Device Tree Overlay Concept, along with loading firmware(fpga).

Configurations

COMMAND - petalinux-config

Enable : DTG Settings ---> [*] Device Tree Overlay

Enable : FPGA Manager ---> [*] Fpga Manager

COMMAND – petalinux-config -c kernel

1. **FPGA Manager Framework**

   Enable : Device Drivers ---> [*] FPGA Configuration Framework

2 Device Tree Overlay ConfigFS

Enable : Device Drivers ---> [*] Device Tree and Open Firmware support

3 **Contiguous Memory Allocator (CMA)**
Enable : Memory Management options ---> [*] Contiguous Memory Allocator (CONFIG_CMA)

Optional for DMA
Enable : Library routines---> [*] DMA Contiguous Memory Allocator (CONFIG_DMA_CMA)

Tools

- **Device Tree Compiler (DTC)**: Version $\geq$ 1.4.4 for overlay support.
- **Bootgen**: Converts `.bit` to `.bin`.

1. Generating the Bitstream File

(a) Convert a **`.bit`** file to a **`.bin`** file using Bootgen:

(b) Command:

(c) **`bootgen -image Full_Bitstream.bif -arch zynq -process_bitstream bin`**

(d) Content of `Full_Bitstream.bif`:

```
all:
{
   design_1_wrapper.bit /* Bitstream file name */
}
```

2. Generating the pl.dtbo File

Automatically generated when we build project through petalinux in the path
<projrct-name>/components/plnx-workspace/device-tree/device-tree/pl.dtsi

Device Tree Overlay Structure

Example `pl.dtsi`:

```
/dts-v1/;
/plugin/;

{
    fragment@0 {
        target = <&fpga_full>;
        __overlay__ {
            firmware-name = "design_1_wrapper.bit.bin";
        };
    };

    fragment@1 {
        target = <&amba>;
        __overlay__ {
            axi_gpio_0: gpio@a0000000 {
                compatible = "xlnx,xps-gpio-1.00.a";
                gpio-controller;
                reg = <0x0 0xa0000000 0x0 0x10000>;
            };
        };
    };
};
```

Compiling the Overlay
Compile `pl.dtsi` to generate `.dtbo`:

```
dtc -O dtb -o pl.dtbo -b 0 -@ pl.dtsi
```

Note: pl.dtbo file generated along with images, when we build the project. If we want changes, we can change in this method.

Now load .dtbo and .bin file into sd card along with images and rootfs.
BOOT the board in usual way.

Steps to follow while board running

Steps to Load Full Bitstream
Once the Linux is up run the below commands to load the required Full Bitstream.
1) Set flags for Full Bitstream.

- **echo 0 > /sys/class/fpga_manager/fpga0/flags**

2) Copy the Full Bitstream (.bin) and pl.dtbo files into firmware folder

- **mount /dev/mmcblk0p1 /media/**
- **mkdir -p /lib/firmware**
- **cp /media/design_1_wrapper.bit.bin /lib/firmware/design_1_wrapper.bit.bin**
- **cp /media/pl.dtbo /lib/firmware/**

3) Apply DTBO (To add device nodes)

- **mkdir /configfs   (in root)**
- **mount -t configfs configfs /configfs**
- **cd /configfs/device-tree/overlays/**
- **mkdir full (in the path)**
- **echo -n "pl.dtbo" > full/path**

4) Steps to remove device nodes

- **rmdir full**

Steps to Re-Load Full Bitstream
1) Remove The overlay file will be applied earlier.

- **rmdir full**

2) Set flags for Full Bitstream.

- **echo 0 > /sys/class/fpga_manager/fpga0/flags**

3) Copy the Full Bitstream and pl.dtbo files into firmware folder

- cp /media/**design_1_wrapper.bit.bin /lib/firmware/design_1_wrapper.bit.bin**
- **cp /media/new-pl.dtbo /lib/firmware/**

3) Apply DTBO

- **mkdir full**
- **echo -n "new-pl.dtbo" > full/path**

Expected Output Using DTBO

```
root@xilinx-zc702-2018_1:/configfs/device-tree/overlays# echo -n
"pl.dtbo" > full/path
fpga_manager fpga0: writing zc702_wrapper.bit.bin to Xilinx Zynq
FPGA Manager
XGpio: /amba/gpio@41200000: registered, base is 902
root@xilinx-zc702-2018_1:/configfs/device-tree/overlays#
```

Using sysfs interface

Once the linux is up run the below commands to load the Bitstream.

1)Set flags for Full Bitstream.

- **echo 0 > /sys/class/fpga_manager/fpga0/flags**

2) Loading Bitstream into PL.

- **mkdir -p /lib/firmware**
- **cp /media/design_1_wrapper.bit.bin /lib/firmware/**
- **echo design_1_wrapper.bit.bin > /sys/class/fpga_manager/fpga0/firmware**

Steps for programming the Encrypted Bitstream

1)Set flags for Encrypted Bitstream

- **echo 0x4 > /sys/class/fpga_manager/fpga0/flags**

2)Load the Bitstream

- **mkdir -p /lib/firmware**
- **cp /media/enrypted.bit.bin /lib/firmware/**
- **echo enrypted.bit.bin > /sys/class/fpga_manager/fpga0/firmware**

Expected Output Using Sysfs

```
root@Xilinx:~# mount /dev/mmcblk0p1 /media/
root@Xilinx:~# mkdir -p /lib/firmware
root@Xilinx:~# echo 0 > /sys/class/fpga_manager/fpga0/flags
root@Xilinx:~# cp /media/system_wrapper.bit.bin /lib/firmware/
root@Xilinx:~# echo system_wrapper.bit.bin >
/sys/class/fpga_manager/fpga0/firmware
[  120.266851] fpga_manager fpga0: writing system_wrapper.bit.bin
to Xilinx Zynq FPGA Manager
root@Xilinx:~# devmem 0xA0000000
0x00000000
```

Verify the device tree update:

cat /proc/device-tree/path/to/updated/node

Testing attlantisian

Test Procedure

Using Device Tree Overlay

Steps to Load Full Bitstream

Once the Linux is up run the below commands to load the required Full Bitstream.

1) Set flags for Full Bitstream.

- **echo 0 > /sys/class/fpga_manager/fpga0/flags**

2) Copy the Full Bitstream (.bin) and pl.dtbo files into firmware folder

- **mount /dev/mmcblk0p1 /media/**
- **mkdir -p /lib/firmware**
- cp /media/**design_1_wrapper.bit.bin /lib/firmware/design_1_wrapper.bit.bin**
- **cp /media/pl.dtbo /lib/firmware/**

3) Apply DTBO (To add device nodes)

- **mkdir /configfs**
- **mount -t configfs configfs /configfs**
- **cd /configfs/device-tree/overlays/**
- **mkdir full**
- **echo -n "pl.dtbo" > full/path**

4) Steps to remove device nodes

- **rmdir full**

Steps to Re-Load Full Bitstream

1) Remove The overlay file will be applied earlier.

- **rmdir full**

2) Set flags for Full Bitstream.

- **echo 0** > **/sys/class/fpga_manager/fpga0/flags**

3) Copy the Full Bitstream and pl.dtbo files into firmware folder

- cp /media/**design_1_wrapper.bit.bin /lib/firmware/design_1_wrapper.bit.bin**
- **cp /media/new-pl.dtbo /lib/firmware/**

3) Apply DTBO

- **mkdir full**
- **echo -n "new-pl.dtbo"** > **full/path**

Expected Output Using DTBO

```
root@xilinx-zc702-2018_1:/configfs/device-tree/overlays# echo -n
"pl.dtbo" > full/path
fpga_manager fpga0: writing zc702_wrapper.bit.bin to Xilinx Zynq
FPGA Manager
XGpio: /amba/gpio@41200000: registered, base is 902
root@xilinx-zc702-2018_1:/configfs/device-tree/overlays#
```

Using sysfs interface

Once the linux is up run the below commands to load the Bitstream.
1)Set flags for Full Bitstream.

- **echo 0** > **/sys/class/fpga_manager/fpga0/flags**

2) Loading Bitstream into PL.

- **mkdir -p /lib/firmware**
- **cp /media/design_1_wrapper.bit.bin /lib/firmware/**
- **echo design_1_wrapper.bit.bin** > **/sys/class/fpga_manager/fpga0/firmware**

Steps for programming the Encrypted Bitstream

1)Set flags for Encrypted Bitstream

- **echo 0x4 > /sys/class/fpga_manager/fpga0/flags**

2)Load the Bitstream

- **mkdir -p /lib/firmware**
- **cp /media/enrypted.bit.bin /lib/firmware/**
- **echo enrypted.bit.bin** > **/sys/class/fpga_manager/fpga0/firmware**

Expected Output Using Sysfs

```
root@Xilinx:~# mount /dev/mmcblk0p1 /media/
root@Xilinx:~#mkdir -p /lib/firmware
```

```
root@Xilinx:~# echo 0 > /sys/class/fpga_manager/fpga0/flags
root@Xilinx:~# cp /media/system_wrapper.bit.bin /lib/firmware/
root@Xilinx:~# echo system_wrapper.bit.bin >
/sys/class/fpga_manager/fpga0/firmware
[  120.266851] fpga_manager fpga0: writing system_wrapper.bit.bin
to Xilinx Zynq FPGA Manager
root@Xilinx:~# devmem 0xA0000000
0x00000000
```