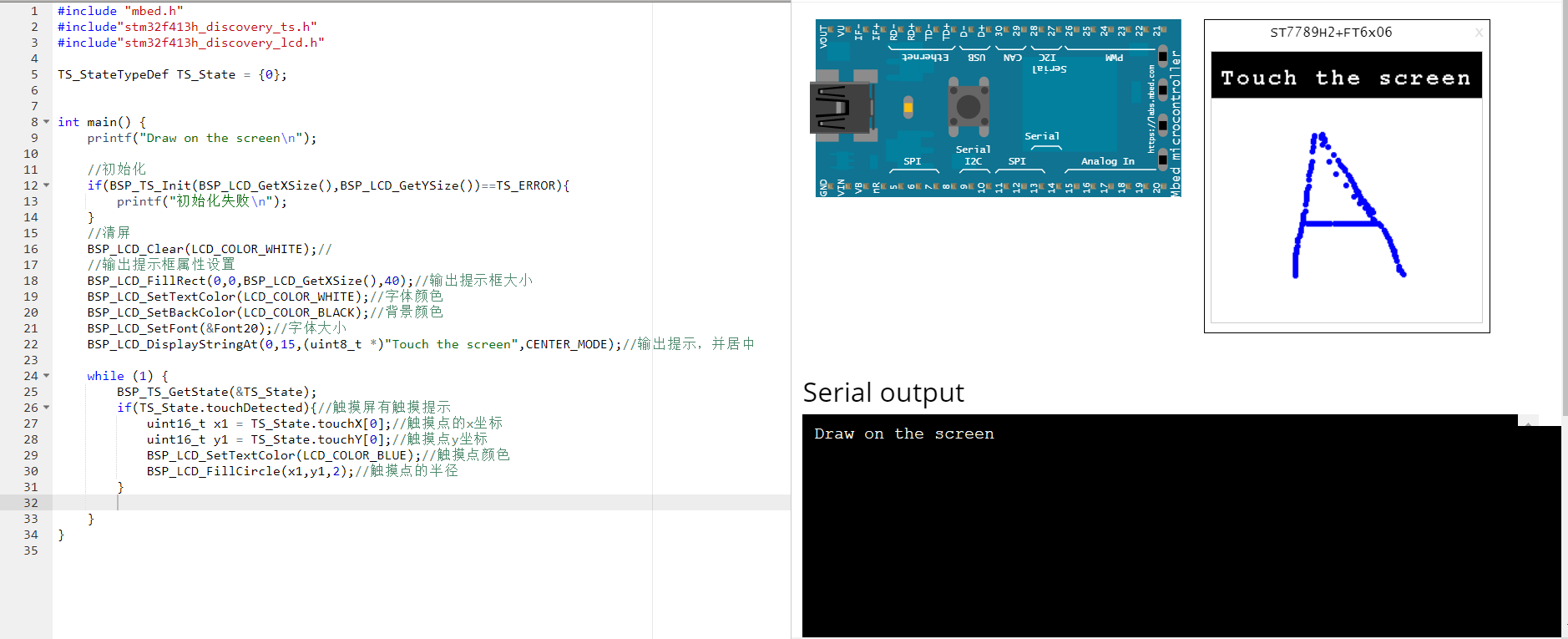
**实验4**

4-1 激活触摸检测并显示在屏幕上绘制的内容。

1. #include "mbed.h"
2. #include"stm32f413h\_discovery\_ts.h"
3. #include"stm32f413h\_discovery\_lcd.h"
5. TS\_StateTypeDef TS\_State = {0};

8. **int** main() {
9. printf("Draw on the screen\n");
11. //初始化
12. **if**(BSP\_TS\_Init(BSP\_LCD\_GetXSize(),BSP\_LCD\_GetYSize())==TS\_ERROR){
13. printf("初始化失败\n");
14. }
15. //清屏
16. BSP\_LCD\_Clear(LCD\_COLOR\_WHITE);//
17. //输出提示框属性设置
18. BSP\_LCD\_FillRect(0,0,BSP\_LCD\_GetXSize(),40);//输出提示框大小
19. BSP\_LCD\_SetTextColor(LCD\_COLOR\_WHITE);//字体颜色
20. BSP\_LCD\_SetBackColor(LCD\_COLOR\_BLACK);//背景颜色
21. BSP\_LCD\_SetFont(&Font20);//字体大小
22. BSP\_LCD\_DisplayStringAt(0,15,(uint8\_t \*)"Touch the screen",CENTER\_MODE);//输出提示，并居中
24. **while** (1) {
25. BSP\_TS\_GetState(&TS\_State);
26. **if**(TS\_State.touchDetected){//触摸屏有触摸提示
27. uint16\_t x1 = TS\_State.touchX[0];//触摸点的x坐标
28. uint16\_t y1 = TS\_State.touchY[0];//触摸点y坐标
29. BSP\_LCD\_SetTextColor(LCD\_COLOR\_BLUE);//触摸点颜色
30. BSP\_LCD\_FillCircle(x1,y1,2);//触摸点的半径
31. }
33. wait\_ms(5);
34. }
35. }



4-2 在触摸屏上绘制4×4键盘，接收来自虚拟键盘的输入，并将其显示在LCD上。

1. #include "mbed.h"
2. #include"C12832.h"
3. #include"stm32f413h\_discovery\_ts.h"
4. #include"stm32f413h\_discovery\_lcd.h"
6. TS\_StateTypeDef TS\_State = {0};//触摸屏
7. C12832 lcd(SPI\_MOSI,SPI\_SCK,SPI\_MISO,p8,p11);//显示屏定义初始化
9. //键盘——二维字符串数组定义
10. **const** **char** \* keyboard[][4] = {
11. {"1","2","3","F"},{"4","5","6","E"},{"7","8","9","D"},{"A","0","B","C"}
12. };
14. **int** c = 0,b = 0;
16. //键盘键背景输出
17. **void** BackPrint(){
18. //背景颜色
19. BSP\_LCD\_SetTextColor(LCD\_COLOR\_LIGHTGRAY);//背景颜色
20. //输出背景框（起点坐标）
21. **for**(**int** i=0;i<4;i++){
22. **for**(**int** j=0;j<4;j++){
23. BSP\_LCD\_FillRect(7.5+j\*(5+52.5),7.5+i\*(5+52.5),52.5,52.5);//起点坐标不同，每个键的背景框大小相同（均为52.5\*52.5）
24. }
25. }
26. **return**;
27. }
28. //键盘字输出
29. **void** CharacterPrint(){
30. //键盘字属性
31. BSP\_LCD\_SetBackColor(LCD\_COLOR\_LIGHTGRAY);//字体背景颜色
32. BSP\_LCD\_SetTextColor(LCD\_COLOR\_WHITE);//字体颜色
33. BSP\_LCD\_SetFont(&Font20);//字体大小
34. //输出
35. **for**(**int** i=0;i<4;i++){
36. **for**(**int** j=0;j<4;j++){
37. BSP\_LCD\_DisplayStringAt(52.5/3+7.5+j\*(5+52.5),52.5/3+7.5+i\*(5+52.5),(uint8\_t \*)keyboard[i][j],LEFT\_MODE);
38. }
39. }
40. }
42. **void** KeyboardPrint(){
43. BackPrint();//背景输出
44. CharacterPrint();//键盘字输出
45. }
47. **void** CharacterTouch(uint16\_t x1,uint16\_t y1){
48. //水平方向判断
49. **if**(x1>7.5 && x1<60) c = 0;
50. **if**(x1>65&&x1<117.5) c =1;
51. **if**(x1>122.5&&x1<175) c =2;
52. **if**(x1>180&&x1<232.5) c =3;
53. //竖直方向判断
54. **if**(y1>7.5 && y1<60) b = 0;
55. **if**(y1>65&&y1<117.5) b =1;
56. **if**(y1>122.5&&y1<175) b =2;
57. **if**(y1>180&&y1<232.5) b =3;
59. }
61. **int** main() {
62. printf("Draw on the screen\n");
63. //初始化
64. **if**(BSP\_TS\_Init(BSP\_LCD\_GetXSize(),BSP\_LCD\_GetYSize())==TS\_ERROR){
65. printf("初始化失败\n");
66. }
67. //清屏
68. BSP\_LCD\_Clear(LCD\_COLOR\_WHITE);
69. BSP\_LCD\_FillRect(0,0,BSP\_LCD\_GetXSize(),BSP\_LCD\_GetYSize());
70. BSP\_LCD\_SetBackColor(LCD\_COLOR\_BLACK);//键盘整体背景颜色
71. KeyboardPrint();//键盘输出
72. **while**(1){
73. BSP\_TS\_GetState(&TS\_State);
74. **if**(TS\_State.touchDetected){//触摸屏有触摸提示
75. uint16\_t x1 = TS\_State.touchX[0];//触摸点的x坐标
76. uint16\_t y1 = TS\_State.touchY[0];//触摸点y坐标
77. CharacterTouch(x1,y1);//根据输入坐标，判断输出哪个键盘字
78. lcd.locate(60,10);
79. lcd.printf(keyboard[b][c]);//注意坐标与二维字符串数组对应关系
80. }
81. wait\_ms(1);
82. }
83. }

