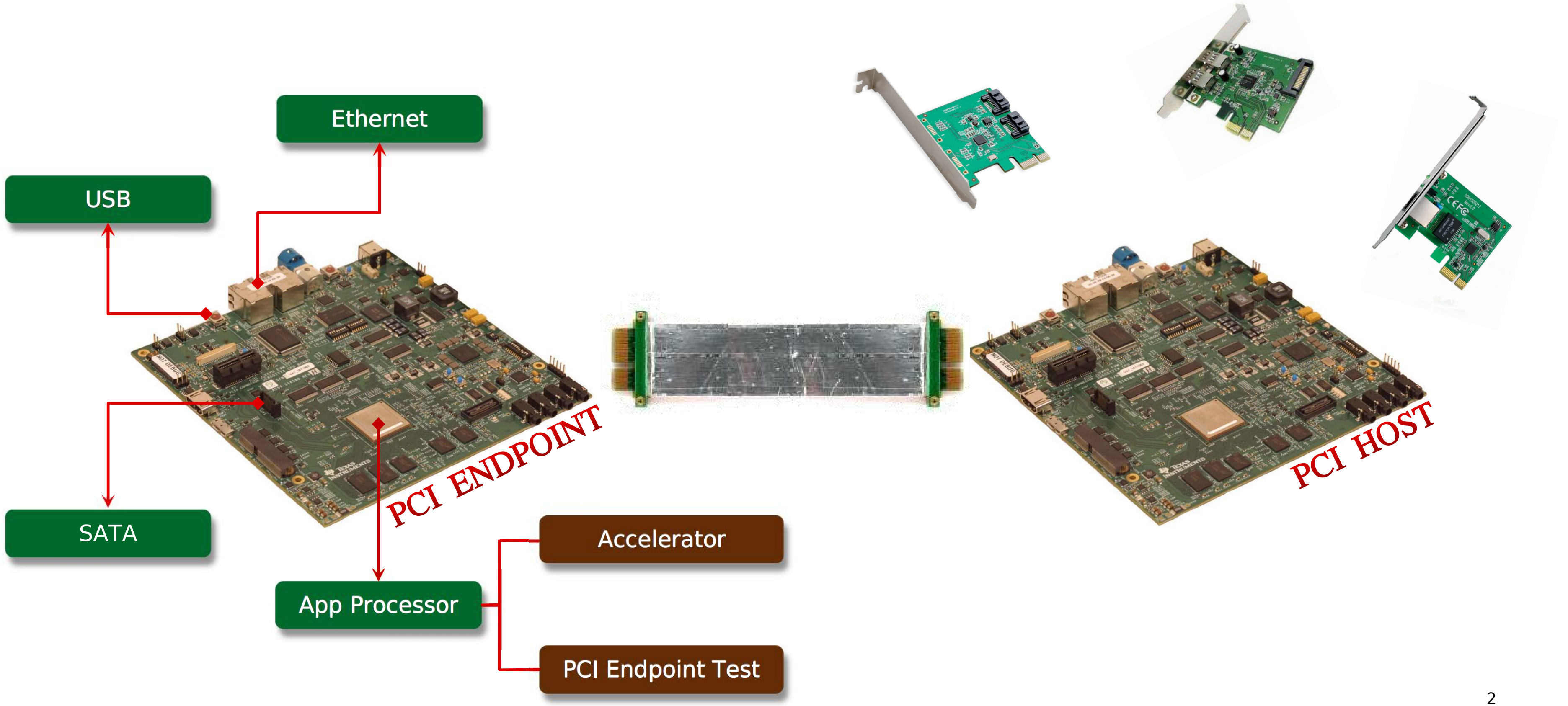
Linux PCI EP Framework

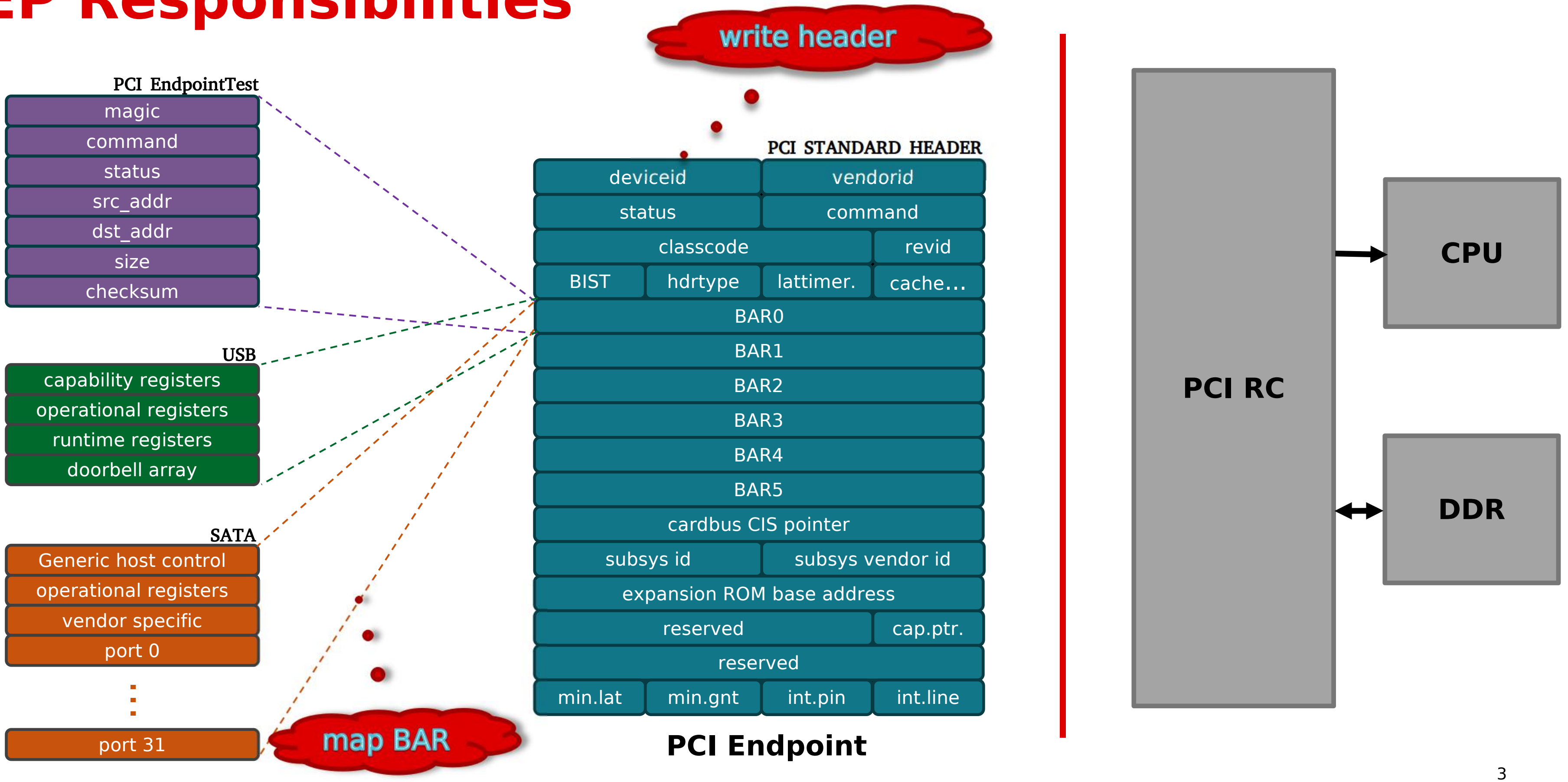
Support for Configurable PCI Endpoint in Linux

KISHON VIJAY ABRAHAM I

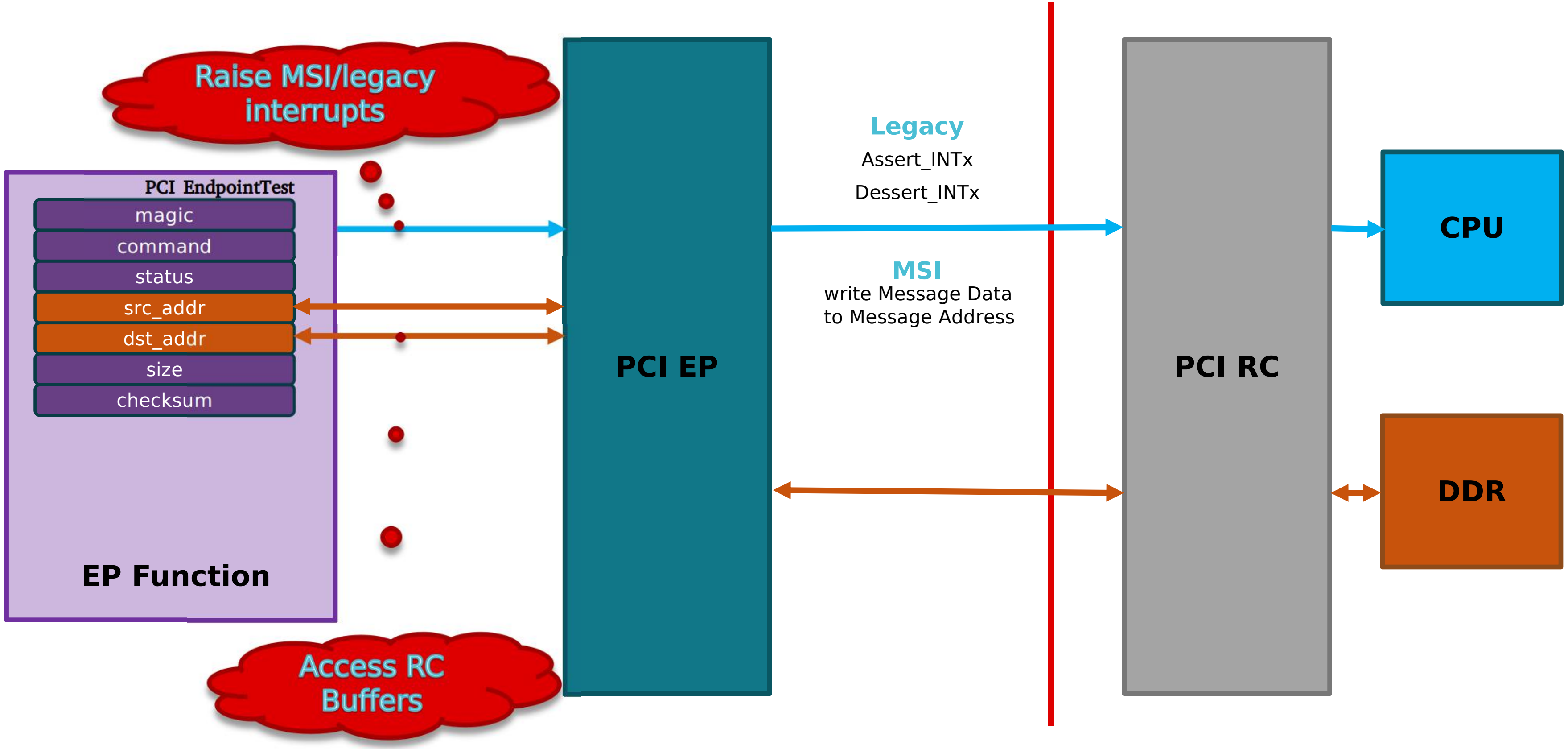
Introduction



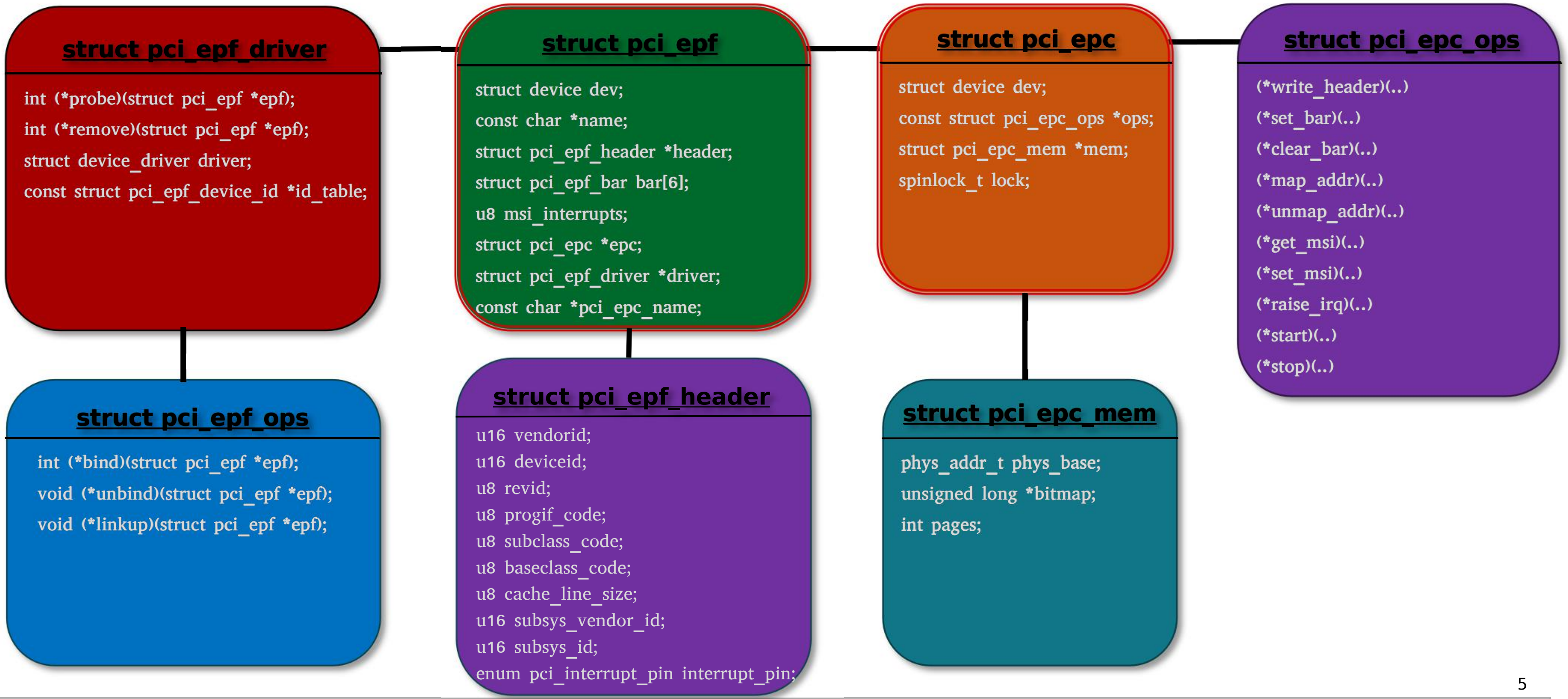
EP Responsibilities



EP Responsibilities



EP Framework



struct pci_epf_driver

```
int (*probe)(struct pci_epf *epf);
int (*remove)(struct pci_epf *epf);
struct device_driver driver;
const struct pci_epf_device_id *id_table;
```

struct pci_epf

```
struct device dev;
const char *name;
struct pci_epf_header *header;
struct pci_epf_bar bar[6];
u8 msi_interrupts;
struct pci_epc *epc;
struct pci_epf_driver *driver;
const char *pci_epc_name;
```

struct pci_epc

```
struct device dev;
const struct pci_epc_ops *ops;
struct pci_epc_mem *mem;
spinlock_t lock;
```

struct pci_epc_ops

```
(*write_header)(..)
(*set_bar)(..)
(*clear_bar)(..)
(*map_addr)(..)
(*unmap_addr)(..)
(*get_msi)(..)
(*set_msi)(..)
(*raise_irq)(..)
(*start)(..)
(*stop)(..)
```

struct pci_epf_ops

```
int (*bind)(struct pci_epf *epf);
void (*unbind)(struct pci_epf *epf);
void (*linkup)(struct pci_epf *epf);
```

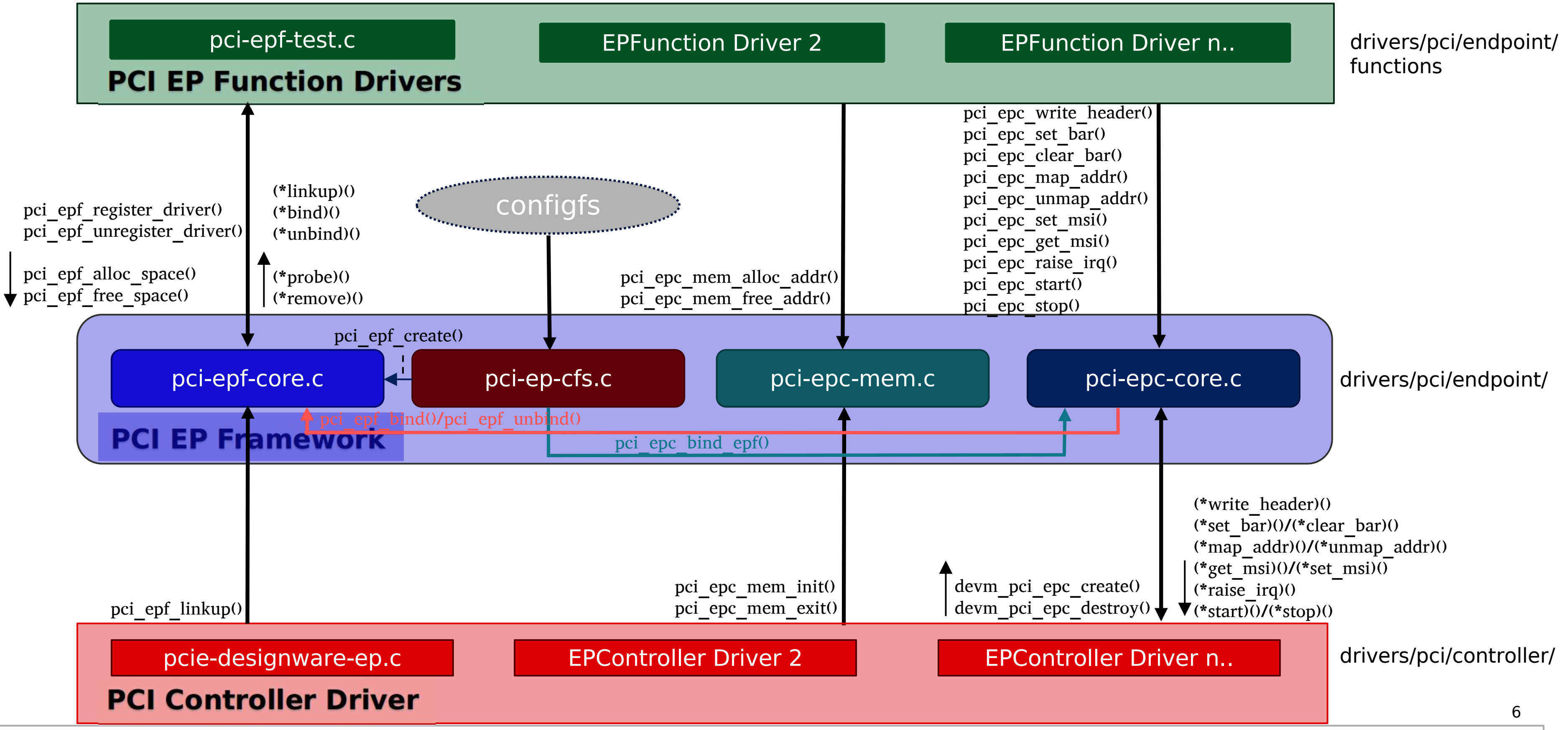
struct pci_epf_header

```
u16 vendorid;
u16 deviceid;
u8 revid;
u8 progif_code;
u8 subclass_code;
u8 baseclass_code;
u8 cache_line_size;
u16 subsys_vendor_id;
u16 subsys_id;
enum pci_interrupt_pin interrupt_pin;
```

struct pci_epc_mem

```
phys_addr_t phys_base;
unsigned long *bitmap;
int pages;
```

EP Framework



devicetree node

```
pcie1_rc: pcie_rc@51000000 {
    compatible = "ti,dra7-pcie";
    reg = <0x51000000 0x1000>, <0x51002000 0x14c>, <0x1000 0x2000>;
    reg-names = "rc_dbics", "ti_conf", "config";
    interrupts = <0 232 0x4>, <0 233 0x4>;
    ti,hwmods = "pcie1";
    phys = <&pcie1_phy>;
    phy-names = "pcie-phy0";
    num-lanes = <1>;
    #address-cells = <3>;
    #size-cells = <2>;
    device_type = "pci";
    ranges = <0x81000000 0 0              0x03000 0 0x00010000
              0x82000000 0 0x20013000 0x13000 0 0xffed000>;
    #interrupt-cells = <1>;
    linux,pci-domain = <0>;
    interrupt-map-mask = <0 0 0 7>;
    interrupt-map = <0 0 0 1 &pcie1_intc 1>,
    ....
};
```

RC Devicetree Node

```
pcie1_ep: pcie_ep@51000000 {
    compatible = "ti,dra7-pcie-ep";
    reg = <0x51000000 0x1000>, <0x51002000 0x14c>,
          <0x51001000 0x80>, <0x1000 0xFFFF000>;
    reg-names = "ep_dbics", "ti_conf", "ep_dbics2", "addr_space";
    interrupts = <0 232 0x4>;
    ti,hwmods = "pcie1";
    phys = <&pcie1_phy>;
    phy-names = "pcie-phy0";
    num-lanes = <1>;
    num-ib-windows = <4>;
    num-ob-windows = <16>;
    syscon-legacy-mode = <&scm_conf1 0x14 2>;
}; `
```

EP Devicetree Node

Configuring PCI Endpoint Function

```
# ls /sys/class/pci_epc/
            51000000.pcie_ep
```

list of PCI Endpoint Controllers

```
# ls /sys/bus/pci-epf/drivers
            pci_epf_test
```

list of PCI Endpoint Function Drivers

```
# mount -t configfs none /sys/kernel/config

# cd /sys/kernel/config/pci_ep/

# mkdir pci_epf_test.0

# cd pci_epf_test.0

# ls
            baseclass_code     function         revid              vendorid
            cache_line_size    interrupt_pin    subclass_code
            deviceid           peripheral       subsys_id
            epc                progif_code      subsys_vendor_id

# cat vendorid
            0xffff

# cat interrupt_pin
            0x0001
```

Creating Endpoint Function Device

```
# echo 0x104c > vendorid

# echo 16 >  msi_interrupts
```

Configuring Endpoint Function Device

```
# echo "51000000.pcie_ep" > epc
```

Binding Endpoint Function with Endpoint Controller

Happy Hacking!

Feedback:
kishon@ti.com
kishonvijayabraham@gmail.com