1

NXKR_YoungSikYang

Exported on 02/27/2024

Table of Contents

1 Driver Model

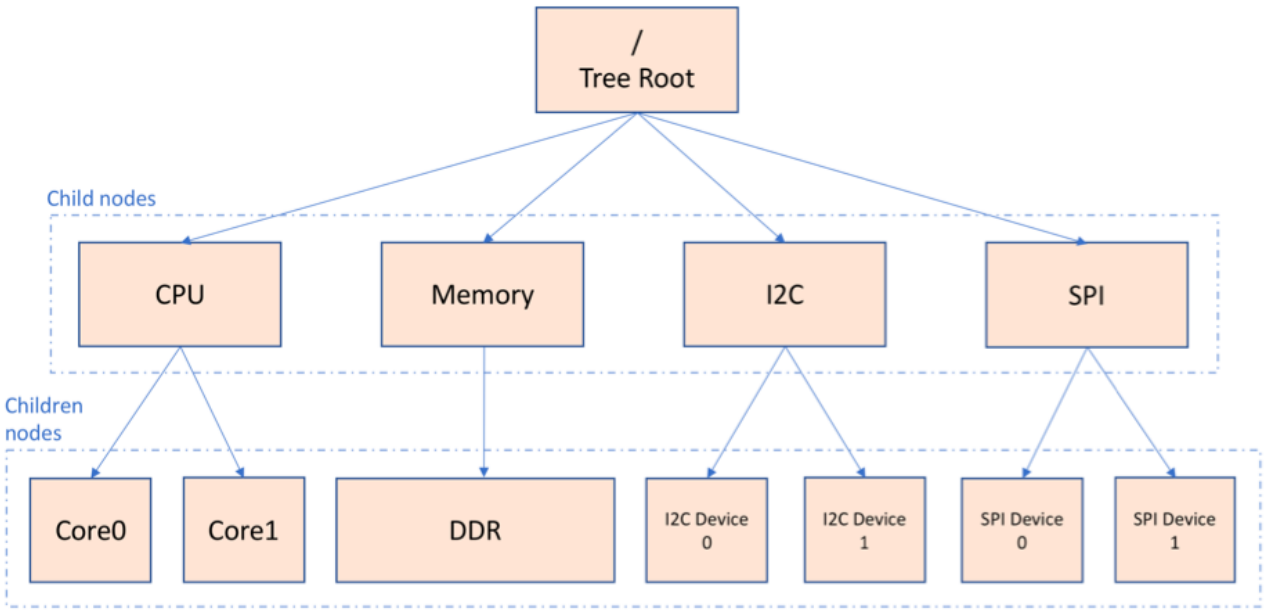
- **Standardization**: The driver model in U-Boot aims to standardize the way in which hardware drivers are written and integrated. Each driver under this model has a standard set of functions. This helps in maintaining consistency across different hardware platforms.

1.1 UCLASS_DRIVER and UCLASS_DEVICE

These are 2 classes within the U-Boot Driver Model:

- **UCLASS_DRIVER**:
    - This is an abstraction class of drivers that allows U-Boot to handle drivers of a similar kind in a uniform way, irrespective of the underlying hardware specifics.
    - Each driver class typically corresponds to a certain type of hardware functionality, like serial I/O, disk interfaces, etc.
- **UCLASS_DEVICE**:
    - Refers to the actual devices that are instantiated based on the drivers(UCLASS_DRIVER).
    - UCLASS_DEVICE allows for the actual hardware-specific operations to be performed. It's where the physical device is managed, and it interacts with the UCLASS_DRIVER to ensure the correct functioning of the hardware.
    - Each device under UCLASS_DEVICE will be initialized according to the specifications of its corresponding UCLASS_DRIVER.

2 Device Tree



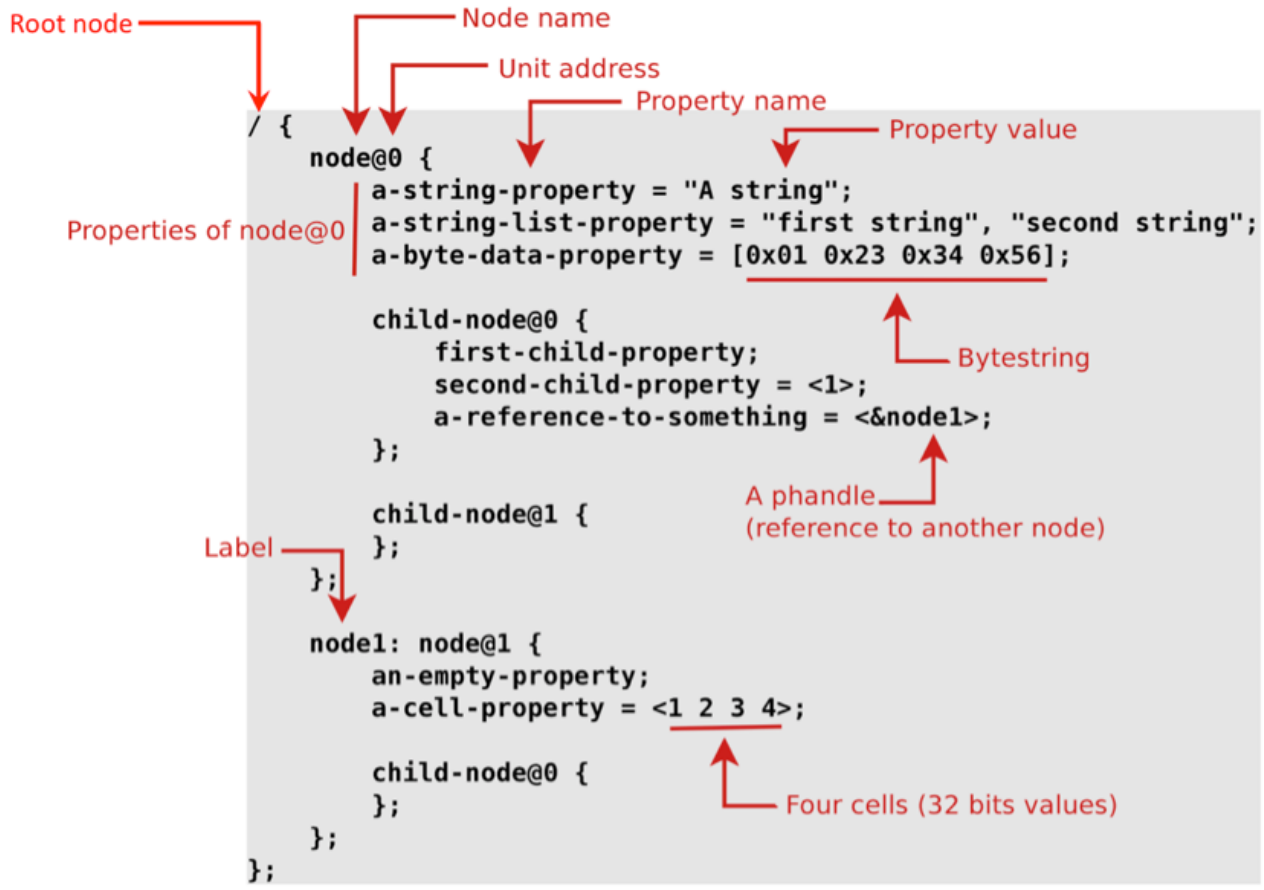
2.1 What is Device Tree?

- **Hardware Description**: The Device Tree is a tree data structure where each node represents a hardware component in a system.
- **Formats**: The Device Tree can exist in two formats:
  - **Device Tree Source (DTS)**: This is a human-readable and editable text file that describes the hardware.
  - **Device Tree Blob (DTB)**: This is a binary file compiled from the DTS, used by the operating system or bootloader.

2.2 Why Use Device Tree?

- **Hardware Abstraction**: The Device Tree provides a way to abstract the hardware details. It allows the software to adapt to different hardware without needing changes in the code. It enables the addition of new hardware configurations without requiring modifications in the kernel source code.
- **Bootloader and Kernel Simplification**: The Device Tree simplifies the bootloader and kernel code by removing the need for hard-coded hardware information. This makes the code cleaner and more maintainable.
- **Device configuration parameters**: The Device Tree includes information about device parameters, interrupt lines, memory addresses, etc., which are crucial for the correct operation of the hardware.

2.3 Device tree structure



3 FDT

3.1 What is Flattened Device Tree (FDT)?

1. **Purpose of FDT**: The Flattened Device Tree, also called `Device Tree Blob(DTB)`, is a compact binary representation of the device tree. It's designed to be easily parsed by software, particularly by the bootloader, and the kernel.
2. **Structure**: The FDT is essentially a serialized version of the device tree in a format that is simpler for software to process.

3.2 Using `fdt_get`, `fdt_set`, `fdt list`

These commands are used in a bootloader or operating system that needs to interact with the FDT

3.2.1 Intialization

fdt ddr [-c] <addr>: Set the fdt location to <addr>

```
bitminer# fdt addr -c
The address of the fdt is 0x4daa1800
bitminer# fdt addr 0x4daa1800
```

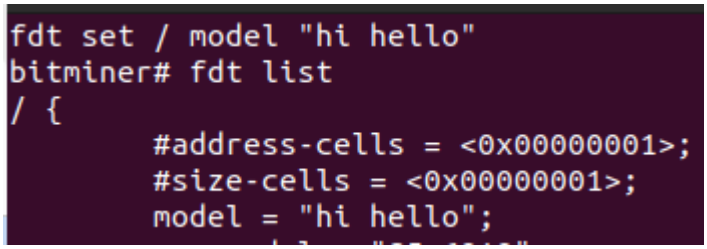
fdt list

This command is used to list the contents of the FDT.

```
bitminer# fdt list
/ {
        #address-cells = <0x00000001>;
        #size-cells = <0x00000001>;
        model = "Bitminer board based on Nexell s5p6818";
        cpu-model = "S5p6818";
        compatible = "nexell,bitminer", "nexell,s5p6818";
        chosen {
        };
        aliases {
        };
        memory {
        };
        mmc@c0069000 {
        };
        mmc@c0068000 {
        };
        mmc@c0062000 {
        };
        ethernet@c0060000 {
        };
        i2c@c00a4000 {
        };
        i2c@c00a5000 {
        };
        i2c@c00a6000 {
        };
        dp@c0102800 {
        };
        dp@c0102c00 {
        };
        usbhost@c0030000 {
        };
        dwc2otg@c0040000 {
        };
        gpio@c001a000 {
        };
        gpio@c001b000 {
        };
        gpio@c001c000 {
        };
        gpio@c001d000 {
        };
        gpio@c001e000 {
        };
        gpio@c0010800 {
        };
        pinctrl@C0010000 {
        };
        i2c_gpio@0 {
        };
        voltage-regulators {
        };
};
```

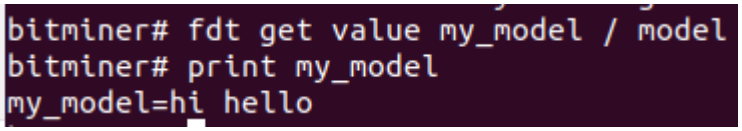
fdt_set

`fdt_set` is used to modify the FDT. This could involve changing a hardware configuration parameter before the kernel is booted.



fdt_get

The `fdt get value` command is used to retrieve a property value from a node in the FDT and store it in a U-Boot environment variable.



4 menuconfig

menuconfig is used to configure options of Linux, U-boot, etc

4.1 Install dependencies

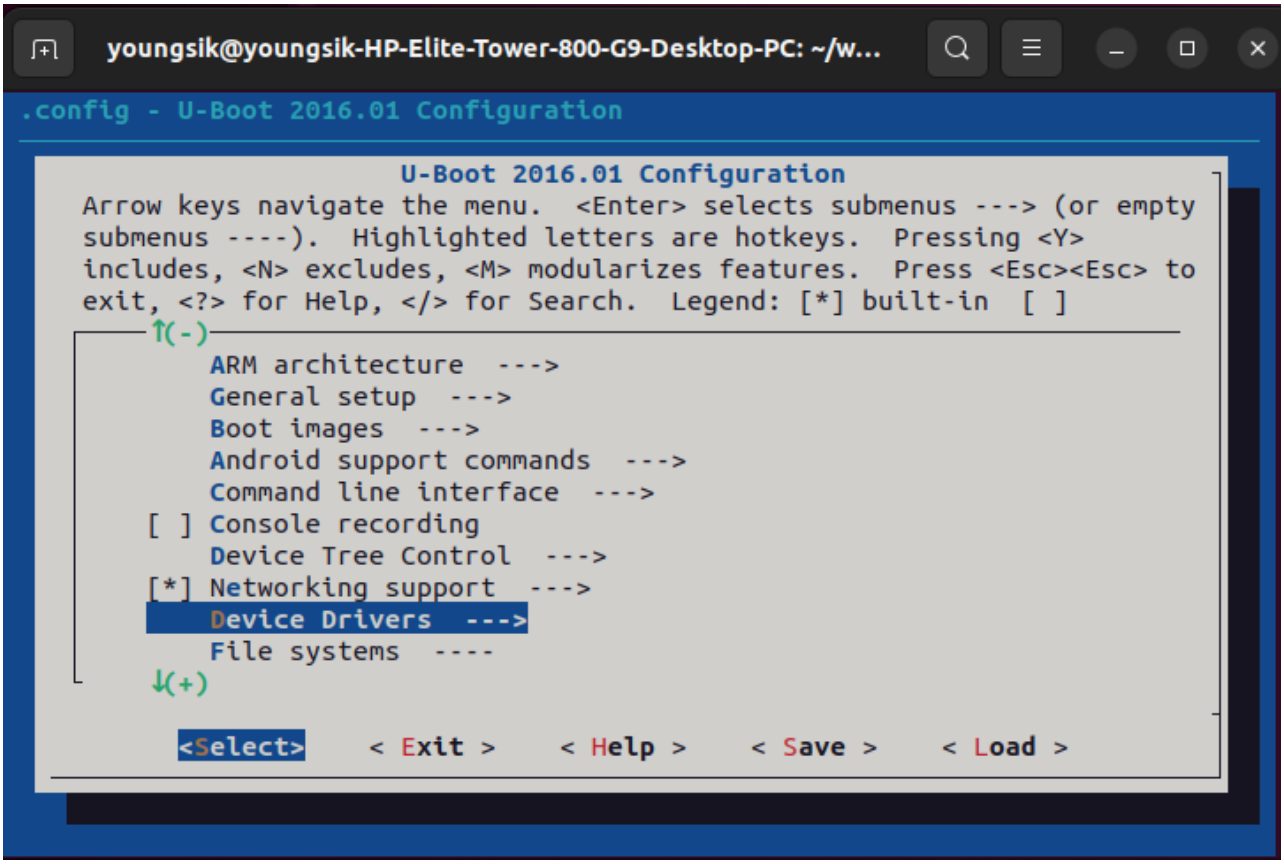
```
sudo apt-get install libncurses5-dev
```

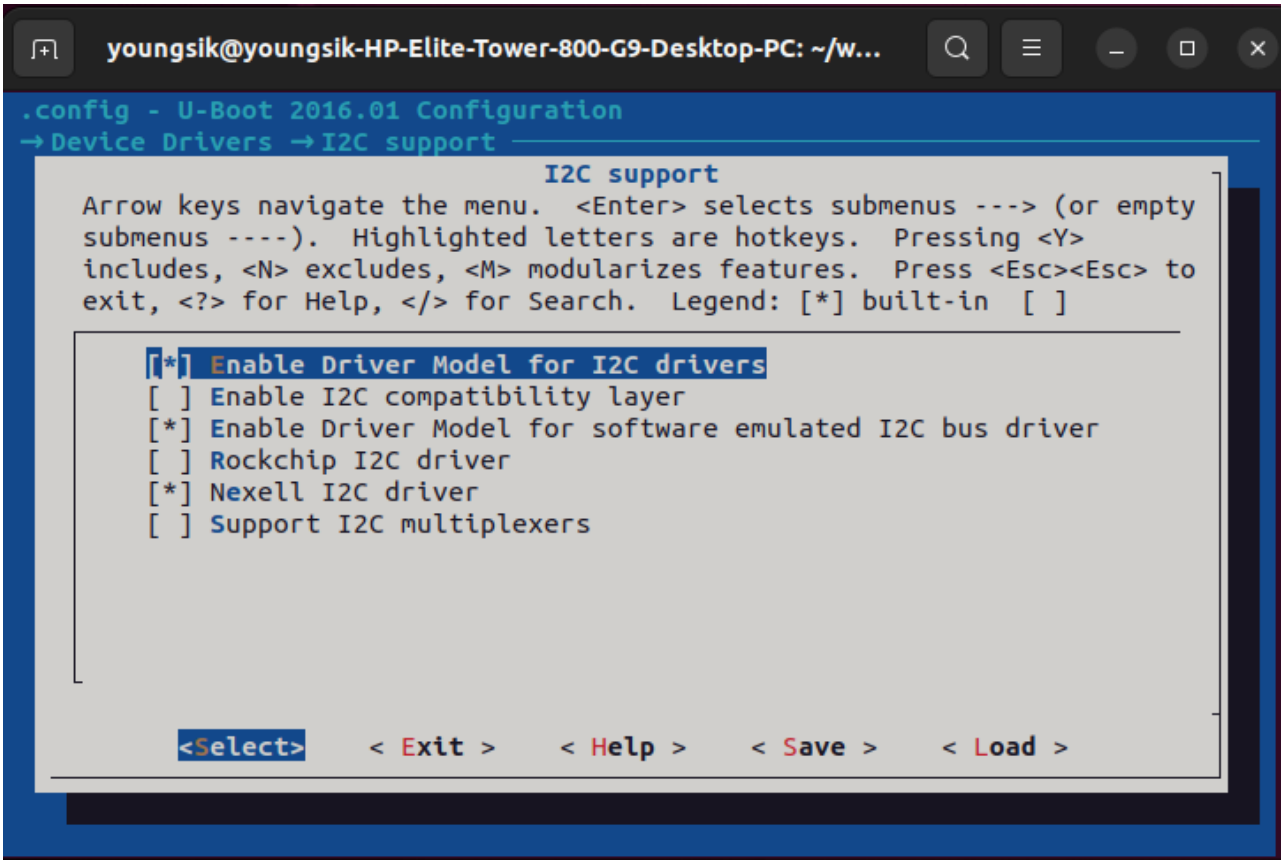
4.2 Run menuconfig

```
cd [workspace]/dunfell-bitminer/sources/boot/u-boot/u-boot-2016.01
make menuconfig
```

4.3 Exclude the i2c device driver

This is done to test how to modify u-boot through menuconfig

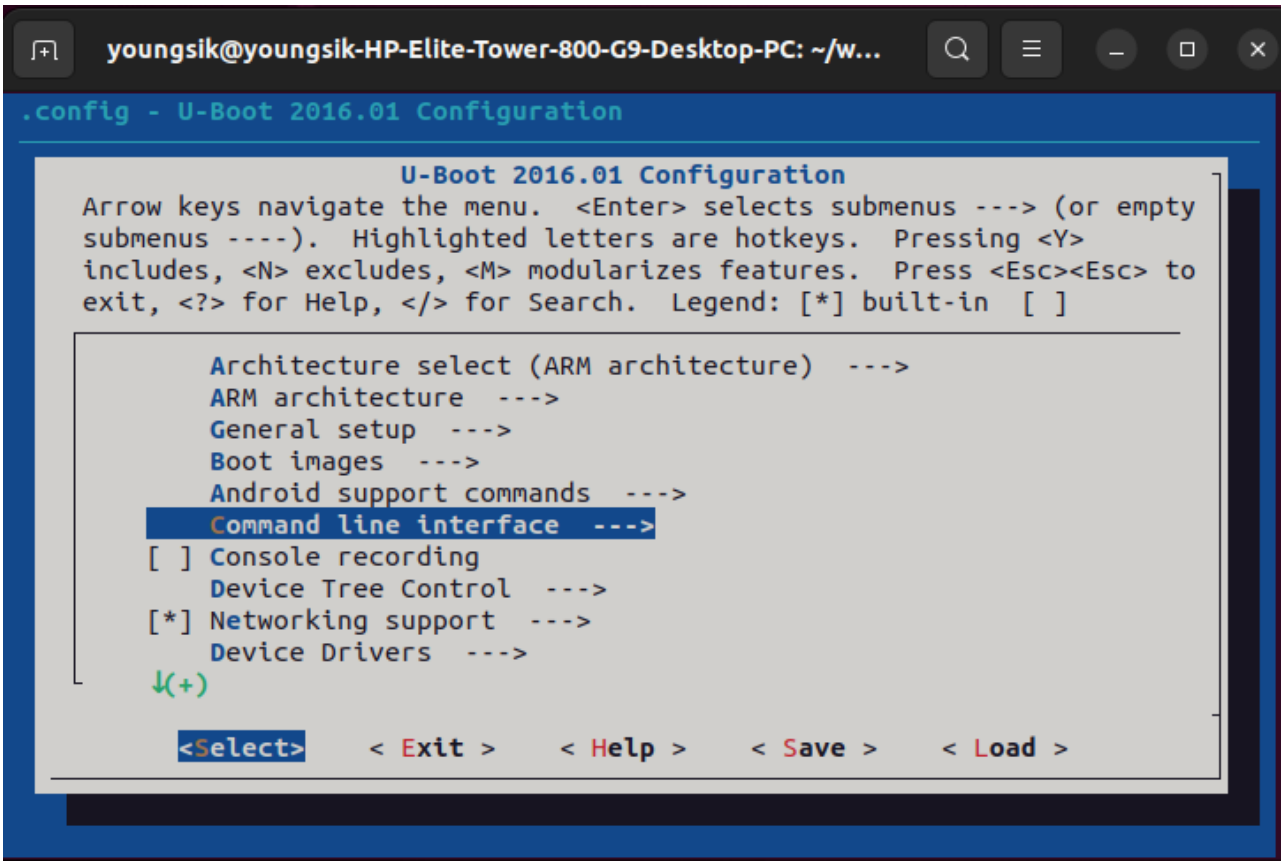

youngsik@youngsik-HP-Elite-Tower-800-G9-Desktop-PC: ~/w...
.config - U-Boot 2016.01 Configuration
U-Boot 2016.01 Configuration
Arrow keys navigate the menu. <Enter> selects submenus ---> (or empty submenus ----). Highlighted letters are hotkeys. Pressing <Y> includes, <N> excludes, <M> modularizes features. Press <Esc><Esc> to exit, <?> for Help, </> for Search. Legend: [*] built-in [ ]
↑(-)
ARM architecture --->
General setup --->
Boot images --->
Android support commands --->
Command line interface --->
[ ] Console recording
Device Tree Control --->
[*] Networking support --->
Device Drivers --->
File systems ----
↓(+)
<Select> < Exit > < Help > < Save > < Load >

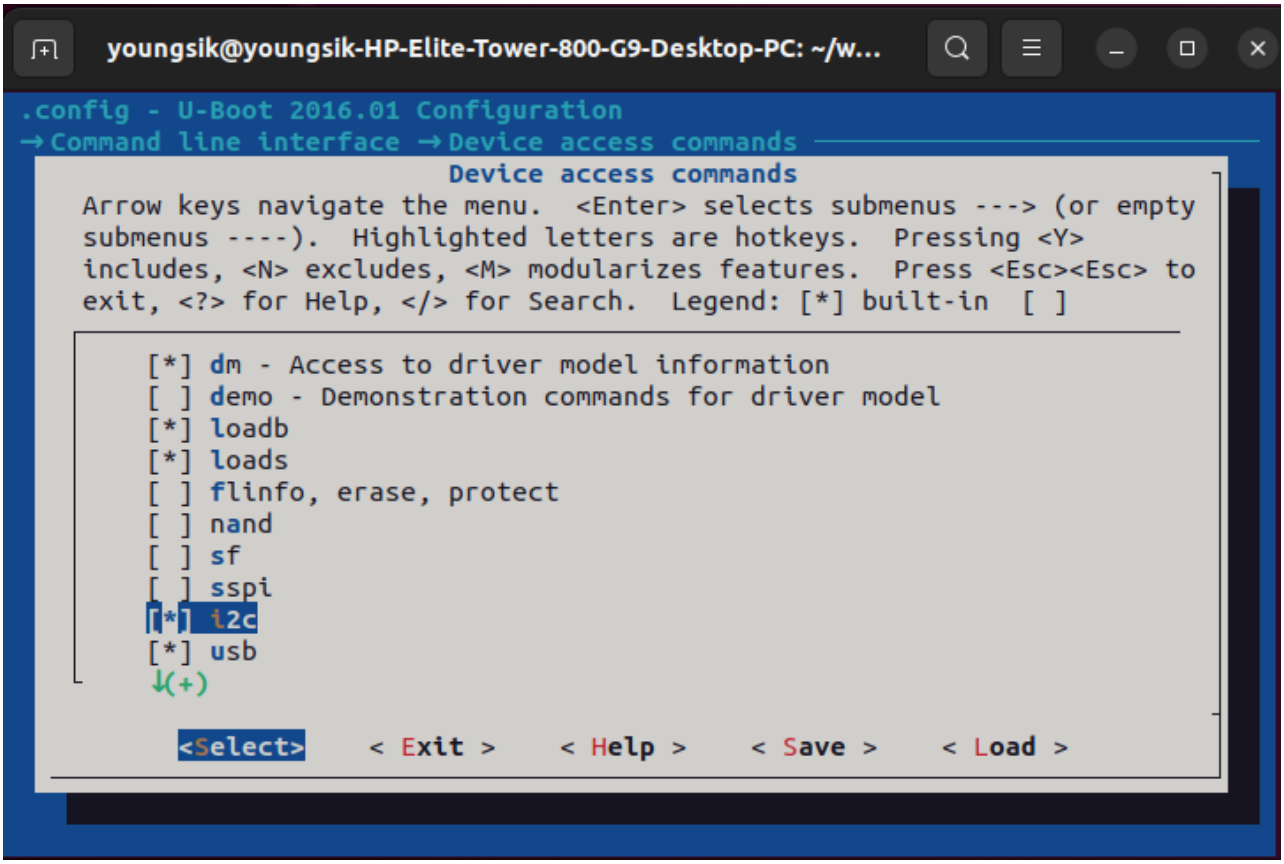


This results in an error in the build.

4.4 Exclude the i2c command

4.4.1 Uncheck the command line in menuconfig



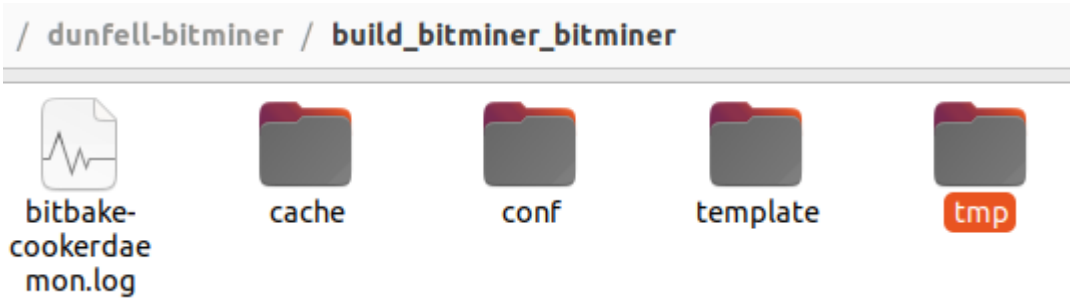


4.4.2 Make a defconfig file and copy it to the configs directory.

```
make savedefconfig
cp defconfig ./configs/s5p6818_bitminer_defconfig
```

The i2c command is excluded in the build.

4.4.3 Delete the `tmp` folder for clean build before building the modified u-boot



4.4.4 Rebuild u-boot

```
bitbake u-boot-nexell
bitbake boot-binary
bitbake optee-build
```

4.5 Check if the i2c command was successfully excluded

```
U-BootDRAM:  219 MiB
POFFHIS: NONE
PONHIS : PWRONPON
CHGS   : CHG OFF
DCDC1  : 1250mV, En, dclim:1303026384, dclimsden:0
DCDC2  : 1200mV, En, dclim:1303026384, dclimsden:0
DCDC3  : 3300mV, En, dclim:1303026384, dclimsden:0
DCDC4  : 1500mV, En, dclim:1303026384, dclimsden:0
DCDC5  : 1500mV, En, dclim:1303026384, dclimsden:0
MMC:   NEXELL DWMMC: 0
In:    serial
Out:   serial
Err:   serial
Net:
Warning: ethernet@c0060000 (eth0) using random MAC address - 86:e2:d9:ef:ec:60
eth0: ethernet@c0060000
Hit any key to stop autoboot:  0
bitminer#
bitminer#
bitminer#
bitminer#
bitminer#
bitminer# i2c
Unknown command 'i2c' - try 'help'
```