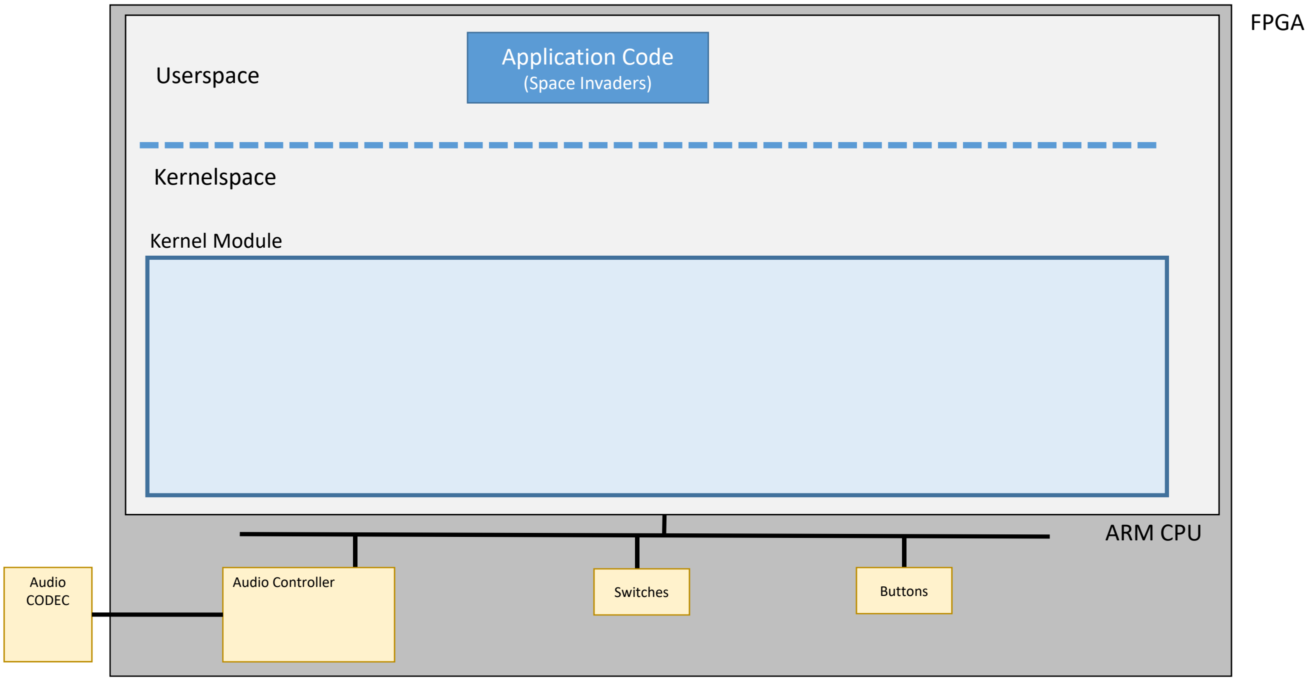
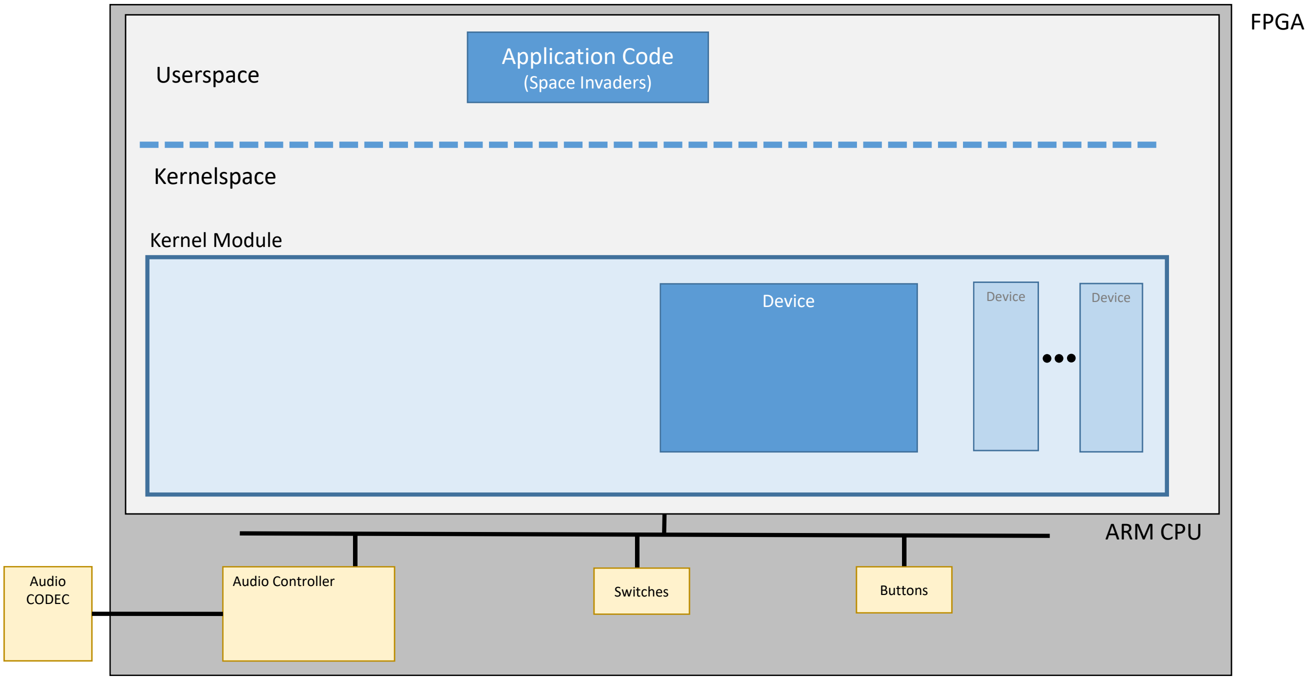
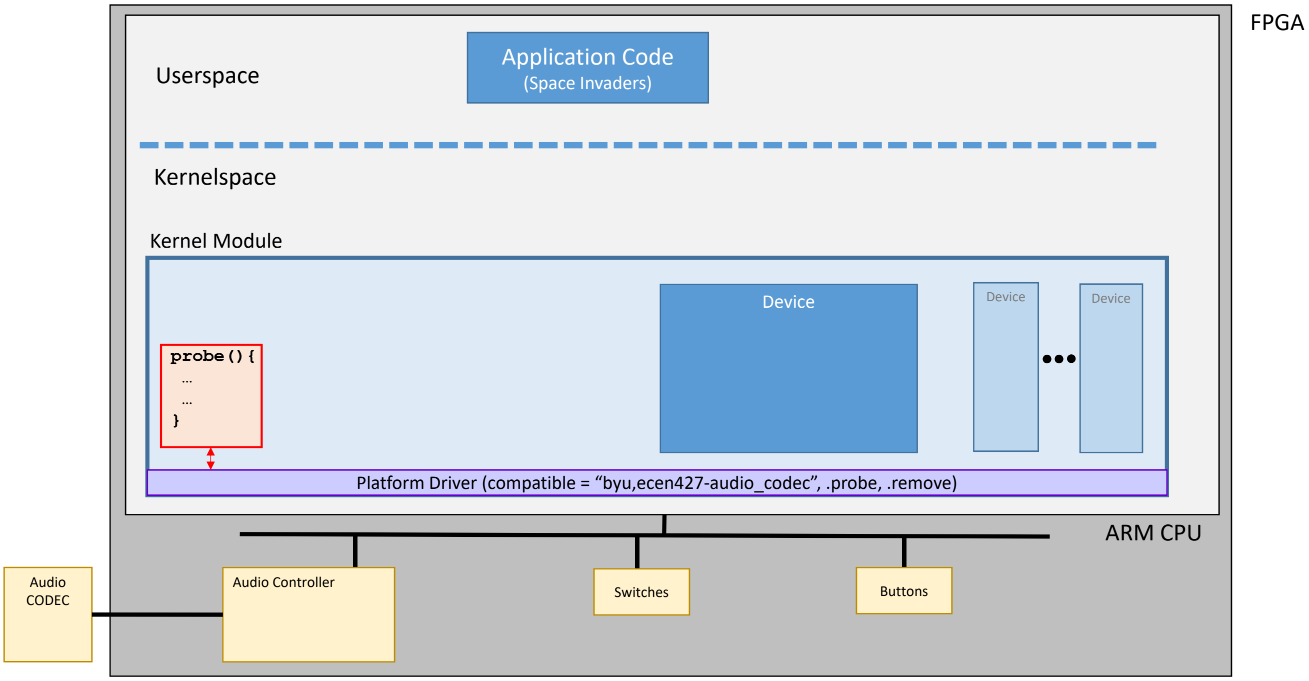
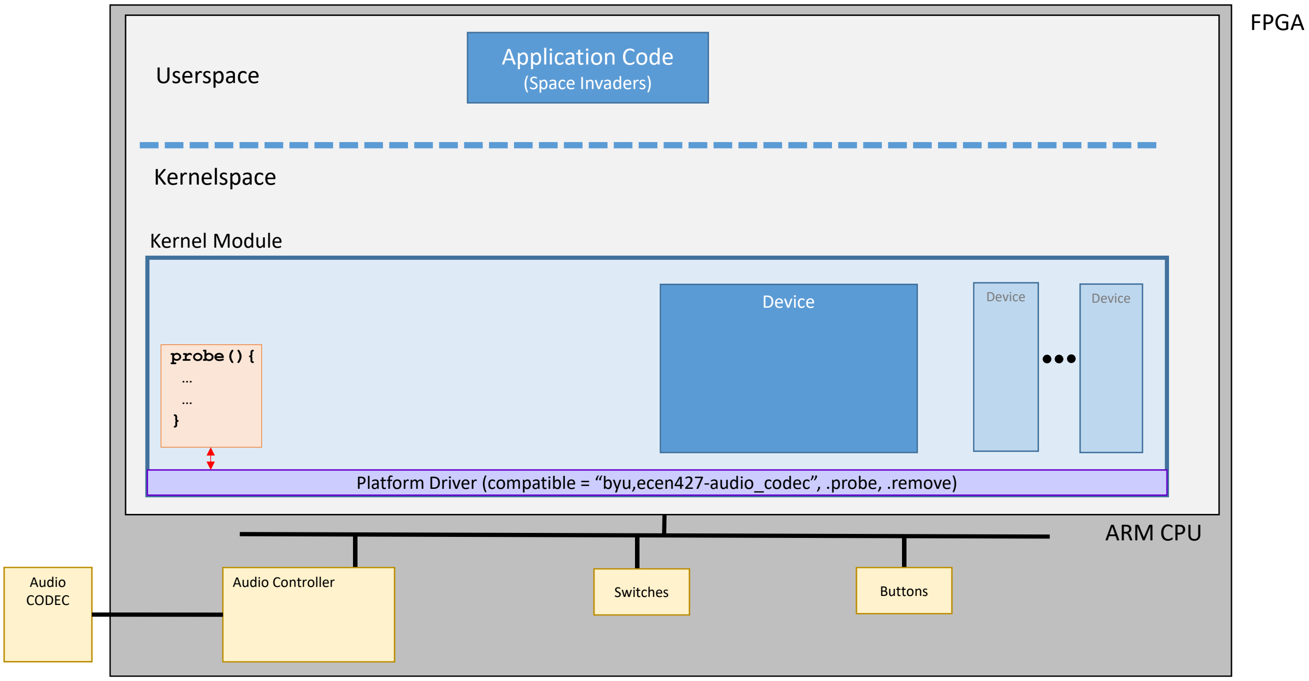

FPGA
Userspace
Application Code
(Space Invaders)
Kernelspace
Kernel Module
ARM CPU
Audio CODEC
Audio Controller
Switches
Buttons


FPGA
Userspace
Application Code
(Space Invaders)
Kernelspace
Kernel Module
Device
Device
•••
Device
ARM CPU
Audio CODEC
Audio Controller
Switches
Buttons



Driver needs to talk to the hardware

1. Need to figure out physical address
2. Need to reserve the physical address
3. Need to get a pointer (virtual address) to the physical address



FPGA

Userspace

Application Code
(Space Invaders)

Kernelspace

Kernel Module

Device

Device

Device

```
probe(){
  …
  …
}
```

Platform Driver (compatible = “byu,ecen427-audio_codec”, .probe, .remove)

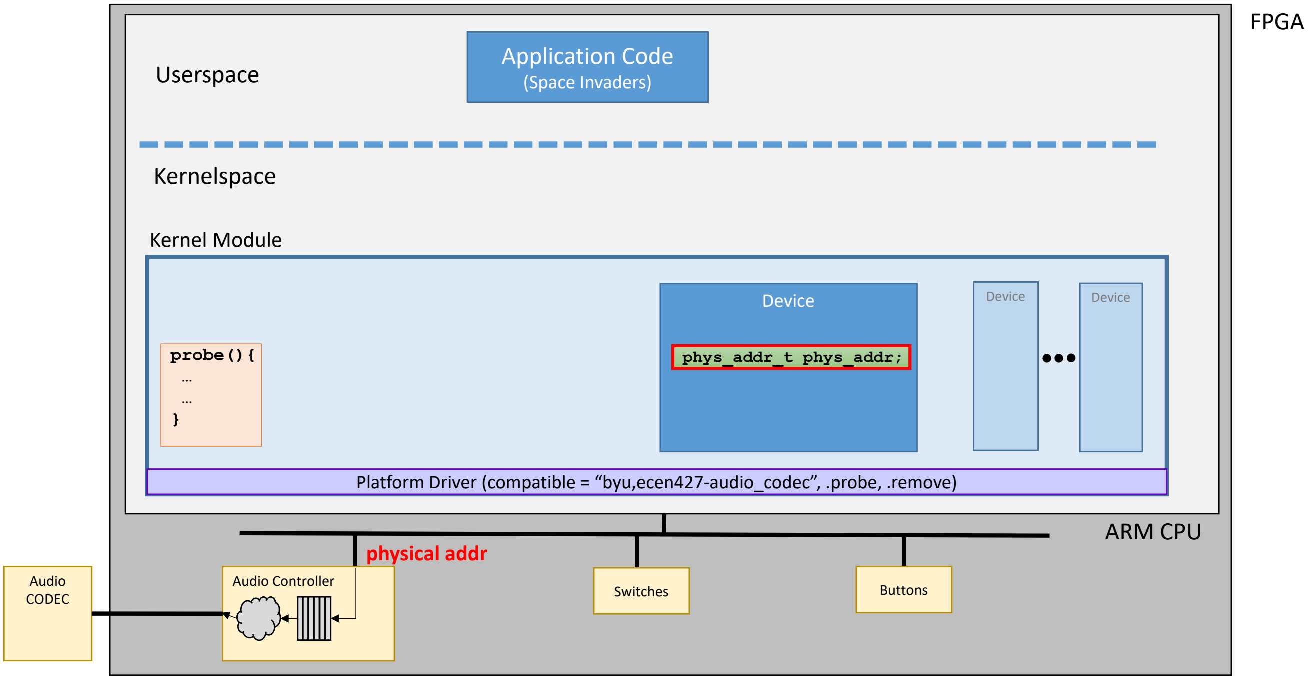
ARM CPU

Audio CODEC

Audio Controller

Switches

Buttons



FPGA

Userspace

Application Code
(Space Invaders)

Kernelspace

Kernel Module

Device

```
phys_addr_t phys_addr;
```

Device

Device

```
probe(){
  …
  …
}
```

Platform Driver (compatible = “byu,ecen427-audio_codec”, .probe, .remove)

ARM CPU

physical addr

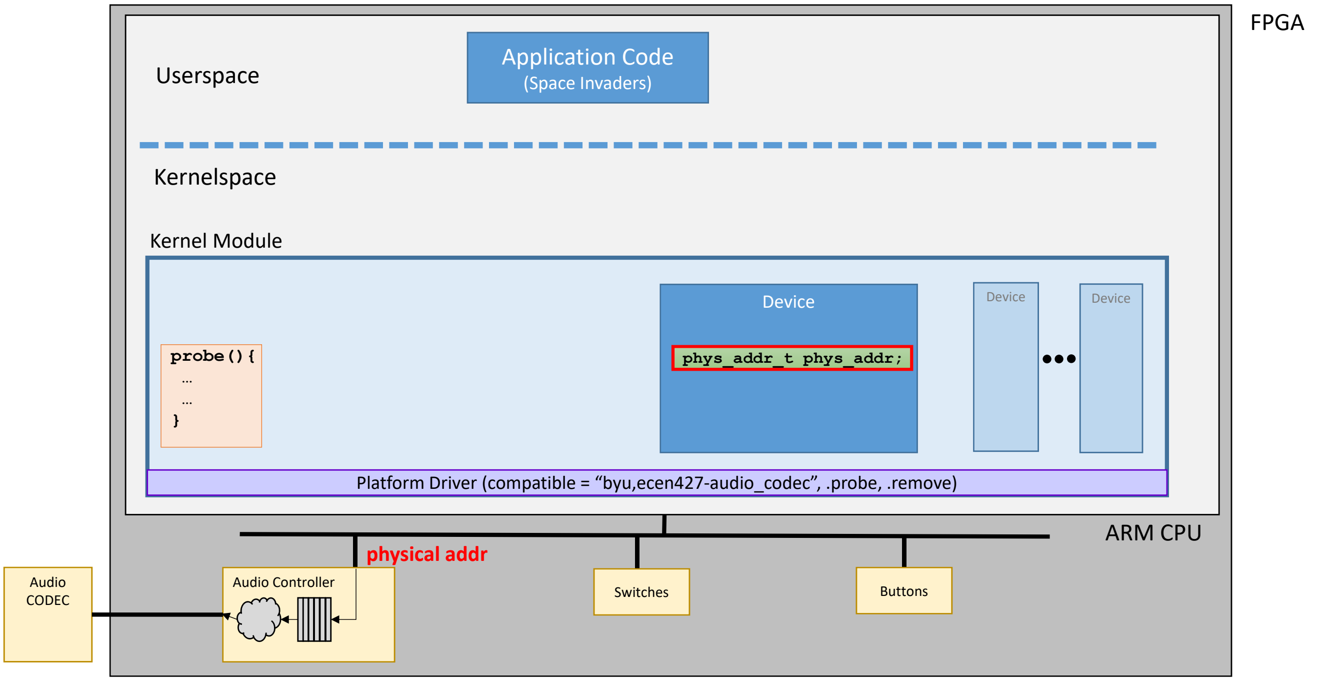
Audio CODEC

Audio Controller

Switches

Buttons

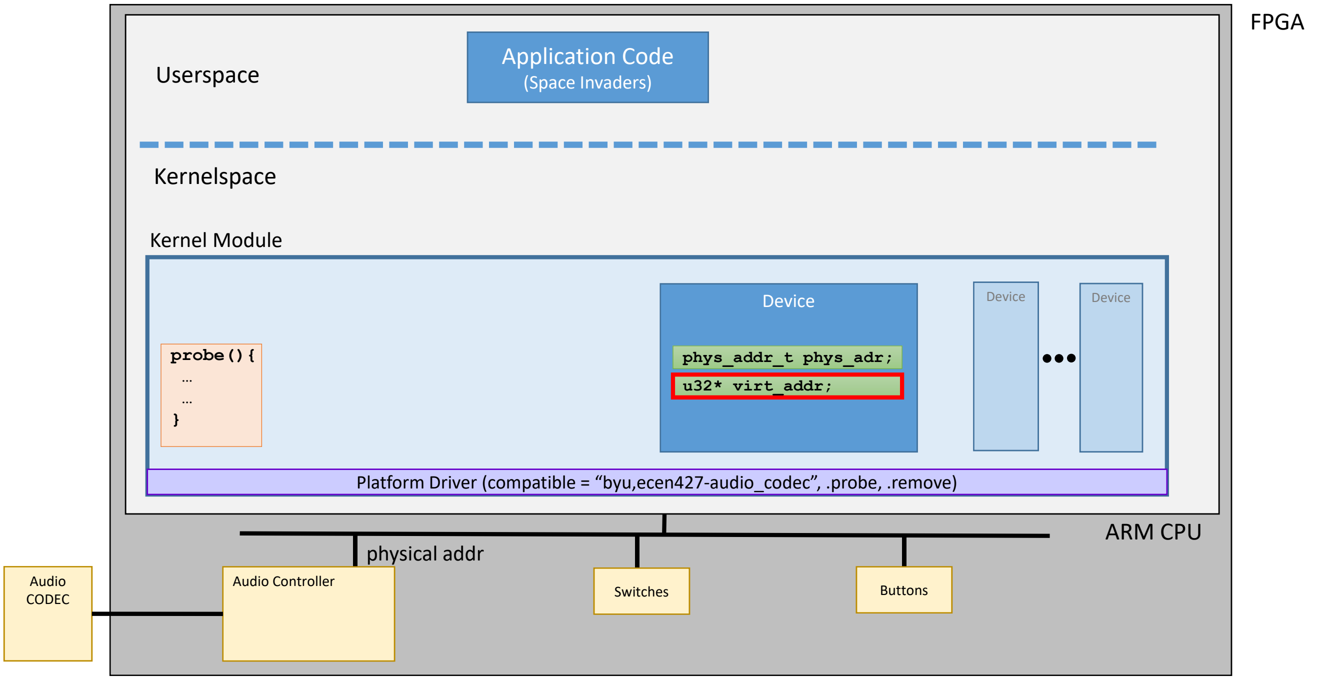
```
platform_get_resource(struct plaform_device * dev, IORESOURCE_MEM, 0);
```



```
probe(){
  …
  …
}
```

```
phys_addr_t phys_addr;
```

```
request_mem_region(phys_addr, size, MODULE_NAME);
```



```
probe(){
  …
  …
}
```

```
phys_addr_t phys_adr;
u32* virt_addr;
```

```
virt_addr = ioremap(phys_addr, size);
```

Driver needs to talk to the hardware

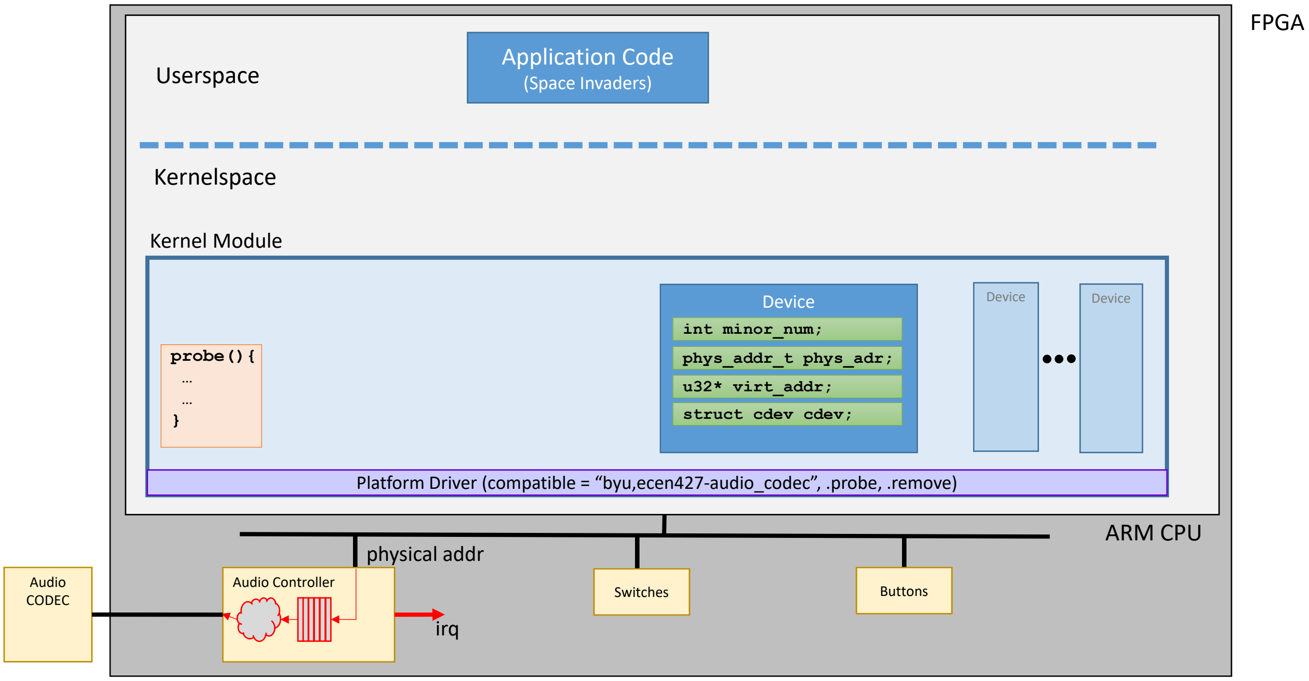
1. Need to figure out physical address

2. Need to reserve the physical address

3. Need to get a pointer (virtual address) to the physical address

4. Talk to the hardware with:
   - iowrite32 (value, virt_addr + offset)
   - ioread32(virt_addr + offset)

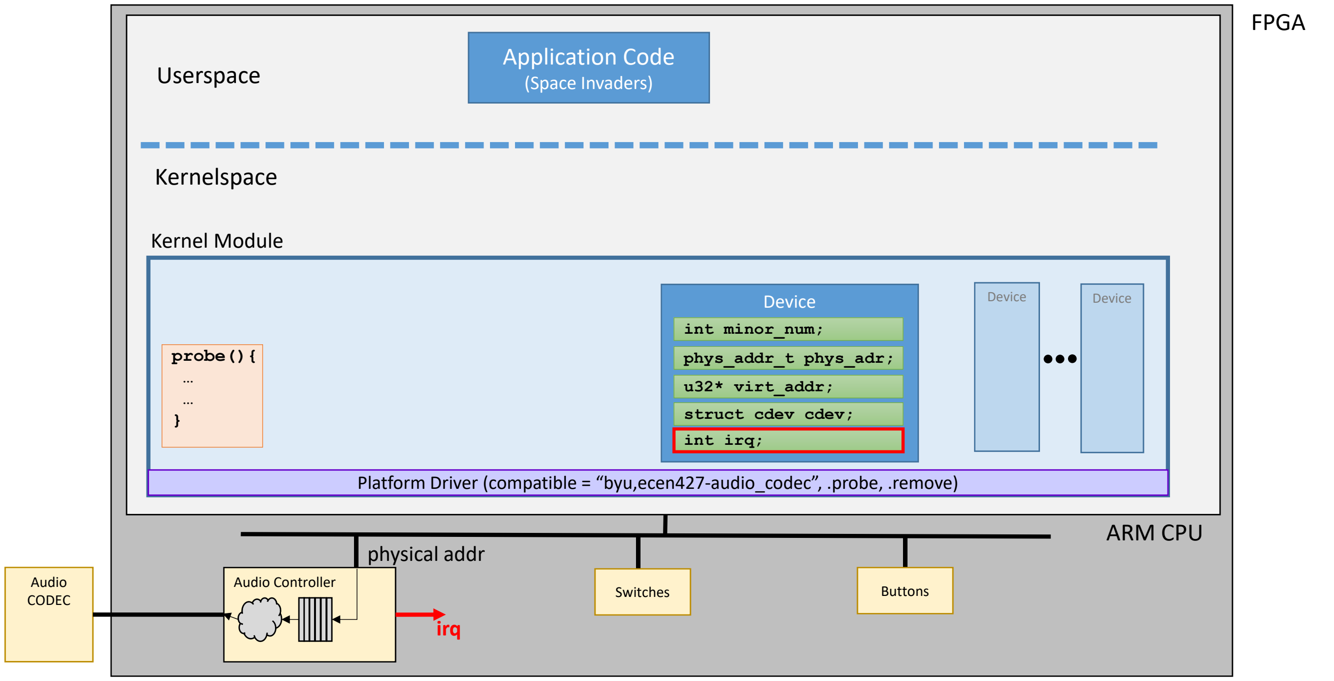
Driver Needs to Handle Interrupts



Driver Needs to Handle Interrupts

1. Get IRQ Number

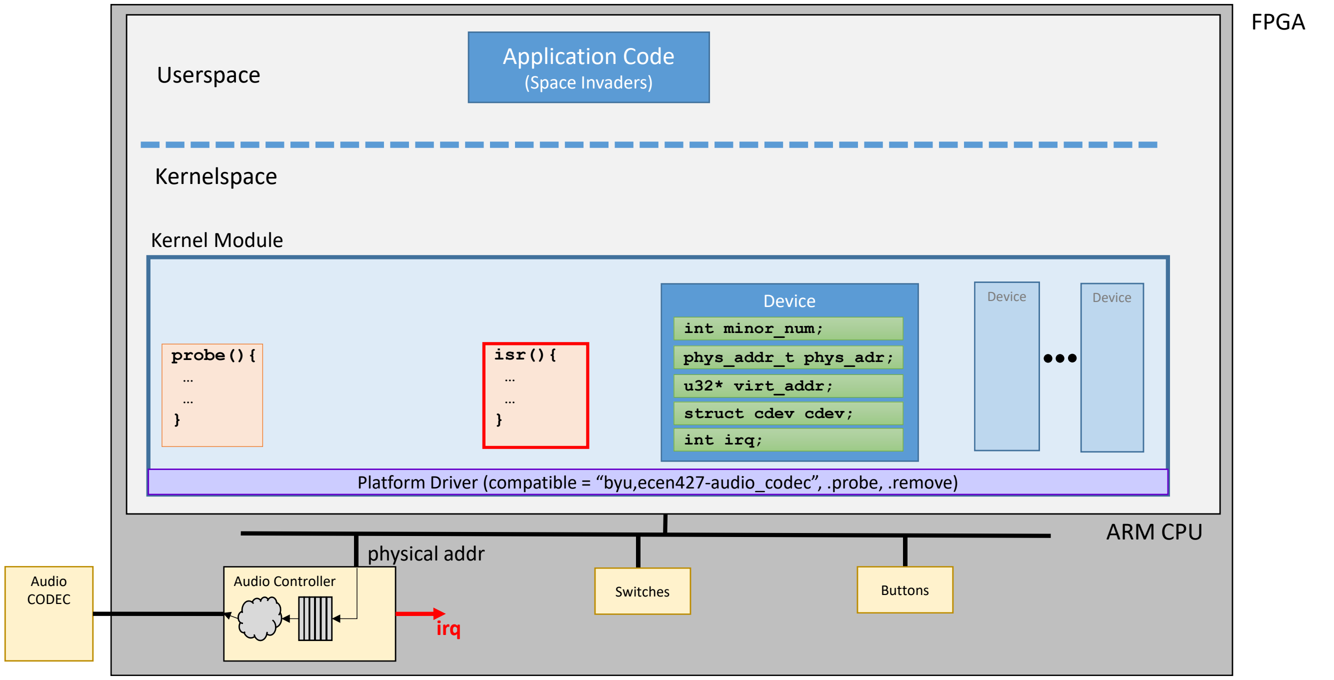
2. Register Interrupt Handler with Linux



```
probe(){
  …
  …
}
```

```
int minor_num;
phys_addr_t phys_adr;
u32* virt_addr;
struct cdev cdev;
int irq;
```

```
irq = platform_get_resource(struct platform_device * dev, IORESOURCE_IRQ, 0);
```



```
probe(){
  …
  …
}
```

```
isr(){
  …
  …
}
```

```
int minor_num;
phys_addr_t phys_adr;
u32* virt_addr;
struct cdev cdev;
int irq;
```

```
request_irq(irq, isr, IRQ_NO_FLAGS, MODULE_NAME, void*)
```

Driver Needs to Handle Interrupts

1. Get IRQ Number

2. Register Interrupt Handler with Linux

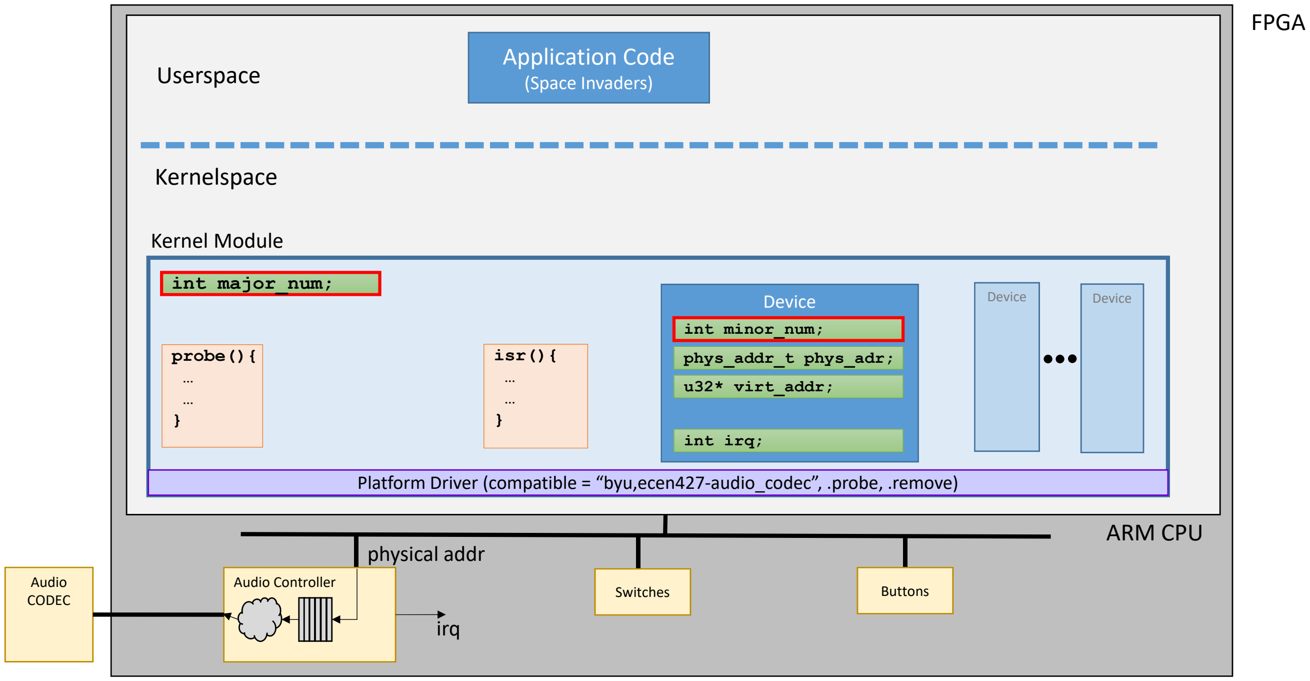
User Code Needs to Talk to Driver

End Goal: Create a device file (/dev/xxx) that we can read() and write() to. (*Recall how you used /dev/uio*)

The device file (/dev/xxx) is an interface to a **character device**.

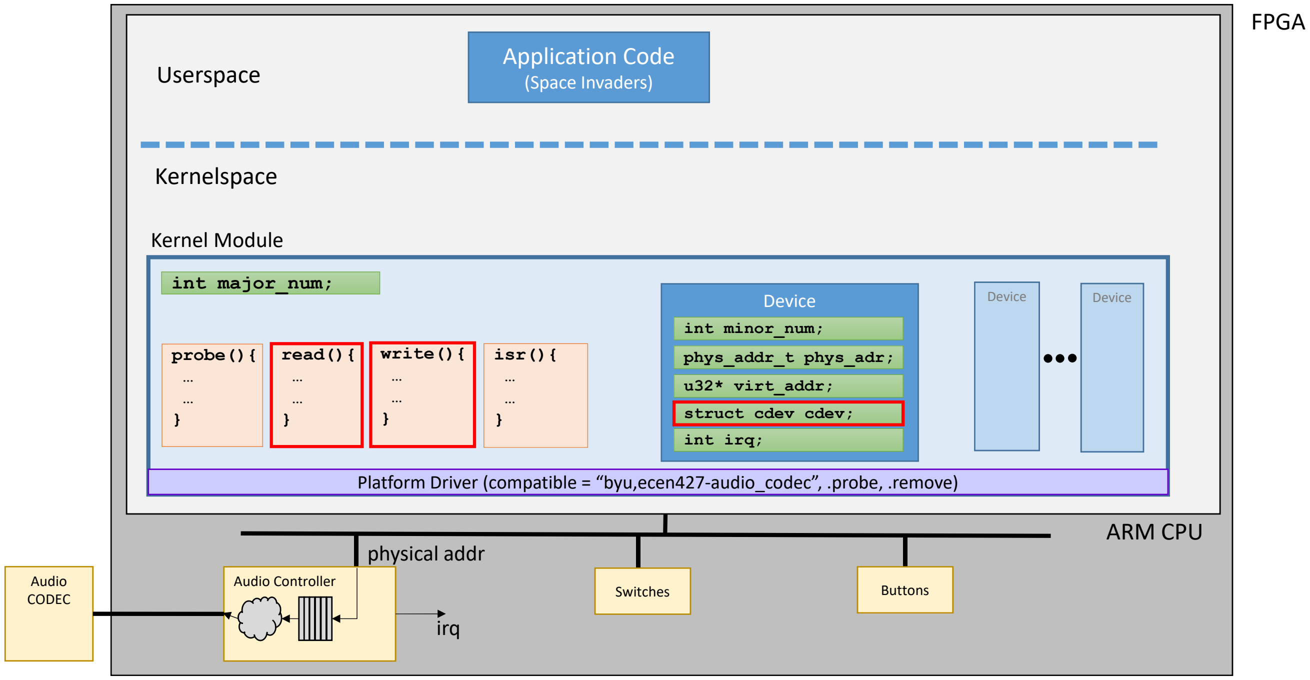
Steps:

1. Create a character device
2. Create a device file



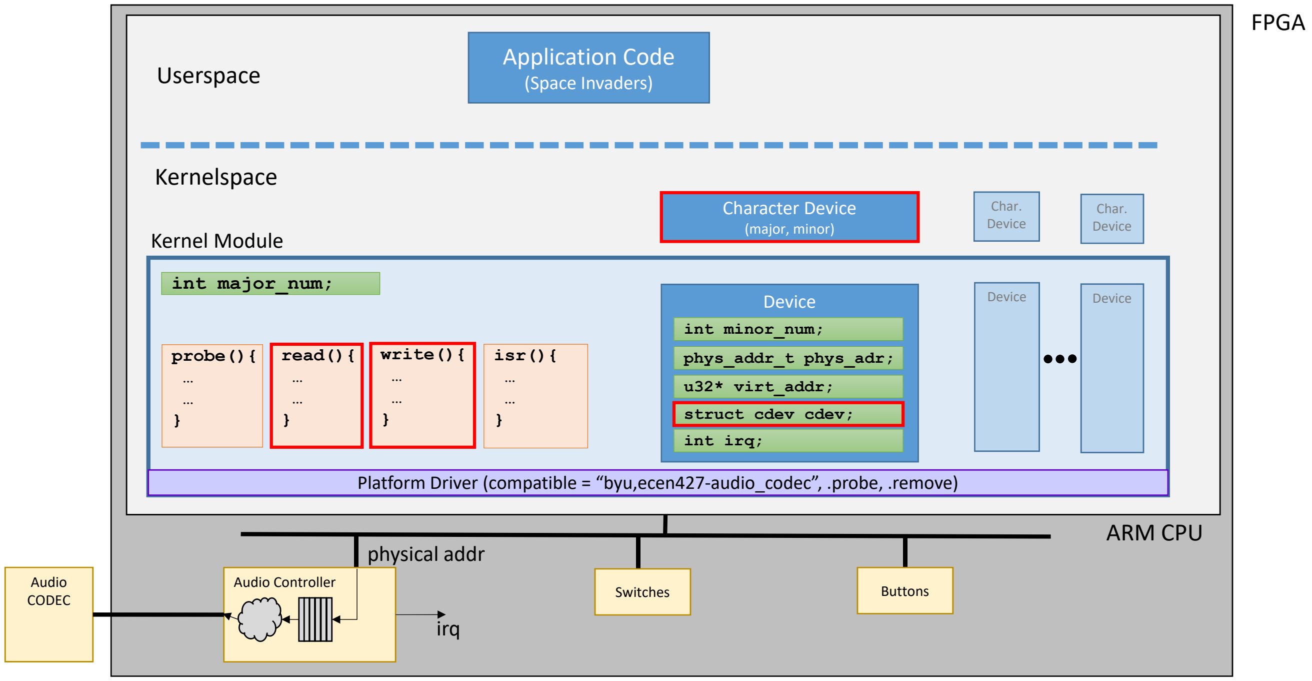
```
int major_num;

probe(){
  …
  …
}

isr(){
  …
  …
}

int minor_num;
phys_addr_t phys_adr;
u32* virt_addr;
int irq;
```

```
alloc_chrdev_region(dev_t * output, minor_start, count, MODULE_NAME)
```



```
int major_num;

probe(){
  …
  …
}

read(){
  …
  …
}

write(){
  …
  …
}

isr(){
  …
  …
}

int minor_num;
phys_addr_t phys_adr;
u32* virt_addr;
struct cdev cdev;
int irq;
```

```
cdev_init (struct cdev*, fops* {.read, .write, .seek})
```



FPGA

Userspace

Application Code
(Space Invaders)

Kernelspace

Character Device
(major, minor)

Char. Device

Char. Device

Kernel Module

int major_num;

probe() { … … }

read() { … … }

write() { … … }

isr() { … … }

Device

int minor_num;

phys_addr_t phys_adr;

u32* virt_addr;

struct cdev cdev;

int irq;

Device

Device

Platform Driver (compatible = "byu,ecen427-audio_codec", .probe, .remove)

ARM CPU

physical addr

Audio CODEC

Audio Controller

irq

Switches

Buttons

`cdev_add (struct cdev*, dev_t, count)`

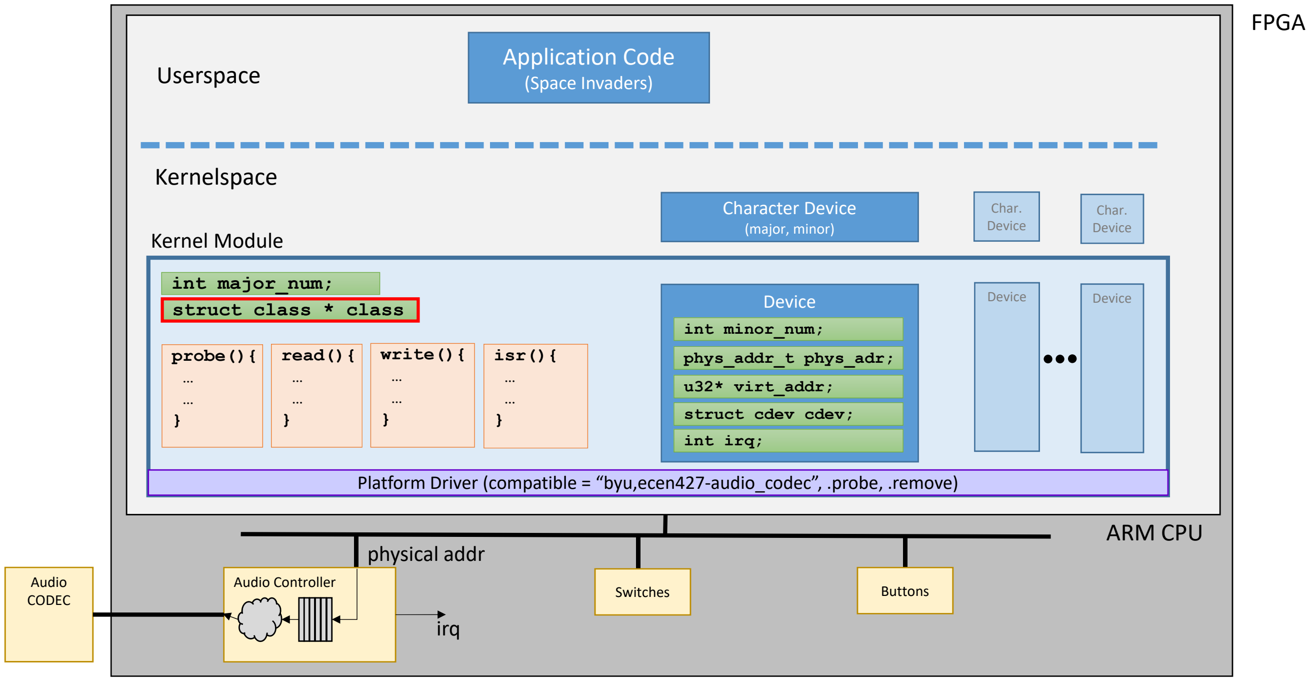
User Code Needs to Talk to Driver

End Goal: Create a device file (/dev/xxx) that we can read() and write() to. (*Recall how you used /dev/uio*)

The device file (/dev/xxx) is an interface to a **character device**.

Steps:

1. Create a character device
2. Create a device file



FPGA

Userspace

Application Code
(Space Invaders)

Kernelspace

Character Device
(major, minor)

Char. Device

Char. Device

Kernel Module

```
int major_num;
struct class * class
```

```
probe(){
 …
 …
}
```

```
read(){
 …
 …
}
```

```
write(){
 …
 …
}
```

```
isr(){
 …
 …
}
```

Device

```
int minor_num;
phys_addr_t phys_adr;
u32* virt_addr;
struct cdev cdev;
int irq;
```

Device

●●●

Device

Platform Driver (compatible = “byu,ecen427-audio_codec”, .probe, .remove)

ARM CPU

physical addr

Audio CODEC

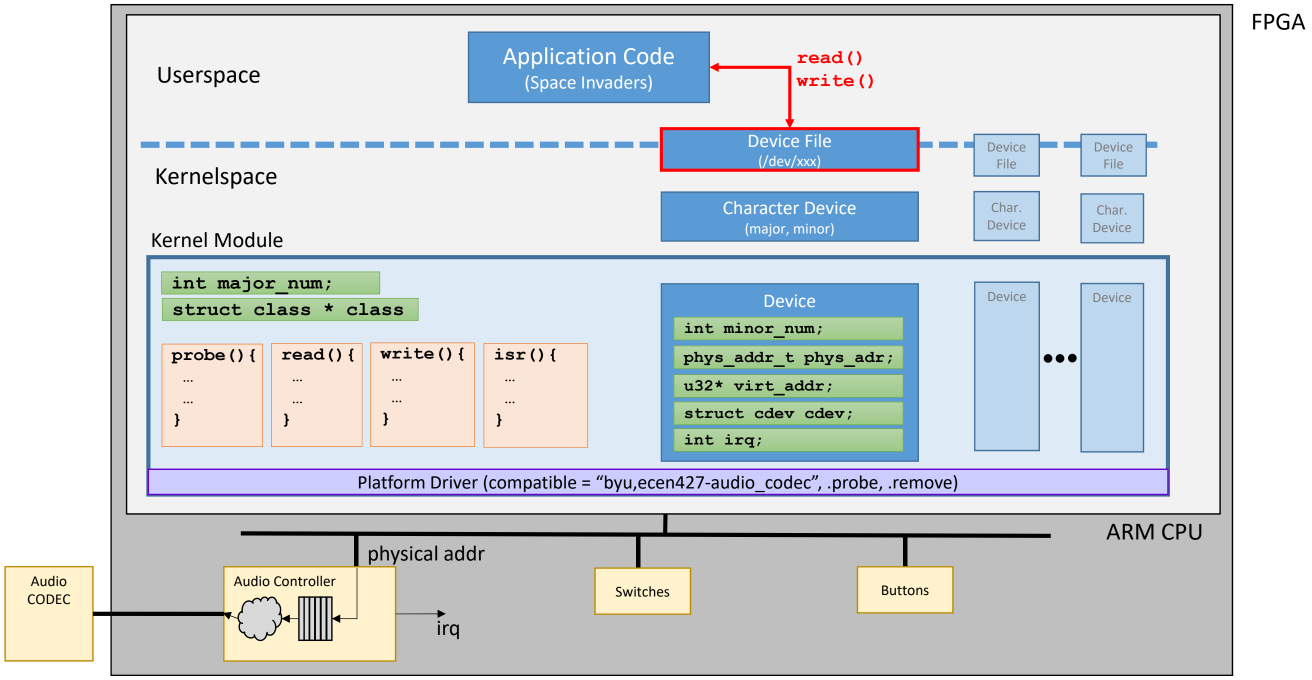
Audio Controller

irq

Switches

Buttons

```
class_create(owner = THIS_MODULE, “my class name”)
```



FPGA

Userspace

Application Code
(Space Invaders)

```
read()
write()
```

Device File
(/dev/xxx)

Device File

Device File

Kernelspace

Character Device
(major, minor)

Char. Device

Char. Device

Kernel Module

```
int major_num;
struct class * class
```

```
probe(){
 …
 …
}
```

```
read(){
 …
 …
}
```

```
write(){
 …
 …
}
```

```
isr(){
 …
 …
}
```

Device

```
int minor_num;
phys_addr_t phys_adr;
u32* virt_addr;
struct cdev cdev;
int irq;
```

Device

●●●

Device

Platform Driver (compatible = “byu,ecen427-audio_codec”, .probe, .remove)

ARM CPU

physical addr

Audio CODEC

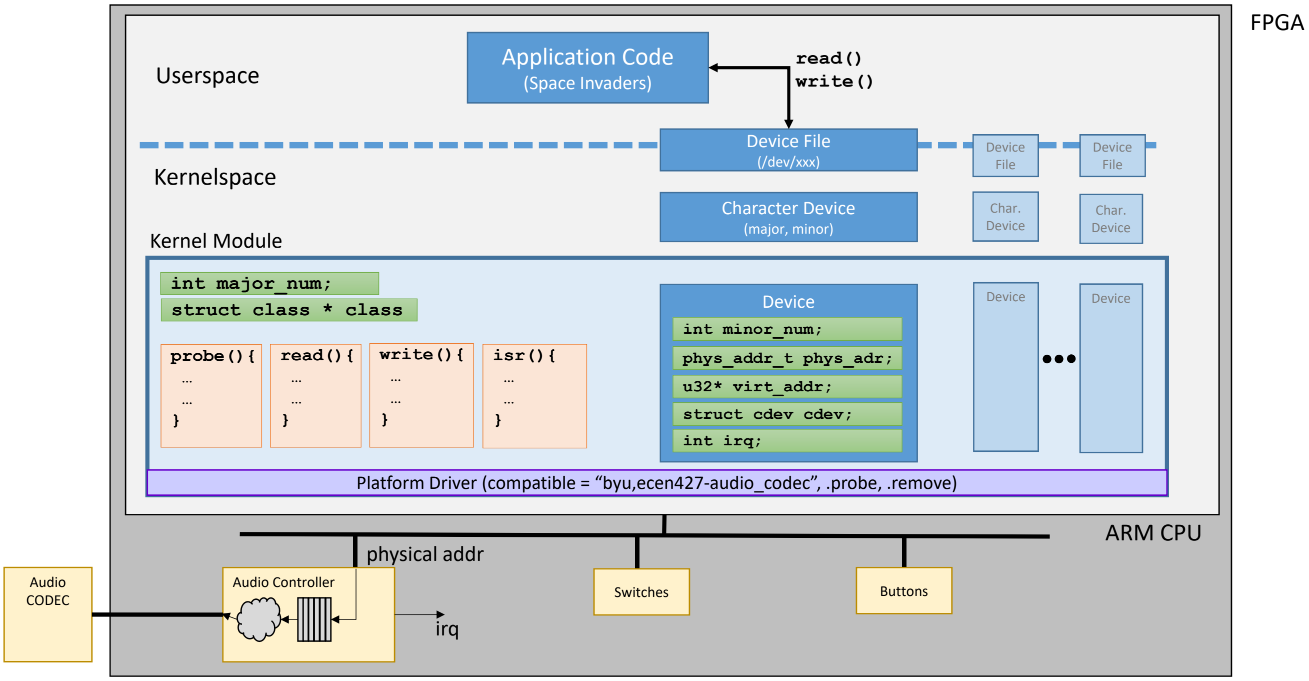
Audio Controller

irq

Switches

Buttons

```
device_create (struct class*, parent = NULL, dev_t, “your device name”)
```



FPGA

Userspace

Application Code
(Space Invaders)

read()
write()

Kernelspace

Device File
(/dev/xxx)

Device File

Device File

Character Device
(major, minor)

Char. Device

Char. Device

Kernel Module

int major_num;
struct class * class

probe(){ … … }

read(){ … … }

write(){ … … }

isr(){ … … }

Device

int minor_num;
phys_addr_t phys_adr;
u32* virt_addr;
struct cdev cdev;
int irq;

Device

•••

Device

Platform Driver (compatible = “byu,ecen427-audio_codec”, .probe, .remove)

ARM CPU

physical addr

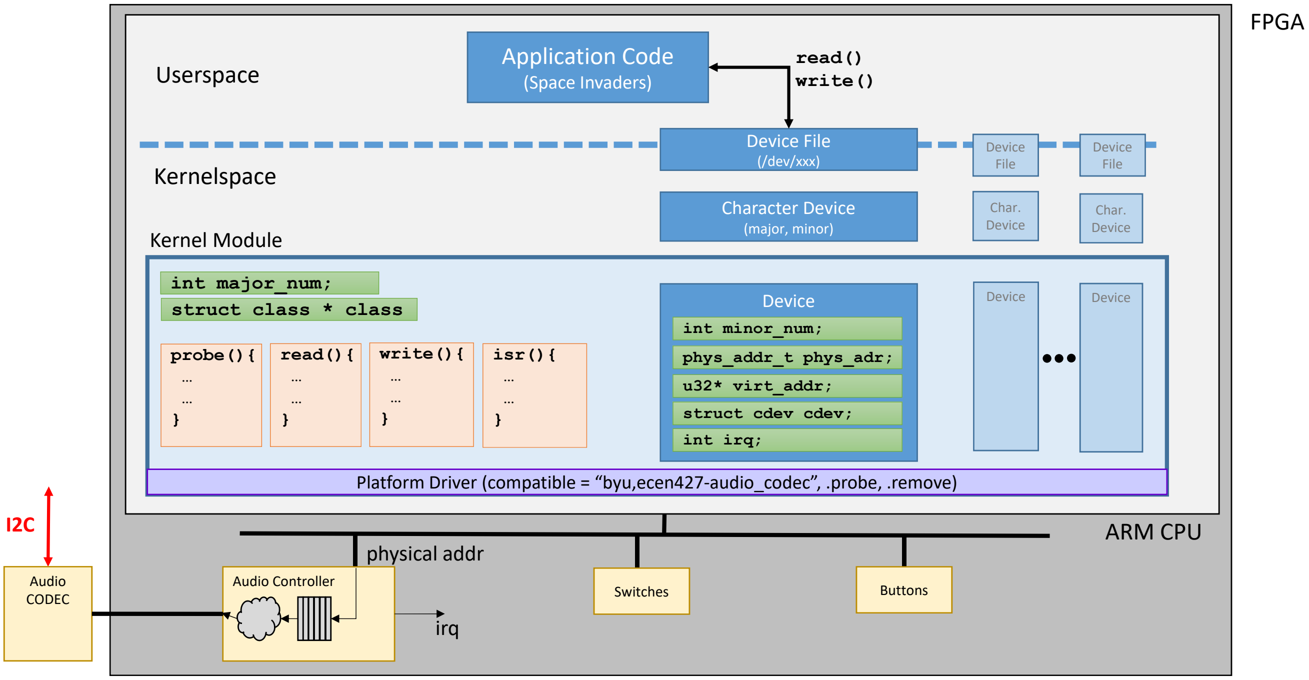
Audio CODEC

Audio Controller

irq

Switches

Buttons



FPGA

Userspace

Application Code
(Space Invaders)

read()
write()

Kernelspace

Device File
(/dev/xxx)

Device File

Device File

Character Device
(major, minor)

Char. Device

Char. Device

Kernel Module

int major_num;
struct class * class

probe(){ … … }

read(){ … … }

write(){ … … }

isr(){ … … }

Device

int minor_num;
phys_addr_t phys_adr;
u32* virt_addr;
struct cdev cdev;
int irq;

Device

•••

Device

Platform Driver (compatible = “byu,ecen427-audio_codec”, .probe, .remove)

ARM CPU

I2C

physical addr

Audio CODEC

Audio Controller

irq

Switches

Buttons