

Uio Driver Modification For AXI-Stream IP

Liu, Yu-Tang

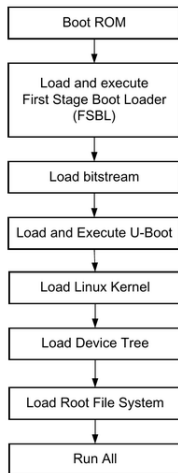
July 13, 2018

Outline

- ▶ Embedded Linux on zedboard
- ▶ UIO driver
- ▶ DMA and AXI4
- ▶ UIO driver modification

Embedded Linux on zedboard

Embedded Linux on zedboard



X16812-041816

Figure 6-1: Linux Boot Process on the Target Platform

Embedded Linux on zedboard

- ▶ Phase 0
 - ▶ Load boot rom
 - ▶ Check JP7-JP11
- ▶ Phase 1
 - ▶ FSBL
- ▶ Phase 2
 - ▶ Software

Embedded Linux on zedboard

- ▶ So we need
 - ▶ Boot.bin
 - ▶ Uboot
 - ▶ devicetree
 - ▶ rootfs

Embedded Linux on zedboard

- ▶ Boot.bin
 - ▶ fsbl.elf
 - ▶ boot.elf
 - ▶ vivado-design.bit

Embedded Linux on zedboard

Uboot

- ▶ Das U-Boot (subtitled "the Universal Boot Loader" and often shortened to U-Boot) is an open source, primary boot loader used in embedded devices to package the instructions to boot the device's operating system kernel.

Embedded Linux on zedboard

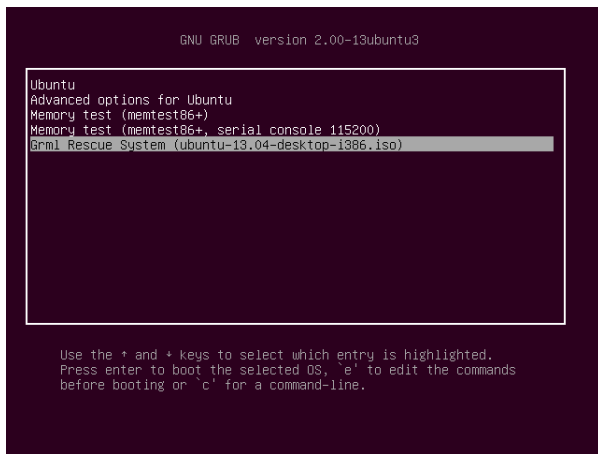


Figure: grub

Embedded Linux on zedboard

devicetree

- ▶ In computing, a device tree (also written devicetree) is a data structure describing the hardware components of a particular computer so that the operating system's kernel can use and manage those components, including the CPU or CPUs, the memory, the buses and the peripherals.

Embedded Linux on zedboard

```
usb@e0003000 {
    compatible = "xlnx,zynq-usb-2.20a", "chipidea,usb2";
    status = "disabled";
    clocks = <0x1 0x1d>;
    interrupt-parent = <0x3>;
    interrupts = <0x0 0x2c 0x4>;
    reg = <0xe0003000 0x1000>;
    phy_type = "ulpi";
};

watchdog@f8005000 {
    clocks = <0x1 0x2d>;
    compatible = "cdns,wdt-r1p2";
    interrupt-parent = <0x3>;
    interrupts = <0x0 0x9 0x1>;
    reg = <0xf8005000 0x1000>;
    timeout-sec = <0xa>;
};

my_AES@43c00000 {
    compatible = "xlnx,my-AES-1.0";
    reg = <0x43c00000 0x1000>;
    interrupts = <0 29 1>;
    interrupt-parent = <0x3>;
    xlnx,s00-axi-addr-width = <0x6>;
    xlnx,s00-axi-data-width = <0x20>;
};

};

phy0 {
    compatible = "ulpi-phy";
    #phy-cells = <0x0>;
    reg = <0xe0002000 0x1000>;
    view-port = <0x170>;
    drv-vbus;
    linux,phandle = <0x6>;
    phandle = <0x6>;
};

};
```

Embedded Linux on zedboard

root file system

- ▶ The root filesystem is the filesystem that is contained on the same partition on which the root directory is located, and it is the filesystem on which all the other filesystems are mounted (i.e., logically attached to the system) as the system is booted up (i.e., started up).

Embedded Linux on zedboard

Now we can run linux on zedboard!

UIO driver

Uio Driver

About UIO

- ▶ only one small kernel module to write and maintain.
- ▶ develop the main part of your driver in user space, with all the tools and libraries you're used to.
- ▶ bugs in your driver won't crash the kernel.
- ▶ updates of your driver can take place without recompiling the kernel.

Uio Driver

How UIO works?

- ▶ Each UIO device is accessed through a device file and several sysfs attribute files. The device file will be called `/dev/uio0` for the first device, and `/dev/uio1`, `/dev/uio2` and so on for subsequent devices.
`/dev/uioX` is used to access the address space of the card.
Just use `mmap()` to access registers or RAM locations of your card.

Uio Driver

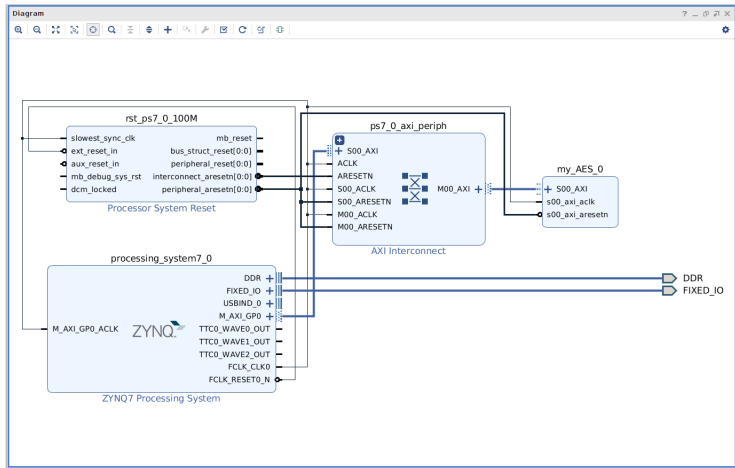
How to use UIO

```
int fd = open("/dev/uio0", O_RDWR);

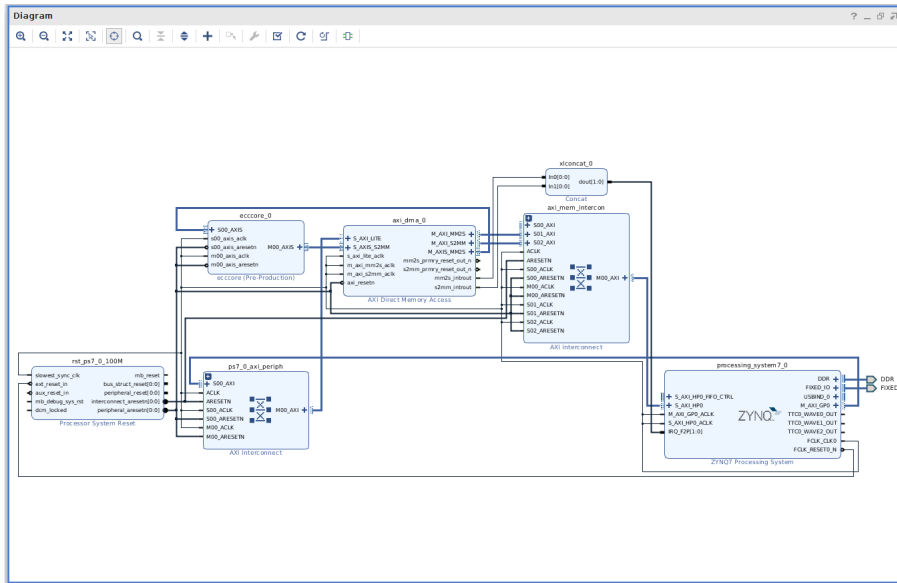
void *ptr = mmap(0, 0x10000, PROT_READ|PROT_WRITE, MAP_SHARED, fd, 0);

volatile uint32_t *ctrl = (uint32_t *)ptr;
volatile uint32_t *memA = (uint32_t *)ptr + 1;
```

Uio Driver



Uio Driver



DMA and AXI4

DMA and AXI4

DMA

- ▶ Direct memory access (DMA) is a feature of computer systems that allows certain hardware subsystems to access main system memory (Random-access memory), independent of the central processing unit (CPU).

DMA and AXI4

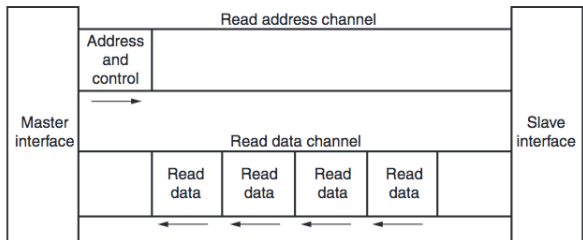
AXI4

- ▶ AMBA® AXI4 (Advanced eXtensible Interface 4) is the fourth generation of the AMBA interface specification from ARM®. Xilinx Vivado Design Suite 2014 and ISE Design Suite 14 extends the Xilinx platform design methodology with the semiconductor industry's first AXI4 Compliant Plug-and-Play IP.

DMA and AXI4

- ▶ AXI4
 - ▶ Traditional memory mapped address/data interface.
 - ▶ Data burst support.
- ▶ AXI4-Lite
 - ▶ Traditional memory mapped address/data interface.
 - ▶ Single data cycle only.
- ▶ AXI4-stream
 - ▶ Data-only burst.

DMA and AXI4



X12076

Figure 1-1: Channel Architecture of Reads

DMA and AXI4

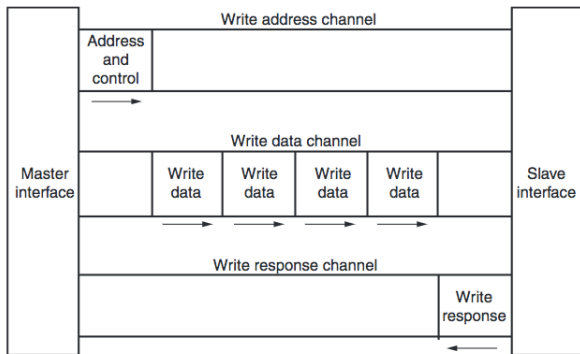
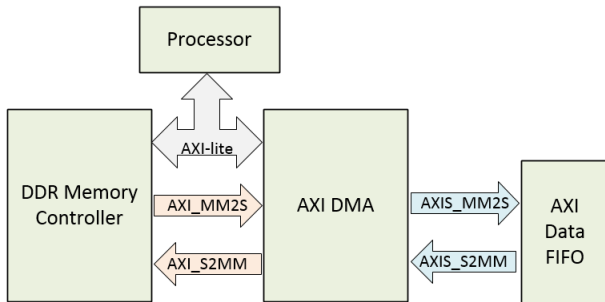


Figure 1-2: Channel Architecture of Writes

DMA and AXI4



Copyright 2014 Xilinx, Inc. All rights reserved.

DMA and AXI4

- ▶ If our custom IP uses AXI-Stream registers, we can't use UIO driver to mount our IP as a device.

DMA and AXI4

But we still want to use our IP in userspace!

DMA and AXI4

But we still want to use our IP in userspace!

->UIO driver modification

UIO driver modification

UIO driver modification

In fact, DMA component is driven by the AXI-DMA module provided by xilinx linux kernel.

So we can use Linux Kernel DMA API in our UIO driver.

UIO driver modification

- ▶ DMA API
 - ▶ `dma_request_slave_channel`
 - ▶ `dmaengine_prep_slave_sg`
 - ▶ `dmaengine_submit`

UIO driver modification

We can create a virtual device node to apply UIO driver.

UIO driver modification

devicetree will be like:

```
xlnx,include-sg;
loopback_dma_mm2s_chan: dma-channel@40410000 {
    compatible = "xlnx,axi-dma-mm2s-channel";
    interrupt-parent = <0 31 4>;
    interrupts = <0 31 4>; // concat port 2
                        // IRQ_F2P[15:0] == [91:84],[68:61]
                        // 2 -> 63 -> 63 - 32 = 31

    xlnx,datawidth = <0x20>; // 32-bit output
    xlnx,sg-length-width = <14>; // Width of Buffer Length Register (configured for 20 bits)

    xlnx,device-id = <0x1>; // what's this for?
};

loopback_dma_s2mm_chan: dma-channel@40410030 {
    compatible = "xlnx,axi-dma-s2mm-channel";
    interrupt-parent = <0 32 4>;
    interrupts = <0 32 4>; // concat port 3
                        // IRQ_F2P[15:0] == [91:84],[68:61]
                        // 3 -> 64 -> 64 - 32 = 32

    xlnx,datawidth = <0x20>; // 32-bit output
    xlnx,sg-length-width = <14>; // Width of Buffer Length Register (configured for 20 bits)

    xlnx,device-id = <0x1>; // what's this for?
};

udma0 {
    compatible = "generic-uio";

    dmas = <&loopback_dma 0 &loopback_dma 1>;
    dma-names = "loop_tx", "loop_rx"; // used when obtaining reference to above DMA core using dma_request_slave_channel()
    ezdma,dirs = <2 1>; // direction of DMA channel: 1 = RX (dev->cpu), 2 = TX (cpu->dev)
};
```

UIO driver modification

recall

```
int fd = open("/dev/uio0", O_RDWR);
```

```
void *ptr = mmap(0, 0x10000, PROT_READ|PROT_WRITE, MAP_SHARED,
```

UIO driver modification

`read()`

- ▶ Interrupts are handled by reading from `/dev/uioX`. A blocking `read()` from `/dev/uioX` will return as soon as an interrupt occurs. You can also use `select()` on `/dev/uioX` to wait for an interrupt. The integer value read from `/dev/uioX` represents the total interrupt count. You can use this number to figure out if you missed some interrupts.

UIO driver modification

`write()`

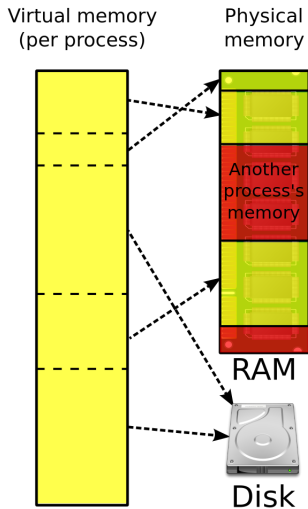
- ▶ UIO also implements a `write()` function, a `write()` to `/dev/uioX` will call the `irqcontrol()` function implemented by the driver. You have to write a 32-bit value that is usually either 0 or 1 to disable or enable interrupts. If a driver does not implement `irqcontrol()`, `write()` will return with `-ENOSYS`.

UIO driver modification

Virtual memory

- ▶ The addresses a program may use to reference memory are distinguished from the addresses the memory system uses to identify physical storage sites, and program generated addresses are translated automatically to the corresponding machine addresses.

UIO driver modification



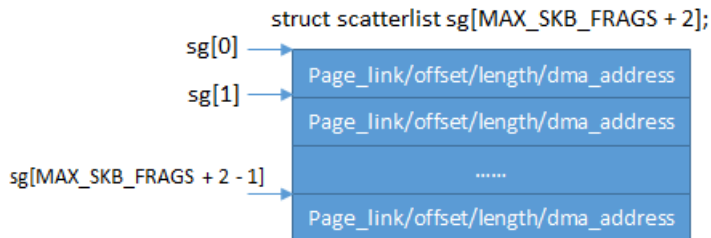
UIO driver modification

Scatter-Gather

Scatter-Gather DMA augments this technique by providing data transfers from one non-contiguous block of memory to another by means of a series of smaller contiguous-block transfers. The Lattice Scatter-Gather DMA Controller core implements a configurable, multi-channel, WISHBONE-compliant DMA controller with scatter-gather capability.

UIO driver modification

Scatterlist



Thanks!