Packet Forwarding Engine (PFE) Ethernet Ports Setting up on LS1012A Platform

LS1012A integrates a hardware packet forwarding engine to provide high performance Ethernet interfaces. This document introduces PFE hardware and software decomposition and data flow, setting up two PFE Ethernet ports to implement Ethernet packets forwarding through PFE, how to modify PFE driver and dts file to set up single PFE Ethernet port on LS1012A custom boards.

PFE hardware Structure

LS1012A PFE includes two Ethernet ports. Both Ethernet ports support SGMII. One port also has the option to support RGMII (10/100M half-duplex mode or 1G full-duplex mode). These configurations conform to IEEE 802.3 and support full duplex and half-duplex operation at 10/100/1000/2500 Mbps. The following figure illustrates the PFE block diagram.

On LS1012A, there are two MDIO controllers coming out from PFE and one MDIO per MAC is provided. The MDIO ports are connected in the chip as the followings.

MDIO1: It has connectivity for external off-chip PHY registers as well as on-chip SGMII1 PSC registers. The selection occurs based on the SCFG_MDIOSELCR[MDIOSEL] bit as follows:
0: MDIO from SerDes
1: MDIO from external Ethernet PHY

MDIO2: It has connectivity for on-chip SGMII2 PCS registers.

PFE Software Decomposition and Data Flow

This section describes Linux driver which enables support for Ethernet on Packet Forwarding Engine (PFE) hardware. EMACs are part of PFE IP, to receive/transmit packets through EMAC interface it should be accessed through PFE interface by programming it.

PFE block in a system level block view, from a network device perspective may be depicted as the follows:
The PFE, MAC and PHY are the hardware blocks, the kernel networking stack along with the network driver are running in the Kernel space, and finally ethtool and iproute are examples of user space tools used for configuring the network devices.

The PFE hardware supports one HIF RX and TX descriptor queues to send and receive packets through PFE. Both network interface traffic is multiplexed and send over HIF queue.

User space packages like ethtool and iproute are used to configure the network device parameters. The ethtool interface is extended to provide support for filer programming. The kernel space module for the network driver is the most important block as it communicates with both the user space and the H/W IP to control the processing of packets.

The basic functionality of any Ethernet driver is to handle the reception of packets from an ingress port (might include checksum calculation, header verification, etc), as well as the transmission of packets on the egress port (might include checksum re-calculation, header manipulation, etc). There are also the device configuration and control functionalities, and device status reporting. When the Ethernet driver is actually implementing these functionalities, it needs to interact with the core (Kernel) as well as the hardware IP (the Ethernet controller).

The PFE Linux kernel module has following two main parts:

**HIF driver layer:** This part of the driver talks with HIF hardware interface and send and receive the packets from it. It receives packets from HIF interface and identifies from which MAC interface it received
and send the packet to corresponding client driver queue. Similarly, if there is any pending packet from client queue to transmit packet it takes and inserts the HIF header and put it into the HIF queue. It uses the NAPI to receive packets and send it to corresponding client queues and triggers client to process packets from the queue.

**HIF/Ethernet client driver:** Ethernet client driver is a hardware independent driver and registers with the HIF driver to transmit and receive packet through HIF interface. For each interface one instance of client driver should be register with the HIF driver layer, other side it registers with Linux kernel stack as network interface. Each client driver will have software queues to communicate with HIF driver layer. Each client driver registers with NAPI and indicate packets to the stack through the NAPI poll.

**Setting up Two PFE Ethernet Ports to Implement Ethernet Packets Forwarding**

Under u-boot, use the U-BOOT command mdio list to display all manageable Ethernet PHYs.

```plaintext
=> mdio list
PFE_MDIO:
  1 - RealTek RTL8211F --- pfe_eth1
  2 - RealTek RTL8211F --- pfe_eth0
```

Define PFE Ethernet ports in the dts file as the following.

```plaintext
&pfe {
  status = "okay";
  ethernet@0 {
    compatible = "fsl,pfe-gemac-port";
    address-cells = <1>;
    size-cells = <0>;
    reg = < 0x0 >; /* GEM_ID */
    fsl,gemac-bus-id = <0x0>; /* BUS_ID */
    fsl,gemac-phy-id = <0x2>; /* PHY_ID */
    fsl,mdio-mux-val = <0x0>;
    local-mac-address = [ 00 1A 2B 3C 4D 5E ];
    phy-mode = "sgmii";
    fsl,pfe-gemac-if-name = "eth0";
    fsl,pfe-phy-if-flags = <0x0>;
    fsl,pfe-gemac-mode = <0x1B00>; /* GEMAC_SW_CONF | GEMAC_SW_FULL_DUPLEX */
    GEMAC_SW_SPEED_1G */
    mdio@0 {
      reg = <0x1>; /* enabled/disabled */
      fsl,mdio-phy-mask = <0xFFFFFFFF>;
    };
  }
  ethernet@1 {
    compatible = "fsl,pfe-gemac-port";
    address-cells = <1>;
    size-cells = <0>;
    reg = < 0x1 >; /* GEM_ID */
    fsl,gemac-bus-id = < 0x1 >; /* BUS_ID */
    fsl,gemac-phy-id = < 0x1 >; /* PHY_ID */
    fsl,mdio-mux-val = <0x0>;
    local-mac-address = [ 00 AA BB CC DD EE ];
    phy-mode = "rgmii";
    fsl,pfe-gemac-if-name = "eth2";
    fsl,pfe-phy-if-flags = <0x0>;
  }
}
```
PFE driver is set up as the following when booting up Linux Kernel.

```
4.045613] pfe: module is from the staging directory, the quality is unknown, you have been warned.
[  4.118057] pfe_hif_init
[  4.123280] pfe_hif_init
[  4.128392] pfe_hif_init
[  4.135571] ext2-fs (mmcblk0p1): warning: mounting unchecked fs, running e2fsck is recommended
[  4.136971] pfe_ctrl_init
[  4.506790] libphy: Concerto MDIO Bus: probed
[  4.516790] pfe_ctrl_timer
[  5.086790] libphy: Concerto MDIO Bus: probed
[  5.092280] pfe_phy_init interface 3
[  5.879007] eth0: pfe_eth_init_one: created interface, baseaddr: c1200000
[  5.887635] pfe_phy_init interface 7
```
Set up Single PFE Ethernet Port on LS1012A Custom Boards

On LS1012A, MDIO from on MAC is used to control both external PHYs, so there is some dependency between these two PFE Ethernet ports. During LS1012 bringing up, some customer only want to use one RGMII Ethernet interface. This section describes how to modify PFE Linux Kernel driver and dts file to support PFE eth0 absence use case.

Define PFE device node in the dts file as the following.

```dts
&pfe {
```
status = "okay";

eternet@0 {
    compatible = "fsl,pfe-gemac-port";
    #address-cells = <1>;
    #size-cells = <0>;
    reg = <0x0>; /* GEM_ID */
    fsl,gemac-bus-id = <0x0>; /* BUS_ID */
    fsl,gemac-phy-id = <0x2>; /* PHY_ID */
    fsl,mdio-mux-val = <0x0>;
    local-mac-address = [00 1A 2B 3C 4D 5E ];
    phy-mode = "sgmii";
    fsl,pfe-gemac-if-name = "eth0";
    fsl,pfe-phy-if-flags = <0x1>;
    fsl,pfe-gemac-mode = <0x1B00>; /* GEMAC_SW_CONF | GEMAC_SW_FULL_DUPLEX | GEMAC_SW_SPEED_1G */

    mdio@0 {
        reg = <0x1>; /* enabled/disabled */
        fsl,mdio-phy-mask = <0xFFFFFFFF9>;
    }
};

ethernet@1 {
    compatible = "fsl,pfe-gemac-port";
    #address-cells = <1>;
    #size-cells = <0>;
    reg = <0x1>; /* GEM_ID */
    fsl,gemac-bus-id = <0x1>; /* BUS_ID */
    fsl,gemac-phy-id = <0x1>; /* PHY_ID */
    fsl,mdio-mux-val = <0x0>; /* PHY_ID */
    fsl,mdio-mux-val = <0x0>;
    local-mac-address = [00 AA BB CC DD EE ];
    phy-mode = "rgmii";
Please modify the function pfe_eth_init_one in PFE drivers/staging/fsl_ppfe/pfe_eth.c as the following.

```c
static int pfe_eth_init_one( struct pfe *pfe, int id )
{
    ...

    /* Copy the station address into the dev structure, */
    memcpy(dev->dev_addr, einfo[id].mac_addr, ETH_ALEN);

    /* Initialize mdio */
    /* if (minfo[id].enabled) { */
    if ((err = pfe_eth_mdio_init(priv, &minfo[id]))) {
        netdev_err(dev, "%s: pfe_eth_mdio_init() failed\n", __func__);
        goto err2;
    }
    /* } */
    ...
    ...
}
```

Linux Kernel boot up log is as the following:

```
[   3.579431] pfe: module is from the staging directory, the quality is unknown, you have been warned.
[   3.581335] pfe_hw_init
[   3.581335] pfe_hw_init
[   3.581335] pfe_hw_init
[   3.581340] CLASS version: 20
[   3.581345] TMU version: 1011231
[   3.581350] BMU1 version: 21
[   3.581355] BMU2 version: 21
[   3.581359] EGPI1 version: 50
```
Verify the PFE RGMII Ethernet interface on the target board.

root@ls1012ardb-32b:~# ifconfig eth0 10.246.68.11
root@ls1012ardb-32b:~# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:04:9f:04:68:b3
          inet addr:10.246.68.11  Bcast:10.255.255.255  Mask:255.0.0.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
root@ls1012ardb-32b:~# ping 10.246.68.13
PING 10.246.68.13 (10.246.68.13) 56(84) bytes of data.
64 bytes from 10.246.68.13: icmp_seq=1 ttl=64 time=0.313 ms
64 bytes from 10.246.68.13: icmp_seq=2 ttl=64 time=0.262 ms
64 bytes from 10.246.68.13: icmp_seq=3 ttl=64 time=0.254 ms
64 bytes from 10.246.68.13: icmp_seq=4 ttl=64 time=0.252 ms
64 bytes from 10.246.68.13: icmp_seq=5 ttl=64 time=0.251 ms

root@ls1012ardb-32b:~# ifconfig eth0 down
root@ls1012ardb-32b:~# ifconfig eth1 10.246.68.12
root@ls1012ardb-32b:~# ping 10.246.68.13
PING 10.246.68.13 (10.246.68.13) 56(84) bytes of data.
64 bytes from 10.246.68.13: icmp_seq=1 ttl=64 time=0.313 ms
64 bytes from 10.246.68.13: icmp_seq=2 ttl=64 time=0.262 ms
64 bytes from 10.246.68.13: icmp_seq=3 ttl=64 time=0.254 ms
64 bytes from 10.246.68.13: icmp_seq=4 ttl=64 time=0.252 ms
64 bytes from 10.246.68.13: icmp_seq=5 ttl=64 time=0.251 ms