设计方案

1 功能：信号灯自动定时变换

1.1 功能简介

a、g 段信号灯为一组，f、b 段信号灯为一组，此后两组信号灯自动交替循环亮起和熄灭，循环时间为 30 秒。

1.2 关键代码

```
int main() {
    int fd = open("/dev/s3c2440_led0",O_RDWR);
    if (fd < 0) {
        printf("####Led device open fail####\n");
        return (-1);
    }

    __asm__(
        "MOV R5,#0X011;"
        "STRB R5,[R4];"
    );

    // 南北红灯：0xffbe    10111110
    // 东西红灯：0xffdd    11011101
    unsigned int STATE[2] = {0xffbe, 0xffdd};

    // 为了演示方便，将自动切换时间设置为 3 秒
    while(1){
        for (int i = 0; i < 2; i++)
        {
            ioctl(fd, 0x12, STATE[i]);
            sleep(3);
        }
    }

    close(fd);
    return 0;
}
```

1.3 设计描述

STATE[0]=0xffbe 表示的是南北红绿灯；

STATE[1]=0xffdd 表示的是东西红绿灯；

通过两层循环：第一层循环保证每时每刻都有一方的灯会亮，第二层循环控制南北红绿灯与东西红绿灯交替点亮；

ioctl()函数用来点亮对应的红绿灯；

为了便于演示，将默认的自动切换时间设置为 3 秒。

2 功能：自动切换的开始和停止

2.1 功能简介

通过用户界面上的按钮控制自动切换的开始和停止。

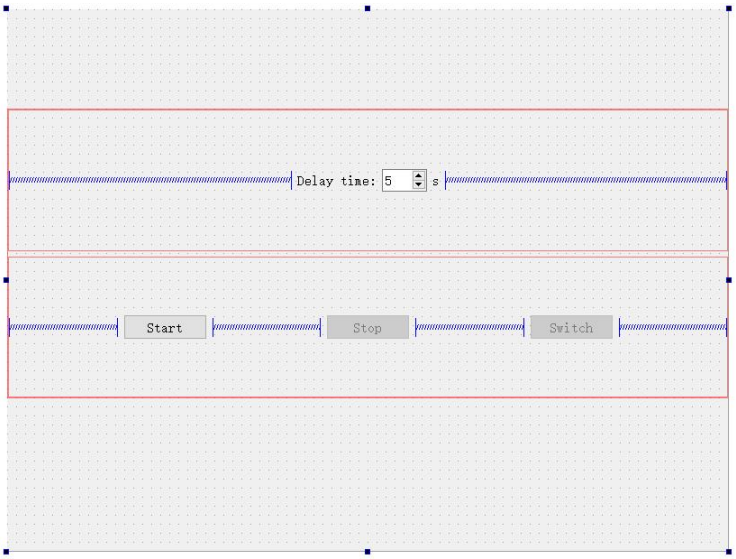
2.2 关键代码

```
class Thread : public QThread
{
    Q_OBJECT

private:
    // North & South: 0xffbe    10111110
    // East  & West:  0xffdd    11011101
    const unsigned int STATES[2];

    int fd;
    int state;
    int delay;

    atomic_bool stopFlag;
    atomic_bool pauseFlag;
    QMutex mutex;
    QWaitCondition condition;

    void openDevice();

protected:
    virtual void run();
    virtual void process();

public:
    Thread();
    Thread(int delay);
    ~Thread();

    void stop();
    void pause();
    void resume();

    void switchState();
    void sleepByDelay();
    void reset();
```

```
};

void Thread::switchState()
{
    state = (state + 1) % 2;
    ioctl(fd, 0x12, STATES[state]);
}

void Thread::sleepByDelay()
{
    QThread::sleep(delay);
}

void Thread::process()
{
    switchState();
    sleepByDelay();
}

void Thread::run()
{
    if (fd < 0)
    {
        printf("Error: Led device didn't open.\n");
        return;
    }

    __asm__(
        "MOV R5,#0X011;"
        "STRB R5,[R4];"
    );

    ioctl(fd, 0x12, STATES[state]);
    sleepByDelay();

    while(!stopFlag)
    {
        process();
        if (pauseFlag)
        {
            mutex.lock();
            condition.wait(&mutex);
            mutex.unlock();
        }
```

```
    }

    pauseFlag = false;
    stopFlag = false;
}

void Thread::stop()
{
    if (QThread::isRunning())
    {
        stopFlag = true;
        condition.wakeAll();
        QThread::quit();
        QThread::wait();
    }
}

void MainWindow::on_startButton_clicked()
{
    int delay = ui->spinBox->value();
    thread = new Thread(delay);
    thread->start();

    ui->spinBox->setDisabled(true);
    ui->startButton->setDisabled(true);
    ui->stopButton->setDisabled(false);
    ui->switchButton->setDisabled(false);
}

void MainWindow::on_stopButton_clicked()
{
    thread->stop();
    thread->reset();
    delete thread;
    thread = NULL;

    ui->spinBox->setDisabled(false);
    ui->startButton->setDisabled(false);
    ui->stopButton->setDisabled(true);
    ui->switchButton->setDisabled(true);
}
```

2.3 用户界面



2.4 设计描述

信号灯的自动切换需要持续运行，为了避免用户界面卡死，建立子线程执行信号灯的相关功能。Thread 是线程类，子线程执行的代码写在 run()函数中。这里为了将主要功能和线程控制分离开来，另外定义了 process()函数，用于执行信号灯的自动切换。

当用户点击界面上的“Start”按钮后，将读取 SpinBox 控件的值作为自动切换的时间间隔，创建 Thread 对象并启动子线程，此时会打开信号灯并开始自动切换。SpinBox 控件默认值为 5，最小值为 1，通过 SpinBox 设置时间间隔可以避免非法输入。

在开启信号灯之前，界面上的“Start”按钮和 SpinBox 控件可用，而“Stop”和“Switch”按钮不可用；点击“Start”按钮开启信号灯后，“Start”按钮和 SpinBox 控件将被禁用，而“Stop”和“Switch”按钮可用。这样可以避免用户误操作。

点击“Stop”按钮，将停止自动切换并关闭信号灯，用户界面上的“Start”按钮和 SpinBox 控件恢复可用，“Stop”和“Switch”按钮禁用。

3 功能：手动切换

3.1 功能简介

通过用户界面手动切换信号灯。

3.2 关键代码

```
void Thread::pause()
{
    if (QThread::isRunning())
    {
        pauseFlag = true;
    }
}
```

```
void Thread::resume()
{
    if (QThread::isRunning())
    {
        pauseFlag = false;
        condition.wakeAll();
    }
}

void MainWindow::on_switchButton_clicked()
{
    thread->pause();
    thread->switchState();
    thread->sleepByDelay();
    thread->resume();
}
```

3.3 设计描述

在自动切换过程中，点击“Switch”按钮，将子线程暂停，执行一次切换并等待指定的时间，在恢复子线程的执行。

三、GitHub 上传地址

https://github.com/zzx-JLU/Traffic-Light