

```

1  #include <OneMsTaskTimer.h>
2
3  int xOutPin = P4_2;
4  int yOutPin = P5_5;
5  int xVal;
6  int yVal;
7
8  int selectPin = PUSH1;
9
10 int xFlag = 0;
11 int yFlag = 0;
12 bool jumpFlag = 0;
13
14 int cnt = 0;
15
16 void setupPlayerActions()
17 {
18     pinMode(selectPin, INPUT_PULLUP);
19     Serial.begin(9600);
20     HeroLocation.x = 0;
21     HeroLocation.y = 1;
22
23     attachInterrupt(digitalPinToInterrupt(selectPin), jumpISR, FALLING);
24 }
25
26 void loopPlayerActions()
27 {
28
29     while(PlayerActionFlag == 0)
30     {
31         delay(10);
32     }
33     PlayerActionFlag = 0;
34
35     //Serial.print("(PA Thread Wokring) ");
36
37
38     /*Serial.println(jumpFlag);
39     delay(1000);
40     Serial.println(jumpFlag);*/
41
42     if(PAS!=gameOver && PAS!=Gamestart && PAS!=GameInit) // detect player movements only
43     if the game is in progress
44     {
45         xVal = analogRead(xOutPin);
46         setXflag(xVal);
47
48         yVal = analogRead(yOutPin);
49         setYflag(yVal);
50     }
51
52     /*Serial.println(xVal);*/
53     //Serial.println(xFlag);
54
55     //Serial.println(yVal);
56     //Serial.println(yFlag);
57
58     PlayerStateProgress();
59
60     /*Serial.print("Hero position: ");
61     Serial.print(HeroLocation.x);
62     Serial.print(" ");
63     Serial.println(HeroLocation.y);*/
64     /*Serial.print("Current Game State: ");
65     Serial.println(PAS);*/
66
67     delay(10);
68 }

```

```

69
70 void jumpISR()
71 {
72     Serial.println("ISR - Jump");
73     jumpFlag = 1;
74 }
75
76 void setXflag(int xVal)
77 {
78     if(xVal > 730 && xVal < 830)
79     {
80         xFlag = 0;
81     }
82     else if(xVal>900)
83     {
84         xFlag = 1;
85     }
86     else if(xVal<533)
87     {
88         xFlag = -1;
89     }
90 }
91
92
93 void setYflag(int yVal)
94 {
95     if(yVal > 730 && yVal < 830)
96     {
97         yFlag = 0;
98     }
99     else if(yVal>900)
100    {
101        yFlag = 1;
102    }
103    else if(yVal<533)
104    {
105        yFlag = -1;
106    }
107 }
108
109 void PlayerStateProgress ()
110 {
111     //switch statements
112     switch(PAS)
113     {
114         case(GameInit):
115             PAS = Gamestart;
116             break;
117         case(Gamestart):
118             if(jumpFlag)
119             {
120                 points = 0;
121                 jumpFlag = 0;
122                 lcd.clear();
123                 PAS = WaitingForAction;
124             }
125             break;
126         case(WaitingForAction):
127             //Serial.println("Transition Waiting for Action");
128             if(xFlag == 1 && HeroLocation.x <= 15)
129             {
130                 PAS = MoveForwards;
131                 //Serial.println("Moving Forwards");
132                 break;
133             }
134             else if(xFlag == -1 && HeroLocation.x >= 0)
135             {
136                 PAS = MoveBackwards;
137                 break;

```

```

138     }
139     else if(yFlag == -1 && HeroLocation.y == 0)
140     {
141         Serial.println("Moving Down");
142         PAS = MoveDown;
143         break;
144     }
145     else if(yFlag == 1 && HeroLocation.y == 1)
146     {
147         Serial.println("Moving Up");
148         PAS = MoveUp;
149         break;
150     }
151     else
152     {
153         PAS = WaitingForAction;
154         break;
155     }
156     case(MoveForwards):
157         PAS = WaitingForAction;
158         break;
159     case(MoveBackwards):
160         PAS = WaitingForAction;
161         break;
162     case(MoveDown):
163         PAS = WaitingForAction;
164         break;
165     case(MoveUp):
166         PAS = WaitingForAction;
167         break;
168     default:
169         PAS = WaitingForAction;
170         break;
171     case(gameOver):
172
173         //Serial.println("Game Over");
174
175         if(jumpFlag)
176         {
177             Serial.println(jumpFlag);
178             jumpFlag = 0;
179             lcd.clear();
180             PAS = Gamestart;
181         }
182         break;
183 }
184
185 //state actions
186 switch(PAS)
187 {
188     case(Gamestart):
189         lcd.setCursor(5, 0);
190         lcd.print("READY?");
191         lcd.setCursor(1, 1);
192         lcd.print("press to start");
193         HeroLocation.x=0;
194         HeroLocation.y=1;
195         break;
196
197     case(MoveForwards):
198         //Serial.println("Move Forward State");
199         xFlag=0;
200         (HeroLocation.x) = (HeroLocation.x) + 1;
201         break;
202
203     case(MoveBackwards):
204         //Serial.println("Move Backward State");
205         xFlag=0;
206         (HeroLocation.x)--;

```

```
207         break;
208
209     case (MoveUp) :
210         yFlag = 0;
211         (HeroLocation.y) = 0;
212         break;
213
214     case (MoveDown) :
215         yFlag = 0;
216         (HeroLocation.y) = 1;
217         break;
218
219     case (gameOver) :
220         for(int i=0; i<obstcount; i++)
221         {
222             obsticals[i].position.x = 16; //reset every obstical on game over
223             obsticals[i].active = 0;
224             obsticals[i].checked = 0;
225         }
226
227         bonus.position.x = 16; //also reset everything else
228         bonus.position.y = 0;
229         bonus.active = 0;
230         nukeCount = 0;
231         count = 0;
232         shieldsInUse = 0;
233         maxShields = 2;
234         deelaay = 1000;
235
236         //pointsx = 15; //also reset points
237         points = 0;
238         break;
239     }
240 }
241
242
```