3

NXKR\_YoungSikYang

Exported on 2024-02-14 17:12:17

Table of Contents

1 Check which GPIO the LED is connected to 4

2 Device tree 5

3 Data sheet 6

3.1 Check which memory address the GPIO is connected to 6

4 GPIO control 7

4.1 Command control 7

4.1.1 Method 1 7

4.1.2 Method 2 7

4.1.3 Before 7

4.1.4 After 7

4.2 Direct control  in the u-boot source 8

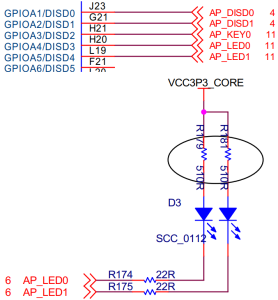
4.2.1 Trace back the found function 8

4.2.2 Add the feature 9

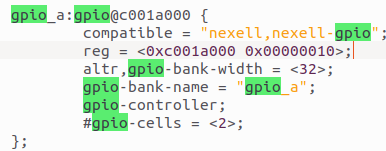
***[Target CPU 의 GPIO 구조이해]***

# Check which GPIO the LED is connected to

 AP\_LED0 >> GPIOA4



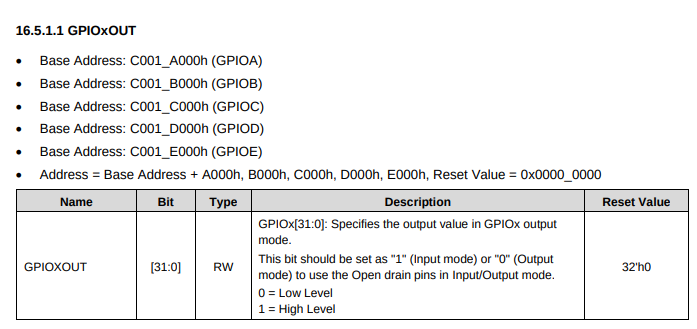
# Device tree



# Data sheet

## Check which memory address the GPIO is connected to

GPIO\_A starts from 0xc001a000



# GPIO control

## Command control

### Method 1

#### Check if defconfig has this line



#### Toggle the GPIO the LED is connected to

|  |
| --- |
| gpio toggle 4 |

### Method 2

#### Turn off the LEDs by writing to the memory location

|  |
| --- |
| md 0xc001a000 # Print  mw 0xc001a000 30 # Write 0x30 = 0b00110000  md 0xc001a000 # Print |

### Before





### After

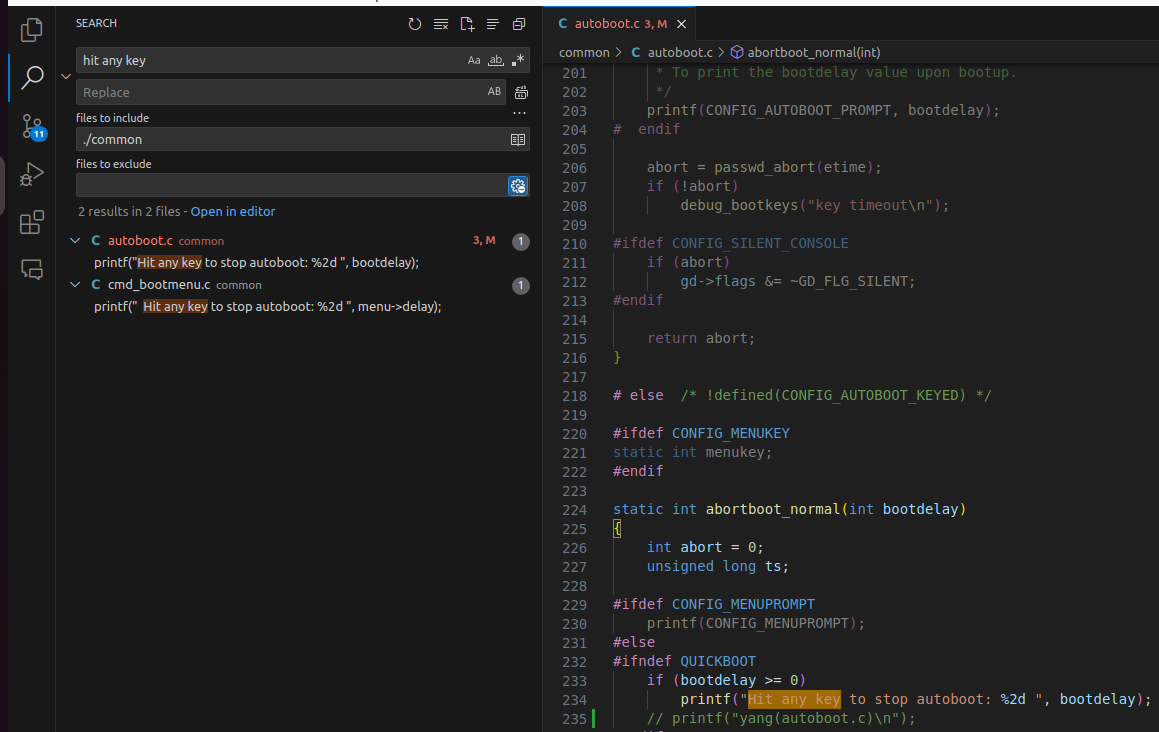




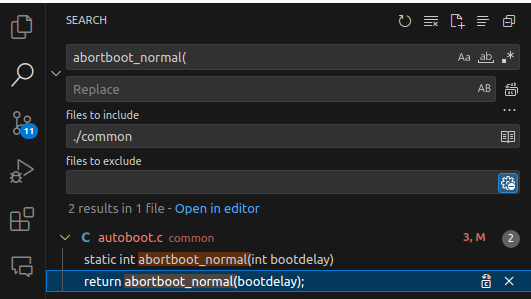
## Direct control  in the u-boot source

Find what function prints this to find the initial booting code





### Trace back the found function



### Add the feature

Run this function below autoboot\_command() in common/main.c

|  |
| --- |
| **void** blink\_LED() {  int i=0, gpio\_value=1;  **for**(i=0; i<10; i++){  printf("hello \n");  // # Use the device driver functions of the GPIO  // gpio\_request(4, "cmd\_gpio");  // gpio\_request(5, "cmd\_gpio");  // gpio\_direction\_output(4, gpio\_value);  // gpio\_direction\_output(5, gpio\_value);  // gpio\_value=!gpio\_value;  // # Bit control  \*(char\*)0xc001a000 ^= 1 << 4;  \*(char\*)0xc001a000 ^= 1 << 5;  mdelay(100);  }  } |