Ethtool - Diagnostic Approach for Network Issues in Linux

Sriranjani P, Ravi Patel

Samsung Semiconductor India R&D Center
Agenda

- Introduction to Ethernet
- Need of debugging and tuning of network
- About MAC Management Counters
- Why Ethtool
- Our Work
- Real Use - Case
- Results – Use case 1
- Results – Use case 2
- Conclusion and Future scope
Introduction to Ethernet

- Standard Ethernet

  - Standard Ethernet: 10 Mbps
  - Fast Ethernet: 100 Mbps
  - Gigabit Ethernet: 1 Gbps
  - Ten – Gigabit Ethernet: 10 Gbps
Automotive Ethernet

(Image source: support.ixiacom.com)
Need of debugging and tuning of Network

- To obtain information about network interface
- To validate the target host is accessible from our machine
- To get information regarding interrupts occurred during data transmission
- To find whether packet got dropped and what error occurred
- To ensure speed and mode of operation is proper
- Different tools available for various purpose like ping, ifconfig, arp, route, ethtool etc.
About MAC Management Counters

- Extension of the register address space of the CSR module
- Contains set of registers for gathering statistics on the received and transmitted packets
- Includes a control register for controlling the behavior of the registers
Why Ethtool

- Open source
- Versatile

Ethtool

- Port
- Duplex
- Stats
- Offload
- Speed
How to install Ethtool

- It can be used in most Linux distributions
- First update APT package
  - `sudo apt update`
- Now install ethtool using,
  - `sudo apt install ethtool --y`
- To install ethtool in Fedora/CentOS systems use,
  - `sudo dnf install python3-ethtool`
Ethtool Architecture Overview

User Space
- Ethtool call from command line
- Ethtool utility

IOCTL Socket

Kernel Space
- IOCTL handler
- Network driver
- Ethtool Ops

Hardware
Kernel Infrastructure details of Ethtool

Ethtool Utility (user-space)

- GET call with to retrieve info: ETHTOOL_MSG_LINKINFO_GET
- Response with data: ETHTOOL_MSG_LINKINFO_GET_REPLY
- SET call with attributes to set: ETHTOOL_MSG_LINKINFO_SET
- Error code or Positive Ack: ETHTOOL_MSG_LINKINFO_NTF
- ACT call with specific control: ETHTOOL_MSG_CABLE_TEST_ACT
- Error code or Positive Ack: ETHTOOL_MSG_CABLE_TEST_NTF

Ethtool (kernel-space)
Ethtool User space Implementation

```
{    .opts = "--show-mmc",
    .func = do_gmmcstats,
    .help = "Show DWC E2OoS MMC register details",
},

static int do_gmmcstats(struct cmd_context *ctx)
{
    return do_gmmc(ctx, ETHTOOL_MMC_STATS, ETH_SS_MMC_STATS, "MMC");
}

static int do_gmmc(struct cmd_context *ctx, int cmd, int stringset,
        const char *name)
{
    struct ethtool_gstrings *strings;
    struct ethtool_guosmmc_stats *stats;
    unsigned int i;
    int err;

    if (ctx->argc != 0)
        exit_bad_args();

    strings = get_stringset(ctx, stringset,
            offsetof(struct ethtool_drvinfo, n_stats),
            0);
```
Ethtool Linux Kernel Implementation

```c
switch (stringset) {
    case ETH_SS_MMC_STATS:
        stmmac_get_mmc_stats(priv, p);
        break;
}
```
Ethtool Linux kernel Implementation

```c
struct dwc_mmc_reg_definitions {
    int index;
    const char *name;
};

static struct dwc_mmc_reg_definitions s_dwc_mmc_cntrl_reg_definitions[BIT_COUNT] = {
    [0], "CNTRST Counter Reset ",
    [1], "CNTSTOPRO Counter Stop Rollover ",
    [2], "RSTONRD Reset on Read ",
    [3], "CNTFREEZ MMC Counter Freeze ",
    [4], "CNTFRST Counters Preset ",
    [5], "CNTFRSTLVL Full Half Preset ",
    [6], "UCDBC Update MMC Counters for dropped Broadcast packets ",
};

static int bit[32];
static int dec_to_bin(int val)
{
    int index;
    for (index = 0; val > 0; index++) {
        bit[index] = val % 2;
        val = val / 2;
    }
    return 0;
}
```

---

Foundry Design Services (FDS) SW 14
static void stm32c_get mmc_stats(struct stm32c_priv *priv, u8 *data) 
{
    u32 val;
    int i,j=0;
    int offset = 0;
    char sample_buf[10000];

    val = readl(priv->ioaddr + DMCE00S MMC_CTRL);
    offset += sprintf(sample_buf + offset, "\tMMC Control val: 0x%x\n", val);
    dec_to_bin(val);
    if (bit[s DWC mmc_cntrl_reg_definitions[0].index] == 1)
        offset += sprintf(sample_buf + offset, "%s: Enabled\n", s DWC mmc_cntrl_reg_definitions[0].name);
    if (bit[s DWC mmc_cntrl_reg_definitions[0].index] == 0)
        offset += sprintf(sample_buf + offset, "%s: disabled\n", s DWC mmc_cntrl_reg_definitions[0].name);
    if (bit[s DWC mmc_cntrl_reg_definitions[1].index] == 1)
        offset += sprintf(sample_buf + offset, "%s: Enabled\n", s DWC mmc_cntrl_reg_definitions[1].name);
    if (bit[s DWC mmc_cntrl_reg_definitions[1].index] == 0)
        offset += sprintf(sample_buf + offset, "%s: disabled\n", s DWC mmc_cntrl_reg_definitions[1].name);
    if (bit[s DWC mmc_cntrl_reg_definitions[2].index] == 1)
        offset += sprintf(sample_buf + offset, "%s: Enabled\n", s DWC mmc_cntrl_reg_definitions[2].name);
    if (bit[s DWC mmc_cntrl_reg_definitions[2].index] == 0)
        offset += sprintf(sample_buf + offset, "%s: disabled\n", s DWC mmc_cntrl_reg_definitions[2].name);
    if (bit[s DWC mmc_cntrl_reg_definitions[3].index] == 1)
        offset += sprintf(sample_buf + offset, "%s: Enabled\n", s DWC mmc_cntrl_reg_definitions[3].name);
    if (bit[s DWC mmc_cntrl_reg_definitions[3].index] == 0)
        offset += sprintf(sample_buf + offset, "%s: disabled\n", s DWC mmc_cntrl_reg_definitions[3].name);
    if (bit[s DWC mmc_cntrl_reg_definitions[4].index] == 1)
        offset += sprintf(sample_buf + offset, "%s: Enabled\n", s DWC mmc_cntrl_reg_definitions[4].name);
    if (bit[s DWC mmc_cntrl_reg_definitions[4].index] == 0)
        offset += sprintf(sample_buf + offset, "%s: disabled\n", s DWC mmc_cntrl_reg_definitions[4].name);
Real Use - Case

- On TX side ifconfig shows these errors,

  TX packets:2009 errors:1360 dropped:0 overruns:792 carrier:574

- Ping flood test with difference packet size shows no packet drops but the error stats increases for each packet when size is more than 100 bytes

- Disable Transmit Status in MTL was set – this caused issue in updating status field of TX descriptor

- Unable to find what error occurred, does it mean S/W won’t be able to find what all transmission error occurred??
Use Case 1:

- mmc_cntrl register value is not show in existing ethtool statistics command which have bit-wise information useful for debugging purpose
- We added additional support for mmc_cntrl register so that user gets more details information about this critical register information in user space
- Output from modified “ethtool” command is shown below:

```
# ./ethtool --show-mmc eth0

MMC Control val: 0x24
   CNTRST Counter Reset : Disable
   CNTSTOPRO Counter Stop Rollover : Disable
   RSTONRD Reset on Read : Enable
   CNTFREEZ MMC Counter Freeze : Disable
   CNTPRST Counters Preset : Disable
   CNTPRSTLVL Full Half Preset : Enable
   UCDBC Update MMC Counters for dropped Broadcast packets : Disable
```
Use Case 2:

- `mmc_rx_ipc_intr_mask` value is shown as a big number in existing `ethtool` command like 1073692671
- It’s difficult for user to decode to get bitwise information

```
$./ethtool --show-mmc eth0
```

```
MMC   IPC Rx Interrupt Mask val: 0x3fff3fff
RXIPv4GPIM MMC Receive IPv4 Good packet counter Interrupt Mask : Enabled
RXIPv4HERFPIIM MMC Receive IPv4 Header Error packet counter Interrupt Mask : Enabled
RXIPv4NPOPAYFIM MMC Receive IPv4 No payload packet counter Interrupt Mask : Enabled
RXIPv4FRAGPIM MMC Receive IPv4 Fragmented packet counter Interrupt Mask : Enabled
RXIPv4UDSBLFIM MMC Receive IPv4 UDP Checksum Disabled packet counter Interrupt Mask : Enabled
RXIPv6GPIM MMC Receive IPv6 Good packet counter Interrupt Mask : Enabled
RXIPv6HERFPIIM MMC Receive IPv6 Header Error packet counter Interrupt Mask : Enabled
RXIPv6NPOPAYFIM MMC Receive IPv6 No Payload packet counter Interrupt Mask : Enabled
RXUDPPIIM MMC Receive UDP Good packet counter Interrupt Mask : Enabled
RXUDPERRPIIM MMC Receive UDP Error packet counter Interrupt Mask : Enabled
RXTCPPIIM MMC Receive TCP Good packet counter Interrupt Mask : Enabled
RXTCPPERRPIIM MMC Receive TCP Error packet counter Interrupt Mask : Enabled
RXICMPGPIIM MMC Receive ICMP Good packet counter Interrupt Mask : Enabled
RXICMPERRPIIM MMC Receive ICMP Error packet counter Interrupt Mask : Enabled
RXIPv4G0I2MM MMC Receive IPv4 Good octet counter Interrupt Mask : Enabled
RXIPv4HERG0I2MM MMC Receive IPv4 Header Error octet counter Interrupt Mask : Enabled
RXIPv6HERG0I2MM MMC Receive IPv6 Header Error octet counter Interrupt Mask : Enabled
RXIPv6NPOPAYG0I2MM MMC Receive IPv6 No Payload octet counter Interrupt Mask : Enabled
RXIPv4FRAGG0I2MM MMC Receive IPv4 Fragmented octet counter Interrupt Mask : Enabled
RXIPv4UDSBLG0I2MM MMC Receive IPv4 UDP Checksum Disabled octet counter Interrupt Mask : Enabled
RXIPv6G0I2MM MMC Receive IPv6 Good octet counter Interrupt Mask : Enabled
RXIPv6HERG0I2MM MMC Receive IPv6 Header Error octet counter Interrupt Mask : Enabled
RXIPv6NPOPAYG0I2MM MMC Receive IPv6 No Payload octet counter Interrupt Mask : Enabled
RXUDP0I2MM MMC Receive UDP Good octet counter Interrupt Mask : Enabled
RXUDPERR0I2MM MMC Receive UDP Error octet counter Interrupt Mask : Enabled
RXTCP0I2MM MMC Receive TCP Good octet counter Interrupt Mask : Enabled
RXTCPPERR0I2MM MMC Receive TCP Error octet counter Interrupt Mask : Enabled
RXICMPPG0I2MM MMC Receive ICMP Good octet counter Interrupt Mask : Enabled
RXICMPERR0I2MM MMC Receive ICMP Error octet counter Interrupt Mask : Enabled
```
We are able to enhance “ethtool” user space utility to capture some of the critical debug registers of DWC EQoS controller and present the data to user-space in human readable format instead of just dumping register values.

We modified the DesignWare EQoS driver (drivers/net/ethernet/stmicro/stmmac/stmmac_ethtool.c) in kernel by adding required ethtool_ops hooks to get all MMC register value, bit wise information of mmc_cntrl, interrupt status and interrupt mask register defined in DWC EQoS IP.

This implementation helping us to get these information via ethtool command line.

Future scope involves implementing this functionality in ethtool for other MMC registers and upstreaming the source code to the Linux mainline community.
Any Questions?
THANK YOU