



厦门大学

《嵌入式系统》

课程实验报告

姓名：苏一涵

学院：信息学院

系：软件工程

专业：软件工程

学号：36720232204041

2025年12月

第8次实验 Qt 与 Android 设计实验

1. 实验设备

- (1) PC 微机
- (2) 嵌入式系统综合实验箱 (FS3399M4)

2. 实验内容

- (1) 必做实验：请从 10 个 Qt 设计实验 (1-1、1-2、1-3、1-4、2-1、2-2、2-3、2-4、3-1、3-2) 中选择 3 个，被控制设备不重复。具体即从 1-1, 1-2, 1-3, 1-4 中选 1 个，2-1, 2-2, 2-3, 2-4 中选 1 个，3-1, 3-2 中选 1 个。
- (2) 必做实验：请从 2 个 Android 设计实验 (4-1、4-2) 任选 1 个。
- (3) 挑战实验：请从 2 个 Android 设计实验 (5-1、5-2) 任选 1 个。
完成挑战实验的，本次实验成绩加 10 分。

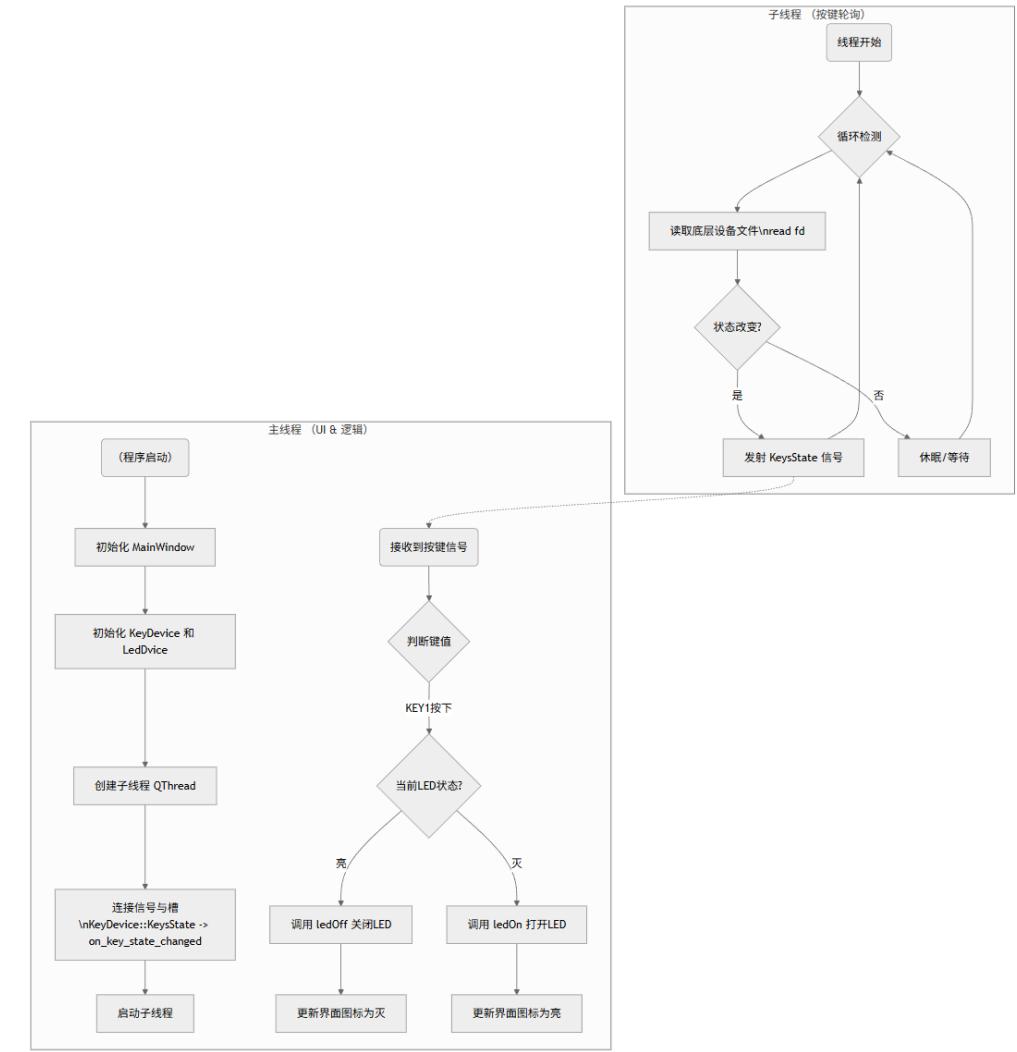
附：设计实验清单

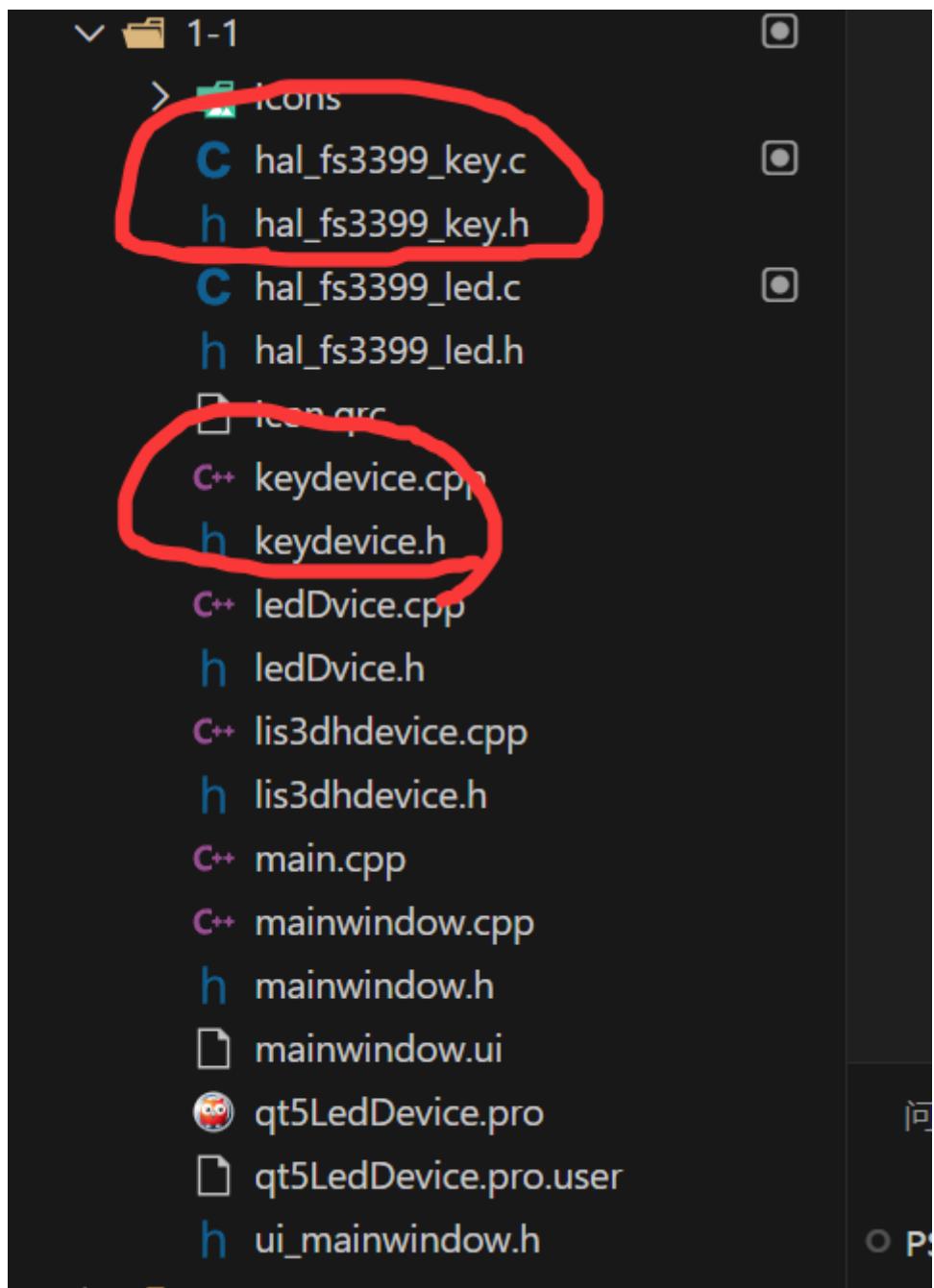
- 第一部分：Qt 设计实验
 - 设计实验 1-1：按键控制 LED 灯

设计思路

在原有的 qt5LedDevice 工程基础上，引入 qt5KeyDevice 工程中的按键驱动文件

流程图如下：





在 mainwindow.cpp 中加入按键的初始化

```
设计实验 > 1-1 > C++ mainwindow.cpp > on_key_state_changed(int)

4 MainWindow::MainWindow(QWidget *parent)
5     : QMainWindow(parent)
6 {
7     QGraphicsScene *scene = new QGraphicsScene(this);
8     view = new QGraphicsView(scene, this);
9     customWidget = new CustomWidget(); // 创建自定义Widget
10
11    // 在构造函数中添加按键初始化代码
12    key = new KeyDevice();
13    keyThread = new QThread();
14    key->moveToThread(keyThread); -----
15
16    connect(keyThread, &QThread::started, key, &KeyDevice::getKeys);
17    connect(key, &KeyDevice::KeysState, this, &MainWindow::on_key_state_changed);
18    keyThread->start();
19
20    QScreen *screen = QApplication::primaryScreen(); // 获取当前屏幕
21    QRect screenGeometry = screen->geometry(); // 获取屏幕尺寸
22    int screenWidth = screenGeometry.width(); // 屏幕宽度
23    int screenHeight = screenGeometry.height(); // 屏幕高度
24}
```

连接

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
```

在 mainwindow.h 中添加按键槽函数声明

```
qt5LedDevice.pro | mainwindow.h ...|mainwindow.cpp ...| qt5
设计实验 > 1-1 > h mainwindow.h > MainWindow
1 #ifndef MAINWINDOW_H
25 class MainWindow : public QMainWindow {
35 public:
36     MainWindow(QWidget *parent = nullptr);
37
38     ~MainWindow();
39
40     // 在 MainWindow 类中添加私有成员
41 private:
42     KeyDevice *key;
43     QThread *keyThread;
44
45     // 添加槽函数声明
46 private slots:
47     void on_key_state_changed(int keyStates);
48
49 private:
50     Ui::MainWindow *ui;
51     CustomWidget *customWidget; // 旋转界面对象
52     Lis3dhDevice *lis3dh; // LIS3DH设备类实例
53     QThread *thread3dh; // 线程处理LIS3DH设备
```

添加对应的槽函数实现

```
// 添加槽函数实现
void MainWindow::on_key_state_changed(int keyStates)
{
    // 假设 KEY1_ON 定义为 1 (具体值请参考 keydevice.h)
    if (keyStates == 1) { // KEY1 按下
        on_led1_clicked(); // 调用现有的LED切换函数
    }
}
```

实验结果如下





- 设计实验 1-2：按键控制蜂鸣器
- 设计实验 1-3：按键控制直流电机

- 设计实验 1-4：按键控制步进电机
- 设计实验 2-1：小键盘控制 LED 灯和蜂鸣器

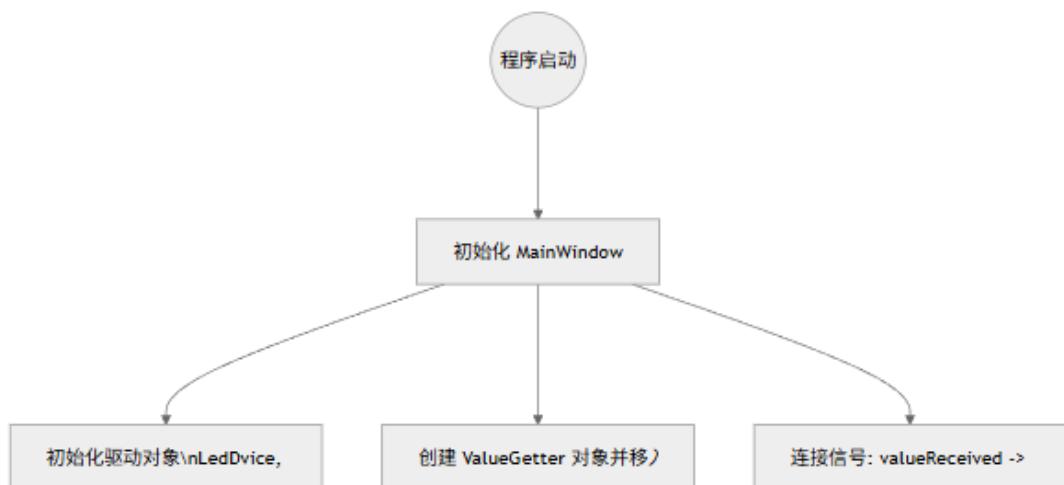
思路：在 `qt5DisplayAndMatrix` 项目中，利用数码管/矩阵键盘驱动读取键值，并引入 LED 和蜂鸣器驱动进行控制。

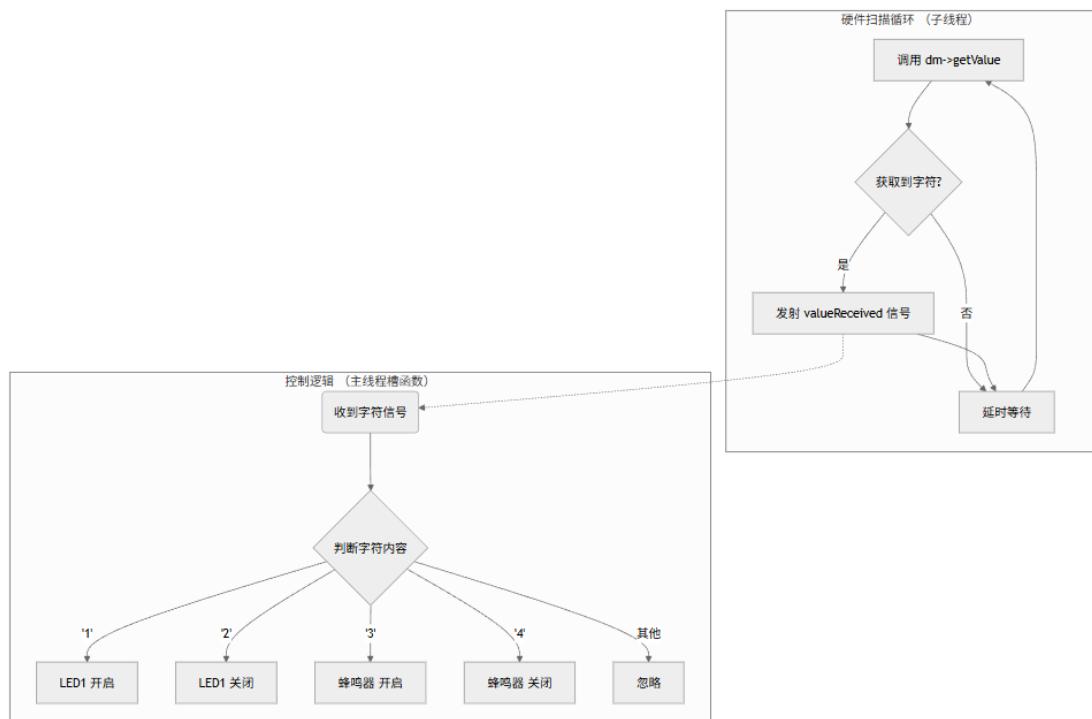
从 `qt5LedDevice` 复制：`ledDvice.cpp`, `ledDvice.h`, `hal_fs3399_led.c`, `hal_fs3399_led.h`

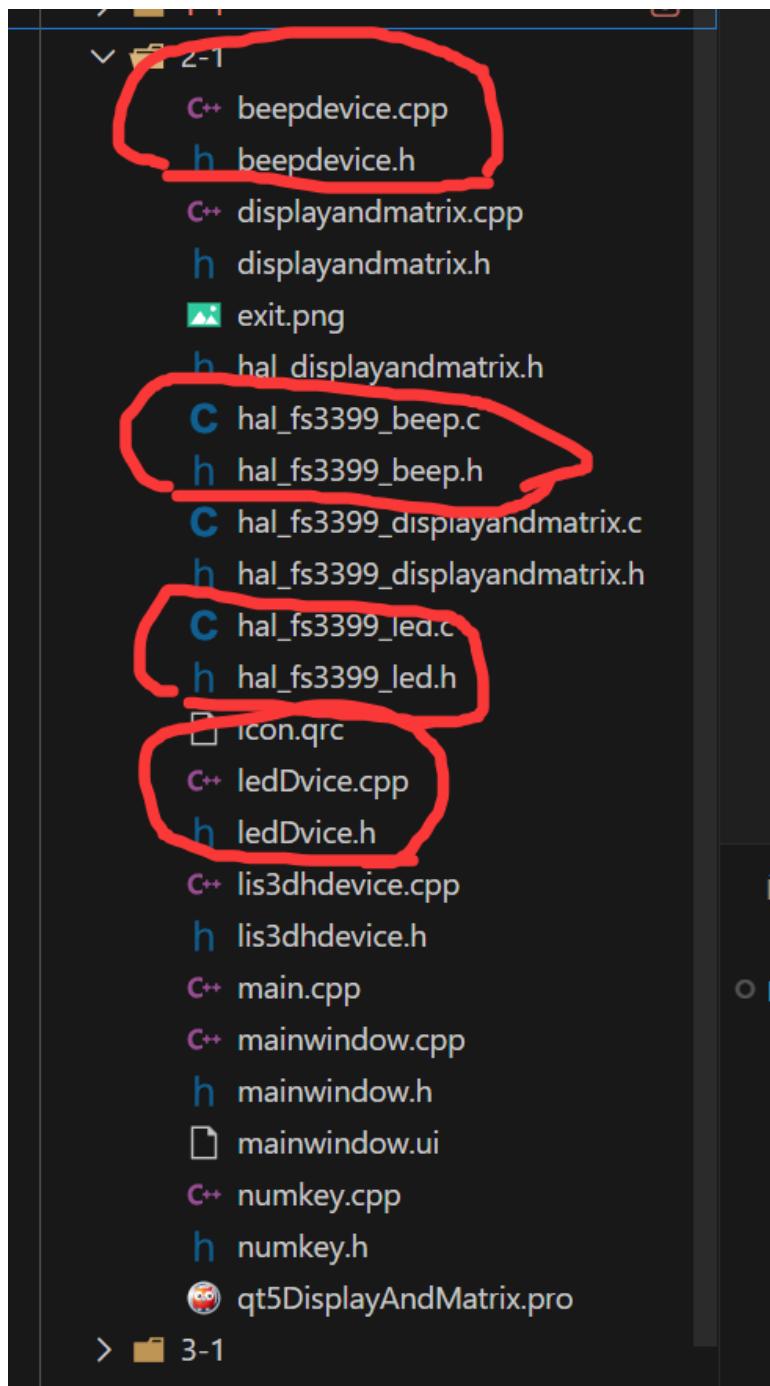
从 `qt5BeepDevice` 复制：`beepdevice.cpp`, `beepdevice.h`,
`hal_fs3399_beep.c`, `hal_fs3399_beep.h`

放入 `qt5DisplayAndMatrix` 文件夹。

流程图如下







在 Mainwindow 类中添加蜂鸣器和 led 灯成员

```
设计实验 > 2-1 > h mainwindow.h > ...
1 #ifndef MAINWINDOW_H
4 #include < QMainWindow>
22 #include "beepdevice.h"
23
24
25 class ValueGetter;
26 class CustomWidget;
27
28 class MainWindow : public QMainWindow
29 {
30     Q_OBJECT
31     private:
32         LedDvice *led;
33         BeepDevice *beep;
34
35     public:
36         MainWindow(QWidget *parent = nullptr);
37         ~MainWindow();
38
39     public slots:
40         void on_key_received(char value);
41 }
```

问题 4 输出 调试控制台 终端 端口 powershell

在 mainwindow.cpp 中初始化

```
设计实验 > 2-1 > C++ mainwindow.cpp > MainWindow(QWidget *)
1 #include "mainwindow.h"
2
3 MainWindow::MainWindow(QWidget *parent)
4     : QMainWindow(parent)
5 {
6     // 构造函数中初始化
7     led = new LedDvice();
8     beep = new BeepDevice();
9
10    // 创建场景
11    QGraphicsScene *scene = new QGraphicsScene(this);
12    view = new QGraphicsView(scene, this);
13
14 }
```

在 mainwindow.h 中添加 on_key_received 槽函数声明

```
1 #ifndef MAINWINDOW_H
28 class MainWindow : public QMainWindow
32     BEEPDEVICE beep,
34
35 public:
36     MainWindow(QWidget *parent = nullptr);
37     ~MainWindow();
38
39 public slots:
40     void on_key_received(char value);
41     void buttonexit();
42
43     void numberButtonClicked(int num);
44
45     void letterButtonClicked();
46
47     void stopGettingValues();
48
49 private:
50     QLCDNumber *Lcd;
51     DisplayandMatrix *dm;
52     QPushButton *m_exitButton1;
53     QString accumulatedStr;
```

实现对应的槽函数

```
151
152
153 void MainWindow::on_key_received(char value)
154 {
155     // 根据接收到的按键值控制 LED 和 蜂鸣器
156     // 假设 '1' 开灯，'2' 关灯，'3' 开蜂鸣器，'4' 关蜂鸣器
157     {
158         if (value == '1') {
159             led->ledOn(LED1);
160         } else if (value == '2') {
161             led->ledOff(LED1);
162         } else if (value == '3') {
163             beep->beepOn();
164         } else if (value == '4') {
165             beep->beepOff();
166         }
167     }
168     this->close();
169     QApplication::quit();
170 }
```

实验结果如下

- 设计实验 2-2：小键盘控制 LED 灯和蜂鸣器
 - 设计实验 2-3：小键盘控制直流电机
 - 设计实验 2-4：小键盘控制步进电机
 - 设计实验 3-1：电子钟

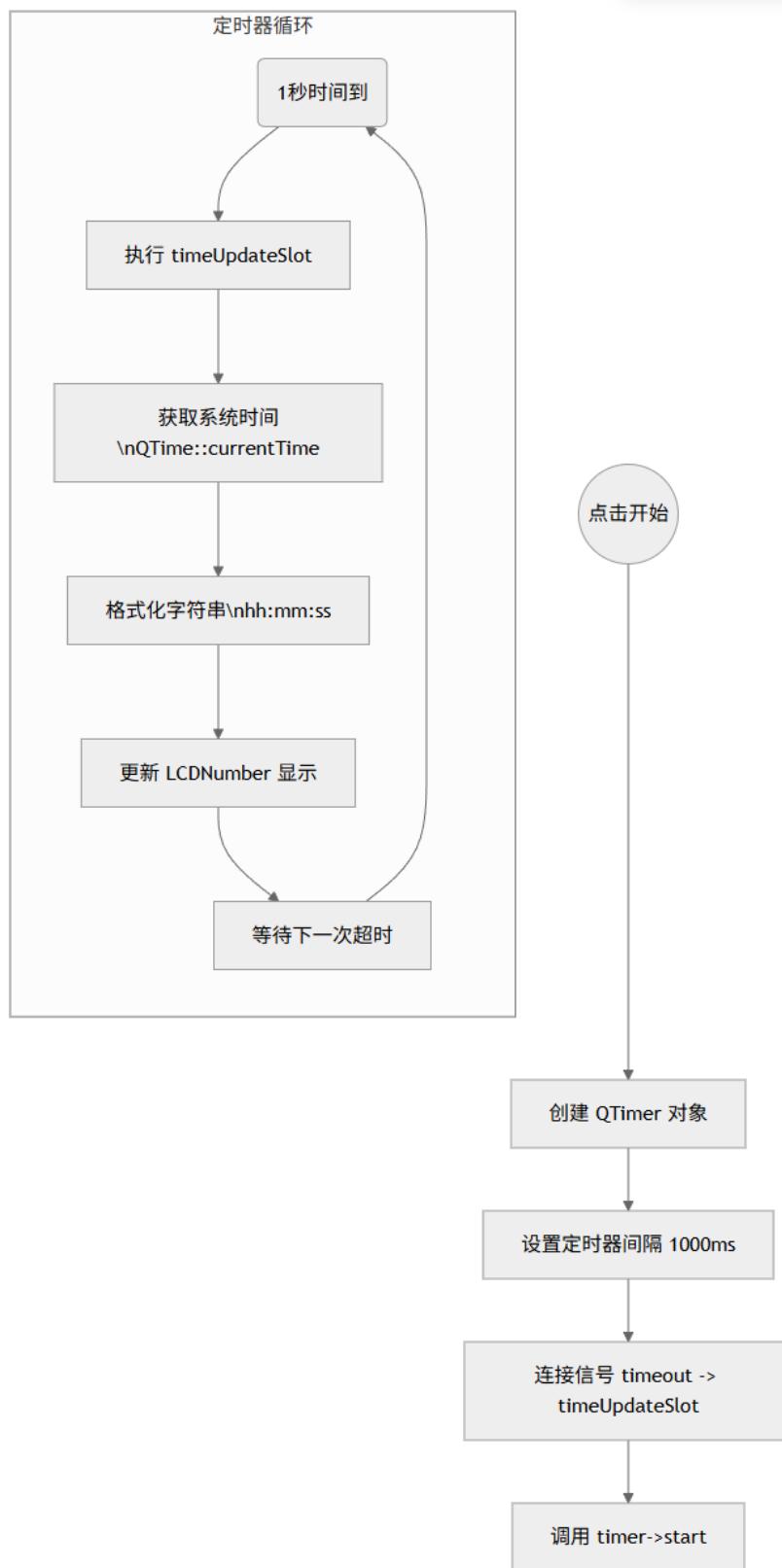
思路：修改 Stopwatch (第一部分) 项目，将其从计时器改为显示当前系统时间。

```
    delete ui;
}
void Timerr::start()
{
    timer = new QTimer(this);
    timer->setSingleShot(false);
    timer->start(1000);
    //超时发出信号
    connect(timer,SIGNAL(timeout()),this,SLOT(timeUpdateSlot()));
    timeUpdateSlot();
    ui->lcdNumber->show();
}
void Timerr::stop()
```

修改显示逻辑，将原来的手动计数逻辑替换为获取系统当前时间并格式化显示。

```
28     timer->stop(),
29 }
30 //超时处理槽函数
31 void Timerr::timeUpdateSlot()
32 {
33     // 获取当前系统时间
34     QTime currentTime = QTime::currentTime();
35     // 格式化为 HH:mm:ss
36     time = currentTime.toString("hh:mm:ss");
37
38     ui->lcdNumber->display(time);
39 }
```

流程图如下



实验结果如下



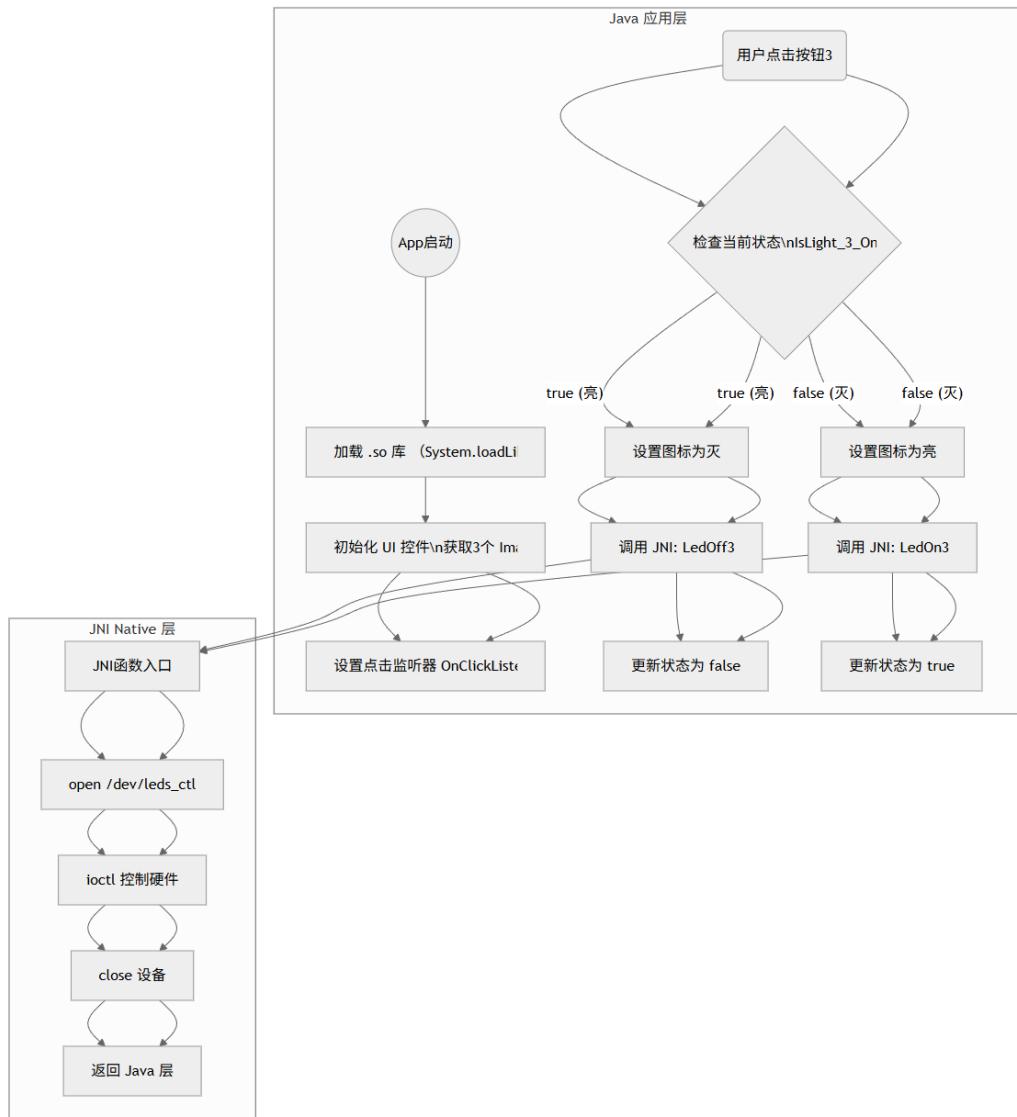
■ 设计实验 3-2：在数码管上显示 ADC 值

第二部分：Android 设计实验

■ 设计实验 4-1：3 个 LED 灯

设计思路：扩展原有的 Android LED 实验（仅控制 2 个灯），使其能控制 3 个灯

流程图如下



JNI 层修改 (native-lib.cpp):

修改宏定义，正确映射硬件 GPIO: LED1('y'), LED2('z'), LED3('x')。

增加 LedOn3 和 LedOff3 的 C++ 实现函数，调用 ioctl 控制设备文件。

硬件映射

```
8  #include <ansi_c.h>
9
10 #define LED1_ON      _IO('y',1)
11 #define LED1_OFF     _IO('y',0)
12 #define LED2_ON      _IO('z',1)
13 #define LED2_OFF     _IO('z',0)
14 #define LED3_ON      _IO('x',1)
15 #define LED3_OFF     _IO('x',0)
16
17 int fd = 0;
```

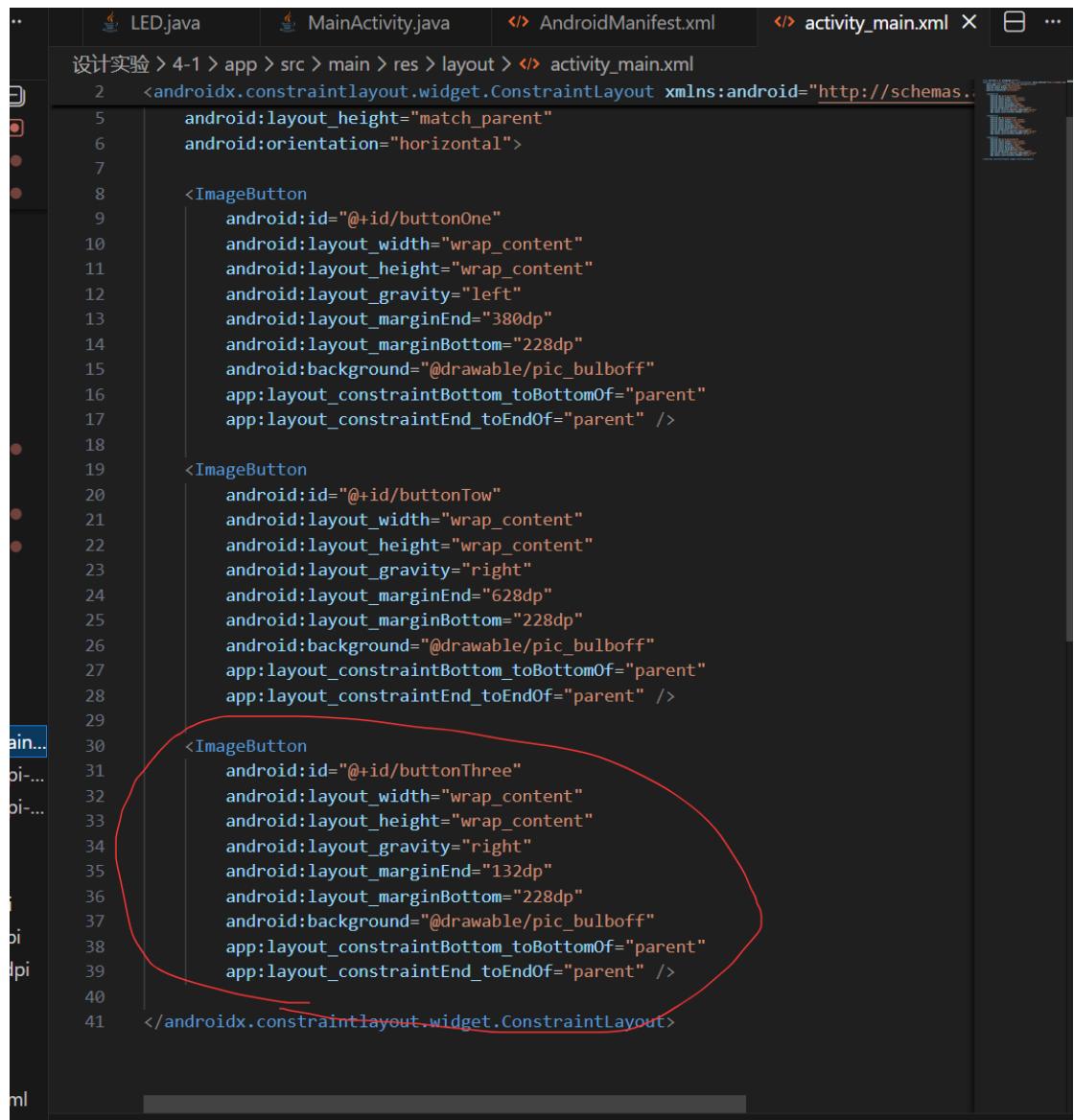
增加 led3 的实现函数

```
设计实验 > 4-1 > app > src > main > cpp > C++ native-lib.cpp > ...
65     |     JNIEnv* env,
66     |     jobject /* this */
67 {
68     |     ioctl(fd, LED2_OFF);
69     |     return 0;
70 }
71
72 extern "C" JNIEXPORT jint JNICALL
73 Java_com_farsight_led_LED_LedOn3(
74     |     JNIEnv* env,
75     |     jobject /* this */
76 {
77     |     ioctl(fd, LED3_ON);
78     |     return 0;
79 }
80
81 extern "C" JNIEXPORT jint JNICALL
82 Java_com_farsight_led_LED_LedOff3(
83     |     JNIEnv* env,
84     |     jobject /* this */
85 {
86     |     ioctl(fd, LED3_OFF);
87     |     return 0;
88 }
89
90 extern "C" JNIEXPORT jint JNICALL
91 Java_com_farsight_led_LED_close(
92     |     JNIEnv* env,
93     |     jobject /* this */
94 {
95     |     close(fd);
96     |     return 0;
}
问题 4 输出 调试控制台 终端 端口 powershell
```

Java 层修改 (LED.java)： 声明新的 Native 接口 LedOn3() 和 LedOff3()。

```
设计实验 > 4-1 > app > src > main > java > com > farsight > led > LED.java
1 package com.farsight.led;
2
3 public class LED
4 {
5     static
6     {
7         System.loadLibrary("led");
8     }
9     public native int open();
10    public native int close();
11    public native int LedOn1();
12    public native int LedOff1();
13    public native int LedOn2();
14    public native int LedOff2();
15    public native int LedOn3(); // Red circle here
16    public native int LedOff3(); // Red circle here
17 }
18
```

UI 层修改 (activity_main.xml)：在布局文件中增加第三个 ImageButton，设置 ID 和初始图片。



```
.. LED.java MainActivity.java AndroidManifest.xml activity_main.xml ...
设计实验 > 4-1 > app > src > main > res > layout > activity_main.xml
2 <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas...
5     android:layout_height="match_parent"
6     android:orientation="horizontal">
7
8     <ImageButton
9         android:id="@+id/buttonOne"
10        android:layout_width="wrap_content"
11        android:layout_height="wrap_content"
12        android:layout_gravity="left"
13        android:layout_marginEnd="380dp"
14        android:layout_marginBottom="228dp"
15        android:background="@drawable/pic_bulboff"
16        app:layout_constraintBottom_toBottomOf="parent"
17        app:layout_constraintEnd_toEndOf="parent" />
18
19     <ImageButton
20         android:id="@+id/buttonTwo"
21        android:layout_width="wrap_content"
22        android:layout_height="wrap_content"
23        android:layout_gravity="right"
24        android:layout_marginEnd="628dp"
25        android:layout_marginBottom="228dp"
26        android:background="@drawable/pic_bulboff"
27        app:layout_constraintBottom_toBottomOf="parent"
28        app:layout_constraintEnd_toEndOf="parent" />
29
30     <ImageButton
31         android:id="@+id/buttonThree"
32        android:layout_width="wrap_content"
33        android:layout_height="wrap_content"
34        android:layout_gravity="right"
35        android:layout_marginEnd="132dp"
36        android:layout_marginBottom="228dp"
37        android:background="@drawable/pic_bulboff"
38        app:layout_constraintBottom_toBottomOf="parent"
39        app:layout_constraintEnd_toEndOf="parent" />
40
41 </androidx.constraintlayout.widget.ConstraintLayout>
```

逻辑层修改 (MainActivity.java): 获取第三个按钮的实例，设置点击监听器。

在点击事件中调用 JNI 接口控制 LED3，并切换按钮的背景图片（亮/灭图标）。

```
4 import android.view.View;
5 import android.widget.ImageButton;
6 import com.farsight.led.databinding.ActivityMainBinding;
7
8 public class MainActivity extends AppCompatActivity
9 {
10     LED led = new LED();
11     boolean IsLight_1_On = false;
12     boolean IsLight_2_On = false;
13     boolean IsLight_3_On = false;
14     private ImageButton LedButton_1;
15     private ImageButton LedButton_2;
16     private ImageButton LedButton_3;
17
18     @Override
19     protected void onCreate(Bundle savedInstanceState)
```

```
67         led.LedOn2();
68         led.close();
69     }
70     IsLight_2_On = !IsLight_2_On;
71 });
72 });
73
74 LedButton_3 = (ImageButton) findViewById(R.id.buttonThree);
75
76 LedButton_3.setOnClickListener(new View.OnClickListener()
77 {
78     @Override
79     public void onClick(View view)
80     {
81         if (IsLight_3_On)
82         {
83             ((ImageButton)view).setBackground(getDrawable(R.drawable.pic_bulboff));
84             led.open();
85             led.LedOff3();
86             led.close();
87         }
88         else
89         {
90             ((ImageButton)view).setBackground(getDrawable(R.drawable.pic_bulbon));
91             led.open();
92             led.LedOn3();
93             led.close();
94         }
95     }
96 });
97 });
98
99
100 }
```

实验结果



详情见打包的视频

- 设计实验 4-2：通过图标控制蜂鸣器
- 设计实验 5-1：2 个按键
- 设计实验 5-2：采集 ADC 值

3. 实验报告具体要求

- 所选择的设计实验的具体设计思路（可利用文字、流程图等进行说明）
- 实验结果（截屏等方式体现）
- 设计实验遇到的问题，如何解决以及心得体会

遇到的问题及解决

问题：LED 灯的编号与控制引脚对应关系混乱，导致控制错位。

解决：查阅实验指导书或原理图，确认了 LED1 对应 GPIO 'y'，LED2 对应 'z'，LED3 对应 'x'，并在 native-lib.cpp 中修正了宏定义。

心得体会

本实验完整走通了 Android App 开发的 JNI 流程(Java Native Interface)。理解了上层 Java 代码是如何一步步穿透到 C++ 层，最终通过 Linux 设备驱动控制硬件的。这种“上层应用+底层驱动”的开发模式是嵌入式 Android 开发的核心技能