmi console was open (hvc0) but got data on different console (hvc1).

During FSP Reset/Reload OPAL closes all consoles. After Reset/Reload complete FSP requests to open hvc1 and sends data on this. If hvc1 registration failed or not opened in host kernel then it will not read data and results in RCU stalls.

Note that this is workaround for older kernel where we don't have separate irq for each console. Latest kernel works fine without this patch.

Since *skiboot-5.9-rc1*:

- FSP/CONSOLE: Limit number of error logging

  Commit c8a7535f (FSP/CONSOLE: Workaround for unresponsive ipmi daemon) added error logging when buffer is full. In some corner cases kernel may call this function multiple time and we may endup logging error again and again.

  This patch fixes it by generating error log only once.

- FSP/CONSOLE: Fix fsp_console_write_buffer_space() call

  Kernel calls fsp_console_write_buffer_space() to check console buffer space availability. If there is enough buffer space to write data, then kernel will call fsp_console_write() to write actual data.

  In some extreme corner cases (like one explained in commit c8a7535f) console becomes full and this function returns 0 to kernel (or space available in console buffer < next incoming data size). Kernel will continue retrying until it gets enough space. So we will start seeing RCU stalls.

  This patch keeps track of previous available space. If previous space is same as current means not enough space in console buffer to write incoming data. It may be due to very high console write operation and slow response from FSP -OR- FSP has stopped processing data (ex: because of ipmi daemon died). At this point we will start timer with timeout of SER_BUFFER_OUT_TIMEOUT (10 secs). If situation is not improved within 10 seconds means something went bad. Lets return OPAL_RESOURCE so that kernel can drop console write and continue.

- FSP/CONSOLE: Close SOL session during R/R

  Presently we are not closing SOL and FW console sessions during R/R. Host will continue to write to SOL buffer during FSP R/R. If there is heavy console write operation happening during FSP R/R (like running *top* command inside console), then at some point console buffer becomes full. fsp_console_write_buffer_space() returns 0 (or less than required space to write data) to host. While one thread is busy writing to console, if some other threads tries to write data to console we may see RCU stalls (like below) in kernel.

```
[ 2082.828363] INFO: rcu_sched detected stalls on CPUs/tasks: { 32} (detected by␣
↪16, t=6002 jiffies, g=23154, c=23153, q=254769)
[ 2082.828365] Task dump for CPU 32:
[ 2082.828368] kworker/32:3    R  running task        0  4637      2 0x00000884
[ 2082.828375] Workqueue: events dump_work_fn
[ 2082.828376] Call Trace:
[ 2082.828382] [c000000f1633fa00] [c00000000013b6b0] console_unlock+0x570/0x600␣
↪(unreliable)
[ 2082.828384] [c000000f1633fae0] [c00000000013ba34] vprintk_emit+0x2f4/0x5c0
[ 2082.828389] [c000000f1633fb60] [c00000000099e644] printk+0x84/0x98
[ 2082.828391] [c000000f1633fb90] [c0000000000851a8] dump_work_fn+0x238/0x250
[ 2082.828394] [c000000f1633fc60] [c0000000000ecb98] process_one_work+0x198/0x4b0
[ 2082.828396] [c000000f1633fcf0] [c0000000000ed3dc] worker_thread+0x18c/0x5a0
[ 2082.828399] [c000000f1633fd80] [c0000000000f4650] kthread+0x110/0x130
[ 2082.828403] [c000000f1633fe30] [c000000000009674] ret_from_kernel_thread+0x5c/
↪0x68
```

  Hence lets close SOL (and FW console) during FSP R/R.

- FSP/CONSOLE: Do not associate unavailable console

  Presently OPAL sends associate/unassociate MBOX command for all FSP serial console (like below OPAL message). We have to check console is available or not before sending this message.

```
[ 5013.227994012,7] FSP: Reassociating HVSI console 1
[ 5013.227997540,7] FSP: Reassociating HVSI console 2
```

- FSP: Disable PSI link whenever FSP tells OPAL about impending R/R

  Commit 42d5d047 fixed scenario where DPO has been initiated, but FSP went into reset before the CEC power down came in. But this is generic issue that can happen in normal shutdown path as well.

  Hence disable PSI link as soon as we detect FSP impending R/R.

- fsp: return OPAL_BUSY_EVENT on failure sending FSP_CMD_POWERDOWN_NORM Also, return OPAL_BUSY_EVENT on failure sending FSP_CMD_REBOOT / DEEP_REBOOT.

  We had a race condition between FSP Reset/Reload and powering down the system from the host:

  Roughly:

#	FSP	Host
1	Power on	
2		Power on
3	(inject EPOW)	
4	(trigger FSP R/R)	
5		Processes EPOW event, starts shutting down
6		calls OPAL_CEC_POWER_DOWN
7	(is still in R/R)	
8		gets OPAL_INTERNAL_ERROR, spins in opal_poll_events
9	(FSP comes back)	
10		spinning in opal_poll_events
11	(thinks host is running)	

The call to OPAL_CEC_POWER_DOWN is only made once as the reset/reload error path for fsp_sync_msg() is to return -1, which means we give the OS OPAL_INTERNAL_ERROR, which is fine, except that our own API docs give us the opportunity to return OPAL_BUSY when trying again later may be successful, and we're ambiguous as to if you should retry on OPAL_INTERNAL_ERROR.

For reference, the linux code looks like this:

```
static void __noreturn pnv_power_off(void)
{
        long rc = OPAL_BUSY;

        pnv_prepare_going_down();

        while (rc == OPAL_BUSY || rc == OPAL_BUSY_EVENT) {
                rc = opal_cec_power_down(0);
                if (rc == OPAL_BUSY_EVENT)
                        opal_poll_events(NULL);
                else
                        mdelay(10);
        }
        for (;;)
                opal_poll_events(NULL);
}
```

Which means that *practically* our only option is to return OPAL_BUSY or OPAL_BUSY_EVENT.

We choose OPAL_BUSY_EVENT for FSP systems as we do want to ensure we're running pollers to communicate with the FSP and do the final bits of Reset/Reload handling before we power off the system.

Since *skiboot-5.8*:

- FSP/NVRAM: Handle "get vNVRAM statistics" command

  FSP sends MBOX command (cmd : 0xEB, subcmd : 0x05, mod : 0x00) to get vNVRAM statistics. OPAL doesn't maintain any such statistics. Hence return FSP_STATUS_INVALID_SUBCMD.

  Fixes these messages appearing in the OPAL log:

```
[16944.384670488,3] FSP: Unhandled message eb0500
[16944.474110465,3] FSP: Unhandled message eb0500
[16945.111280784,3] FSP: Unhandled message eb0500
[16945.293393485,3] FSP: Unhandled message eb0500
```

- fsp: Move common prints to trace

These two prints just end up filling the skiboot logs on any machine that's been booted for more than a few hours.

They have never been useful, so make them trace level. They were: :: SURV: Received heartbeat acknowledge from FSP SURV: Sending the heartbeat command to FSP

BMC based systems

- hw/lpc-uart: read from RBR to clear character timeout interrupts

  When using the aspeed SUART, we see a condition where the UART sends continuous character timeout interrupts. This change adds a (heavily commented) dummy read from the RBR to clear the interrupt condition on init.

  This was observed on p9dsu systems, but likely applies to other systems using the SUART.

- astbmc: Add methods for handing Device Tree based slots e.g. ones from HDAT on POWER9.

General

Since *skiboot-5.9-rc5*:

- p8-i2c: Further timeout reworks

  This patch reworks the way timeouts are set so that rather than imposing a hard deadline based on the transaction length it uses a kick-the-can-down-the-road approach where the timeout will be reset each time data is written to or received from the master. This fits better with the actual failure modes that timeouts are designed to handle, such as unusually slow or broken devices.

  Additionally this patch moves all the special case detection out of the timeout handler. This is help to improve the robustness of the driver and prepare for a more substantial rework of the driver as a whole later on.

- npu: Fix broken fast reset

  0679f61244b "fast-reset: by default (if possible)" broke NPU - now the NV links does not get enabled after reboot.

  This disables fast reboot for NPU machines till a better solution is found.

Since *skiboot-5.9-rc2*:

- Improvements to vpd device tree entries

  Previously we would miss some properties

Since *skiboot-5.9-rc1*:

- hw/p8-i2c: Fix deadlock in p9_i2c_bus_owner_change

  When debugging a system where Linux was taking soft lockup errors with two CPUs stuck in OPAL:

CPU0	CPU1
lock	
p8_i2c_recover	
opal_handle_interrupt	sync_timer cancel_timer p9_i2c_bus_owner_change occ_p9_interrupt xive_source_interrupt opal_handle_interrupt

  p8_i2c_recover() is a timer, and is stuck trying to take master->lock. p9_i2c_bus_owner_change() has taken master->lock, but then is stuck waiting for all timers to complete. We deadlock.

  Fix this by using cancel_timer_async().

- opal/cpu: Mark the core as bad while disabling threads of the core.

  If any of the core fails to sync its TB during chipTOD initialization, all the threads of that core are disabled. But this does not make linux kernel to ignore the core/cpus. It crashes while bringing them up with below backtrace:

```
[   38.883898] kexec_core: Starting new kernel
cpu 0x0: Vector: 300 (Data Access) at [c0000003f277b730]
    pc: c0000000001b9890: internal_create_group+0x30/0x304
    lr: c0000000001b9880: internal_create_group+0x20/0x304
    sp: c0000003f277b9b0
   msr: 900000000280b033
   dar: 40
 dsisr: 40000000
  current = 0xc0000003f9f41000
  paca    = 0xc00000000fe00000   softe: 0        irq_happened: 0x01
    pid   = 2572, comm = kexec
Linux version 4.13.2-openpower1 (jenkins@p89) (gcc version 6.4.0 (Buildroot 2017.
↪08-00006-g319c6e1)) #1 SMP Wed Sep 20 05:42:11 UTC 2017
enter ? for help
[c0000003f277b9b0] c0000000008a8780 (unreliable)
[c0000003f277ba50] c00000000041c3ac topology_add_dev+0x2c/0x40
[c0000003f277ba70] c00000000006b078 cpuhp_invoke_callback+0x88/0x170
[c0000003f277bac0] c00000000006b22c cpuhp_up_callbacks+0x54/0xb8
[c0000003f277bb10] c00000000006bc68 cpu_up+0x11c/0x168
[c0000003f277bbc0] c00000000002f0e0 default_machine_kexec+0x1fc/0x274
[c0000003f277bc50] c00000000002e2d8 machine_kexec+0x50/0x58
[c0000003f277bc70] c0000000000de4e8 kernel_kexec+0x98/0xb4
[c0000003f277bce0] c00000000008b0f0 SyS_reboot+0x1c8/0x1f4
[c0000003f277be30] c00000000000b118 system_call+0x58/0x6c
```

Since *skiboot-5.8*:

- ipmi: Convert common debug prints to trace

  OPAL logs messages for every IPMI request from host. Sometime OPAL console is filled with only these messages. This path is pretty stable now and we have enough logs to cover bad path. Hence lets convert these debug message to trace/info message. Examples are:

```
[ 1356.423958816,7] opal_ipmi_recv(cmd: 0xf0 netfn: 0x3b resp_size: 0x02)
[ 1356.430774496,7] opal_ipmi_send(cmd: 0xf0 netfn: 0x3a len: 0x3b)
[ 1356.430797392,7] BT: seq 0x20 netfn 0x3a cmd 0xf0: Message sent to host
[ 1356.431668496,7] BT: seq 0x20 netfn 0x3a cmd 0xf0: IPMI MSG done
```

- libflash/file: Handle short read()s and write()s correctly

  Currently we don't move the buffer along for a short read() or write() and nor do we request only the remaining amount.

- hw/p8-i2c: Rework timeout handling

  Currently we treat a timeout as a hard failure and will automatically fail any transations that hit their timeout. This results in unnecessarily failing I2C requests if interrupts are dropped, etc. Although these are bad things that we should log we can handle them better by checking the actual hardware status and completing the transation if there are no real errors. This patch reworks the timeout handling to check the status and continue the transaction if it can. if it can while logging an error if it detects a timeout due to a dropped interrupt.

- core/flash: Only expect ELF header for BOOTKERNEL partition flash resource

  When loading a flash resource which isn't signed (secure and trusted boot) and which doesn't have a subpartition, we assume it's the BOOTKERNEL since previously this was the only such resource. Thus we also assumed it had an ELF header which we parsed to get the size of the partition rather than trusting the actual_size field in

the FFS header. A previous commit (9727fe3 DT: Add ibm,firmware-versions node) added the version resource which isn't signed and also doesn't have a subpartition, thus we expect it to have an ELF header. It doesn't so we print the error message "FLASH: Invalid ELF header part VERSION".

It is a fluke that this works currently since we load the secure boot header unconditionally and this happen to be the same size as the version partition. We also don't update the return code on error so happen to return OPAL_SUCCESS.

To make this explicitly correct; only check for an ELF header if we are loading the BOOTKERNEL resource, otherwise use the partition size from the FFS header. Also set the return code on error so we don't erroneously return OPAL_SUCCESS. Add a check that the resource will fit in the supplied buffer to prevent buffer overrun.

- flash: Support adding the no-erase property to flash

  The mbox protocol explicitly states that an erase is not required before a write. This means that issuing an erase from userspace, through the mtd device, and back returns a successful operation that does nothing. Unfortunately, this makes userspace tools unhappy. Linux MTD devices support the MTD_NO_ERASE flag which conveys that writes do not require erases on the underlying flash devices. We should set this property on all of our devices which do not require erases to be performed.

  NOTE: This still requires a linux kernel component to set the MTD_NO_ERASE flag from the device tree property.

Utilities

Since *skiboot-5.9-rc1*: - opal-prd: Fix memory leak

Since *skiboot-5.8*:

- external/gard: Clear entire guard partition instead of entry by entry

  When using the current implementation of the gard tool to ecc clear the entire GUARD partition it is done one gard record at a time. While this may be ok when accessing the actual flash this is very slow when done from the host over the mbox protocol (on the order of 4 minutes) because the bmc side is required to do many read, erase, writes under the hood.

  Fix this by rewriting the gard tool reset_partition() function. Now we allocate all the erased guard entries and (if required) apply ecc to the entire buffer. Then we can do one big erase and write of the entire partition. This reduces the time to clear the guard partition to on the order of 4 seconds.

- opal-prd: Fix opal-prd command line options

  HBRT OCC reset interface depends on service processor type.

  - FSP: reset_pm_complex()
  - BMC: process_occ_reset()

  We have both *occ* and *pm-complex* command line interfaces. This patch adds support to dispaly appropriate message depending on system type.

SP	Command	Action
FSP	opal-prd occ	display error message
FSP	opal-prd pm-complex	Call pm_complex_reset()
BMC	opal-prd occ	Call process_occ_reset()
BMC	opal-prd pm-complex	display error message

- opal-prd: detect service processor type and then make appropriate occ reset call.

- pflash: Fix erase command for unaligned start address

  The erase_range() function handles erasing the flash for a given start address and length, and can handle an unaligned start address and length. However in the unaligned start address case we are incorrectly calculating the remaining size which can lead to incomplete erases.

  If we're going to update the remaining size based on what the start address was then we probably want to do that before we overide the origin start address. So rearrange the code so that this is indeed the case.

- external/gard: Print an error if run on an FSP system

Simulators

- mambo: Add mambo socket program

  This adds a program that can be run inside a mambo simulator in linux userspace which enables TCP sockets to be proxied in and out of the simulator to the host.

  Unlike mambo bogusnet, it's requires no linux or skiboot specific drivers/infrastructure to run.

  Run inside the simulator:

  - to forward host ssh connections to sim ssh server: `./mambo-socket-proxy -h 10022 -s 22`, then connect to port 10022 on your host with `ssh -p 10022 localhost`
  - to allow http proxy access from inside the sim to local http proxy: `./mambo-socket-proxy -b proxy.mynetwork -h 3128 -s 3128`

  Multiple connections are supported.

- idle: disable stop*_lite POWER9 idle states for Mambo platform

  Mambo prior to Mambo.7.8.21 had a bug where the stop idle instruction with PSSCR[ESL]=PSSCR[EC]=0 would resume with MSR set as though it had taken a system reset interrupt.

  Linux currently executes this instruction with MSR already set that way, so the problem went unnoticed. A proposed patch to Linux changes that, and causes the idle code to crash. Work around this by disabling lite stop states for the mambo platform for now.

Contributors

- 209 csets from 32 developers
- 2 employers found
- A total of 9619 lines added, 1612 removed (delta 8007)

Extending the analysis done for some previous releases, we can see our trends in code review across versions:

Release	csets	Ack %	Reviews %	Tested %	Reported %
5.0	329	15 (5%)	20 (6%)	1 (0%)	0 (0%)
5.1	372	13 (3%)	38 (10%)	1 (0%)	4 (1%)
5.2-rc1	334	20 (6%)	34 (10%)	6 (2%)	11 (3%)
5.3-rc1	302	36 (12%)	53 (18%)	4 (1%)	5 (2%)
5.4	361	16 (4%)	28 (8%)	1 (0%)	9 (2%)
5.5	408	11 (3%)	48 (12%)	14 (3%)	10 (2%)
5.6	87	12 (14%)	6 (7%)	5 (6%)	2 (2%)
5.7	232	30 (13%)	32 (14%)	5 (2%)	2 (1%)
5.8	157	13 (8%)	36 (23%)	2 (1%)	6 (4%)
5.9	209	15 (7%)	78 (37%)	3 (1%)	10 (5%)

The review count here is largely bogus, there was a series of 25 whitespace patches that got "Reviewed-by" and if we exclude them, we're back to 14%, which is more like what I'd expect.

Developers with the most changesets

Developer	#	%
Stewart Smith	28	(13.4%)
Vasant Hegde	25	(12.0%)
Joel Stanley	25	(12.0%)
Michael Neuling	24	(11.5%)
Oliver O'Halloran	20	(9.6%)
Benjamin Herrenschmidt	16	(7.7%)
Nicholas Piggin	12	(5.7%)
Akshay Adiga	8	(3.8%)
Madhavan Srinivasan	7	(3.3%)
Reza Arbab	6	(2.9%)
Mahesh Salgaonkar	3	(1.4%)
Claudio Carvalho	3	(1.4%)
Suraj Jitindar Singh	3	(1.4%)
Sam Bobroff	3	(1.4%)
Shilpasri G Bhat	2	(1.0%)
Michael Ellerman	2	(1.0%)
Andrew Donnellan	2	(1.0%)
Vaibhav Jain	2	(1.0%)
Jeremy Kerr	2	(1.0%)
Cyril Bur	2	(1.0%)
Christophe Lombard	2	(1.0%)
Daniel Black	2	(1.0%)
Alexey Kardashevskiy	1	(0.5%)
Alistair Popple	1	(0.5%)
Anton Blanchard	1	(0.5%)
Guilherme G. Piccoli	1	(0.5%)
John W Walthour	1	(0.5%)
Anju T Sudhakar	1	(0.5%)
Balbir Singh	1	(0.5%)
Russell Currey	1	(0.5%)
William A. Kennington III	1	(0.5%)
Sukadev Bhattiprolu	1	(0.5%)

Developers with the most changed lines

Developer	#	%
Akshay Adiga	2731	(27.9%)
Oliver O'Halloran	1512	(15.5%)
Stewart Smith	1355	(13.9%)
Nicholas Piggin	929	(9.5%)
Vasant Hegde	827	(8.5%)
Michael Neuling	719	(7.4%)
Benjamin Herrenschmidt	522	(5.3%)

Continued on next page

Table 4.10 – continued from previous page

Developer	#	%
Madhavan Srinivasan	180	(1.8%)
Sam Bobroff	172	(1.8%)
Christophe Lombard	170	(1.7%)
Mahesh Salgaonkar	166	(1.7%)
Andrew Donnellan	125	(1.3%)
Joel Stanley	70	(0.7%)
Reza Arbab	64	(0.7%)
Claudio Carvalho	51	(0.5%)
Suraj Jitindar Singh	42	(0.4%)
Alistair Popple	28	(0.3%)
Jeremy Kerr	25	(0.3%)
Michael Ellerman	21	(0.2%)
Cyril Bur	18	(0.2%)
Shilpasri G Bhat	17	(0.2%)
Vaibhav Jain	8	(0.1%)
Daniel Black	6	(0.1%)
William A. Kennington III	4	(0.0%)
Sukadev Bhattiprolu	4	(0.0%)
Alexey Kardashevskiy	3	(0.0%)
John W Walthour	3	(0.0%)
Balbir Singh	3	(0.0%)
Guilherme G. Piccoli	2	(0.0%)
Anton Blanchard	1	(0.0%)
Anju T Sudhakar	1	(0.0%)
Russell Currey	1	(0.0%)

Developers with the most lines removed

Developer	#	%
Alistair Popple	28	(1.7%)

Developers with the most signoffs

Developer	#	%
Stewart Smith	180	(97.8%)
Shilpasri G Bhat	2	(1.1%)
Mukesh Ojha	1	(0.5%)
Michael Neuling	1	(0.5%)
Total	184	(100%)

Developers with the most reviews

Developer	#	%
Michael Neuling	25	(32.5%)
Russell Currey	25	(32.5%)
Vaidyanathan Srinivasan	9	(11.7%)
Oliver O'Halloran	4	(5.2%)
Andrew Donnellan	3	(3.9%)
Frederic Barrat	2	(2.6%)
Suraj Jitindar Singh	2	(2.6%)
Vasant Hegde	2	(2.6%)
Andrew Jeffery	1	(1.3%)
Samuel Mendoza-Jonas	1	(1.3%)
Alexey Kardashevskiy	1	(1.3%)
Cyril Bur	1	(1.3%)
Akshay Adiga	1	(1.3%)
Total	77	(100%)

Developers with the most test credits

Developer	#	%
Pridhiviraj Paidipeddi	3	(100.0%)

Developers who gave the most tested-by credits

Developer	#	%
Vasant Hegde	2	(66.7%)
Michael Neuling	1	(33.3%)

Developers with the most report credits

Developer	#	%
Pridhiviraj Paidipeddi	6	(60.0%)
Andrew Donnellan	1	(10.0%)
Stewart Smith	1	(10.0%)
Shriya	1	(10.0%)
Robert Lippert	1	(10.0%)
Total	10	(100%)

Developers who gave the most report credits

Developer	#	%
Stewart Smith	3	(30.0%)
Suraj Jitindar Singh	3	(30.0%)
Vasant Hegde	2	(20.0%)
Michael Neuling	1	(10.0%)
Madhavan Srinivasan	1	(10.0%)
Total	10	(100%)

Changesets and Employers

Top changeset contributors by employer:

Employer	#	%
IBM	208	(99.5%)
Google	1	(0.5%)

Top lines changed by employer:

Employer	#	%
IBM	9776	(100.0%)
Google	4	(0.0%)

Employers with the most signoffs (total 184):

Employer	#	%
IBM	184	(100.0%)

Employers with the most hackers (total 32):

Employer	#	%
IBM	31	(96.9%)
Google	1	(3.1%)

4.1.69 skiboot-5.9-rc1

skiboot v5.9-rc1 was released on Wednesday October 11th 2017. It is the first release candidate of skiboot 5.9, which will become the new stable release of skiboot following the 5.8 release, first released August 31st 2017.

skiboot v5.9-rc1 contains all bug fixes as of *skiboot-5.4.7* and *skiboot-5.1.21* (the currently maintained stable releases). We do not currently expect to do any 5.8.x stable releases.

For how the skiboot stable releases work, see *Skiboot stable tree rules and releases* for details.

The current plan is to cut the final 5.9 by October 17th, with skiboot 5.9 being for all POWER8 and POWER9 platforms in op-build v1.20 (Due October 18th). This release will be targetted to early POWER9 systems.

Over skiboot-5.8, we have the following changes:

New Features

POWER8

- fast-reset by default (if possible)

  Currently, this is limited to POWER8 systems.

  A normal reboot will, rather than doing a full IPL, go through a fast reboot procedure. This reduces the "reboot to petitboot" time from minutes to a handful of seconds.

POWER9

- POWER9 power management during boot

  Less power should be consumed during boot.

- OPAL_SIGNAL_SYSTEM_RESET for POWER9

  This implements OPAL_SIGNAL_SYSTEM_RESET, using scom registers to quiesce the target thread and raise a system reset exception on it. It has been tested on DD2 with stop0 ESL=0 and ESL=1 shallow power saving modes.

  DD1 is not implemented because it is sufficiently different as to make support difficult.

- Enable deep idle states for POWER9
  - SLW: Add support for p9_stop_api

    p9_stop_api's are used to set SPR state on a core wakeup form a deeper low power state. p9_stop_api uses low level platform formware and self-restore microcode to restore the sprs to requested values.

    Code is taken from : https://github.com/open-power/hostboot/tree/master/src/import/chips/p9/procedures/utils/stopreg
  - SLW: Removing timebase related flags for stop4

    When a core enters stop4, it does not loose decrementer and time base. Hence removing flags OPAL_PM_DEC_STOP and OPAL_PM_TIMEBASE_STOP.
  - SLW: Allow deep states if homer address is known

    Use a common variable has_wakeup_engine instead of has_slw to tell if the: - SLW image is populated in case of power8 - CME image is populated in case of power9

    Currently we expect CME to be loaded if homer address is known (except for simulators)
  - SLW: Configure self-restore for HRMOR

    Make a stop api call using libpore to restore HRMOR register. HRMOR needs to be cleared so that when thread exits stop, they arrives at linux system_reset vector (0x100).
  - SLW: Add opal_slw_set_reg support for power9

    This OPAL call is made from Linux to OPAL to configure values in various SPRs after wakeup from a deep idle state.
- PHB4: CAPP recovery

  CAPP recovery is initiated when a CAPP Machine Check is detected. The capp recovery procedure is initiated via a Hypervisor Maintenance interrupt (HMI).

CAPP Machine Check may arise from either an error that results in a PHB freeze or from an internal CAPP error with CAPP checkstop FIR action. An error that causes a PHB freeze will result in the link down signal being asserted. The system continues running and the CAPP and PSL will be re-initialized.

This implements CAPP recovery for POWER9 systems

- Add `wafer-location` property for POWER9

  Extract wafer-location from ECID and add property under xscom node. - bits 64:71 are the chip x location (7:0) - bits 72:79 are the chip y location (7:0)

  Sample output:

```
[root@wsp xscom@623fc00000000]# lsprop ecid
ecid              019a00d4 03100718 852c0000 00fd7911
[root@wsp xscom@623fc00000000]# lsprop wafer-location
wafer-location    00000085 0000002c
```

- Add `wafer-id` property for POWER9

  Wafer id is derived from ECID data. - bits 4:63 are the wafer id (ten 6 bit fields each containing a code)

  Sample output:

```
[root@wsp xscom@623fc00000000]# lsprop ecid
ecid              019a00d4 03100718 852c0000 00fd7911
[root@wsp xscom@623fc00000000]# lsprop wafer-id
wafer-id          "6Q0DG340SO"
```

- Add `ecid` property under `xscom` node for POWER9. Sample output:

```
[root@wsp xscom@623fc00000000]# lsprop ecid
ecid              019a00d4 03100718 852c0000 00fd7911
```

- Add ibm,firmware-versions device tree node

  In P8, hostboot provides mini device tree. It contains `/ibm,firmware-versions` node which has various firmware component version details.

  In P9, OPAL is building device tree. This patch adds support to parse VERSION section of PNOR and create `/ibm,firmware-versions` device tree node.

  Sample output:

```
/sys/firmware/devicetree/base/ibm,firmware-versions # lsprop .
occ               "6a00709"
skiboot           "v5.7-rc1-p344fb62"
buildroot         "2017.02.2-7-g23118ce"
capp-ucode        "9c73e9f"
petitboot         "v1.4.3-p98b6d83"
sbe               "02021c6"
open-power        "witherspoon-v1.17-128-gf1b53c7-dirty"
....
....
```

POWER9

- Disable Transactional Memory on Power9 DD 2.1

Update pa_features_p9[] to disable TM (Transactional Memory). On DD 2.1 TM is not usable by Linux without other workarounds, so skiboot must disable it.

- xscom: Do not print error message for 'chiplet offline' return values

  xscom_read/write operations returns CHIPLET_OFFLINE when chiplet is offline. Some multicast xscom_read/write requests from HBRT results in xscom operation on offline chiplet(s) and printing below warnings in OPAL console:

```
[ 135.036327572,3] XSCOM: Read failed, ret = -14
[ 135.092689829,3] XSCOM: Read failed, ret = -14
```

  Some SCOM users can deal correctly with this error code (notably opal-prd), so the error message is (in practice) erroneous.

- IMC: Fix the core_imc_event_mask

  CORE_IMC_EVENT_MASK is a scom that contains bits to control event sampling for different machine state for core imc. The current event-mask setting sample events only on host kernel (hypervisor) and host userspace.

  Patch to enable the sampling of events in other machine states (like guest kernel and guest userspace).

- IMC: Update the nest_pmus array with occ/gpe microcode uav updates

  OOC/gpe nest microcode maintains the list of individual nest units supported. Sync the recent updates to the UAV with nest_pmus array.

  For reference occ/gpr microcode link for the UAV: https://github.com/open-power/occ/blob/master/src/occ_gpe1/gpe1_24x7.h

- Parse IOSLOT information from HDAT

  Add structure definitions that describe the physical PCIe topology of a system and parse them into the device-tree based PCIe slot description.

- idle: user context state loss flags fix for stop states

  The "lite" stop variants with PSSCR[ESL]=PSSCR[EC]=1 do not lose user context, while the non-lite variants do (ESL: enable state loss).

  Some of the POWER9 idle states had these wrong.

CAPI

- POWER9 DD2 update

  The CAPI initialization sequence has been updated in DD2. This patch adapts to the changes, retaining compatibility with DD1. The patch includes some changes to DD1 fix-ups as well.

- Load CAPP microcode for POWER9 DD2.0 and DD2.1

- capi: Mask Psl Credit timeout error for POWER9

  Mask the PSL credit timeout error in CAPP FIR Mask register bit(46). As per the h/w team this error is now deprecated and shouldn't cause any fir-action for P9.

NVLINK2

A notabale change is that we now generate the device tree description of NVLINK based on the HDAT we get from hostboot. Since Hostboot will generate HDAT based on VPD, you now *MUST* have correct VPD programmed or we

will *default* to a Sequoia layout, which will lead to random problems if you are not booting a Sequoia Witherspoon planar. In the case of booting with old VPD and/or Hostboot, we print a **giant scary warning** in order to scare you.

- npu2: Read slot label from the HDAT link node

  Binding GPU to emulated NPU PCI devices is done using the slot labels since the NPU devices do not have a patching slot node we need to copy the label in here.

- npu2: Copy link speed from the npu HDAT node

  This needs to be in the PCI device node so the speed of the NVLink can be passed to the GPU driver.

- npu2: hw-procedures: Add settings to PHY_RESET

  Set a few new values in the PHY_RESET procedure, as specified by our updated programming guide documentation.

- Parse NVLink information from HDAT

  Add the per-chip structures that descibe how the A-Bus/NVLink/OpenCAPI phy is configured. This generates the npu@xyz nodes for each chip on systems that support it.

- npu2: Add vendor cap for IRQ testing

  Provide a way to test recoverable data link interrupts via a new vendor capability byte.

- npu2: Enable recoverable data link (no-stall) interrupts

  Allow the NPU2 to trigger "recoverable data link" interrupts.

- npu2: Implement basic FLR (Function Level Reset)
- npu2: hw-procedures: Update PHY DC calibration procedure
- npu2: hw-procedures: Change rx_pr_phase_step value

XIVE

- xive: Fix opal_xive_dump_tm() to access W2 properly. The HW only supported limited access sizes.
- xive: Make opal_xive_allocate_irq() properly try all chips

  When requested via OPAL_XIVE_ANY_CHIP, we need to try all chips. We first try the current one (on which the caller sits) and if that fails, we iterate all chips until the allocation succeeds.

- xive: Fix initialization & cleanup of HW thread contexts

  Instead of trying to "pull" everything and clear VT (which didn't work and caused some FIRs to be set), instead just clear and then set the PTER thread enable bit. This has the side effect of completely resetting the corresponding thread context.

  This fixes the spurrious XIVE FIRs reported by PRD and fircheck

- xive: Add debug option for detecting misrouted IPI in emulation

  This is high overhead so we don't enable it by default even in debug builds, it's also a bit messy, but it allowed me to detect and debug a locking issue earlier so it can be useful.

- xive: Increase the interrupt "gap" on debug builds

  We normally allocate IPIs from 0x10. Make that 0x1000 on debug builds to limit the chances of overlapping with Linux interrupt numbers which makes debugging code that confuses them easier.

  Also add a warning in emulation if we get an interrupt in the queue whose number is below the gap.

- xive: Fix locking around cache scrub & watch

  Thankfully the missing locking only affects debug code and init code that doesn't run concurrently. Also adds a DEBUG option that checks the lock is properly held.

- xive: Workaround HW issue with scrub facility

  Without this, we sometimes don't observe from a CPU the values written to the ENDs or NVTs via the cache watch.

- xive: Add exerciser for cache watch/scrub facility in DEBUG builds
- xive: Make assertion in xive_eq_for_target() more informative
- xive: Add debug code to check initial cache updates
- xive: Ensure pressure relief interrupts are disabled

  We don't use them and we hijack the VP field with their configuration to store the EQ reference, so make sure the kernel or guest can't turn them back on by doing MMIO writes to ACK#

- xive: Don't try setting the reserved ACK# field in VPs

  That doesn't work, the HW doesn't implement it in the cache watch facility anyway.

- xive: Remove useless memory barriers in VP/EQ inits

  We no longer update "live" memory structures, we use a temporary copy on the stack and update the actual memory structure using the cache watch, so those barriers are pointless.

PHB4

- phb4: Mask RXE_ARB: DEC Stage Valid Error

  Change the inits to mask out the RXE ARB: DEC Stage Valid Error (bit 370. This has been a fatal error but should be informational only.

  This update will be in the next version of the phb4 workbook.

- phb4: Add additional adapter to retrain whitelist

  The single port version of the ConnectX-5 has a different device ID 0x1017. Updated descriptions to match pciutils database.

- PHB4: Default to PCIe GEN3 on POWER9 DD2.00

  You can use the NVRAM override for DD2.00 screened parts.

- phb4: Retrain link if degraded

  On P9 Scale Out (Nimbus) DD2.0 and Scale in (Cumulus) DD1.0 (and below) the PCIe PHY can lockup causing training issues. This can cause a degradation in speed or width in ~5% of training cases (depending on the card). This is fixed in later chip revisions. This issue can also cause PCIe links to not train at all, but this case is already handled.

  This patch checks if the PCIe link has trained optimally and if not, does a full PHB reset (to fix the PHY lockup) and retrain.

  One complication is some devices are known to train degraded unless device specific configuration is performed. Because of this, we only retrain when the device is in a whitelist. All devices in the current whitelist have been testing on a P9DSU/Boston, ZZ and Witherspoon.

  We always gather information on the link and print it in the logs even if the card is not in the whitelist.

For testing purposes, there's an nvram to retry all PCIe cards and all P9 chips when a degraded link is detected. The new option is 'pci-retry-all=true' which can be set using: *nvram -p ibm,skiboot –update-config pci-retry-all=true*. This option may increase the boot time if used on a badly behaving card.

IBM FSP platforms

- FSP/NVRAM: Handle "get vNVRAM statistics" command

  FSP sends MBOX command (cmd : 0xEB, subcmd : 0x05, mod : 0x00) to get vNVRAM statistics. OPAL doesn't maintain any such statistics. Hence return FSP_STATUS_INVALID_SUBCMD.

  Fixes these messages appearing in the OPAL log:

```
[16944.384670488,3] FSP: Unhandled message eb0500
[16944.474110465,3] FSP: Unhandled message eb0500
[16945.111280784,3] FSP: Unhandled message eb0500
[16945.293393485,3] FSP: Unhandled message eb0500
```

- fsp: Move common prints to trace

  These two prints just end up filling the skiboot logs on any machine that's been booted for more than a few hours.

  They have never been useful, so make them trace level. They were: :: SURV: Received heartbeat acknowledge from FSP SURV: Sending the heartbeat command to FSP

BMC based systems

- hw/lpc-uart: read from RBR to clear character timeout interrupts

  When using the aspeed SUART, we see a condition where the UART sends continuous character timeout interrupts. This change adds a (heavily commented) dummy read from the RBR to clear the interrupt condition on init.

  This was observed on p9dsu systems, but likely applies to other systems using the SUART.

- astbmc: Add methods for handing Device Tree based slots e.g. ones from HDAT on POWER9.

General

- ipmi: Convert common debug prints to trace

  OPAL logs messages for every IPMI request from host. Sometime OPAL console is filled with only these messages. This path is pretty stable now and we have enough logs to cover bad path. Hence lets convert these debug message to trace/info message. Examples are:

```
[ 1356.423958816,7] opal_ipmi_recv(cmd: 0xf0 netfn: 0x3b resp_size: 0x02)
[ 1356.430774496,7] opal_ipmi_send(cmd: 0xf0 netfn: 0x3a len: 0x3b)
[ 1356.430797392,7] BT: seq 0x20 netfn 0x3a cmd 0xf0: Message sent to host
[ 1356.431668496,7] BT: seq 0x20 netfn 0x3a cmd 0xf0: IPMI MSG done
```

- libflash/file: Handle short read()s and write()s correctly

  Currently we don't move the buffer along for a short read() or write() and nor do we request only the remaining amount.

- hw/p8-i2c: Rework timeout handling

  Currently we treat a timeout as a hard failure and will automatically fail any transations that hit their timeout. This results in unnecessarily failing I2C requests if interrupts are dropped, etc. Although these are bad things that we should log we can handle them better by checking the actual hardware status and completing the transation if there are no real errors. This patch reworks the timeout handling to check the status and continue the transaction if it can. if it can while logging an error if it detects a timeout due to a dropped interrupt.

- core/flash: Only expect ELF header for BOOTKERNEL partition flash resource

  When loading a flash resource which isn't signed (secure and trusted boot) and which doesn't have a subpartition, we assume it's the BOOTKERNEL since previously this was the only such resource. Thus we also assumed it had an ELF header which we parsed to get the size of the partition rather than trusting the actual_size field in the FFS header. A previous commit (9727fe3 DT: Add ibm,firmware-versions node) added the version resource which isn't signed and also doesn't have a subpartition, thus we expect it to have an ELF header. It doesn't so we print the error message "FLASH: Invalid ELF header part VERSION".

  It is a fluke that this works currently since we load the secure boot header unconditionally and this happen to be the same size as the version partition. We also don't update the return code on error so happen to return OPAL_SUCCESS.

  To make this explicitly correct; only check for an ELF header if we are loading the BOOTKERNEL resource, otherwise use the partition size from the FFS header. Also set the return code on error so we don't erroneously return OPAL_SUCCESS. Add a check that the resource will fit in the supplied buffer to prevent buffer overrun.

- flash: Support adding the no-erase property to flash

  The mbox protocol explicitly states that an erase is not required before a write. This means that issuing an erase from userspace, through the mtd device, and back returns a successful operation that does nothing. Unfortunately, this makes userspace tools unhappy. Linux MTD devices support the MTD_NO_ERASE flag which conveys that writes do not require erases on the underlying flash devices. We should set this property on all of our devices which do not require erases to be performed.

  NOTE: This still requires a linux kernel component to set the MTD_NO_ERASE flag from the device tree property.

Utilities

- external/gard: Clear entire guard partition instead of entry by entry

  When using the current implementation of the gard tool to ecc clear the entire GUARD partition it is done one gard record at a time. While this may be ok when accessing the actual flash this is very slow when done from the host over the mbox protocol (on the order of 4 minutes) because the bmc side is required to do many read, erase, writes under the hood.

  Fix this by rewriting the gard tool reset_partition() function. Now we allocate all the erased guard entries and (if required) apply ecc to the entire buffer. Then we can do one big erase and write of the entire partition. This reduces the time to clear the guard partition to on the order of 4 seconds.

- opal-prd: Fix opal-prd command line options

  HBRT OCC reset interface depends on service processor type.

  - FSP: reset_pm_complex()
  - BMC: process_occ_reset()

  We have both *occ* and *pm-complex* command line interfaces. This patch adds support to dispaly appropriate message depending on system type.

SP	Command	Action
FSP	opal-prd occ	display error message
FSP	opal-prd pm-complex	Call pm_complex_reset()
BMC	opal-prd occ	Call process_occ_reset()
BMC	opal-prd pm-complex	display error message

- opal-prd: detect service processor type and then make appropriate occ reset call.
- pflash: Fix erase command for unaligned start address

  The erase_range() function handles erasing the flash for a given start address and length, and can handle an unaligned start address and length. However in the unaligned start address case we are incorrectly calculating the remaining size which can lead to incomplete erases.

  If we're going to update the remaining size based on what the start address was then we probably want to do that before we overide the origin start address. So rearrange the code so that this is indeed the case.
- external/gard: Print an error if run on an FSP system

Simulators

- mambo: Add mambo socket program

  This adds a program that can be run inside a mambo simulator in linux userspace which enables TCP sockets to be proxied in and out of the simulator to the host.

  Unlike mambo bogusnet, it's requires no linux or skiboot specific drivers/infrastructure to run.

  Run inside the simulator:

  - to forward host ssh connections to sim ssh server: `./mambo-socket-proxy -h 10022 -s 22`, then connect to port 10022 on your host with `ssh -p 10022 localhost`
  - to allow http proxy access from inside the sim to local http proxy: `./mambo-socket-proxy -b proxy.mynetwork -h 3128 -s 3128`

  Multiple connections are supported.
- idle: disable stop*_lite POWER9 idle states for Mambo platform

  Mambo prior to Mambo.7.8.21 had a bug where the stop idle instruction with PSSCR[ESL]=PSSCR[EC]=0 would resume with MSR set as though it had taken a system reset interrupt.

  Linux currently executes this instruction with MSR already set that way, so the problem went unnoticed. A proposed patch to Linux changes that, and causes the idle code to crash. Work around this by disabling lite stop states for the mambo platform for now.

4.1.70 skiboot-5.9-rc2

skiboot v5.9-rc2 was released on Monday October 16th 2017. It is the second release candidate of skiboot 5.9, which will become the new stable release of skiboot following the 5.8 release, first released August 31st 2017.

skiboot v5.9-rc2 contains all bug fixes as of *skiboot-5.4.8* and *skiboot-5.1.21* (the currently maintained stable releases). We do not currently expect to do any 5.8.x stable releases.

For how the skiboot stable releases work, see *Skiboot stable tree rules and releases* for details.

The current plan is to cut the final 5.9 by October 17th, with skiboot 5.9 being for all POWER8 and POWER9 platforms in op-build v1.20 (Due October 18th). This release will be targetted to early POWER9 systems.

Over *skiboot-5.9-rc1*, we have the following changes:

- opal-prd: Fix memory leak
- hdata/i2c: update the list of known i2c devs

  This updates the list of known i2c devices - as of HDAT spec v10.5e - so that they can be properly identified during the hdat parsing.

- hdata/i2c: log unknown i2c devices

  An i2c device is unknown if either the i2c device list is outdated or the device is marked as unknown (0xFF) in the hdat.

- opal/cpu: Mark the core as bad while disabling threads of the core.

  If any of the core fails to sync its TB during chipTOD initialization, all the threads of that core are disabled. But this does not make linux kernel to ignore the core/cpus. It crashes while bringing them up with below backtrace:

```
[   38.883898] kexec_core: Starting new kernel
cpu 0x0: Vector: 300 (Data Access) at [c0000003f277b730]
    pc: c0000000001b9890: internal_create_group+0x30/0x304
    lr: c0000000001b9880: internal_create_group+0x20/0x304
    sp: c0000003f277b9b0
   msr: 900000000280b033
   dar: 40
 dsisr: 40000000
  current = 0xc0000003f9f41000
  paca    = 0xc00000000fe00000   softe: 0        irq_happened: 0x01
    pid   = 2572, comm = kexec
Linux version 4.13.2-openpower1 (jenkins@p89) (gcc version 6.4.0 (Buildroot 2017.
↪08-00006-g319c6e1)) #1 SMP Wed Sep 20 05:42:11 UTC 2017
enter ? for help
[c0000003f277b9b0] c0000000008a8780 (unreliable)
[c0000003f277ba50] c00000000041c3ac topology_add_dev+0x2c/0x40
[c0000003f277ba70] c00000000006b078 cpuhp_invoke_callback+0x88/0x170
[c0000003f277bac0] c00000000006b22c cpuhp_up_callbacks+0x54/0xb8
[c0000003f277bb10] c00000000006bc68 cpu_up+0x11c/0x168
[c0000003f277bbc0] c00000000002f0e0 default_machine_kexec+0x1fc/0x274
[c0000003f277bc50] c00000000002e2d8 machine_kexec+0x50/0x58
[c0000003f277bc70] c0000000000de4e8 kernel_kexec+0x98/0xb4
[c0000003f277bce0] c00000000008b0f0 SyS_reboot+0x1c8/0x1f4
[c0000003f277be30] c00000000000b118 system_call+0x58/0x6c
```

- hw/imc: pause microcode at boot

  IMC nest counters has both in-band (ucode access) and out of band access to it. Since not all nest counter configurations are supported by ucode, out of band tools are used to characterize other configuration.

  So it is prefer to pause the nest microcode at boot to aid the nest out of band tools. If the ucode not paused and OS does not have IMC driver support, then out to band tools will race with ucode and end up getting undesirable values. Patch to check and pause the ucode at boot.

  OPAL provides APIs to control IMC counters. OPAL_IMC_COUNTERS_INIT is used to initialize these counters at boot. OPAL_IMC_COUNTERS_START and OPAL_IMC_COUNTERS_STOP API calls should be used to start and pause these IMC engines. *doc/opal-api/opal-imc-counters.rst* details the OPAL APIs and their usage.

- xive: Fix VP free block group mode false-positive parameter check

  The check to ensure the buddy allocation idx is aligned to its allocation order was not taking into account the allocation split. This would result in opal_xive_free_vp_block failures despite giving the same value as returned by opal_xive_alloc_vp_block.

E.g., starting then stopping 4 KVM guests gives the following pattern in the host:

```
opal_xive_alloc_vp_block(5)=0x45000020
opal_xive_alloc_vp_block(5)=0x45000040
opal_xive_alloc_vp_block(5)=0x45000060
opal_xive_alloc_vp_block(5)=0x45000080
opal_xive_free_vp_block(0x45000020)=-1
opal_xive_free_vp_block(0x45000040)=0
opal_xive_free_vp_block(0x45000060)=-1
opal_xive_free_vp_block(0x45000080)=0
```

- hw/p8-i2c: Fix deadlock in p9_i2c_bus_owner_change

  When debugging a system where Linux was taking soft lockup errors with two CPUs stuck in OPAL:

CPU0	CPU1
lock	
p8_i2c_recover	
opal_handle_interrupt	sync_timer cancel_timer p9_i2c_bus_owner_change occ_p9_interrupt xive_source_interrupt opal_handle_interrupt

  p8_i2c_recover() is a timer, and is stuck trying to take master->lock. p9_i2c_bus_owner_change() has taken master->lock, but then is stuck waiting for all timers to complete. We deadlock.

  Fix this by using cancel_timer_async().

- FSP/CONSOLE: Limit number of error logging

  Commit c8a7535f (FSP/CONSOLE: Workaround for unresponsive ipmi daemon) added error logging when buffer is full. In some corner cases kernel may call this function multiple time and we may endup logging error again and again.

  This patch fixes it by generating error log only once.

- FSP/CONSOLE: Fix fsp_console_write_buffer_space() call

  Kernel calls fsp_console_write_buffer_space() to check console buffer space availability. If there is enough buffer space to write data, then kernel will call fsp_console_write() to write actual data.

  In some extreme corner cases (like one explained in commit c8a7535f) console becomes full and this function returns 0 to kernel (or space available in console buffer < next incoming data size). Kernel will continue retrying until it gets enough space. So we will start seeing RCU stalls.

  This patch keeps track of previous available space. If previous space is same as current means not enough space in console buffer to write incoming data. It may be due to very high console write operation and slow response from FSP -OR- FSP has stopped processing data (ex: because of ipmi daemon died). At this point we will start timer with timeout of SER_BUFFER_OUT_TIMEOUT (10 secs). If situation is not improved within 10 seconds means something went bad. Lets return OPAL_RESOURCE so that kernel can drop console write and continue.

- FSP/CONSOLE: Close SOL session during R/R

  Presently we are not closing SOL and FW console sessions during R/R. Host will continue to write to SOL buffer during FSP R/R. If there is heavy console write operation happening during FSP R/R (like running *top* command inside console), then at some point console buffer becomes full. fsp_console_write_buffer_space() returns 0 (or less than required space to write data) to host. While one thread is busy writing to console, if some other threads tries to write data to console we may see RCU stalls (like below) in kernel.

```
[ 2082.828363] INFO: rcu_sched detected stalls on CPUs/tasks: { 32} (detected by␣
↪16, t=6002 jiffies, g=23154, c=23153, q=254769)
[ 2082.828365] Task dump for CPU 32:
[ 2082.828368] kworker/32:3    R  running task        0  4637      2 0x00000884
[ 2082.828375] Workqueue: events dump_work_fn
[ 2082.828376] Call Trace:
[ 2082.828382] [c000000f1633fa00] [c00000000013b6b0] console_unlock+0x570/0x600␣
↪(unreliable)
[ 2082.828384] [c000000f1633fae0] [c00000000013ba34] vprintk_emit+0x2f4/0x5c0
[ 2082.828389] [c000000f1633fb60] [c00000000099e644] printk+0x84/0x98
[ 2082.828391] [c000000f1633fb90] [c0000000000851a8] dump_work_fn+0x238/0x250
[ 2082.828394] [c000000f1633fc60] [c0000000000ecb98] process_one_work+0x198/0x4b0
[ 2082.828396] [c000000f1633fcf0] [c0000000000ed3dc] worker_thread+0x18c/0x5a0
[ 2082.828399] [c000000f1633fd80] [c0000000000f4650] kthread+0x110/0x130
[ 2082.828403] [c000000f1633fe30] [c000000000009674] ret_from_kernel_thread+0x5c/
↪0x68
```

Hence lets close SOL (and FW console) during FSP R/R.

- FSP/CONSOLE: Do not associate unavailable console

  Presently OPAL sends associate/unassociate MBOX command for all FSP serial console (like below OPAL message). We have to check console is available or not before sending this message.

```
[ 5013.227994012,7] FSP: Reassociating HVSI console 1
[ 5013.227997540,7] FSP: Reassociating HVSI console 2
```

- FSP: Disable PSI link whenever FSP tells OPAL about impending R/R

  Commit 42d5d047 fixed scenario where DPO has been initiated, but FSP went into reset before the CEC power down came in. But this is generic issue that can happen in normal shutdown path as well.

  Hence disable PSI link as soon as we detect FSP impending R/R.

- fsp: return OPAL_BUSY_EVENT on failure sending FSP_CMD_POWERDOWN_NORM Also, return OPAL_BUSY_EVENT on failure sending FSP_CMD_REBOOT / DEEP_REBOOT.

  We had a race condition between FSP Reset/Reload and powering down the system from the host:

  Roughly:

#	FSP	Host
1	Power on	
2		Power on
3	(inject EPOW)	
4	(trigger FSP R/R)	
5		Processes EPOW event, starts shutting down
6		calls OPAL_CEC_POWER_DOWN
7	(is still in R/R)	
8		gets OPAL_INTERNAL_ERROR, spins in opal_poll_events
9	(FSP comes back)	
10		spinning in opal_poll_events
11	(thinks host is running)	

  The call to OPAL_CEC_POWER_DOWN is only made once as the reset/reload error path for fsp_sync_msg() is to return -1, which means we give the OS OPAL_INTERNAL_ERROR, which is fine, except that our own API docs give us the opportunity to return OPAL_BUSY when trying again later may be successful, and we're ambiguous as to if you should retry on OPAL_INTERNAL_ERROR.

For reference, the linux code looks like this:

```
static void __noreturn pnv_power_off(void)
{
        long rc = OPAL_BUSY;

        pnv_prepare_going_down();

        while (rc == OPAL_BUSY || rc == OPAL_BUSY_EVENT) {
                rc = opal_cec_power_down(0);
                if (rc == OPAL_BUSY_EVENT)
                        opal_poll_events(NULL);
                else
                        mdelay(10);
        }
        for (;;)
                opal_poll_events(NULL);
}
```

Which means that *practically* our only option is to return OPAL_BUSY or OPAL_BUSY_EVENT.

We choose OPAL_BUSY_EVENT for FSP systems as we do want to ensure we're running pollers to communicate with the FSP and do the final bits of Reset/Reload handling before we power off the system.

4.1.71 skiboot-5.9-rc3

skiboot v5.9-rc3 was released on Wednesday October 18th 2017. It is the third release candidate of skiboot 5.9, which will become the new stable release of skiboot following the 5.8 release, first released August 31st 2017.

skiboot v5.9-rc3 contains all bug fixes as of *skiboot-5.4.8* and *skiboot-5.1.21* (the currently maintained stable releases). We do not currently expect to do any 5.8.x stable releases.

For how the skiboot stable releases work, see *Skiboot stable tree rules and releases* for details.

The current plan is to cut the final 5.9 by October 20th, with skiboot 5.9 being for all POWER8 and POWER9 platforms in op-build v1.20 (Due October 18th). This release will be targetted to early POWER9 systems.

Over *skiboot-5.9-rc2*, we have the following changes:

- Improvements to vpd device tree entries

  Previously we would miss some properties

- Revert "npu2: Add vendor cap for IRQ testing"

  This reverts commit 9817c9e29b6fe00daa3a0e4420e69a97c90eb373 which seems to break setting the PCI dev flag and the link number in the PCIe vendor specific config space. This leads to the device driver attempting to re-init the DL when it shouldn't which can cause HMI's.

- hw/imc: Fix IMC Catalog load for DD2.X processors

- cpu: Add OPAL_REINIT_CPUS_TM_SUSPEND_DISABLED

  Add a new CPU reinit flag, "TM Suspend Disabled", which requests that CPUs be configured so that TM (Transactional Memory) suspend mode is disabled.

  Currently this always fails, because skiboot has no way to query the state. A future hostboot change will add a mechanism for skiboot to determine the status and return an appropriate error code.

4.1.72 skiboot-5.9-rc4

skiboot v5.9-rc4 was released on Thursday October 19th 2017. It is the fourth release candidate of skiboot 5.9, which will become the new stable release of skiboot following the 5.8 release, first released August 31st 2017.

skiboot v5.9-rc4 contains all bug fixes as of *skiboot-5.4.8* and *skiboot-5.1.21* (the currently maintained stable releases). We do not currently expect to do any 5.8.x stable releases.

For how the skiboot stable releases work, see *Skiboot stable tree rules and releases* for details.

The current plan is to cut the final 5.9 by October 20th, with skiboot 5.9 being for all POWER8 and POWER9 platforms in op-build v1.20 (Due October 18th, so we're running a bit behind there). This release will be targetted to early POWER9 systems.

Over *skiboot-5.9-rc3*, we have the following changes:

- phb4: Fix PCIe GEN4 on DD2.1 and above

  In this change: eef0e197ab PHB4: Default to PCIe GEN3 on POWER9 DD2.00

  We clamped DD2.00 parts to GEN3 but unfortunately this change also applies to DD2.1 and above.

  This fixes this to only apply to DD2.00.

- occ-sensors : Add OCC inband sensor region to exports (useful for debugging)

Two SRESET fixes:

- core: direct-controls: Fix clearing of special wakeup

  'special_wakeup_count' is incremented on successfully asserting special wakeup. So we will never clear the special wakeup if we check 'special_wakeup_count' to be zero. Fix this issue by checking the 'special_wakeup_count' to 1 in dctl_clear_special_wakeup().

- core/direct-controls: increase special wakeup timeout on POWER9

  Some instances have been observed where the special wakeup assert times out. The current timeout is too short for deeper sleep states. Hostboot uses 100ms, so match that.

4.1.73 skiboot-5.9-rc5

skiboot v5.9-rc5 was released on Monday October 23rd 2017 approximately 32,000ft above somewhere north of Tucson, Arizona. It is the fifth release candidate of skiboot 5.9, which will become the new stable release of skiboot following the 5.8 release, first released August 31st 2017.

skiboot v5.9-rc5 contains all bug fixes as of *skiboot-5.4.8* and *skiboot-5.1.21* (the currently maintained stable releases). We do not currently expect to do any 5.8.x stable releases.

For how the skiboot stable releases work, see *Skiboot stable tree rules and releases* for details.

The current plan is to cut the final 5.9 very shortly, with skiboot 5.9 being for all POWER8 and POWER9 platforms in op-build v1.20 (Due October 18th, so we're running a bit behind there). This release will be targetted to early POWER9 systems.

Over *skiboot-5.9-rc3*, we have the following changes:

- opal/hmi: Workaround Power9 hw logic bug for couple of TFMR TB errors.
- opal/hmi: Fix TB reside and HDEC parity error recovery for power9
- phb4: Escalate freeze to fence to avoid checkstop

  Freeze events such as MMIO loads can cause the PHB to lose it's limited powerbus credits. If all credits are used and a further MMIO will cause a checkstop.

To work around this, we escalate the troublesome freeze events to a fence. The fence will cause a full PHB reset which resets the powerbus credits and avoids the checkstop.

- phb4: Update some init registers

  New inits based on next PHB4 workbook. Increases some timeouts to avoid some spurious error conditions.

- phb4: Enable PHB MMIO in phb4_root_port_init()

  Linux EEH flow is somewhat broken. It saves the PCIe config space of the PHB on boot, which it then uses to restore on EEH recovery. It does this to restore MMIO bars and some other pieces.

  Unfortunately this save is done before any drivers are bound to devices under the PHB. A number of other things are configured in the PHB after drivers start, hence some configuration space settings aren't saved correctly. These include bus master and MMIO bits in the command register.

  Linux tried to hack around this in this linux commit `bf898ec5cb` powerpc/eeh: Enable PCI_COMMAND_MASTER for PCI bridges This sets the bus master bit but ignores the MMIO bit.

  Hence we lose MMIO after a full PHB reset. This causes the next MMIO access to the device to fail and for us to perform a PE freeze recovery, which still doesn't set the MMIO bit and hence we still fail.

  This works around this by forcing MMIO on during phb4_root_port_init().

  With this we can recovery from a PHB fence event on POWER9.

- phb4: Reduce link degraded message log level to debug

  If we hit this message we'll retry and fix the problem. If we run out of retries and can't fix the problem, we'll still print a log message at error level indicating a problem.

- phb4: Fix GEN3 for DD2.00

  In this fix: 62ac7631ae phb4: Fix PCIe GEN4 on DD2.1 and above

  We fixed DD2.1 GEN4 but broke DD2.00 as GEN3.

  This fixes DD2.00 back to GEN3. This time for sure!

4.1.74 skiboot-5.9.1

skiboot 5.9.1 was released on Tuesday November 14th, 2017. It replaces *skiboot-5.9* as the current stable release in the 5.9.x series.

Over *skiboot-5.9*, we have two NPU2 (NVLink2) fixes and two XIVE bug fixes:

- npu2: hw-procedures: Refactor reset_ntl procedure

  Change the implementation of reset_ntl to match the latest programming guide documentation.

- npu2: hw-procedures: Add phy_rx_clock_sel()

  Change the RX clk mux control to be done by software instead of HW. This avoids glitches caused by changing the mux setting.

- xive: Fix ability to clear some EQ flags

  We could never clear "unconditional notify" and "escalate"

- xive: Update inits for DD2.0

  This updates some inits based on information from the HW designers. This includes enabling some new DD2.0 features that we don't yet exploit.

4.1.75 skiboot-5.9.2

skiboot 5.9.2 was released on Thursday November 16th, 2017. It replaces *skiboot-5.9.1* as the current stable release in the 5.9.x series.

Over *skiboot-5.9.1*, we have a few PHB4 (PCI) fixes, an i2c fix for POWER9 platforms to avoid conflicting with the OCC use and an important NPU2 (NVLink2) fix.

- phb4: Fix lane equalisation setting

  Fix cut and paste from phb3. The sizes have changes now we have GEN4, so the check here needs to change also

  Without this we end up with the default settings (all '7') rather than what's in HDAT.

- phb4: Fix PE mapping of M32 BAR

  The M32 BAR is the PHB4 region used to map all the non-prefetchable or 32-bit device BARs. It's supposed to have its segments remapped via the MDT and Linux relies on that to assign them individual PE#.

  However, we weren't configuring that properly and instead used the mode where PE# == segment#, thus causing EEH to freeze the wrong device or PE#.

- phb4: Fix lost bit in PE number on config accesses

  A PE number can be up to 9 bits, using a uint8_t won't fly..

  That was causing error on config accesses to freeze the wrong PE.

- phb4: Update inits

  New init value from HW folks for the fence enable register.

  This clears bit 17 (CFG Write Error CA or UR response) and bit 22 (MMIO Write DAT_ERR Indication) and sets bit 21 (MMIO CFG Pending Error)

- npu2: Move to new GPU memory map

  There are three different ways we configure the MCD and memory map.

  1. Old way (current way) Skiboot configures the MCD and puts GPUs at 4TB and below
  2. New way with MCD Hostboot configures the MCD and skiboot puts GPU at 4TB and above
  3. New way without MCD No one configures the MCD and skiboot puts GPU at 4TB and below

  The change keeps option 1 and adds options 2 and 3.

  The different configurations are detected using certain scoms (see patch).

  Option 1 will go away eventually as it's a configuration that can cause xstops or data integrity problems. We are keeping it around to support existing hostboot.

  Option 2 supports only 4 GPUs and 512GB of memory per socket.

  Option 3 supports 6 GPUs and 4TB of memory but may have some performance impact.

- p8-i2c: Don't write the watermark register at init

  On P9 the I2C master is shared with the OCC. Currently the watermark values are set once at init time which is bad for two reasons:

  1. We don't take the OCC master lock before setting it. Which may cause issues if the OCC is currently using the master.
  2. The OCC might change the watermark levels and we need to reset them.

  Change this so that we set the watermark value when a new transaction is started rather than at init time.

4.1.76 skiboot-5.9.3

skiboot 5.9.3 was released on Wednesday November 22nd, 2017. It replaces *skiboot-5.9.2* as the current stable release in the 5.9.x series.

Over *skiboot-5.9.2*, we have one NPU2/NVLink2 fix that causes the machine to crash hard in the event of hardware error rather than crash mysteriously later on whenever the NVLink2 links are used.

That fix is:

- npu2: hw-procedures: Add check_credits procedure

  As an immediate mitigator for a current hardware glitch, add a procedure that can be used to validate NTL credit values. This will be called as a safeguard to check that link training succeeded.

  Assert that things are exactly as we expect, because if they aren't, the system will experience a catastrophic failure shortly after the start of link traffic.

4.1.77 skiboot-5.9.4

skiboot 5.9.4 was released on Wednesday November 29th, 2017. It replaces *skiboot-5.9.3* as the current stable release in the 5.9.x series.

Over *skiboot-5.9.3*, we have one NPU2/NVLink2 fix that works around a potential glitch (the one *skiboot-5.9.3* would hard crash on rather than let a system continue to run until it mysteriously crashed later on).

That fix is in two parts:

- npu2: hw-procedures: Change phy_rx_clock_sel values to recover from a potential glitch.
- npu2: hw-procedures: Manipulate IOVALID during training

  Ensure that the IOVALID bit for this brick is raised at the start of link training, in the reset_ntl procedure.

  Then, to protect us from a glitch when the PHY clock turns off or gets chopped, lower IOVALID for the duration of the phy_reset and phy_rx_dccal procedures.

4.1.78 skiboot-5.9.5

skiboot 5.9.5 was released on Wednesday December 13th, 2017. It replaces *skiboot-5.9.4* as the current stable release in the 5.9.x series.

Over *skiboot-5.9.4*, we have a few bug fixes, they are:

- Fix *extremely* rare race in timer code.
- xive: Ensure VC informational FIRs are masked

  Some HostBoot versions leave those as checkstop, they are harmless and can sometimes occur during normal operations.

- xive: Fix occasional VC checkstops in xive_reset

  The current workaround for the scrub bug described in __xive_cache_scrub() has an issue in that it can leave dirty invalid entries in the cache.

  When cleaning up EQs or VPs during reset, if we then remove the underlying indirect page for these entries, the XIVE will checkstop when trying to flush them out of the cache.

  This replaces the existing workaround with a new pair of workarounds for VPs and EQs:

  - The VP one does the dummy watch on another entry than the one we scrubbed (which does the job of pushing old stores out) using an entry that is known to be backed by a permanent indirect page.
  - The EQ one switches to a more efficient workaround which consists of doing a non-side-effect ESB load from the EQ's ESe control bits.

- io: Add load_wait() helper

  This uses the standard form twi/isync pair to ensure a load is consumed by the core before continuing. This can be necessary under some circumstances for example when having the following sequence:

  - Store reg A
  - Load reg A (ensure above store pushed out)
  - delay loop
  - Store reg A

  IE, a mandatory delay between 2 stores. In theory the first store is only guaranteed to rach the device after the load from the same location has completed. However the processor will start executing the delay loop without waiting for the return value from the load.

  This construct enforces that the delay loop isn't executed until the load value has been returned.

- xive: Do not return a trigger page for an escalation interrupt

  This is bogus, we don't support them. (Thankfully the callers didn't actually try to use this on escalation interrupts).

- xive: Mark a freed IRQ's IVE as valid and masked

  Removing the valid bit means a FIR will trip if it's accessed inadvertently. Under some circumstances, the XIVE will speculatively access an IVE for a masked interrupt and trip it. So make sure that freed entries are still marked valid (but masked).

- hw/nx: Fix NX BAR assignments

  The NX rng BAR is used by each core to source random numbers for the DARN instruction. Currently we configure each core to use the NX rng of the chip that it exists on. Unfortunately, the NX can be deconfigured by hostboot and in this case we need to use the NX of a different chip.

  This patch moves the BAR assignments for the NX into the normal nx-rng init path. This lets us check if the normal (chip local) NX is active when configuring which NX a core should use so that we can fallback gracefully.

4.1.79 skiboot-5.9.6

skiboot 5.9.6 was released on Friday December 15th, 2017. It replaces *skiboot-5.9.5* as the current stable release in the 5.9.x series.

Over *skiboot-5.9.5*, we have a few bug fixes, they are:

- sensors: occ: Skip counter type of sensors

  Don't add counter type of sensors to device-tree as they don't fit into hwmon sensor interface.

- p9_stop_api updates to support IMC across deep stop states.

- opal/xscom: Add recovery for lost core wakeup scom failures.

  Due to a hardware issue where core responding to scom was delayed due to thread reconfiguration, leaves the SCOM logic in a state where the subsequent scom to that core can get errors. This is affected for Core PC scom registers in the range of 20010A80-20010ABF

The solution is if a xscom timeout occurs to one of Core PC scom registers in the range of 20010A80-20010ABF, a clearing scom write is done to 0x20010800 with data of '0x00000000' which will also get a timeout but clears the scom logic errors. After the clearing write is done the original scom operation can be retried.

The scom timeout is reported as status 0x4 (Invalid address) in HMER[21-23].

CHAPTER

FIVE

INDICES AND TABLES

- genindex
- modindex
- search