Minidump

Mukesh Ojha <quic_mojha@quicinc.com> (Staff Engineer)
Elliot Berman <quic_eberman@quicinc.com> (Senior Engineer)

Qualcomm Innovation Center, Inc.

Problem Statement

Devices in engineering mode frequently provide a mechanism for generating full system DDR dump. However, for end user devices, it is not feasible to capture the entire DDR content and transfer them electronically. Now a days, typical size of DDR on a premium tier phones is 12 GB and growing, so the problem is going to be worse.

What is Minidump?

- Minidump provides a mechanism to nominate regions of memory that should be included in the DDR dump
- Minidump could be triggered by kernel panic or by other subsystem crash (NOC error, watchdog bite, firmware panic, more)
  - Firmware does the minidump capture

Why Minidump?

- Minidump is primarily for a device deployed to the end user
- Deployed devices don’t want to send full DDR dump
- Crashes could be due to range of hardware or software bugs
- Kernel may not have opportunity to run but it may contain useful information
  - At Qualcomm, ~30% of crashes don’t come from kernel panic
- The amount of useful and predefined data collected is limited. Hence, ability to debug issue off-target is limited.

What about kdump?

✓ Flexibility: any userspace to collect dump and save/send it

What about kdump?

✓ Flexibility: whatever userspace you want to collect dump and save/send it

✕ Memory overhead: large reserved memory region for crash kernel (128MB for arm64)

What about kdump?

✓ Flexibility: whatever userspace you want to collect dump and save/send it

✗ Memory overhead: large reserved memory region for crash kernel (128MB for arm64)

✗ Two kernel boots to come back to normal operation (once to crash kernel, then normal boot)

What about kdump?

✓ Flexibility: whatever userspace you want to collect dump and save/send it

✗ Memory overhead: large reserved memory region for crash kernel (128MB for arm64)

✗ Two kernel boots to come back to normal operation (once to crash kernel, then normal boot)

✗ Only kernel triggers kdump

What about pstore?

✓ Low overhead during dump collection

- No extra kernel boots, userspace can collect dump after reboot

What about pstore?

✓ Low overhead during dump collection

- No extra kernel boots, userspace can collect dump after reboot

✗ Collection time overhead to copy to oops device

What about pstore?

✓ Low overhead during dump collection

- No extra kernel boots, userspace can collect dump after reboot

✗ Collection time overhead to copy to oops device

✗ Requires memory reservation for a ramoops region

What about pstore?

✓ Low overhead during dump collection

- No extra kernel boots, userspace can collect dump after reboot

✗ Collection time overhead to copy to oops device

✗ Requires memory reservation for a ramoops region

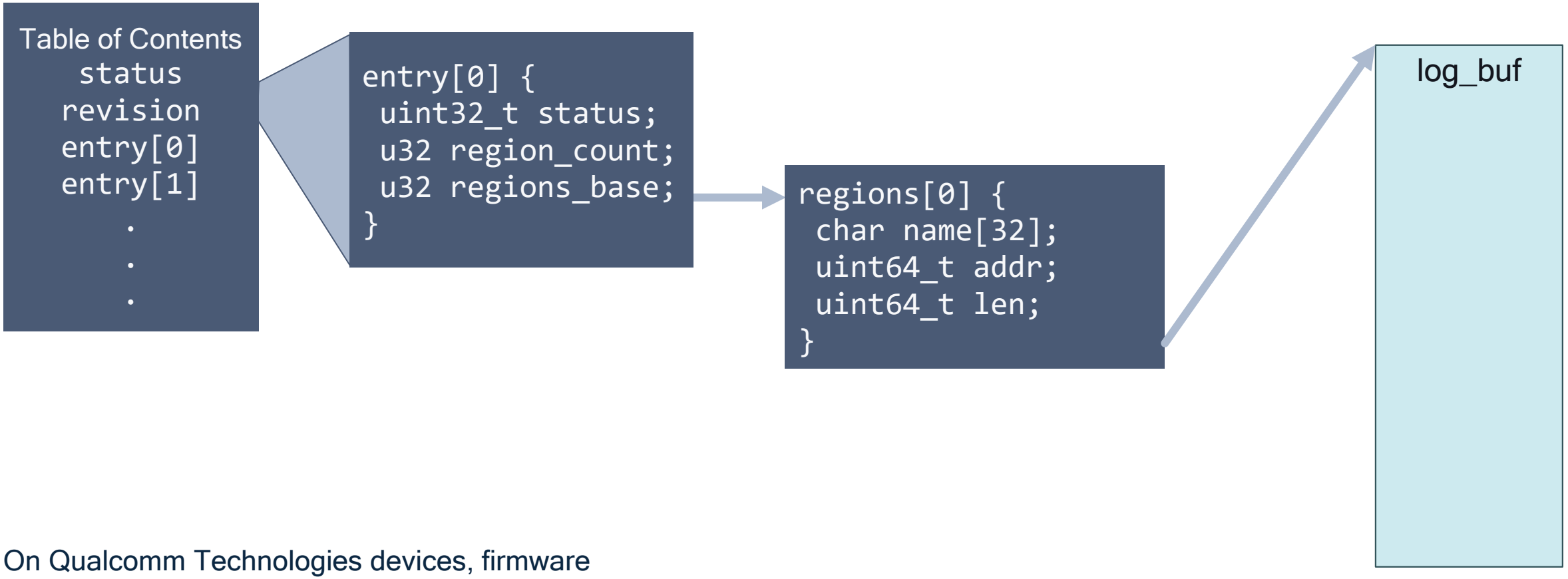
✗ Limited to memory that Linux can reliably copy from before crash

Minidump Differences

- Minimize memory overhead to capture a small dump
  - Maintain table of physical addresses and sizes
  - No need to copy dmesg buffers, ftrace buffers, etc.
- Reduced collection time overhead: firmware captures the minidump regions directly to storage medium

Minidump Table

Note: some fields omitted for brevity



On Qualcomm Technologies devices, firmware traverses the Table of Contents, creates ELF with the regions, and dumps the ELF to storage medium (typically eMMC/UFS/SD card)

Minidump for Application processor (APSS)

- Older firmware implementation of the minidump has put restriction on the number of APSS regions could register is 200 based on the time it takes to collect all the regions and reset to next boot.
- Upfront memory is allocated for MAX_LIMIT region and its physical address saved in APSS TOC.
- Any kernel client can fill qcom_minidump_region and use minidump register API to register with the minidump.

```
/**
  * struct qcom_minidump_region - APSS Minidump
region information
  *
  * @name: Entry name, Minidump will dump binary
with this name.
  * @virt_addr: Virtual address of the entry.
  * @phys_addr: Physical address of the entry to
dump.
  * @size: Number of byte to dump from @address
location,
  * and it should be 4 byte aligned.
  */
struct qcom_minidump_region {
         char     name[MAX_NAME_LENGTH];
         void     *virt_addr;
         phys_addr_t        phys_addr;
         size_t   size;
};
```

Minidump patches status in upstream

- Sent v5 of Minidump driver patches along with the support to enable it for Qualcomm SoCs.

  https://lore.kernel.org/lkml/1694429639-21484-1-git-send-email-quic_mojha@quicinc.com/

- Current intention is to collect all the existing pstore records like dmesg, ftrace, console, pmsg.

Minidump support for Remote processors

- Registering minidump tables for remote processors like ADSP, CDSP, modem are already supported in kernel
  - Each remote processor maintains known list of segment to be dumped during recovery
- Core dump is read by user space on getting Uevent notification from devcoredump framework via devcd addition.
- qcom_minidump() in driver/remoteproc/qcom_common.c is high level API which takes care of remote processor's minidump.

Some debug information aimed to be captured in Minidump

- Initial boot up dmesg logs
- MMIO tracing
- IRQ statistics
- Timer summary
- Run queue information
- Dump stack of each core in any kind of crash
- Dump stack of task which are stuck in D state from a very long time
- Memory/Slab/page owner information

Minidump outside Qualcomm

- Useful for other SoC vendors?
- Useful for resource constrained virtual machines?

Thank you

Qualcomm

Follow us on:

For more information, visit us at:

qualcomm.com & qualcomm.com/blog

Minidump for Application processor (APSS)

- Enablement of minidump is controlled through a secure register.
  - Minidump bit need to be set to enable minidump mode.
  - Default mode is set to full dump. (already supported in up stream)
  - Collects both full dump and minidump if both bits are set.

- Minidump download type is controlled through IMEM cookies.
  - Default type is set to usb, dump gets downloaded to PC connected through USB to the SoC.
  - Can be change to type ufs/emmc/sd-card where dump gets copied to the target dedicated partition.