BMW IDCEvo :: Boot time KPI marker

1. Introduction: -

This feature is used to log time stamp starting from Boot loader stage till Kernel layer. Time stamp values are stored into DRAM in a particular format. Once OS boots up, we can read RAM logs to see boot up time of various stages and also init time of various kernel services and drivers.

To read time stamp logs from RAM, there is a driver in Linux kernel which exposes a file in procfs i.e. "/proc/bo

ttime". A user can read this file to retrieve all time stamp data logged in RAM.

2. Time stamp logging procedure: -

Time stamp starts from Boot loader stage, for this just after power ON, ROM code initializes internal RTC of SOC. But Time Stamp is initialized by EPBL.

Boot steps along with Boot Time logging is explained below.

Step I. ROM code (BL0) initialize RTC Clock setting.

Step II. BL0 initiate UFS setting and loads Boot loader - 1 (BL1) from UFS to internal SRAM and jump entry fu

nction of BL1.

Step III. BL1 that covers core/power initialization, loads EPBL from UFS to internal SRAM and jump entry function of EPBL.

EPBL (Exynos Primary BootLoader) is a small split version of EL3 monitor to load in advance.

Step IV. EPBL logs first RTC clock that is EPBL's begin time and loads Boot loader - 2 (BL2), it is explained below for both EVT0 and EVT1.

EVT0: EPBL initialize time stamp area of NPU SRAM and log first RTC clock that is begin time of EPBL. EPBL load BL2 from UFS to NPU SRAM and jump entry function of BL2.

EVT1: EPBL initialize time stamp area of ABOX SRAM and log first RTC clock that is begin time of EPBL. EPBL load BL2 from UFS to ABOX SRAM and jump entry function of BL2.

Step V. BL2 log RTC Clock that is start time of BL2.
BL2 initialize DRAM setting and BL2 loads LK from UFS to DRAM by requesting EPBL.
Before jumping EPBL, BL2 logs RTC Clock that is end time of BL2.

Step VI. EPBL load EL3 monitor (EL3-mon) from UFS to DRAM and jump to EL3-mon entry function.
EL3-mon initializes Secure feature and jump to entry function of LK.

Note: - EL3-mon covers secure/non-secure world configuration.

Step VII. LK logs its start and end time stamp before jumping to kernel or hypervisor.

EVT0: LK logs RTC clock that is start time of LK and LK copies time stamp data from NPU SRAM to DRAM before jumping to kernel or hypervisor, LK logs RTC time that is end time of LK.

It is shown in figure - 1.

EVT1: LK logs RTC clock that is start time of LK and LK copies time stamp data from ABOX SRAM to DRAM before jump to kernel or hypervisor, LK logs RTC time that is end time of LK.

It is shown in figure - 2.

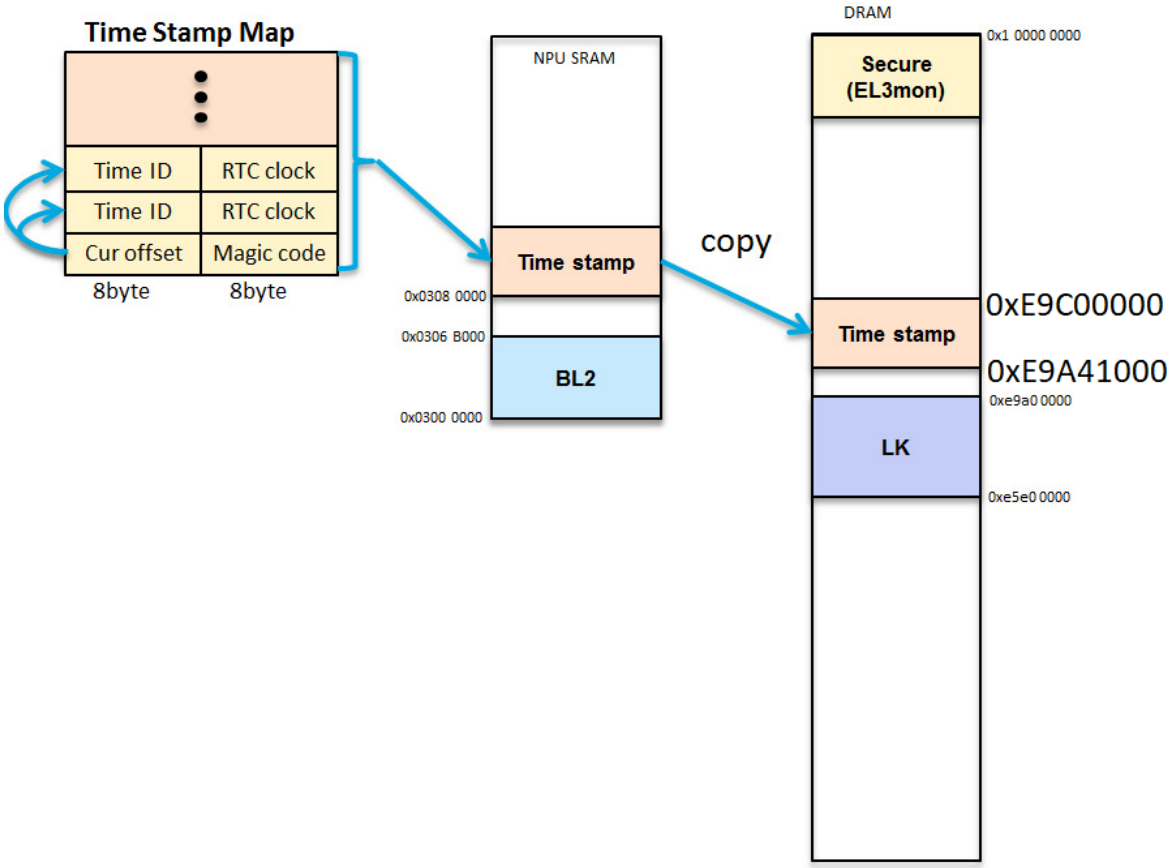


Figure - 1: EVT0

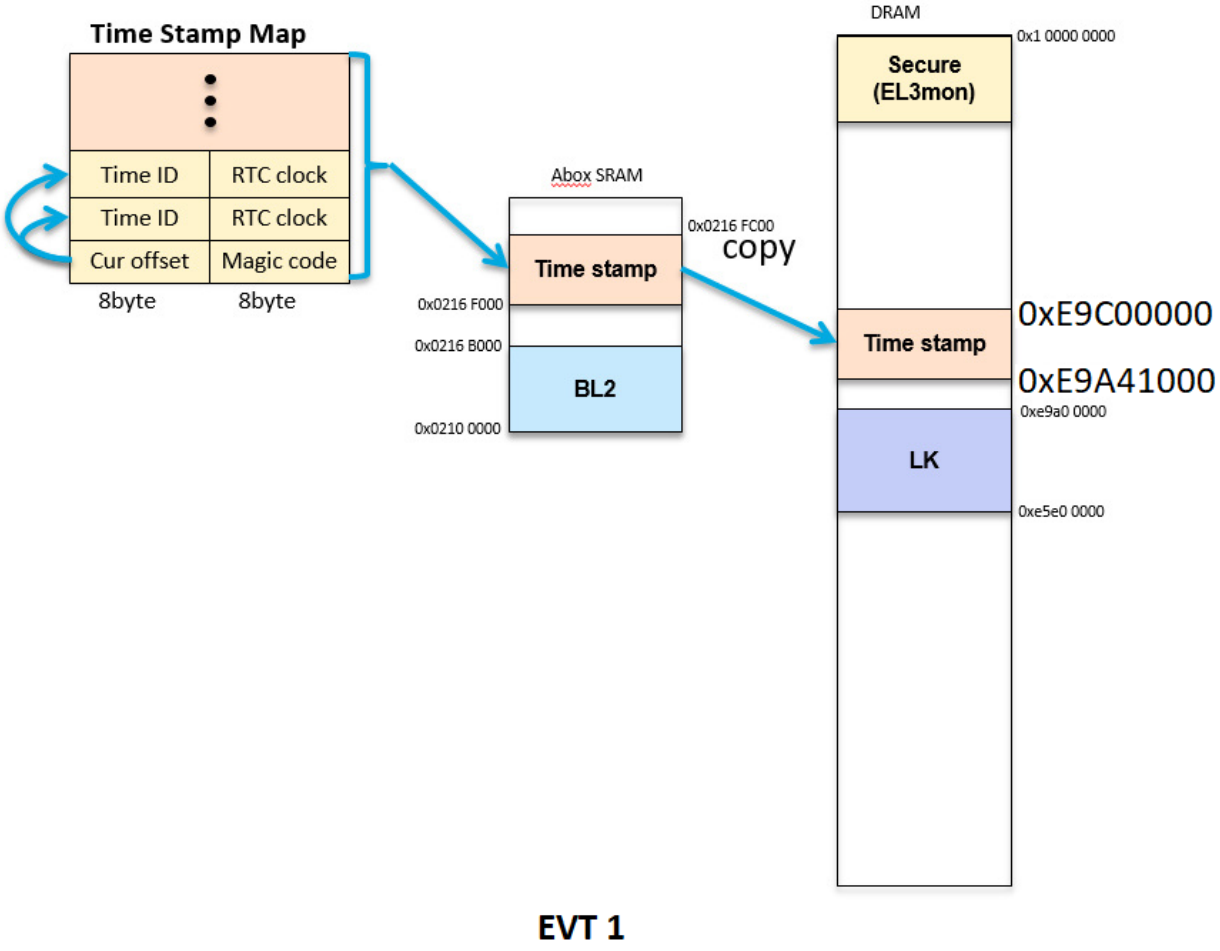


Figure - 2: EVT1

2.1 SRAM Layout: -

- NPU SRAM Layout (EVT 0): -

Module or Core	Base Address
Core 0	0x0308 1000
SFI	0x0308 2000

- ABOX SRAM Layout (EVT 1): -

Module or Core	Base Address
Core 0	0x0216 F800
SFI	0x0216 F000

3. Boot Time Stamp Structure or Format: -

3.1 Time stamp data structure

Each time stamp record is of 16 bytes. It constitutes of three fields, as explained below.

a. Category ID (4 Bytes) - This filed is used to identify type of OS or VM image or Boot loader stage (from EPBL). Example - EPBL, BL2, Bare Linux, Hypervisor, SYS, SFI, VM2 etc.

b. Internal ID (4 Bytes) - For each of the Category IDs, there is a set of Internal ID which is used to identify various sub components or stages. For, example for Category ID equal to SYS, there can be various Internal IDs like Kernel Init, MMU Init, PCIe driver Init etc.

c. Real Time Clock (8 Bytes) - This filed contains RTC time stamp, which is the value read by hardware register of internal RTC of the SOC.

This time stamp data structure of 16 bytes is shown in figure - 3.

Category ID (4 Bytes)	Internal ID (4 Bytes)	Real Time Clock (8 Bytes)

Time ID (Category ID + Internal ID)

Figure - 3: Time Stamp Data Structure

3.2 Time stamp map

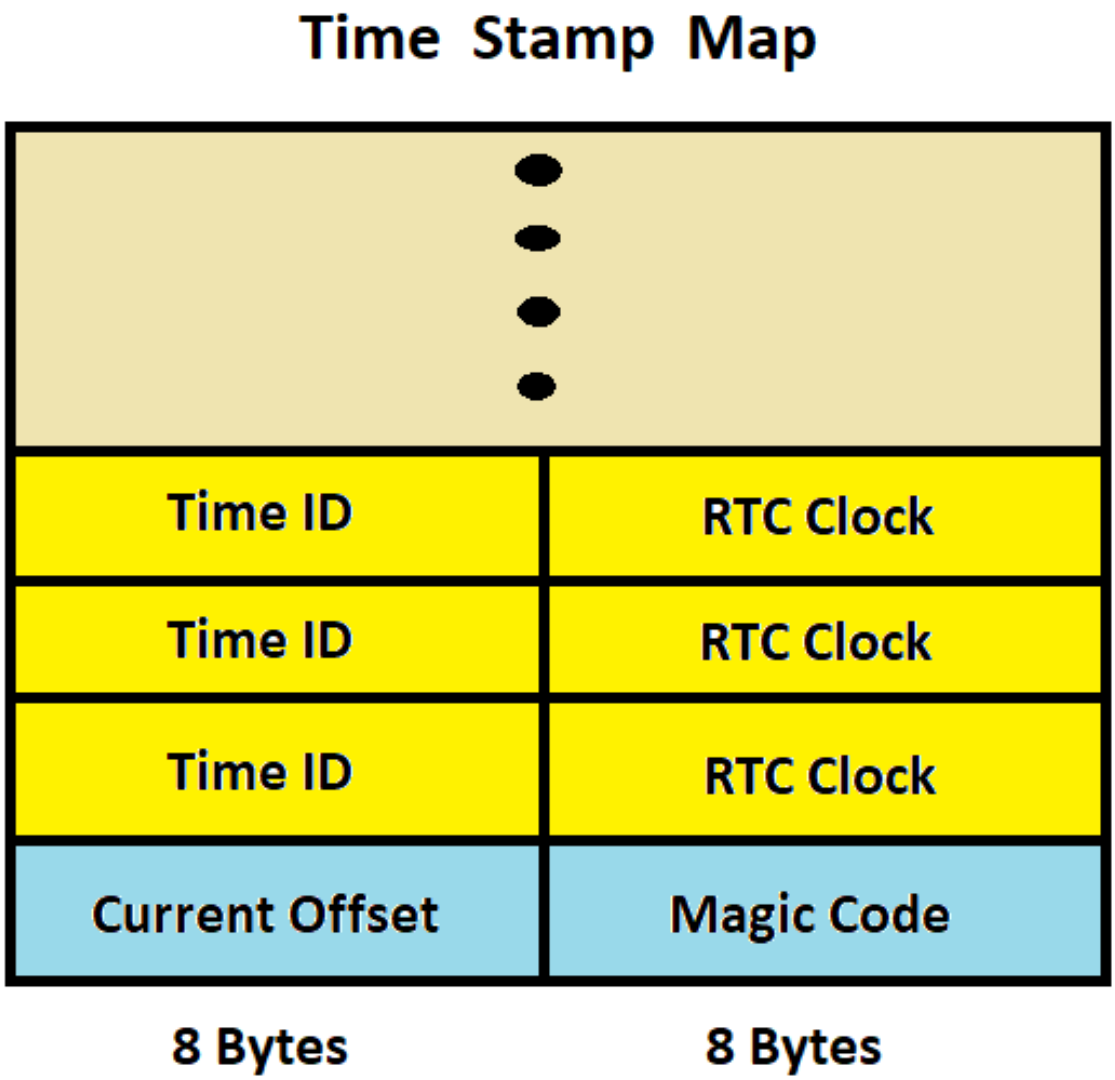


Figure - 4: Time Stamp Map

From figure - 4 above, we can see that at the beginning of every time stamp area, there is a Magic Code of 8 bytes its value is equal to 0x0BAD FAFA. Using Magic code driver recognises whether Boot Time Log is initialised or not.

Here there is also a field called "Current Offset", it resides at the beginning or time base area of any Time Stamp region. Main purpose of "Current Offset" field is to give index information to Boot Time KPI driver, whenever driver wants to write time stamp data then it

first reads "Current Offset" i.e. address of last time stamp data, then driver writes new time stamp data at address equal to (Current Offset + 16), as every time stamp data takes 16 bytes.

Region in yellow colour in figure - 4 is 16 bytes time stamp data, which is explained in figure 3.

3.3 Category ID (4 Bytes)

Category ID field of 4 bytes (figure - 3) in time stamp data field decides boot steps. Every boot step is encoded into 4 bytes category ID, it is shown in below table.

Table - 1: Category ID

Boot Step	Category ID
EPBL	0x1000 0000
BL2	0x2000 0000
EL3_MON	0x3000 0000
LK	0x4000 0000
BARE Linux	0x5000 0000
Hypervisor	0x6000 0000
SYS	0x7000 0000
IVI	0x8000 0000
AND	0x9000 0000
QNX	0xA000 0000
SFI	0xB000 0000
Domain VM2	0xC000 0000
Domain VM3	0xD000 0000
Domain VM4	0xE000 0000
Domain VM5	0xF000 0000

3.4 Internal ID (4 Bytes)

For each boot stages like EPBL, BL2, LK, OS/Kernel, there is Internal ID defined, to identify various sub stages or components init timings. Here only Hypervisor and Kernel/OS internal ID is mentioned as below.

Table 2. Internal ID of SFI (Safety Island)

Internal ID of SFI	Description
0x10000	SFI BL Entry
0x20000	SFI BL Image Loading Start
0x30000	SFI BL Image Loading End
0x40000	SFI BL Image Authenticating Start
0x50000	SFI BL Image Authenticating End
0x60000	SFI BL Image Jump

0x70000	SFI BL Image Finish

Table 3. Internal ID of HV (Hypervisor)

Internal ID of HV	Description
0xAA01	Start HV
0xAA02	Jump to GuestOS
0xAA04	Finish init. and waiting GuestOS loading by LK
0xAA08	GuestOS binary ready by LK

Table 4. Internal ID of SYS/IVI

Internal ID of SYS/IVI	Description
0xB000	MMU Init Done
0xB001	Kernel Init Done
0xB002	PCIe Driver Init Done
0xB003	Graphics Driver Init Done
0xB004	Ethernet Driver Init Done
0xB005	Initialize Memory
0xB006	Initialize Cache
0xB007	Initialize PTable
0xB008	Initialize VM Alloc
0xB009	Initialize IO remap
0xB00A	Initialize ESPFIX
0xB00B	Initialize PIT
0xB00C	Initialize Schedule
0xB00D	Initialize Console
0xB00E	Initialize Mem Enc
0xB100	Start Kernel
0xB110	Setup Arch Start
0xB111	Boot Mem Init Start
0xB112	Boot Mem Init Done
0xB113	Initialize Kernel Sparse
0xB114	Initialize Kernel Zone
0xB120	MMU Init Start
0xB200	Initialize Mem Leak
0xB201	Initialize Dbg Late
0xB202	Initialize Timer
0xB300	Thread Start
0xB310	Default Module Start
0xB311	Mount Root FS Done

0xB312	Execute Boot Args
0xB313	Execute Boot Args Done
0xB314	Another Working Start
0xB315	Drivers Start
0xB316	Drivers Done
0xB317	UFS Start
0xB318	UFS Done
0xB319	Initialize Platform Populate
0xB31A	Platform Populate Done
0xB31B	Internal Drivers Start
0xB500	Initialize ACPM MFC bus
0xB501	ACPM MFD bus Done
0xB700	Initialize Trace FS
0xB701	Trace FS Done
0xC000	Secondary Kernel Start
0xC100	Secondary Pre SMP InitCall
0xC200	Initialize Secondary SMP
0xCFFF	CPU core Idle

To know Internal ID of other Boot loader stages, refer the pdf file attached here.

4. DRAM Layout for Boot Time logging

In DRAM, **the address range allocated for Boot time memory logging is: 0xE9A4 1000 to 0xE9C0 0000**. **So, total size of memory allocated for boot time KPI marker is 1830912 bytes or 1788 KB.** For each CPU core **8 KB** of memory is used.

For each of VM sectors like VM2, VM3, VM4 and VM5, **64 KB** of memory is used.

CPU core logging size = 8KB

SFI area logging sized = 8KB

VM sector logging size = 64KB

Base address of various core and VM logging area is shown in table below.

Table 5. Core and VM area base address.

Core or VM region	Base address
Core 0	0xE9A4 1000
Core 1	0xE9A4 3000
Core 2	0xE9A4 5000
Core 3	0xE9A4 7000
Core 4	0xE9A4 9000
Core 5	0xE9A4 B000
Core 6	0xE9A4 D000
Core 7	0xE9A4 F000
SFI	0xE9A5 1000
SFI_reserved	0xE9A5 3000
VM2	0xE9A6 3000
VM3	0xE9A7 3000
VM4	0xE9A8 3000
VM5	0xE9A9 3000
Bare	0xE9AA 3000

4.1 DRAM Layout

This DRAM layout explains how Time stamp region (0xE9A4 1000 to 0xE9C0 0000) is divided into various sub regions to store time stamp data of various CPU cores and VM images or OSes. Base address of Core and VM region

is shown in Table - 5 above.

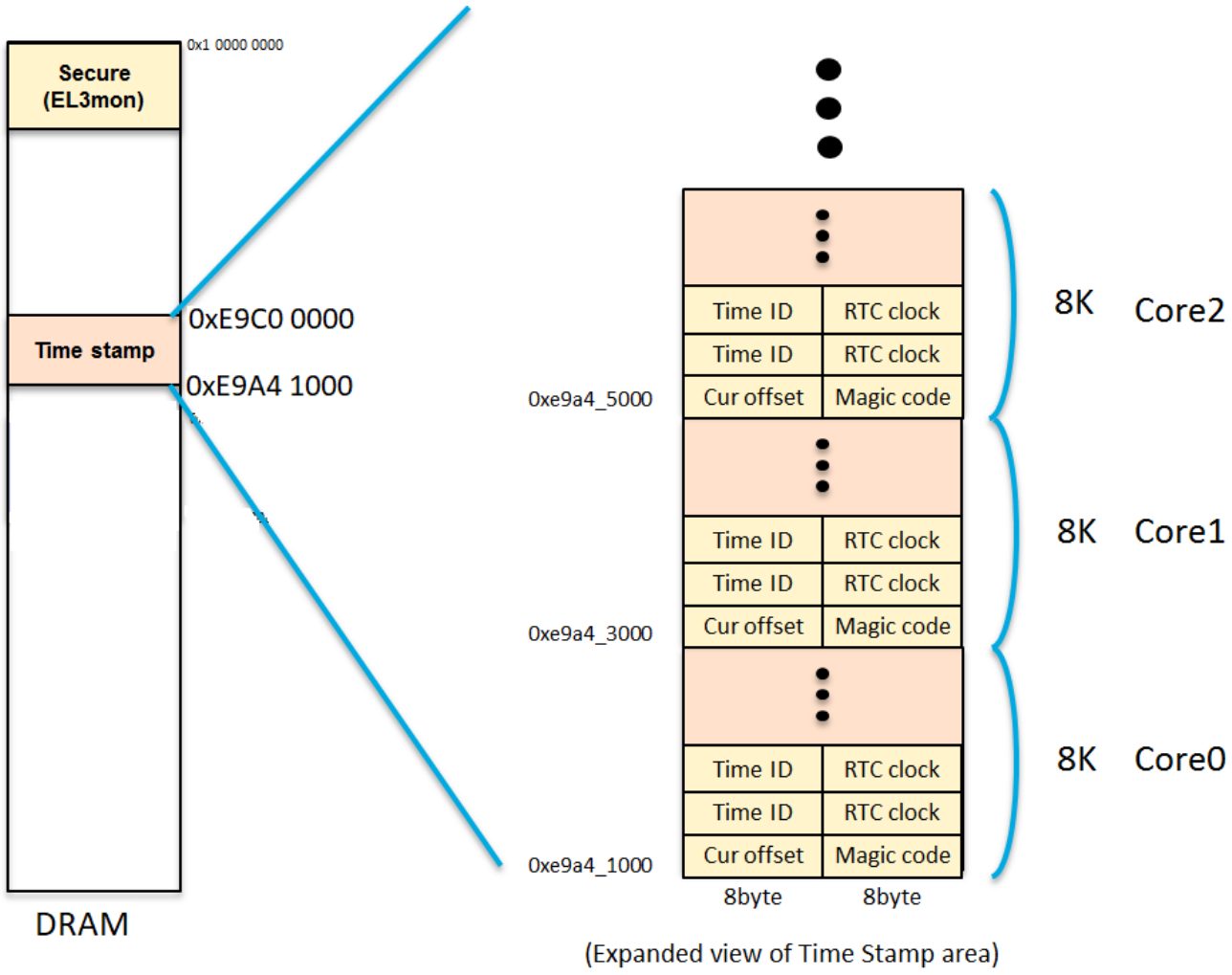


Figure - 5: DRAM Layout

4.2 Algorithm to log/write time stamp: -

Step I. Find time base of any CPU core region or VM region.

Step II. Read the value at time base, MSB 8 bytes as Current offset and next 8 bytes as Magic Code.

Step III. If value at Magic code field does not match with 0x0BAD FAFA, then it means memory log region is not initialized and write "Current Offset " filed as zero.

Step IV. Calculate "Next Offset" as new address to write new time stamp data.

Next Offset = Current Offset + 16

Example: - If last time stamp data is at address offset = 96, then Current Offset = 96. So, Next Offset to write new stamp data will be

Next Offset = 96 + 16 = 112

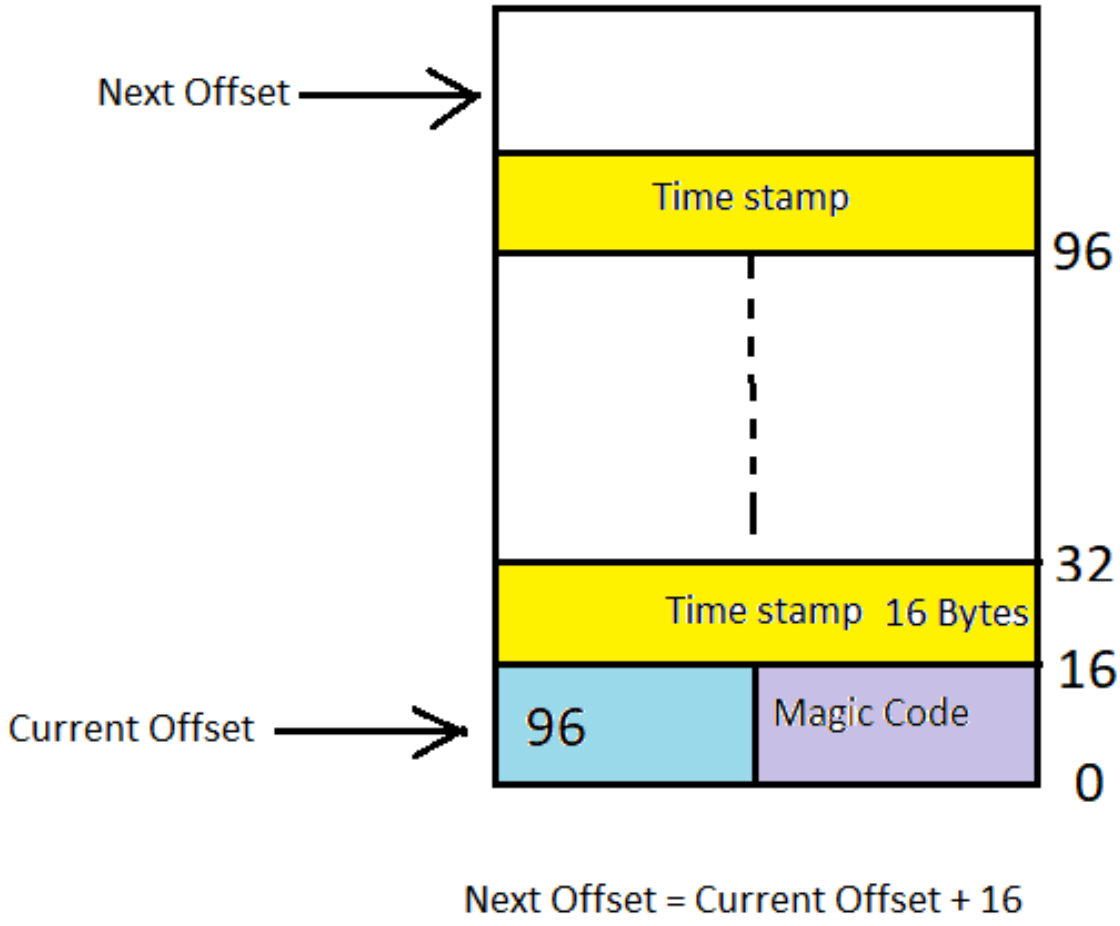


Figure - 6: Writing new time stamp data

Step V. Write VM Category ID, Internal ID and 8 bytes time stamp data at Next Offset address.

Step VI. Write "Next Offset" value calculated in Step IV at "Current Offset" address area, highlighted as cyan colour in Figure - 6.

Step VII. Repeat the step from beginning to write any new time stamp data.

5. Retrieving Log Data

To read time stamp data, Boot Time KPI driver exposes a file "boottime" in procfs. If we read this file, then it gives memory dump comprising of encoded Category ID, Internal ID and Time stamp in hex. So, to make it readable, there is a parser which decodes

this memory dump into understandable data.

Command to read time stamp log is as below.

#cat /proc/boottime | btime.py

Output: -

```
---------------------------------- [[[ Host Boot Time Table (2048)Hz]]] ----------------------------------
-- Core [0] ----------------------------------------------------------------------------------------------
EPBL                          Start Time (  113 clk/  0 sec 055.664 ms) dur (   63 clk/  0 sec 030.761 ms)
 BL2                          BEGIN Time (  176 clk/  0 sec 086.425 ms) dur (    0 clk/  0 sec 000.000 ms)
 BL2                        set pmic Time (  176 clk/  0 sec 086.425 ms) dur (    3 clk/  0 sec 001.464 ms)
 BL2                        init-ect Time (  179 clk/  0 sec 087.890 ms) dur (    6 clk/  0 sec 002.929 ms)
 BL2                  cp acpm-dbg2iram Time (  185 clk/  0 sec 090.820 ms) dur (    0 clk/  0 sec 000.000 ms)
 BL2                 cp acpm-frame2iram Time (  185 clk/  0 sec 090.820 ms) dur (    3 clk/  0 sec 001.464 ms)
```

```
BL2                cp acpm-plugin2iram Time (  188 clk/  0 sec 092.285 ms) dur (   16 clk/  0 sec 007.812 ms)
BL2                    cp dbg2sram Time (  204 clk/  0 sec 100.097 ms) dur (    5 clk/  0 sec 002.441 ms)
BL2                   init-cmu-dmc Time (  209 clk/  0 sec 102.539 ms) dur (    1 clk/  0 sec 000.488 ms)
BL2                    Enable APM Time (  210 clk/  0 sec 103.027 ms) dur (    7 clk/  0 sec 003.417 ms)
BL2               Wait for Enable APM Time (  217 clk/  0 sec 106.445 ms) dur (    5 clk/  0 sec 002.441 ms)
BL2                  init DRAM CTRL Time (  222 clk/  0 sec 108.886 ms) dur (   41 clk/  0 sec 020.019 ms)
BL2                 after DRAM CTRL Time (  263 clk/  0 sec 128.906 ms) dur (    0 clk/  0 sec 000.000 ms)
BL2                cp acpm-dbg2dram Time (  263 clk/  0 sec 128.906 ms) dur (    0 clk/  0 sec 000.000 ms)
BL2                         NA Time (  263 clk/  0 sec 128.906 ms) dur (   15 clk/  0 sec 007.324 ms)
BL2                    cp ect2dram Time (  278 clk/  0 sec 136.230 ms) dur (    3 clk/  0 sec 001.464 ms)
BL2                     UFS GearUp Time (  281 clk/  0 sec 137.695 ms) dur (    7 clk/  0 sec 003.417 ms)
BL2                load bootloader Time (  288 clk/  0 sec 141.113 ms) dur (   26 clk/  0 sec 012.695 ms)
BL2                 EL3 mon Start Time (  314 clk/  0 sec 153.808 ms) dur (   27 clk/  0 sec 013.183 ms)
LK                  Initial BL_MP Time (  341 clk/  0 sec 166.992 ms) dur (    1 clk/  0 sec 000.488 ms)
LK                Start enabling MMU Time (  342 clk/  0 sec 167.480 ms) dur (    6 clk/  0 sec 002.929 ms)
LK                 End enabling MMU Time (  348 clk/  0 sec 170.410 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                Start dcache flush Time (  348 clk/  0 sec 170.410 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                 End dcache flush Time (  348 clk/  0 sec 170.410 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                Start Enable Cores Time (  348 clk/  0 sec 170.410 ms) dur (    4 clk/  0 sec 001.953 ms)
LK               Wait4LoadingBinaries Time (  352 clk/  0 sec 172.363 ms) dur (  313 clk/  0 sec 152.832 ms)
LK                Jump to hypervisor Time (  665 clk/  0 sec 325.195 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                Start disable MMU Time (  665 clk/  0 sec 325.195 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                 End disable MMU Time (  665 clk/  0 sec 325.195 ms) dur ( 1827 clk/  0 sec 892.089 ms)
VM2               DPU_Driver Init Done Time ( 2492 clk/  1 sec 217.285 ms) dur (    2 clk/  0 sec 000.976 ms)
VM2               DPU_Driver Init Done Time ( 2494 clk/  1 sec 218.261 ms) duration not applicable
-- Core [1] ----------------------------------------------------------------------------------------------------
LK                Start enabling MMU Time (  352 clk/  0 sec 172.363 ms) dur (    1 clk/  0 sec 000.488 ms)
LK                 End enabling MMU Time (  353 clk/  0 sec 172.851 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                    Enter Core Time (  353 clk/  0 sec 172.851 ms) dur (    0 clk/  0 sec 000.000 ms)
LK               Wait4DecidingBootMode Time (  353 clk/  0 sec 172.851 ms) dur (  167 clk/  0 sec 081.542 ms)
LK               Wait 4 Core0 Jump2 HYP Time (  520 clk/  0 sec 254.394 ms) dur (  147 clk/  0 sec 071.777 ms)
LK                    Core Jump2Hyp Time (  667 clk/  0 sec 326.171 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                Start disable MMU Time (  667 clk/  0 sec 326.171 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                 End disable MMU Time (  667 clk/  0 sec 326.171 ms) dur ( 1864 clk/  0 sec 910.156 ms)
VM2              Ethernet Driver Init Done Time ( 2531 clk/  1 sec 236.328 ms) dur (  651 clk/  0 sec 317.871 ms)
VM2                   Kernel Init Done Time ( 3182 clk/  1 sec 554.199 ms) duration not applicable
-- Core [2] ----------------------------------------------------------------------------------------------------
LK                Start enabling MMU Time (  352 clk/  0 sec 172.363 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                 End enabling MMU Time (  352 clk/  0 sec 172.363 ms) dur (    1 clk/  0 sec 000.488 ms)
LK                    Enter Core Time (  353 clk/  0 sec 172.851 ms) dur (    0 clk/  0 sec 000.000 ms)
LK               Wait4DecidingBootMode Time (  353 clk/  0 sec 172.851 ms) dur (  167 clk/  0 sec 081.542 ms)
LK               Wait 4 Core0 Jump2 HYP Time (  520 clk/  0 sec 254.394 ms) dur (  147 clk/  0 sec 071.777 ms)
LK                    Core Jump2Hyp Time (  667 clk/  0 sec 326.171 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                Start disable MMU Time (  667 clk/  0 sec 326.171 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                 End disable MMU Time (  667 clk/  0 sec 326.171 ms) dur ( 1611 clk/  0 sec 786.621 ms)
VM3                   Kernel Init Done Time ( 2278 clk/  1 sec 112.792 ms) duration not applicable
-- Core [3] ----------------------------------------------------------------------------------------------------
LK                Start enabling MMU Time (  351 clk/  0 sec 171.875 ms) dur (    1 clk/  0 sec 000.488 ms)
LK                 End enabling MMU Time (  352 clk/  0 sec 172.363 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                    Enter Core Time (  352 clk/  0 sec 172.363 ms) dur (    0 clk/  0 sec 000.000 ms)
LK               Wait4DecidingBootMode Time (  352 clk/  0 sec 172.363 ms) dur (  168 clk/  0 sec 082.031 ms)
LK               Wait 4 Core0 Jump2 HYP Time (  520 clk/  0 sec 254.394 ms) dur (  147 clk/  0 sec 071.777 ms)
LK                    Core Jump2Hyp Time (  667 clk/  0 sec 326.171 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                Start disable MMU Time (  667 clk/  0 sec 326.171 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                 End disable MMU Time (  667 clk/  0 sec 326.171 ms) duration not applicable
-- Core [4] ----------------------------------------------------------------------------------------------------
LK                Start enabling MMU Time (  351 clk/  0 sec 171.875 ms) dur (    1 clk/  0 sec 000.488 ms)
LK                 End enabling MMU Time (  352 clk/  0 sec 172.363 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                    Enter Core Time (  352 clk/  0 sec 172.363 ms) dur (   31 clk/  0 sec 015.136 ms)
LK               start init secure OS Time (  383 clk/  0 sec 187.500 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                init secure OS done Time (  383 clk/  0 sec 187.500 ms) dur (    0 clk/  0 sec 000.000 ms)
LK               Wait4DecidingBootMode Time (  383 clk/  0 sec 187.500 ms) dur (  137 clk/  0 sec 066.894 ms)
LK               Wait 4 Core0 Jump2 HYP Time (  520 clk/  0 sec 254.394 ms) dur (  147 clk/  0 sec 071.777 ms)
LK                    Core Jump2Hyp Time (  667 clk/  0 sec 326.171 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                Start disable MMU Time (  667 clk/  0 sec 326.171 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                 End disable MMU Time (  667 clk/  0 sec 326.171 ms) duration not applicable
-- Core [5] ----------------------------------------------------------------------------------------------------
LK                Start enabling MMU Time (  351 clk/  0 sec 171.875 ms) dur (    1 clk/  0 sec 000.488 ms)
LK                 End enabling MMU Time (  352 clk/  0 sec 172.363 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                    Enter Core Time (  352 clk/  0 sec 172.363 ms) dur (    0 clk/  0 sec 000.000 ms)
LK               Wait4DecidingBootMode Time (  352 clk/  0 sec 172.363 ms) dur (  168 clk/  0 sec 082.031 ms)
LK               Wait 4 Core0 Jump2 HYP Time (  520 clk/  0 sec 254.394 ms) dur (  147 clk/  0 sec 071.777 ms)
LK                    Core Jump2Hyp Time (  667 clk/  0 sec 326.171 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                Start disable MMU Time (  667 clk/  0 sec 326.171 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                 End disable MMU Time (  667 clk/  0 sec 326.171 ms) duration not applicable
-- Core [6] ----------------------------------------------------------------------------------------------------
LK                Start enabling MMU Time (  351 clk/  0 sec 171.875 ms) dur (    1 clk/  0 sec 000.488 ms)
LK                 End enabling MMU Time (  352 clk/  0 sec 172.363 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  Initial BL_AUTH Time (  352 clk/  0 sec 172.363 ms) dur (    0 clk/  0 sec 000.000 ms)
```

```
LK                  init-early-platform Time (  352 clk/  0 sec 172.363 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  init-early-target Time (  352 clk/  0 sec 172.363 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  init-lk-heap Time (  352 clk/  0 sec 172.363 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  loading secure OS Time (  352 clk/  0 sec 172.363 ms) dur (   31 clk/  0 sec 015.136 ms)
LK                loading secure OS done Time (  383 clk/  0 sec 187.500 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  AUTH load ldfw Time (  383 clk/  0 sec 187.500 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  AUTH load ldfw Time (  383 clk/  0 sec 187.500 ms) dur (   66 clk/  0 sec 032.226 ms)
LK                  End loading ldfw Time (  449 clk/  0 sec 219.726 ms) dur (  540 clk/  0 sec 263.671 ms)
LK                Wait4DecidingBootMode Time (  989 clk/  0 sec 483.398 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  Wait 4 Loading HYP Time (  989 clk/  0 sec 483.398 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                Wait 4 Loading binaries Time (  989 clk/  0 sec 483.398 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  Auth end Jump2Hyp Time (  989 clk/  0 sec 483.398 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  Start disable MMU Time (  989 clk/  0 sec 483.398 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  End disable MMU Time (  989 clk/  0 sec 483.398 ms) duration not applicable
-- Core [7] ----------------------------------------------------------------------------------------------------
LK                  Start enabling MMU Time (  351 clk/  0 sec 171.875 ms) dur (    1 clk/  0 sec 000.488 ms)
LK                  End enabling MMU Time (  352 clk/  0 sec 172.363 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  Initial LK Time (  352 clk/  0 sec 172.363 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                request loading secure OS Time (  352 clk/  0 sec 172.363 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                req loading secure OS done Time (  352 clk/  0 sec 172.363 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  request loading ldfw Time (  352 clk/  0 sec 172.363 ms) dur (   31 clk/  0 sec 015.136 ms)
LK                request loading ldfw done Time (  383 clk/  0 sec 187.500 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  LK Early Func Time (  383 clk/  0 sec 187.500 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  init-early-thread Time (  383 clk/  0 sec 187.500 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  init-early-arch Time (  383 clk/  0 sec 187.500 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  init-early-platform Time (  383 clk/  0 sec 187.500 ms) dur (   35 clk/  0 sec 017.089 ms)
LK                  init-early-target Time (  418 clk/  0 sec 204.589 ms) dur (    4 clk/  0 sec 001.953 ms)
LK                  init-lk-heap Time (  422 clk/  0 sec 206.542 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  init-little-kernel Time (  422 clk/  0 sec 206.542 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  start bootstrap-2 Time (  422 clk/  0 sec 206.542 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  LK Platform Func Time (  422 clk/  0 sec 206.542 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  init-arch Time (  422 clk/  0 sec 206.542 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  start-init-platform Time (  422 clk/  0 sec 206.542 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  init-platform-pmic Time (  422 clk/  0 sec 206.542 ms) dur (   11 clk/  0 sec 005.371 ms)
LK                  init-platform-tmu Time (  433 clk/  0 sec 211.914 ms) dur (   13 clk/  0 sec 006.347 ms)
LK                init-platform-config-ufs Time (  446 clk/  0 sec 218.261 ms) dur (    7 clk/  0 sec 003.417 ms)
LK                  init-platform-pit Time (  453 clk/  0 sec 221.679 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  init-platform-pit Time (  453 clk/  0 sec 221.679 ms) dur (    1 clk/  0 sec 000.488 ms)
LK                init-platform-show-acpm-ver Time (  454 clk/  0 sec 222.167 ms) dur (   25 clk/  0 sec 012.207 ms)
LK                  Wait4Loadingldfw Time (  479 clk/  0 sec 234.375 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  init-platform-ufs Time (  479 clk/  0 sec 234.375 ms) dur (    5 clk/  0 sec 002.441 ms)
LK                  init-target Time (  484 clk/  0 sec 236.816 ms) dur (   29 clk/  0 sec 014.160 ms)
LK                  start app Time (  513 clk/  0 sec 250.976 ms) dur (    7 clk/  0 sec 003.417 ms)
LK                  sel boot mode Time (  520 clk/  0 sec 254.394 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  write dramtrain Time (  520 clk/  0 sec 254.394 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  NA Time (  520 clk/  0 sec 254.394 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  load hyp dtb Time (  520 clk/  0 sec 254.394 ms) dur (   13 clk/  0 sec 006.347 ms)
LK                  configure dtb Time (  533 clk/  0 sec 260.742 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  load sys dtb Time (  533 clk/  0 sec 260.742 ms) dur (   28 clk/  0 sec 013.671 ms)
LK                  adjust loglevel Time (  561 clk/  0 sec 274.414 ms) dur (    1 clk/  0 sec 000.488 ms)
LK                loads header of boot.img Time (  562 clk/  0 sec 274.902 ms) dur (    4 clk/  0 sec 001.953 ms)
LK                  load dtb Time (  566 clk/  0 sec 276.855 ms) dur (    1 clk/  0 sec 000.488 ms)
LK                  overlay dtbo Time (  567 clk/  0 sec 277.343 ms) dur (   43 clk/  0 sec 020.996 ms)
LK                  load audio fw Time (  610 clk/  0 sec 298.339 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  hyp reset-audio Time (  610 clk/  0 sec 298.339 ms) dur (    7 clk/  0 sec 003.417 ms)
LK                  adjust loglevel Time (  617 clk/  0 sec 301.757 ms) dur (    8 clk/  0 sec 003.906 ms)
LK                  load hypervisor Time (  625 clk/  0 sec 305.664 ms) dur (   40 clk/  0 sec 019.531 ms)
LK                  Start dcache flush Time (  665 clk/  0 sec 325.195 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  End dcache flush Time (  665 clk/  0 sec 325.195 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  load sys kernel Time (  665 clk/  0 sec 325.195 ms) dur (  212 clk/  0 sec 103.515 ms)
LK                load android kernel Time (  877 clk/  0 sec 428.710 ms) dur (   86 clk/  0 sec 041.992 ms)
LK                  SFI start Time (  963 clk/  0 sec 470.703 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  Check QSPI Time (  963 clk/  0 sec 470.703 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  CPY 2 temp DRAM Time (  963 clk/  0 sec 470.703 ms) dur (    4 clk/  0 sec 001.953 ms)
LK                  CPY 2 Secure SRAM Time (  967 clk/  0 sec 472.656 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  CPY 2 Secure SRAM Time (  967 clk/  0 sec 472.656 ms) dur (   20 clk/  0 sec 009.765 ms)
LK                  CPY 2 Secure SRAM Time (  987 clk/  0 sec 482.421 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  SFI reset Time (  987 clk/  0 sec 482.421 ms) dur (    2 clk/  0 sec 000.976 ms)
LK                  SFI reset Time (  989 clk/  0 sec 483.398 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  LK End Jump2Hyp Time (  989 clk/  0 sec 483.398 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  Start disable MMU Time (  989 clk/  0 sec 483.398 ms) dur (    0 clk/  0 sec 000.000 ms)
LK                  End disable MMU Time (  989 clk/  0 sec 483.398 ms) dur ( 1874 clk/  0 sec 915.039 ms)
VM3               Ethernet Driver Init Done Time ( 2863 clk/  1 sec 398.437 ms) dur (    4 clk/  0 sec 001.953 ms)
VM3                 DPU_Driver Init Done Time ( 2867 clk/  1 sec 400.390 ms) duration not applicable
----------------------------------------------------------------------------------------------------------------
```

---------- [[[Sfi Boot Time Table (2048)Hz]]] ----------
-- Core [0] ----------
SFI SFI BL Entry Time (989 clk/ 0 sec 483.398 ms) dur (0 clk/ 0 sec 000.000 ms)
SFI SFI BL Image loading Start Time (989 clk/ 0 sec 483.398 ms) dur (8 clk/ 0 sec 003.906 ms)
SFI SFI BL Image loading End Time (997 clk/ 0 sec 487.304 ms) dur (0 clk/ 0 sec 000.000 ms)
SFI SFI BL Image auth Start Time (997 clk/ 0 sec 487.304 ms) dur (16 clk/ 0 sec 007.812 ms)
SFI SFI BL Image auth End Time (1013 clk/ 0 sec 495.117 ms) dur (0 clk/ 0 sec 000.000 ms)
SFI SFI BL Image Jump Time (1013 clk/ 0 sec 495.117 ms) duration not applicable
----

---------- [[[VM2 Boot Time Table (2048)Hz]]] ----------
VM2 DPU_Driver Init Done Time (2492 clk/ 1 sec 217.285 ms) dur (2 clk/ 0 sec 000.976 ms)
VM2 DPU_Driver Init Done Time (2494 clk/ 1 sec 218.261 ms) dur (37 clk/ 0 sec 018.066 ms)
VM2 Ethernet Driver Init Done Time (2531 clk/ 1 sec 236.328 ms) dur (651 clk/ 0 sec 317.871 ms)
VM2 ***Kernel Init Done Time (3182 clk/ 1 sec 554.199 ms)*** *duration not applicable*

---------- [[[VM3 Boot Time Table (2048)Hz]]] ----------
VM3 ***Kernel Init Done Time (2278 clk/ 1 sec 112.792 ms)*** *dur (585 clk/ 0 sec 285.644 ms)*
VM3 Ethernet Driver Init Done Time (2863 clk/ 1 sec 398.437 ms) dur (4 clk/ 0 sec 001.953 ms)
VM3 DPU_Driver Init Done Time (2867 clk/ 1 sec 400.390 ms) duration not applicable
----

Result: -

From above output we can see that we are able to read time stamp data starting from boot loader stage till OS bootup.

We can see that Kernel Init time for **VM2 i.e. SYS is 1 sec 554.199ms.**

Kernel Init time for **VM3 i.e. IVI is 1 sec 112.792 ms.**

[Above test result is with **ES2 Artifact - 132 dated 05-08-2022** with additional changes on SYS and IVI kernel image to enable Boot Time KPI Marker.]

6. Boot Time KPI Marker Driver: -

To enable compilation of "Boot Time KPI Driver", below config macros should be enabled.

```
CONFIG_EXYNOS_BOOTTIME_LOG=y

CONFIG_EXYNOS_KERNEL_BOOTTIME=y
```

DTS changes needed to enable Boot time KPI Marker is as below.

```
&boottime_mem {
        status = "okay";
};

&boottime_log {
        status = "okay";
};
```

Driver source code and role: -

a. **drivers/soc/samsung/exynos-kernel-boottime.c** : This source file contains very important function "exynos_kernel_boottime_log( )". This is the function which is called in various drivers or kernel routines to take timestamp data and write into RAM. This API does time stamp for various CPU cores and VM image. Apart from this API, there is another helper functions to support this API. So, we can say this code is responsible for writing or logging time stamp data.
b. **drivers/soc/samsung/exynos-boottimelog.c:** This is the main driver source code which is registered as platform driver. Once platform bus, reads DTB with appropriate compatible string, then its probe() function is invoked, which gets physical address of timestamp region in DRAM i.e. (0xE9A4 1000) and gives virtual address. Then it creates procfs file (/proc/boottime) with necessary fops to give read access from user space.

Usage: -

To take time stamp for any driver or kernel routine, we have to call API "exynos_kernel_boottime_log( )". For example this function is used in source file "init/main.c" as exynos_kernel_boottime_log(KERNEL_INIT_DONE).

In kernel source file <kernel_root_folder>/init/main.c, there is a function kernel_init_freeable() upto which all root file system, devtmpfs etc is mounted. To get complete kernel initialization time (i.e. before init process) we call exynos_kernel_boottime_log(KERNEL_INIT_DONE) just after function call "rcu_end_inkernel_boot()" in init/main.c.

This API needs one argument, to log with Internal ID (shown in Table - 4 above). The value of these arguments are defined in a header file - "include/linux /soc/samsung/exynosauto9-boottimelog.h".

7. Procedure to add custom marker for debugging: -

If we want to measure time taken to execute some lines of code, then we have to call exynos_kernel_boottime_log() API with proper argument as below.

```
exynos_kernel_boottime_log(XYZ_INIT_START);

/*Lines of code whose time is to measure*/

.....

.....

exynos_kernel_boottime_log(XYZ_INIT_END);
```

We have to define macros for our custom marker in header file - "include/linux/soc/samsung/exynosauto9-boottimelog.h", as below and we should ensure that value of these macros do not conflict with other values.

For example -

```
#define XYZ_INIT_START          0xb325
#define XYZ_INIT_END            0xb326
```

Now, when we read the log, the parser has to decode the corresponding time stamp, so we have to add entries for our marker (defined above) in parser script also.

Modify below parser file - "<Parser_folder>/local/boottime-viewer/lib/define/boottimelable.py" to add markers and corresponding string to be printed on log, explained as below for example.

File - local/boottime-viewer/lib/define/boottimelable.py

```
+ local/boottime-viewer/lib/define/boottimelable.py:277:     ,["KER",    0xb325, "XYZ debug start"]
+ local/boottime-viewer/lib/define/boottimelable.py:278:     ,["KER",    0xb326, "XYZ debug end"]
```

After editing above file in host/development PC, we should copy entire Parser directory to the target, if parser is installed as part of root file system, then we can modify parser directly on target. No compilation/cross-compilation of parser is required, as it is Python script.

The complete folder of parser is attached here.

We can run parser from above package for Boot time KPI marker log, as below.

```
# cat /proc/boottime | ./usr/local/boottime-viewer/btime.py
```

or, if parser is installed, then run command as usual.

8. Gerrits to apply to enable Boot-KPI driver in Android

1. https://androidhub.harman.com/c/Android/kernel/+/317231
2. https://androidhub.harman.com/c/and_ivi_dts-idcevo-la/+/317234

9. Procedure to build Boot KPI Marker driver for Android -

As Android Boot-KPI Marker is debug feature, so to build it separate config fragment file - "exynosautov920_idcevo_la_and_gki_debug.fragment" is added.

For Android Boot-KPI marker to work, below changes are done in build configuration file in Android Code base.

File name - <Android_Code_Base>/Android-kernel/exynos/build.config.gki.pf_vBmw_b1

Change to do in above file:

```
PRE_DEFCONFIG_CMDS="
.....
+ ${KERNEL_DIR}/arch/arm64/configs/exynosautov920_idcevo_la_and_gki_debug.fragment \
"
```

Note: Only one line is added, which is starting with "+" symbol shown above.

10. Debug RTC platform device used in Boot KPI driver -

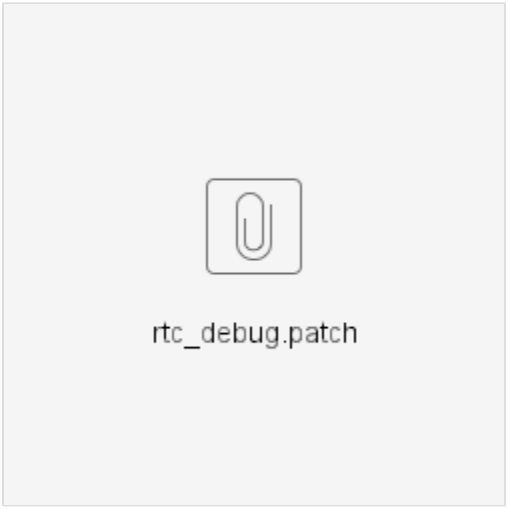
While working with Boot-KPI driver, it is observed that RTC hardware device used for Boot-KPI, is not configured properly or disabled. It can be debugged using a debug patch as below.

V920 SOC ID:	0x0A920000
RTC Module base address in v920 SOC:	0x118C0000
RTC_CURTICCNT_0 (gives tick count):	Offset: 0x90
RTC_TICCON_0(configures RTC):	Offset: 0x38

Complete details of RTC_TICCON_0 register can be found in SOC TRM, Section: - 23.10.1.3, Page - 4732, SOC TRM Revision - 0.6, Dated : September 2022.

If content of RTC_TICCON_0 is 0x09 i.e. 0b01001, then it indicates RTC is configured properly with RTC increment clock - 2048Hz and RTC module is enabled.

Debug patch output: -



Important Note: -

1) To take Boot Time KPI Marker log, it is mandatory to do complete RESET of the system, by turning off External Power Supply. If we do Reset using push button of Debug/Edge connector, then it does not turn off internal RTC of SOC used for time stamping and hence it will show Boot Time due to Edge Connector Reset plus the time the hardware was started in last Power On Session.

2) In Boot Time KPI log, the timing data which precedes with string "Time" in format " (N clk/ A sec B ms) " is absolute time and the one which precedes with string "dur" is delta time that is difference between next time stamp and current time stamp.