

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA
RECHERCHE SCIENTIFIQUE



UNIVERSITE TUNIS EL MANAR



INSTITUT SUPERIEUR D'INFORMATIQUE

Rapport Mini Projet

Développement de logiciels système

**Sujet: développement d'un Pilote pour
commander des LED a partir les GPIO du
Raspberry Pi**

Par

Mohamed Firas Mejri

&

Rami Ben Sliman

2 ème MP2L-DL

Année Universitaire 2018/2019

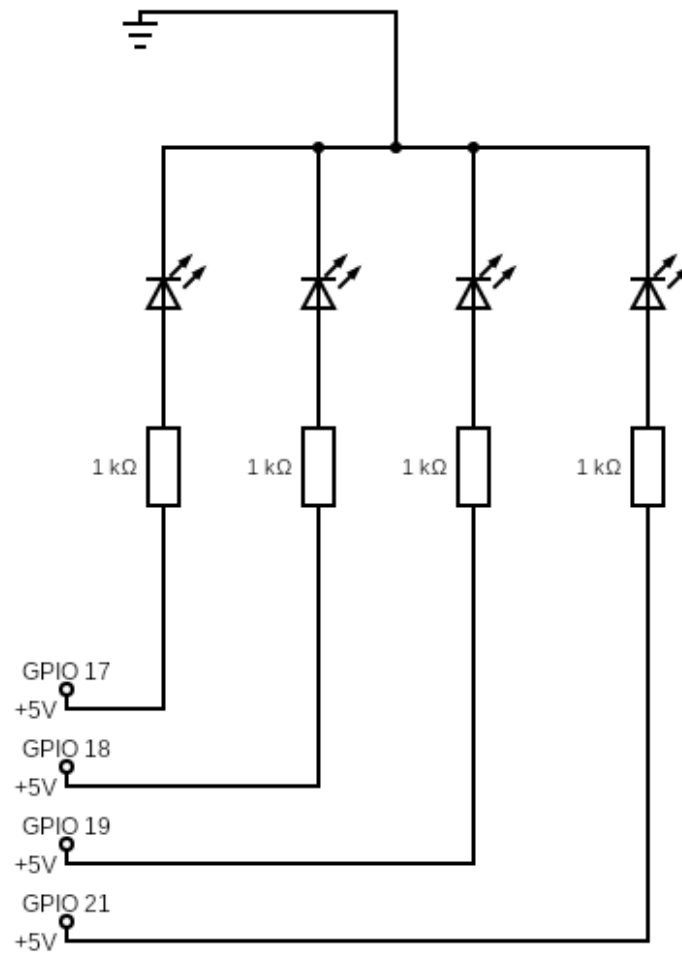
I. Introduction:

1. Sujet:

le travail présenté dans ce rapport décrit le processus de la réalisation d'un pilote pour commander une simple carte électronique qui contient 4 diodes LED à travers les GPIO de la carte raspberry Pi 3.

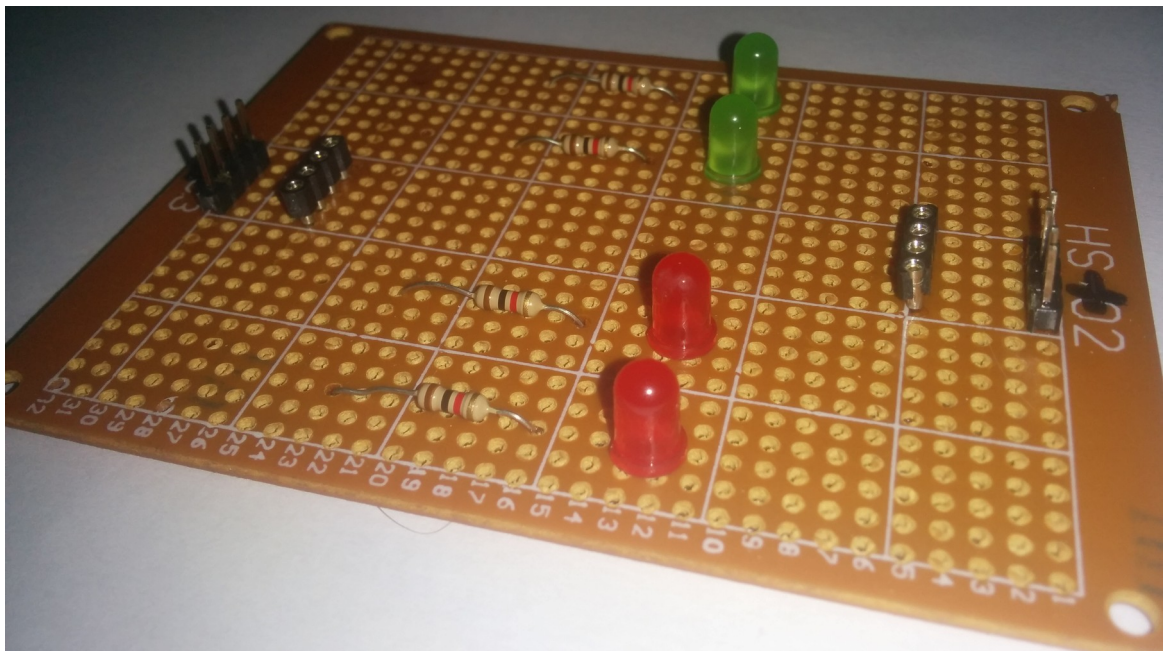
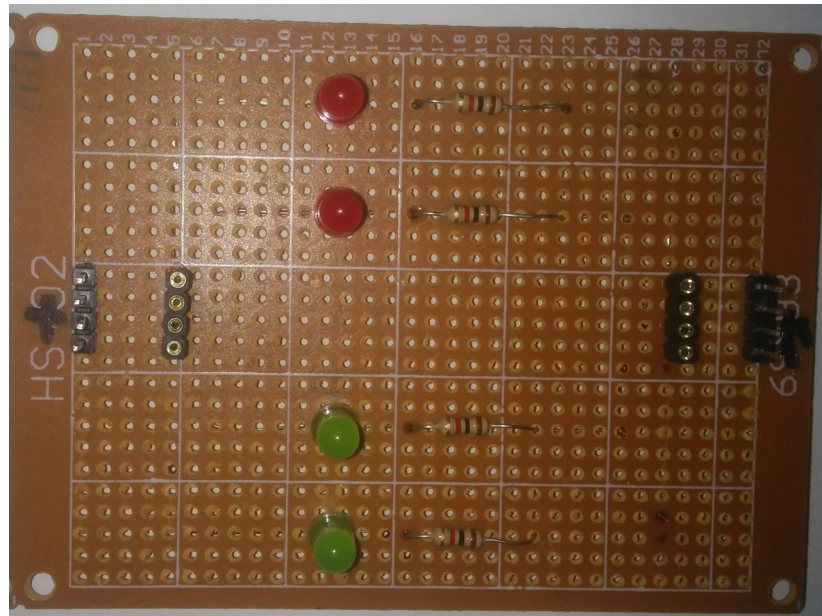
II. Partie Matérielle :

1. Schéma du circuit de la carte des LED :

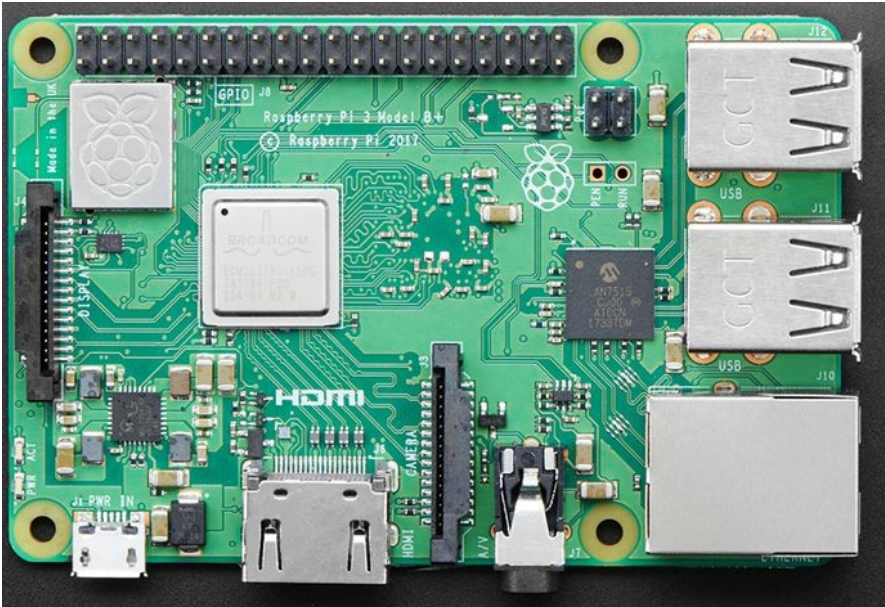


2. Matériel utiliser :

Circuit électronique : 4 diodes LED, 4 Resistors 1k, carte de maquette.



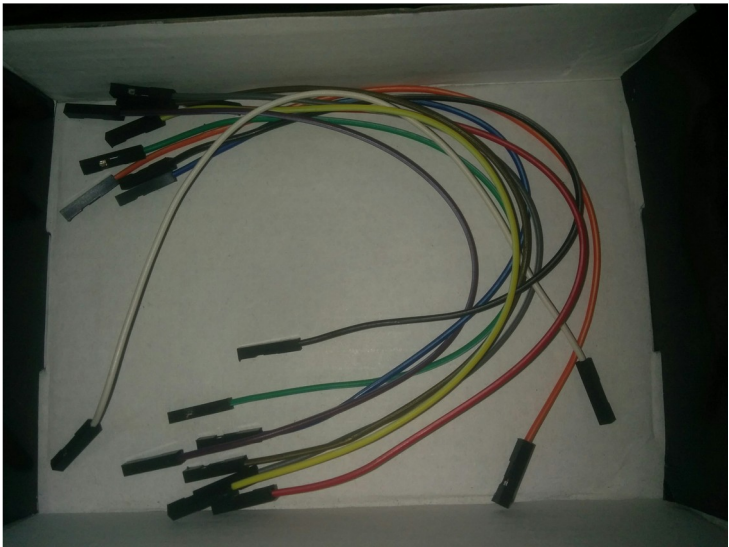
Carte Raspberry Pi 3:



| | | | | | | | | | | | | | | | | | | | | | |
|-----------------------|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|--|
| 5v Power | 2 | 5 | 6 | 9 | 10 | 22 | 14 | 16 | 18 | 20 | 22 | 24 | 26 | 29 | 30 | 32 | 34 | 36 | 38 | 40 | |
| 5v Power | 4 | | | 3 | | | | | | | | | | | | | | | | | |
| Ground | | 3 | 5 | 7 | 9 | 11 | 13 | 15 | 17 | 19 | 21 | 23 | 25 | 27 | 28 | 31 | 33 | 35 | 37 | 39 | |
| GPI014 - UART TxD | | | | | | | | | | | | | | | | | | | | | |
| GPI015 - UART RxD | | | | | | | | | | | | | | | | | | | | | |
| GPI018 - PCM_CLK/PWM0 | | | | | | | | | | | | | | | | | | | | | |
| Ground | | | | | | | | | | | | | | | | | | | | | |
| GPI023 | | | | | | | | | | | | | | | | | | | | | |
| GPI024 | | | | | | | | | | | | | | | | | | | | | |
| Ground | | | | | | | | | | | | | | | | | | | | | |
| GPI025 | | | | | | | | | | | | | | | | | | | | | |
| GPI08 - SPI_CEO_N | | | | | | | | | | | | | | | | | | | | | |
| GPI07 - SPI_CEO_N | | | | | | | | | | | | | | | | | | | | | |
| ID_SC - I2C IO EEPROM | | | | | | | | | | | | | | | | | | | | | |
| Ground | | | | | | | | | | | | | | | | | | | | | |
| GPI012 | | | | | | | | | | | | | | | | | | | | | |
| Ground | | | | | | | | | | | | | | | | | | | | | |
| GPI016 | | | | | | | | | | | | | | | | | | | | | |
| GPI020 | | | | | | | | | | | | | | | | | | | | | |
| GPI021 | | | | | | | | | | | | | | | | | | | | | |
| 3.3v Power | 1 | 3 | 5 | 7 | 9 | 11 | 13 | 15 | 17 | 19 | 21 | 23 | 25 | 27 | 28 | 31 | 33 | 35 | 37 | 39 | |
| GPI02 - SDA1 - I2C | | | | | | | | | | | | | | | | | | | | | |
| GPI03 - SCL1 - I2C | | | | | | | | | | | | | | | | | | | | | |
| GPI04 | | | | | | | | | | | | | | | | | | | | | |
| Ground | | | | | | | | | | | | | | | | | | | | | |
| GPI017 | | | | | | | | | | | | | | | | | | | | | |
| GPI027 | | | | | | | | | | | | | | | | | | | | | |
| GPI022 | | | | | | | | | | | | | | | | | | | | | |
| 3.3v Power | | | | | | | | | | | | | | | | | | | | | |
| GPI010 - SPI_MOSI | | | | | | | | | | | | | | | | | | | | | |
| GPI09 - SPI_MISO | | | | | | | | | | | | | | | | | | | | | |
| GPI011 - SPI_CLK | | | | | | | | | | | | | | | | | | | | | |
| Ground | | | | | | | | | | | | | | | | | | | | | |
| ID_SD - I2C IO EEPROM | | | | | | | | | | | | | | | | | | | | | |
| GPI05 | | | | | | | | | | | | | | | | | | | | | |
| GPI06 | | | | | | | | | | | | | | | | | | | | | |
| GPI013 | | | | | | | | | | | | | | | | | | | | | |
| GPI019 | | | | | | | | | | | | | | | | | | | | | |
| GPI026 | | | | | | | | | | | | | | | | | | | | | |
| Ground | | | | | | | | | | | | | | | | | | | | | |

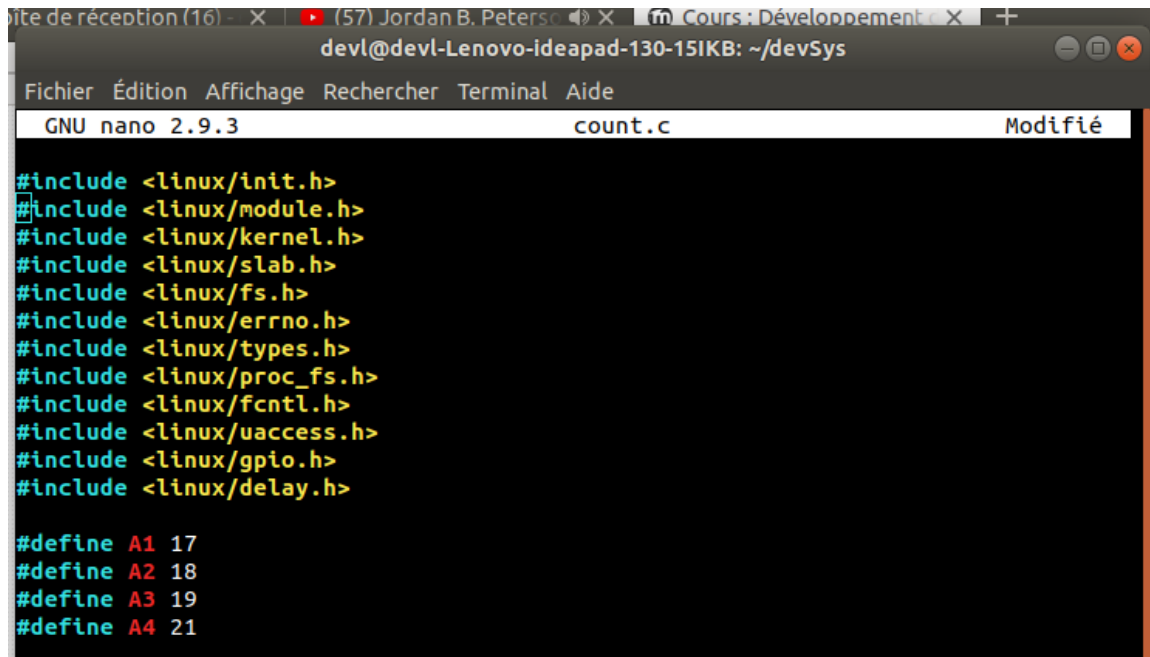
Pi Model B/B+ Pinout

Jumper cables :



III. Partie Logiciel :

1. Importation des bibliothèques nécessaires et définition des pin des GPIO :

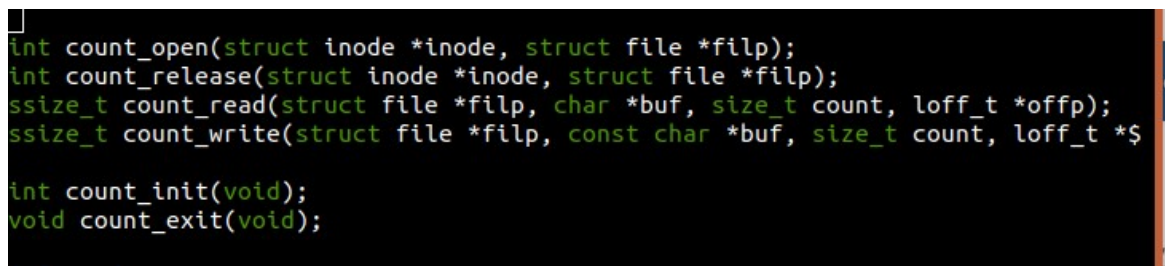


```
devl@devl-Lenovo-ideapad-130-151KB: ~/devSys
Fichier  Édition  Affichage  Rechercher  Terminal  Aide
GNU nano 2.9.3 count.c Modifié

#include <linux/init.h>
#include <linux/module.h>
#include <linux/kernel.h>
#include <linux/slab.h>
#include <linux/fs.h>
#include <linux/errno.h>
#include <linux/types.h>
#include <linux/proc_fs.h>
#include <linux/fcntl.h>
#include <linux/uaccess.h>
#include <linux/gpio.h>
#include <linux/delay.h>

#define A1 17
#define A2 18
#define A3 19
#define A4 21
```

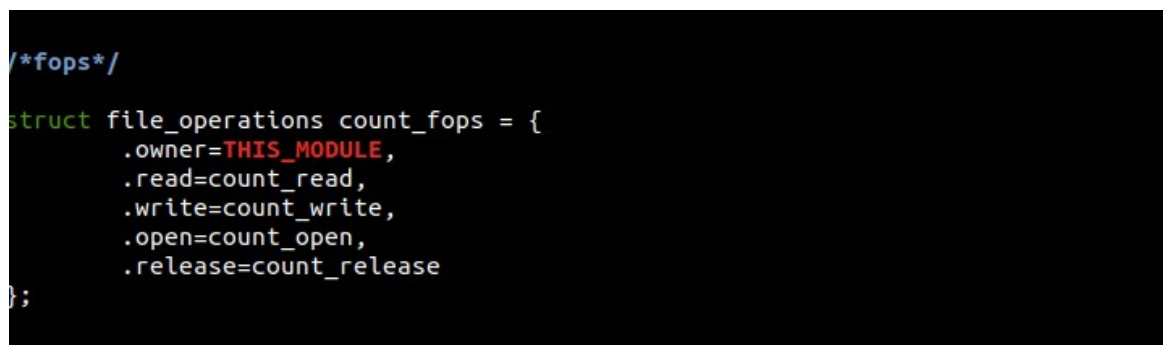
2. Déclaration des prototypes des fonctions usuelles du Module :



```
int count_open(struct inode *inode, struct file *filp);
int count_release(struct inode *inode, struct file *filp);
ssize_t count_read(struct file *filp, char *buf, size_t count, loff_t *offp);
ssize_t count_write(struct file *filp, const char *buf, size_t count, loff_t *$

int count_init(void);
void count_exit(void);
```

3. Définition de la structure file_operations :



```
/*fops*/

struct file_operations count_fops = {
    .owner=THIS_MODULE,
    .read=count_read,
    .write=count_write,
    .open=count_open,
    .release=count_release
};
```

4. information du kernel a propos des fonctions d'init et d'exit e définition des variables globales :

```
/* init | exit */  
  
module_init(count_init);  
module_exit(count_exit);  
  
/* gloabal vars */  
  
int count_major = 60;  
int count_minor = 0;  
dev_t dev;
```

5. Implémentation d'Init :

```
/* implementation init */  
  
int count_init(){  
  
    int result;  
    dev = MKDEV(count_major, count_minor);  
    result = register_chrdev_region(dev, 1, "counter");  
    if(result<0){  
        printk(KERN_WARNING,"majeur non dispo");  
        return result;  
    }  
  
    printk(KERN_INFO "compteur: initialisation des GPIO gpio...");  
  
    gpio_request(A1, "a1");  
    gpio_request(A2, "a2");  
    gpio_request(A3, "a3");  
    gpio_request(A4, "a4");  
  
    gpio_direction_output(A1, 1);  
    gpio_direction_output(A2, 1);  
    gpio_direction_output(A3, 1);  
    gpio_direction_output(A4, 1);  
  
    return 0;  
}
```

^G Aide ^O Écrire ^W Chercher ^K Couper ^J Justifier
^X Quitter ^P Lire fich ^\ Remplacer ^U Coller ^T Orthographe

6. Implémentation d'exit :

```
void count_exit(void){  
    unregister_chrdev_region(dev, 1);  
    /* libération des GPIO */  
    gpio_free(A1);  
    gpio_free(A2);  
    gpio_free(A3);  
    gpio_free(A4);  
}
```

7. Implémentation de Open et Release :

```
int count_open(struct inode *inode, struct file *filp){  
    return 0;  
}  
  
int count_release(struct inode *inode, struct file *filp){  
    return 0;  
}
```

8. Implémentation de Read :

```
ssize_t count_read(struct file *filp, char *buf, size_t count, loff_t *offp){  
    gpio_set_value(A1, 0);  
    gpio_set_value(A2, 0);  
    gpio_set_value(A3, 0);  
    gpio_set_value(A4, 0);  
    msleep_interruptible(1500);  
  
    gpio_set_value(A1, 1);  
    gpio_set_value(A2, 0);  
    gpio_set_value(A3, 0);  
    gpio_set_value(A4, 0);  
    msleep_interruptible(1500);  
  
    gpio_set_value(A1, 1);  
    gpio_set_value(A2, 1);  
    gpio_set_value(A3, 0);  
    gpio_set_value(A4, 0);  
    msleep_interruptible(1500);  
  
    gpio_set_value(A1, 1);  
    gpio_set_value(A2, 1);  
    gpio_set_value(A3, 1);  
    gpio_set_value(A4, 0);  
    msleep_interruptible(1500);  
  
    gpio_set_value(A1, 1);  
    gpio_set_value(A2, 1);  
    gpio_set_value(A3, 1);  
    gpio_set_value(A4, 1);  
  
    return 1;  
}
```

9. Implémentation de Write :

```
ssize_t count_write(struct file *filp, const char *buf, size_t count, loff_t *$

    int i = 0;

    gpio_set_value(A1, 0);
    gpio_set_value(A2, 0);
    gpio_set_value(A3, 0);
    gpio_set_value(A4, 0);
    msleep_interruptible(1500);

    while( i < 5 ){
        gpio_set_value(A1, 1);
        gpio_set_value(A2, 0);
        gpio_set_value(A3, 0);
        gpio_set_value(A4, 0);
        msleep_interruptible(1500);

        gpio_set_value(A1, 0);
        gpio_set_value(A2, 1);
        gpio_set_value(A3, 0);
        gpio_set_value(A4, 0);
        msleep_interruptible(1500);

        gpio_set_value(A1, 1);
        gpio_set_value(A2, 1);
        gpio_set_value(A3, 0);
        gpio_set_value(A4, 0);
        msleep_interruptible(1500);

        gpio_set_value(A1, 0);
        gpio_set_value(A2, 0);
        gpio_set_value(A3, 1);
        gpio_set_value(A4, 0);
        msleep_interruptible(1500);

        gpio_set_value(A1, 1);
        gpio_set_value(A2, 0);
        gpio_set_value(A3, 1);
        gpio_set_value(A4, 0);
        msleep_interruptible(1500);

        gpio_set_value(A1, 1);
        gpio_set_value(A2, 0);
        gpio_set_value(A3, 1);
        gpio_set_value(A4, 0);
        msleep_interruptible(1500);

        gpio_set_value(A1, 0);
        gpio_set_value(A2, 1);
        gpio_set_value(A3, 1);
        gpio_set_value(A4, 0);
        msleep_interruptible(1500);

        gpio_set_value(A1, 1);
        gpio_set_value(A2, 1);
        gpio_set_value(A3, 1);
        gpio_set_value(A4, 0);
        msleep_interruptible(1500);

        gpio_set_value(A1, 0);
        gpio_set_value(A2, 0);
        gpio_set_value(A3, 0);
        gpio_set_value(A4, 1);
        msleep_interruptible(1500);

        gpio_set_value(A1, 1);
        gpio_set_value(A2, 0);
        gpio_set_value(A3, 0);
        gpio_set_value(A4, 1);
        msleep_interruptible(1500);
```



```
    gpio_set_value(A1, 1);
    gpio_set_value(A2, 0);
    gpio_set_value(A3, 0);
    gpio_set_value(A4, 1);
    msleep_interruptible(1500);

    gpio_set_value(A1, 0);
    gpio_set_value(A2, 1);
    gpio_set_value(A3, 0);
    gpio_set_value(A4, 1);
    msleep_interruptible(1500);

    gpio_set_value(A1, 1);
    gpio_set_value(A2, 1);
    gpio_set_value(A3, 0);
    gpio_set_value(A4, 1);
    msleep_interruptible(1500);

    gpio_set_value(A1, 0);
    gpio_set_value(A2, 0);
    gpio_set_value(A3, 1);
    gpio_set_value(A4, 1);
    msleep_interruptible(1500);

    gpio_set_value(A1, 1);
    gpio_set_value(A2, 0);
    gpio_set_value(A3, 1);
    gpio_set_value(A4, 1);
    msleep_interruptible(1500);
```

```
    gpio_set_value(A1, 1);
    gpio_set_value(A2, 0);
    gpio_set_value(A3, 1);
    gpio_set_value(A4, 1);
    msleep_interruptible(1500);

    gpio_set_value(A1, 0);
    gpio_set_value(A2, 1);
    gpio_set_value(A3, 1);
    gpio_set_value(A4, 1);
    msleep_interruptible(1500);

    gpio_set_value(A1, 1);
    gpio_set_value(A2, 1);
    gpio_set_value(A3, 1);
    gpio_set_value(A4, 1);

    i++;
}
return 1;
}
```