## UFO

Kenneth Noronha #50068016  
EE500 Embedded Systems Howework#3

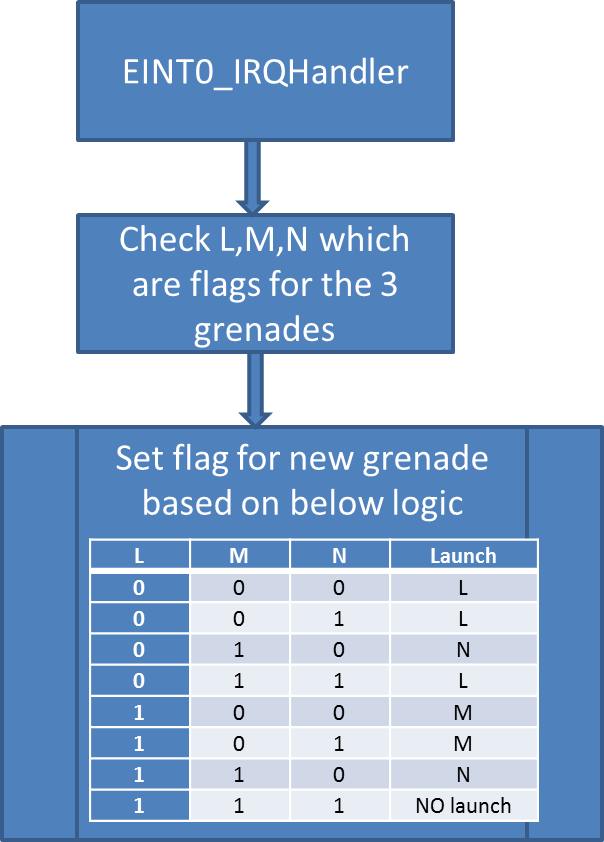
## Introduction

The report is for the UFO project. This is an application on the MCB1700 board. The UFO is at the bottom center of the screen and is moved using the joystick using the left and right buttons. The UFO can shoot grenades a maximum of three at a time. The INT0 button is used for the same. The timer0 is used to capture movements.

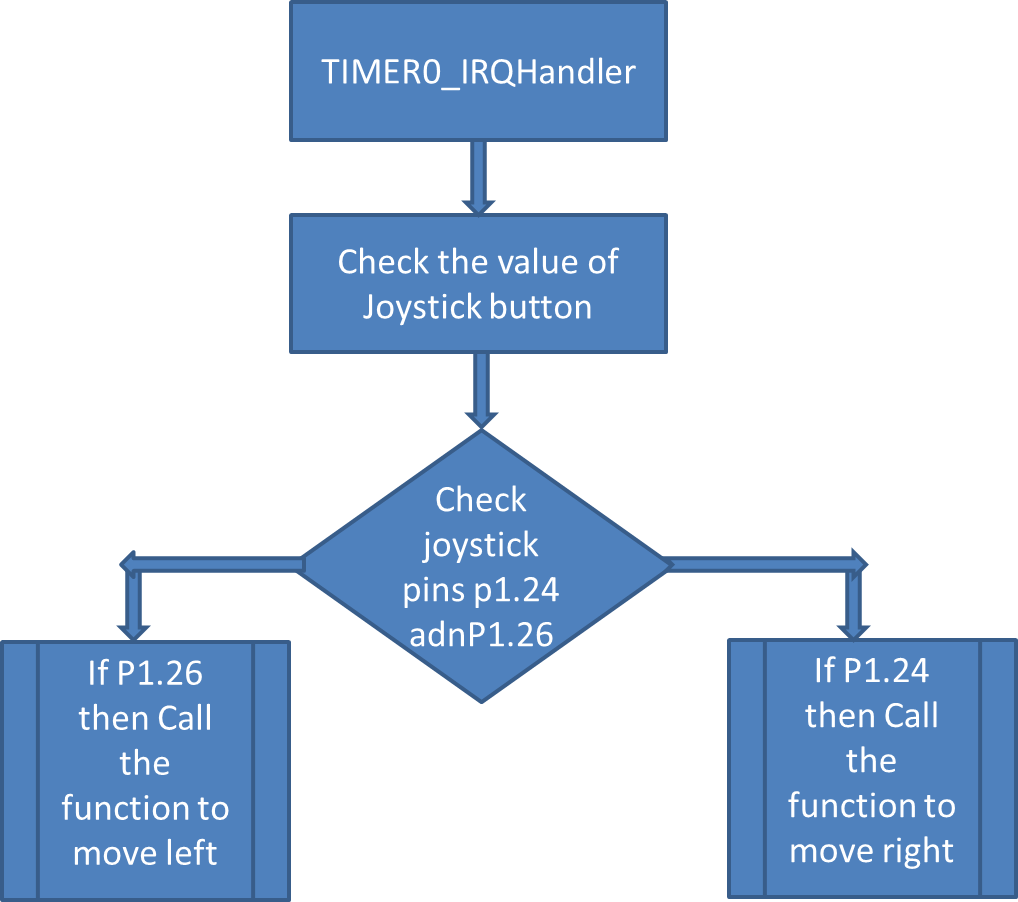
## Unique features

1. Logic table used for launching of grenade.
2. Parallel drawing. Only one loop.
3. Instead of erasing and redrawing the entire UFO/grenade we only redraw and erase parts which have to be changed making the code more efficient and responsive.
4. Sound is used when any button is pressed.

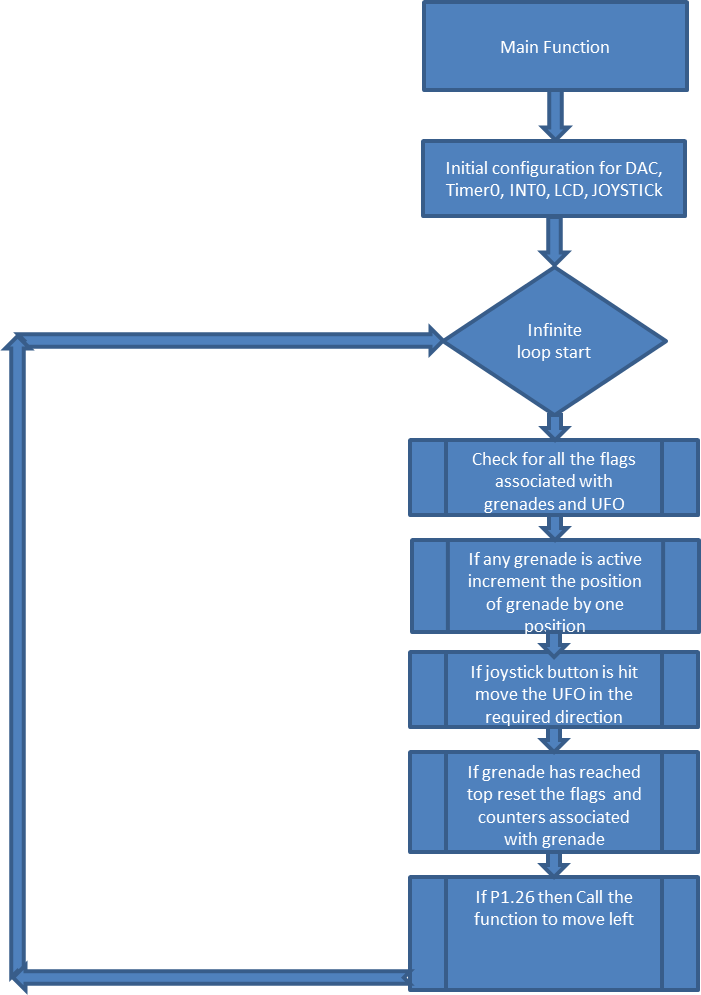
**INT0 Interrupt**



**TIMER0 Interrupt**



**Main Function**

****

## Initialization

This is done in the main function.  
Timer0

* Power up Timer 0: LPC\_SC->PCONP |= 1 << 1;
* Clock for timer = CCLK: LPC\_SC->PCLKSEL0 |= 1 << 2;
* Give a value for Match Control Register: LPC\_TIM0->MR0 = 1 << 22;
* Interrupt on Match 0 compare: LPC\_TIM0->MCR |= 1 << 0
* Reset timer on Match 0 : LPC\_TIM0->MCR |= 1 << 1
* Manually Reset Timer 0 : LPC\_TIM0->TCR |= 1 << 1
* Stop resetting the timer : LPC\_TIM0->TCR &= ~(1 << 1);
* Enable source of interrupt for timer0: NVIC\_EnableIRQ(TIMER0\_IRQn);
* Start timer LPC\_TIM0->TCR |= 1 << 0;

INT0

* Power up: LPC\_SC->PCONP |= ( 1 << 15 );
* Set pin P2.10 as EINT0: LPC\_PINCON->PINSEL4 = 0x00100000;
* Enable source of interrupt for INT0: NVIC\_EnableIRQ(EINT0\_IRQn);

LCD

* Call the GLCD function GLCD\_Init();
* Clear the LCD screen : GLCD\_Clear(Black);

Joystick

* Power up GPIO. LPC\_SC->PCONP |= ( 1 << 15 );
* Mode : LPC\_PINCON->PINSEL3 &= ~((3<< 8)|(3<<14)|(3<<16)|(3<<18)|(3<<20));
* Set the direction as input LPC\_GPIO1->FIODIR &= ~((1<<24)|(1<<26));

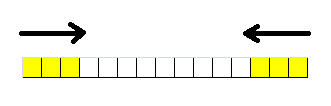
DAC for sound

* Power up DAC.
* set p0.26 to DAC output: LPC\_PINCON->PINSEL1 = 0x00200000;
* System clock update. Load DACR with value to be converted to analog

## Draw Straight line functions

The grenade and UFO are both drawn using the basic function to draw a line. We have two functions for line one which draws a starting line in x direction and the other which draws a start line in Y direction.

We basically draw a line by drawing pixels from both directions.



64 void KEN\_draw\_straightlineX( int x0, int y0, int x1){

65

66 // first obtain absolute distance

67 int dx = x0<x1 ? (x1-x0) : (x0-x1); // basically dx=abs(x1-x0)

68 int k = dx/2+1; // Get k value to get distance by 2.

69 while(k>0) {

70 // call of GCLD function;

71 GLCD\_PutPixel(x0,y0); // Put pixels from one direction

72 GLCD\_PutPixel(x1,y0); // Put pixels from opposite direction

73 x1--; // decrement in one direction

74 x0++; // increment on other direction

75 k--;

76 }

77 }

78

79 void KEN\_draw\_straightlineY( int x0, int y0, int y1) {

80

81 // first obtain absolute distance

82 int dy = y0<y1 ? (y1-y0) : (y0-y1); // basically dx=abs(y1-y0)

83 int k = dy/2+1; // Get k value to get distance by 2.

84 // call of GCLD function;

85 while(k>0) {

86 GLCD\_PutPixel(x0,y0); // Put pixels from one direction

87 GLCD\_PutPixel(x0,y1); // Put pixels from opposite direction

88 y1--; // decrement in one direction

89 y0++; // increment on other direction

90 k--;

91 }

92 }

## UFO initial drawing

Now we draw the UFO using the lines. We have divided the UFO into blocks of 5\*5 pixels.



All the blocks are drawn simultaneously. In each pass of the loop one line corresponding to each block is drawn using the KEN\_draw\_straightlineY explained earlier.

42void KEN\_draw\_UFO( int x0, int y0){

43 int k = 0 ;

44

45 while (k < 5) {

46

47 // Draw the first set of blue blocks

48 GLCD\_SetTextColor(Blue);

49 KEN\_draw\_straightlineY( x0-15,y0,x0-15,y0+4);

50 KEN\_draw\_straightlineY( x0-10,y0,x0-10,y0+4);

51 KEN\_draw\_straightlineY( x0-5,y0,x0-5,y0+4);

52 KEN\_draw\_straightlineY( x0,y0,x0,y0+4);

53 KEN\_draw\_straightlineY( x0+4,y0,x0+4,y0+4);

54 KEN\_draw\_straightlineY( x0+9,y0,x0+9,y0+4);

55 // Draw the second set of blue blocks

56 GLCD\_SetTextColor(Blue);

57 KEN\_draw\_straightlineY( x0-25,y0+5,x0-25,y0+9);

58 KEN\_draw\_straightlineY( x0-20,y0+5,x0-20,y0+9);

59 KEN\_draw\_straightlineY( x0-15,y0+5,x0-15,y0+9);

60 KEN\_draw\_straightlineY( x0-10,y0+5,x0-10,y0+9);

61 KEN\_draw\_straightlineY( x0-5,y0+5,x0-5,y0+9);

62 KEN\_draw\_straightlineY( x0,y0+5,x0,y0+9);

63 KEN\_draw\_straightlineY( x0+4,y0+5,x0+4,y0+9);

64 KEN\_draw\_straightlineY( x0+9,y0+5,x0+9,y0+9);

65 KEN\_draw\_straightlineY( x0+14,y0+5,x0+14,y0+9);

66 KEN\_draw\_straightlineY( x0+19,y0+5,x0+19,y0+9);

67 // Draw the tthird set of consecutive red and blue blocks

68 GLCD\_SetTextColor(Red);

69 KEN\_draw\_straightlineY( x0-35,y0+10,x0-35,y0+14);

70 GLCD\_SetTextColor(Yellow);

71 KEN\_draw\_straightlineY( x0-30,y0+10,x0-30,y0+14);

72 GLCD\_SetTextColor(Red);

73 KEN\_draw\_straightlineY( x0-25,y0+10,x0-25,y0+14);

74 GLCD\_SetTextColor(Yellow);

75 KEN\_draw\_straightlineY( x0-20,y0+10,x0-20,y0+14);

76 GLCD\_SetTextColor(Red);

77 KEN\_draw\_straightlineY( x0-15,y0+10,x0-15,y0+14);

78 GLCD\_SetTextColor(Yellow);

79 KEN\_draw\_straightlineY( x0-10,y0+10,x0-10,y0+14);

80 GLCD\_SetTextColor(Red);

81 KEN\_draw\_straightlineY( x0-5,y0+10,x0-5,y0+14);

82 GLCD\_SetTextColor(Yellow);

83 KEN\_draw\_straightlineY( x0,y0+10,x0,y0+14);

84 GLCD\_SetTextColor(Red);

85 KEN\_draw\_straightlineY( x0+4,y0+10,x0+4,y0+14);

86 GLCD\_SetTextColor(Yellow);

87 KEN\_draw\_straightlineY( x0+9,y0+10,x0+9,y0+14);

88 GLCD\_SetTextColor(Red);

89 KEN\_draw\_straightlineY( x0+14,y0+10,x0+14,y0+14);

90 GLCD\_SetTextColor(Yellow);

91 KEN\_draw\_straightlineY( x0+19,y0+10,x0+19,y0+14);

92 GLCD\_SetTextColor(Red);

93 KEN\_draw\_straightlineY( x0+24,y0+10,x0+24,y0+14);

94 GLCD\_SetTextColor(Yellow);

95 KEN\_draw\_straightlineY( x0+29,y0+10,x0+29,y0+14);

96 // Draw the fourth set of blue blocks

97 GLCD\_SetTextColor(Blue);

98 KEN\_draw\_straightlineY( x0-25,y0+15,x0-25,y0+19);

99 KEN\_draw\_straightlineY( x0-20,y0+15,x0-20,y0+19);

100 KEN\_draw\_straightlineY( x0-15,y0+15,x0-15,y0+19);

101 KEN\_draw\_straightlineY( x0-10,y0+15,x0-10,y0+19);

102 KEN\_draw\_straightlineY( x0-5,y0+15,x0-5,y0+19);

103 KEN\_draw\_straightlineY( x0,y0+15,x0,y0+19);

104 KEN\_draw\_straightlineY( x0+4,y0+15,x0+4,y0+19);

105 KEN\_draw\_straightlineY( x0+9,y0+15,x0+9,y0+19);

106 KEN\_draw\_straightlineY( x0+14,y0+15,x0+14,y0+19);

107 KEN\_draw\_straightlineY( x0+19,y0+15,x0+19,y0+19);

108 // Draw the fifth set of blue blocks

109 KEN\_draw\_straightlineY( x0-15,y0+20,x0-15,y0+24);

110 KEN\_draw\_straightlineY( x0-10,y0+20,x0-10,y0+24);

111 KEN\_draw\_straightlineY( x0-5,y0+20,x0-5,y0+24);

112 KEN\_draw\_straightlineY( x0,y0+20,x0,y0+24);

113 KEN\_draw\_straightlineY( x0+4,y0+20,x0+4,y0+24);

114 KEN\_draw\_straightlineY( x0+9,y0+20,x0+9,y0+24);

115

116 x0++;

117 k++;

118 }

120 }

## Grenade initial drawing

The initial grenade drawing is also done using the line functions.



256 void KEN\_draw\_grenade( int x0, int y0){

257 int k ;

258

259 while ( k < 10){

260 KEN\_draw\_straightlineX( x0 - 7, y0+19, x0 + 7, y0+19); // straight line

261 KEN\_draw\_straightlineX( x0 - 1, y0+9, x0 + 1, y0+9); // straight line for stem

262 k++;

263 y0--;

264 }

265 }

## UFO redraw when left and right buttons of joystick is hit

When the UFO needs to be moved to the right or to the left depending on the button pressed on the joystick we need to redraw the UFO at the new position and erase the UFO in the old position.

It is more efficient however to draw/erase only segments of UFO pertaining to new position which have changed instead of the whole UFO itself.

Below is how we will move the UFO to the right. Similarly we will move the UFO to the left. Also the new 5\*5 block are drawn using vertical lines from left to right during right motion and right to left when moving left position.



void KEN\_draw\_UFORIGHT( int x0, int y0)

122 {

123 int k = 0 ;

124 while (k < 5)

125 {

126 // Erase the blocks to the left

127 GLCD\_SetTextColor(Black);

128 KEN\_draw\_straightlineY( x0-15,y0,x0-15,y0+4);

129 KEN\_draw\_straightlineY( x0-25,y0+5,x0-25,y0+9);

130 KEN\_draw\_straightlineY( x0-35,y0+10,x0-35,y0+14);

131 KEN\_draw\_straightlineY( xr0-15,y0+20,x0-15,y0+24);

132 // draw blue boxes to the right

133 GLCD\_SetTextColor(Blue);

134 KEN\_draw\_straightlineY( x0+14,y0,x0+14,y0+4);

135 KEN\_draw\_straightlineY( x0+24,y0+5,x0+24,y0+9);

136 KEN\_draw\_straightlineY( x0+24,y0+15,x0+24,y0+19);

137 KEN\_draw\_straightlineY( x0+14,y0+20,x0+14,y0+24);

138

139

140 //redraw red and yellow boxes

141 GLCD\_SetTextColor(Red);

142 KEN\_draw\_straightlineY( x0-30,y0+10,x0-30,y0+14);

143 GLCD\_SetTextColor(Yellow);

144 KEN\_draw\_straightlineY( x0-25,y0+10,x0-25,y0+14);

145 GLCD\_SetTextColor(Red);

146 KEN\_draw\_straightlineY( x0-20,y0+10,x0-20,y0+14);

147 GLCD\_SetTextColor(Yellow);

148 KEN\_draw\_straightlineY( x0-15,y0+10,x0-15,y0+14);

149 GLCD\_SetTextColor(Red);

150 KEN\_draw\_straightlineY( x0-10,y0+10,x0-10,y0+14);

151 GLCD\_SetTextColor(Yellow);

152 KEN\_draw\_straightlineY( x0-5,y0+10,x0-5,y0+14);

153 GLCD\_SetTextColor(Red);

154 KEN\_draw\_straightlineY( x0,y0+10,x0,y0+14);

155 GLCD\_SetTextColor(Yellow);

156 KEN\_draw\_straightlineY( x0+4,y0+10,x0+4,y0+14);

157 GLCD\_SetTextColor(Red);

158 KEN\_draw\_straightlineY( x0+9,y0+10,x0+9,y0+14);

159 GLCD\_SetTextColor(Yellow);

160 KEN\_draw\_straightlineY( x0+14,y0+10,x0+14,y0+14);

161 GLCD\_SetTextColor(Red);

162 KEN\_draw\_straightlineY( x0+19,y0+10,x0+19,y0+14);

163 GLCD\_SetTextColor(Yellow);

164 KEN\_draw\_straightlineY( x0+24,y0+10,x0+24,y0+14);

165 GLCD\_SetTextColor(Red);

166 KEN\_draw\_straightlineY( x0+29,y0+10,x0+29,y0+14);

167 GLCD\_SetTextColor(Yellow);

168 KEN\_draw\_straightlineY( x0+34,y0+10,x0+34,y0+14);

169

170 x0++;

171 k++;

172 }

173 }

174

175 void KEN\_draw\_UFOLEFT( int x0, int y0)

176 {

177

178 int k = 0 ;

179 while (k < 5)

180 {

181 // Erase the blocks to the right

182 GLCD\_SetTextColor(Black);

183 KEN\_draw\_straightlineY( x0+14,y0,x0+14,y0+4);

184 KEN\_draw\_straightlineY( x0+24,y0+5,x0+24,y0+9);

185 KEN\_draw\_straightlineY( x0+34,y0+10,x0+34,y0+14);

186 KEN\_draw\_straightlineY( x0+24,y0+15,x0+24,y0+19);

187 KEN\_draw\_straightlineY( x0+14,y0+20,x0+14,y0+24);

188

189 // redraw blue boxes to the left

190 GLCD\_SetTextColor(Blue);

191 KEN\_draw\_straightlineY( x0-15,y0,x0-20,y0+4);

192 KEN\_draw\_straightlineY( x0-25,y0+5,x0-25,y0+9);

193 KEN\_draw\_straightlineY( x0-25,y0+15,x0-25,y0+19);

194 KEN\_draw\_straightlineY( x0-15,y0+20,x0-20,y0+24);

195

196 //redraw red and yellow boxes

197 GLCD\_SetTextColor(Red);

198 KEN\_draw\_straightlineY( x0-40+5,y0+10,x0-40,y0+14);

199 GLCD\_SetTextColor(Yellow);

200 KEN\_draw\_straightlineY( x0-35+5,y0+10,x0-35,y0+14);

201 GLCD\_SetTextColor(Red);

202 KEN\_draw\_straightlineY( x0-30+5,y0+10,x0-30,y0+14);

203 GLCD\_SetTextColor(Yellow);

204 KEN\_draw\_straightlineY( x0-25+5,y0+10,x0-25,y0+14);

205 GLCD\_SetTextColor(Red);

206 KEN\_draw\_straightlineY( x0-20+5,y0+10,x0-20,y0+14);

207 GLCD\_SetTextColor(Yellow);

208 KEN\_draw\_straightlineY( x0-15+5,y0+10,x0-15,y0+14);

209 GLCD\_SetTextColor(Red);

210 KEN\_draw\_straightlineY( x0-10+5,y0+10,x0-10,y0+14);

211 GLCD\_SetTextColor(Yellow);

212 KEN\_draw\_straightlineY( x0-5+5,y0+10,x0-5,y0+14);

213 GLCD\_SetTextColor(Red);

214 KEN\_draw\_straightlineY( x0+5,y0+10,x0,y0+14);

215 GLCD\_SetTextColor(Yellow);

216 KEN\_draw\_straightlineY( x0+4+5,y0+10,x0+4,y0+14);

217 GLCD\_SetTextColor(Red);

218 KEN\_draw\_straightlineY( x0+9+5,y0+10,x0+9,y0+14);

219 GLCD\_SetTextColor(Yellow);

220 KEN\_draw\_straightlineY( x0+14+5,y0+10,x0+14,y0+14);

221 GLCD\_SetTextColor(Red);

222 KEN\_draw\_straightlineY( x0+19+5,y0+10,x0+19,y0+14);

223 GLCD\_SetTextColor(Yellow);

224 KEN\_draw\_straightlineY( x0+24+5,y0+10,x0+24,y0+14);

225

226 x0--;

227 k++;

228 }

229 }

## Grenade redrawn

Just as we redrew the UFO right and left the grenade also is redrawn by erasing and drawing only parts of the UFO which are being changed.



244

245 void KEN\_draw\_grenadeM( int x0, int y0){

246 int k ;

247 while ( k < 5){

248 GLCD\_SetTextColor(Black);

249 KEN\_draw\_straightlineX( x0 - 7, y0+19, x0 + 7, y0+19); // straight line for base of

250 GLCD\_SetTextColor(Magenta);

251 KEN\_draw\_straightlineX( x0 - 1, y0+1, x0 + 1, y0+1); // New stem drawn

252 KEN\_draw\_straightlineX( x0 - 7, y0+9, x0 + 7, y0+9); // New base bottom drawn

253 k++;

254 y0--;

255 }

256 }

## Sound function

This is used to generate a sound each time a grenade is launched or the UFO position is changed.

274 void sound (int x0){

275

276 while(x0){

277 x0--;

278 LPC\_DAC->DACR = (i << 6) | DAC\_BIAS;

279 i++;

280 for(g = 10000; m > 1; m--);

281 if ( i == DATA\_LENGTH ) {

283 i = 0;

284 }

285 }

286 }

## Joystick buttons capture

The joystick button is checked in the timer 0 interrupt function. First the timer is reset. Next we check the FIOPIN register to check the value of pins corresponding to the right and left of joystick. If right button is pressed we set the right flag and if left button is pressed we set the left flag. The sound function is also called.

192 void TIMER0\_IRQHandler(void)

193 {

194

195 // IR is "Interrupt Register".

196 if ( (LPC\_TIM0->IR & 0x01) == 0x01 ){ // if MR0 interrupt (this is a sanity check)

198 LPC\_TIM0->IR |= 1 << 0; // Clear MR0 interrupt flag (

200 kbd\_val = ~(LPC\_GPIO1->FIOPIN >> 20) & KBD\_MASK;

201

202 if (kbd\_val == 0x10 ){ // check if joystick right button is hit

203 right = 1 ; //right movement lag

204 sound(1000); // sound function

205 kbd\_val = 0;

206 }

207 if (kbd\_val == 0x40 ){ // check if joystick left button is hit

208 left = 1; //left movement flag

209 sound(1000); // sound function

210 kbd\_val = 0;

211 }

212 }

213 }

## Logic for Grenade launch

When the INT0 external interrupt occurs because of the pressing of the INT0 button the function EINT0\_IRQHandler (void) is called.

First we clear the interrupt by making LPC\_SC->EXTINT = 1.

Here we check the current status of grenades. We have three grenades and they have a corresponding flag "L","M","N". For example if L = 1 means that L grenade is currently active.

We decide to launch a new grenade based on the logic table below.

|  |  |  |  |
| --- | --- | --- | --- |
| **L** | **M** | **N** | **Launch** |
| 0 | 0 | 0 | L |
| 0 | 0 | 1 | L |
| 0 | 1 | 0 | N |
| 0 | 1 | 1 | L |
| 1 | 0 | 0 | M |
| 1 | 0 | 1 | M |
| 1 | 1 | 0 | N |
| 1 | 1 | 1 | NO launch |

Here for example if L = 1 and M = 1 and N = 0 . It means that grenade L and M are currently active and grenade N is not. Hence we launce grenade N be making the flag N = 1. If all the grenades L,M and N are active then no grenade is launched.

At the same time we capture the X coordinate of the UFO and pass it on to a variable corresponding to the grenade L,M and N depending on which grenade is being launched.

The sound function is also called when we a grenade is launched. The grenade is initially drawn using the function KEN\_draw\_grenade(int x, int y);

238 **void** EINT0\_IRQHandler (**void**) {

240 LPC\_SC->EXTINT = 1; /\* clear interrupt \*/

241

243 **if** ( l == 0 & m == 0 & n == 0){

244 l = 1; // launch grenade by making flag =1

245 d = q; // save the UFO x coordinate into a new variable

246 sound(10000); // sound function

247 KEN\_draw\_grenade(d,190); // Draw grenade initially

248 }

249

250 **else if** ( l == 0 & m == 0 & n == 1){

251 l = 1; // launch grenade by making flag =1

252 d = q; // save the UFO x coordinate into a new variable

253 sound(10000); // sound function

254 KEN\_draw\_grenade(d,190); // Draw grenade initially

255 }

256

257 **else if** ( l == 0 & m == 1 & n == 0){

258 n = 1; // launch grenade by making flag =1

259 f = q; // save the UFO x coordinate into a new variable

260 KEN\_draw\_grenade(f,190); // Draw grenade initially

261 sound(10000); // sound function

262 }

263

264 **else if** ( l == 0 & m == 1 & n == 1){

265 l = 1; // launch grenade by making flag =1

266 d = q; // save the UFO x coordinate into a new variable

267 KEN\_draw\_grenade(d,190); // Draw grenade initially

268 sound(10000); // sound function

269 }

270

271 **else if** ( l == 1 & m == 0 & n == 0){

272 m = 1; // launch grenade by making flag =1

273 e = q; // save the UFO x coordinate into a new variable

274 KEN\_draw\_grenade(e,190); // Draw grenade initially

275 sound(10000); // sound function

276 }

277

278 **else if** ( l == 1 & m == 0 & n == 1){

279 m = 1; // launch grenade by making flag =1

280 e = q; // save the UFO x coordinate into a new variable

281 KEN\_draw\_grenade(e,190); // Draw grenade initially

282 sound(10000); // sound function

283 }

284

285 **else if** ( l == 1 & m == 1 & n == 0){

286 n = 1; // launch grenade by making flag =1

287 f = q; // save the UFO x coordinate into a new variable

288 sound(10000); // sound function

289 KEN\_draw\_grenade(f,190); // Draw grenade initially

290 }

291

292 **else if** ( l == 1 & m == 1 & n == 1){

293 // since all the grenades are active no new grenade to be launched

294 }

295 }

## The while Loop in main function

Inside the while we will check for all the flags corresponding to the joystick buttons and grenades. When the UFO position has to be changed we call the UFO right and UFO left function and reset the flag and change the global variable which store the central coordinate for the UFO.

The grenades are moved inside the while loop. Each pass of the while loop changes the position of all the grenades which are active by one position which is by 5 pixels. The three grenades have three vertical movement coordinates which are reset one the grenade moves to the top. Also once the grenade moves to the top we reset the grenade flag and the captured UFO horizontal position.

while (1)

95 {

96

97

98 //check if anything needs to be done at all

99 while( l == 1 | m == 1 | n == 1 | left == 1 | right == 1){

100 // if the joystick right button has been pressed

101 if ( right == 1){

102 right = 0; //Reset the right flag

103 if ( q <= 310){ //As long as we have not reached right most point

104 KEN\_draw\_UFORIGHT( q, r);// Move UFO by one position

105 q=q + 5; // Update the position

106 }

107

108 }

109 // if the joystick left button has been pressed

110 if ( left == 1){

111 left = 0; //Reset the left flag

112 if( q >= 5){ //As long as we have not reached most left point

113 KEN\_draw\_UFOLEFT( q, r);// Move UFO by one position

114 q=q - 5; // Update the position

115 }

116

117 }

118 // if grenade L is active

119 if ( l == 1){

120 KEN\_draw\_grenadeM(d,190-y); //Redraw one instance of grenade

121 y+= 5; // Update the position

122 }

123 // if grenade M is active

124 if ( m == 1){

125 KEN\_draw\_grenadeM(e,190-z); //Redraw one instance of grenade

126 z+= 5; // Update the position

127 }

128 // if grenade N is active

129 if ( n == 1){

130 KEN\_draw\_grenadeM(f,190-x); //Redraw one instance of grenade

131 x+= 5; // Update the position

132 }

133

134 // check if any of the grenades ahs reached the top

135 if( y >= 190 | z >= 190 | x >= 190){

136

137 if( y >= 190 ){

138 GLCD\_SetTextColor(Black); // Clear the grenade

139 KEN\_draw\_grenade(d,0);

140 y =0; //reset the vertical counter

141 l=0; //Reset the grenade flag

142 d = 0; //reset UFO position capture

143 }

144 if( z >= 190 ){

145 GLCD\_SetTextColor(Black); // Clear the grenade

146 KEN\_draw\_grenade(e,0);

147 z =0; //reset the vertical counter

148 m=0; //Reset the grenade flag

149 e = 0; //reset UFO position capture

150 }

151 if( x >= 190 ){

152 GLCD\_SetTextColor(Black); // Clear the grenade

153 KEN\_draw\_grenade(f,0);

154 x =0; //reset the vertical counter

155 n=0; //Reset the grenade flag

156 f = 0; //reset UFO position capture

157 }

158 }

159

160 }

161

162 }

163 }

## The complete code

## Main.c

1 #include "GLCD.h"

2 #include "LPC17xx.h"

3 #include "KEN\_UTILS.h"

4 #include "type.h"

5 #include "dac.h"

6 #include "dma.h"

7

8 extern volatile uint32\_t DACDMA0Done, DACDMA1Done;

9 volatile int i, k,x,y,z,l = 0,m = 0,n = 0,q,r,left = 0, right = 0,d,e,f,g ;

10 int kbd\_val;

11

12

13 #define KBD\_MASK 0x79

14

15 int main (void) {

17 // DAC configuration

18 SystemClockUpdate();

19 //Timer0 configuration

20 LPC\_SC->PCONP |= 1 << 1;

21 LPC\_SC->PCLKSEL0 |= 1 << 2;

22 LPC\_TIM0->MR0 = 1 << 22;

23 LPC\_TIM0->MCR |= 1 << 0;

24 LPC\_TIM0->MCR |= 1 << 1;

25 LPC\_TIM0->TCR |= 1 << 1;

26 LPC\_TIM0->TCR &= ~(1 << 1);

27 NVIC\_EnableIRQ(TIMER0\_IRQn);

28 LPC\_TIM0->TCR |= 1 << 0;

29 //GPIO

30 LPC\_SC->PCONP |= ( 1 << 15 );

31 //INT0

32 LPC\_PINCON->PINSEL4 = 0x00100000;

33 LPC\_GPIOINT->IO2IntEnF = 0x200;

34 LPC\_SC->EXTMODE = 0x00000001;

35 LPC\_SC->EXTPOLAR = 0;

36 NVIC\_EnableIRQ(EINT0\_IRQn);

37

38 //LCD

39 GLCD\_Init();

40 GLCD\_Clear(Black);

41

42 //Joystick

43 LPC\_PINCON->PINSEL3 &= ~((3<< 8)|(3<<14)|(3<<16)|(3<<18)|(3<<20));

44 // P1.20, P1.23, …, P1.26 are GPIO (Joystick)

45 LPC\_GPIO1->FIODIR &= ~((1<<20)|(1<<23)|(1<<24)|(1<<25)|(1<<26));

46 // P1.20, P1.23, ..., P1.26 are input (Joystick)

47

48 // Initial UFO drawn

49 q = 160;

50 r = 210;

51 KEN\_draw\_UFO( q, r);

52

71

72

73

74 while (1) {

78 //check if anything needs to be done at all

79 while( l == 1 | m == 1 | n == 1 | left == 1 | right == 1){

80 // if the joystick right button has been pressed

81 if ( right == 1){

82 right = 0; //Reset the right flag

83 if ( q <= 310){ //As long as we have not reached right most point

84 KEN\_draw\_UFORIGHT( q, r);// Move UFO by one position

85 q=q + 5; // Update the position

86 }

87

88 }

89 // if the joystick left button has been pressed

90 if ( left == 1){

91 left = 0; //Reset the left flag

92 if( q >= 5){ //As long as we have not reached most left point

93 KEN\_draw\_UFOLEFT( q, r);// Move UFO by one position

94 q=q - 5; // Update the position

95 }

96

97 }

98 // if grenade L is active

99 if ( l == 1){

100 KEN\_draw\_grenadeM(d,190-y); //Redraw one instance of grenade

101 y+= 5; // Update the position

102 }

103 // if grenade M is active

104 if ( m == 1){

105 KEN\_draw\_grenadeM(e,190-z); //Redraw one instance of grenade

106 z+= 5; // Update the position

107 }

108 // if grenade N is active

109 if ( n == 1){

110 KEN\_draw\_grenadeM(f,190-x); //Redraw one instance of grenade

111 x+= 5; // Update the position

112 }

113

114 // check if any of the grenades ahs reached the top

115 if( y >= 190 | z >= 190 | x >= 190){

116

117 if( y >= 190 ){

118 GLCD\_SetTextColor(Black); // Clear the grenade

119 KEN\_draw\_grenade(d,0);

120 y =0; //reset the vertical counter

121 l=0; //Reset the grenade flag

122 d = 0; //reset UFO position capture

123 }

124 if( z >= 190 ){

125 GLCD\_SetTextColor(Black); // Clear the grenade

126 KEN\_draw\_grenade(e,0);

127 z =0; //reset the vertical counter

128 m=0; //Reset the grenade flag

129 e = 0; //reset UFO position capture

130 }

131 if( x >= 190 ){

132 GLCD\_SetTextColor(Black); // Clear the grenade

133 KEN\_draw\_grenade(f,0);

134 x =0; //reset the vertical counter

135 n=0; //Reset the grenade flag

136 f = 0; //reset UFO position capture

137 }

138 }

140 }

142 }

143 }

144 void TIMER0\_IRQHandler(void)

145 {

146

147 // IR is "Interrupt Register".

148 if ( (LPC\_TIM0->IR & 0x01) == 0x01 ){ // if MR0 interrupt (this is a sanity check);

149

150 LPC\_TIM0->IR |= 1 << 0; // Clear MR0 interrupt flag

151 LPC\_GPIO1->FIOPIN ^= 1 << 29; // Toggle the LED (see lab1)

152 kbd\_val = ~(LPC\_GPIO1->FIOPIN >> 20) & KBD\_MASK;

153

154 if (kbd\_val == 0x10 ){ // check if joystick right button is hit

155 right = 1 ; //right movement lag

156 sound(1000); // sound function

157 kbd\_val = 0;

158 }

159 if (kbd\_val == 0x40 ){ // check if joystick left button is hit

160 left = 1; //left movement flag

161 sound(1000); // sound function

162 kbd\_val = 0;

163 }

164 }

165 }

166

167

168 void EINT0\_IRQHandler (void)

169 {

170

171 LPC\_SC->EXTINT = 1; /\* clear interrupt \*/

172

173

174

175 if ( l == 0 & m == 0 & n == 0){

176 l = 1; // launch grenade by making flag =1

177 d = q; // save the UFO x coordinate into a new variable

178 sound(10000); // sound function

179 KEN\_draw\_grenade(d,190); // Draw grenade initially

180 }

181

182 else if ( l == 0 & m == 0 & n == 1){

183 l = 1; // launch grenade by making flag =1

184 d = q; // save the UFO x coordinate into a new variable

185 sound(10000); // sound function

186 KEN\_draw\_grenade(d,190); // Draw grenade initially

187 }

188

189 else if ( l == 0 & m == 1 & n == 0){

190 n = 1; // launch grenade by making flag =1

191 f = q; // save the UFO x coordinate into a new variable

192 KEN\_draw\_grenade(f,190); // Draw grenade initially

193 sound(10000); // sound function

194 }

195

196 else if ( l == 0 & m == 1 & n == 1){

197 l = 1; // launch grenade by making flag =1

198 d = q; // save the UFO x coordinate into a new variable

199 KEN\_draw\_grenade(d,190); // Draw grenade initially

200 sound(10000); // sound function

201 }

202

203 else if ( l == 1 & m == 0 & n == 0){

204 m = 1; // launch grenade by making flag =1

205 e = q; // save the UFO x coordinate into a new variable

206 KEN\_draw\_grenade(e,190); // Draw grenade initially

207 sound(10000); // sound function

208 }

209

210 else if ( l == 1 & m == 0 & n == 1){

211 m = 1; // launch grenade by making flag =1

212 e = q; // save the UFO x coordinate into a new variable

213 KEN\_draw\_grenade(e,190); // Draw grenade initially

214 sound(10000); // sound function

215 }

216

217 else if ( l == 1 & m == 1 & n == 0){

218 n = 1; // launch grenade by making flag =1

219 f = q; // save the UFO x coordinate into a new variable

220 sound(10000); // sound function

221 KEN\_draw\_grenade(f,190); // Draw grenade initially

222 }

223

224 else if ( l == 1 & m == 1 & n == 1){

225 // since all the grenades are active no new grenade to be launched

226 }

227 }

228

229 void sound (int x0){

230

231 while(x0){

232 x0--;

233 LPC\_DAC->DACR = (i << 6) | DAC\_BIAS;

234 i++;

235 for(g = 10000; m > 1; m--);

236 if ( i == DATA\_LENGTH ) {

238 i = 0;

239 }

240 }

241 }

## File : kenneth\_util.c

2 #include <stdio.h>

3 #include "LPC17xx.H" /\* LPC17xx definitions \*/

4 #include "GLCD.h"

5 #include "KEN\_UTILS.h"

6

7 #define \_\_FI 1 /\* Font index 16x24 \*/

8

9

10 **void** KEN\_draw\_straightlineX( **int** x0, **int** y0, **int** x1)

11 {

12 // first obtain absolute distance

13 **int** dx = x0<x1 ? (x1-x0) : (x0-x1); // basically dx=abs(x1-x0)

14 **int** k = dx/2+1; // Get k value to get distance by 2.

15 **while**(k>0) {

16 // call of GCLD function;

17 GLCD\_PutPixel(x0,y0); // Put pixels from one direction

18 GLCD\_PutPixel(x1,y0); // Put pixels from opposite direction

19 x1--; // decrement in one direction

20 x0++; // increment on other direction

21 k--;

22 }

23 }

24

25 **void** KEN\_draw\_straightlineY( **int** x0, **int** y0, **int** x1, **int** y1)

26 {

27 // first obtain absolute distance

28 **int** dy = y0<y1 ? (y1-y0) : (y0-y1); // basically dx=abs(y1-y0)

29 **int** k = dy/2+1; // Get k value to get distance by 2.

30 // call of GCLD function;

31 **while**(k>0) {

32 GLCD\_PutPixel(x0,y0); // Put pixels from one direction

33 GLCD\_PutPixel(x0,y1); // Put pixels from opposite direction

34 y1--; // decrement in one direction

35 y0++; // increment on other direction

36 k--;

37 }

38 }

41 **void** KEN\_draw\_UFO( **int** x0, **int** y0)

42 {

43 **int** k = 0 ;

44

45 **while** (k < 5)

46 {

47 // Draw the first set of blue blocks

48 GLCD\_SetTextColor(Blue);

49 KEN\_draw\_straightlineY( x0-15,y0,x0-15,y0+4);

50 KEN\_draw\_straightlineY( x0-10,y0,x0-10,y0+4);

51 KEN\_draw\_straightlineY( x0-5,y0,x0-5,y0+4);

52 KEN\_draw\_straightlineY( x0,y0,x0,y0+4);

53 KEN\_draw\_straightlineY( x0+4,y0,x0+4,y0+4);

54 KEN\_draw\_straightlineY( x0+9,y0,x0+9,y0+4);

55 // Draw the second set of blue blocks

56 GLCD\_SetTextColor(Blue);

57 KEN\_draw\_straightlineY( x0-25,y0+5,x0-25,y0+9);

58 KEN\_draw\_straightlineY( x0-20,y0+5,x0-20,y0+9);

59 KEN\_draw\_straightlineY( x0-15,y0+5,x0-15,y0+9);

60 KEN\_draw\_straightlineY( x0-10,y0+5,x0-10,y0+9);

61 KEN\_draw\_straightlineY( x0-5,y0+5,x0-5,y0+9);

62 KEN\_draw\_straightlineY( x0,y0+5,x0,y0+9);

63 KEN\_draw\_straightlineY( x0+4,y0+5,x0+4,y0+9);

64 KEN\_draw\_straightlineY( x0+9,y0+5,x0+9,y0+9);

65 KEN\_draw\_straightlineY( x0+14,y0+5,x0+14,y0+9);

66 KEN\_draw\_straightlineY( x0+19,y0+5,x0+19,y0+9);

67 // Draw the tthird set of consecutive red and blue blocks

68 GLCD\_SetTextColor(Red);

69 KEN\_draw\_straightlineY( x0-35,y0+10,x0-35,y0+14);

70 GLCD\_SetTextColor(Yellow);

71 KEN\_draw\_straightlineY( x0-30,y0+10,x0-30,y0+14);

72 GLCD\_SetTextColor(Red);

73 KEN\_draw\_straightlineY( x0-25,y0+10,x0-25,y0+14);

74 GLCD\_SetTextColor(Yellow);

75 KEN\_draw\_straightlineY( x0-20,y0+10,x0-20,y0+14);

76 GLCD\_SetTextColor(Red);

77 KEN\_draw\_straightlineY( x0-15,y0+10,x0-15,y0+14);

78 GLCD\_SetTextColor(Yellow);

79 KEN\_draw\_straightlineY( x0-10,y0+10,x0-10,y0+14);

80 GLCD\_SetTextColor(Red);

81 KEN\_draw\_straightlineY( x0-5,y0+10,x0-5,y0+14);

82 GLCD\_SetTextColor(Yellow);

83 KEN\_draw\_straightlineY( x0,y0+10,x0,y0+14);

84 GLCD\_SetTextColor(Red);

85 KEN\_draw\_straightlineY( x0+4,y0+10,x0+4,y0+14);

86 GLCD\_SetTextColor(Yellow);

87 KEN\_draw\_straightlineY( x0+9,y0+10,x0+9,y0+14);

88 GLCD\_SetTextColor(Red);

89 KEN\_draw\_straightlineY( x0+14,y0+10,x0+14,y0+14);

90 GLCD\_SetTextColor(Yellow);

91 KEN\_draw\_straightlineY( x0+19,y0+10,x0+19,y0+14);

92 GLCD\_SetTextColor(Red);

93 KEN\_draw\_straightlineY( x0+24,y0+10,x0+24,y0+14);

94 GLCD\_SetTextColor(Yellow);

95 KEN\_draw\_straightlineY( x0+29,y0+10,x0+29,y0+14);

96 // Draw the fourth set of blue blocks

97 GLCD\_SetTextColor(Blue);

98 KEN\_draw\_straightlineY( x0-25,y0+15,x0-25,y0+19);

99 KEN\_draw\_straightlineY( x0-20,y0+15,x0-20,y0+19);

100 KEN\_draw\_straightlineY( x0-15,y0+15,x0-15,y0+19);

101 KEN\_draw\_straightlineY( x0-10,y0+15,x0-10,y0+19);

102 KEN\_draw\_straightlineY( x0-5,y0+15,x0-5,y0+19);

103 KEN\_draw\_straightlineY( x0,y0+15,x0,y0+19);

104 KEN\_draw\_straightlineY( x0+4,y0+15,x0+4,y0+19);

105 KEN\_draw\_straightlineY( x0+9,y0+15,x0+9,y0+19);

106 KEN\_draw\_straightlineY( x0+14,y0+15,x0+14,y0+19);

107 KEN\_draw\_straightlineY( x0+19,y0+15,x0+19,y0+19);

108 // Draw the fifth set of blue blocks

109 KEN\_draw\_straightlineY( x0-15,y0+20,x0-15,y0+24);

110 KEN\_draw\_straightlineY( x0-10,y0+20,x0-10,y0+24);

111 KEN\_draw\_straightlineY( x0-5,y0+20,x0-5,y0+24);

112 KEN\_draw\_straightlineY( x0,y0+20,x0,y0+24);

113 KEN\_draw\_straightlineY( x0+4,y0+20,x0+4,y0+24);

114 KEN\_draw\_straightlineY( x0+9,y0+20,x0+9,y0+24);

115

116 x0++;

117 k++;

118 }

119

120 }

121 **void** KEN\_draw\_UFORIGHT( **int** x0, **int** y0) {

122

123 **int** k = 0 ;

124 **while** (k < 5) {

126 // Erase the blocks to the left

127 GLCD\_SetTextColor(Black);

128 KEN\_draw\_straightlineY( x0-15,y0,x0-15,y0+4);

129 KEN\_draw\_straightlineY( x0-25,y0+5,x0-25,y0+9);

130 KEN\_draw\_straightlineY( x0-35,y0+10,x0-35,y0+14);

131 KEN\_draw\_straightlineY( xr0-15,y0+20,x0-15,y0+24);

132 // draw blue boxes to the right

133 GLCD\_SetTextColor(Blue);

134 KEN\_draw\_straightlineY( x0+14,y0,x0+14,y0+4);

135 KEN\_draw\_straightlineY( x0+24,y0+5,x0+24,y0+9);

136 KEN\_draw\_straightlineY( x0+24,y0+15,x0+24,y0+19);

137 KEN\_draw\_straightlineY( x0+14,y0+20,x0+14,y0+24);

138

139

140 //redraw red and yellow boxes

141 GLCD\_SetTextColor(Red);

142 KEN\_draw\_straightlineY( x0-30,y0+10,x0-30,y0+14);

143 GLCD\_SetTextColor(Yellow);

144 KEN\_draw\_straightlineY( x0-25,y0+10,x0-25,y0+14);

145 GLCD\_SetTextColor(Red);

146 KEN\_draw\_straightlineY( x0-20,y0+10,x0-20,y0+14);

147 GLCD\_SetTextColor(Yellow);

148 KEN\_draw\_straightlineY( x0-15,y0+10,x0-15,y0+14);

149 GLCD\_SetTextColor(Red);

150 KEN\_draw\_straightlineY( x0-10,y0+10,x0-10,y0+14);

151 GLCD\_SetTextColor(Yellow);

152 KEN\_draw\_straightlineY( x0-5,y0+10,x0-5,y0+14);

153 GLCD\_SetTextColor(Red);

154 KEN\_draw\_straightlineY( x0,y0+10,x0,y0+14);

155 GLCD\_SetTextColor(Yellow);

156 KEN\_draw\_straightlineY( x0+4,y0+10,x0+4,y0+14);

157 GLCD\_SetTextColor(Red);

158 KEN\_draw\_straightlineY( x0+9,y0+10,x0+9,y0+14);

159 GLCD\_SetTextColor(Yellow);

160 KEN\_draw\_straightlineY( x0+14,y0+10,x0+14,y0+14);

161 GLCD\_SetTextColor(Red);

162 KEN\_draw\_straightlineY( x0+19,y0+10,x0+19,y0+14);

163 GLCD\_SetTextColor(Yellow);

164 KEN\_draw\_straightlineY( x0+24,y0+10,x0+24,y0+14);

165 GLCD\_SetTextColor(Red);

166 KEN\_draw\_straightlineY( x0+29,y0+10,x0+29,y0+14);

167 GLCD\_SetTextColor(Yellow);

168 KEN\_draw\_straightlineY( x0+34,y0+10,x0+34,y0+14);

169

170 x0++;

171 k++;

172 }

173 }

174

175 **void** KEN\_draw\_UFOLEFT( **int** x0, **int** y0)

176 {

177

178 **int** k = 0 ;

179 **while** (k < 5)

180 {

181 // Erase the blocks to the right

182 GLCD\_SetTextColor(Black);

183 KEN\_draw\_straightlineY( x0+14,y0,x0+14,y0+4);

184 KEN\_draw\_straightlineY( x0+24,y0+5,x0+24,y0+9);

185 KEN\_draw\_straightlineY( x0+34,y0+10,x0+34,y0+14);

186 KEN\_draw\_straightlineY( x0+24,y0+15,x0+24,y0+19);

187 KEN\_draw\_straightlineY( x0+14,y0+20,x0+14,y0+24);

188

189 // redraw blue boxes to the left

190 GLCD\_SetTextColor(Blue);

191 KEN\_draw\_straightlineY( x0-15,y0,x0-20,y0+4);

192 KEN\_draw\_straightlineY( x0-25,y0+5,x0-25,y0+9);

193 KEN\_draw\_straightlineY( x0-25,y0+15,x0-25,y0+19);

194 KEN\_draw\_straightlineY( x0-15,y0+20,x0-20,y0+24);

195

196 //redraw red and yellow boxes

197 GLCD\_SetTextColor(Red);

198 KEN\_draw\_straightlineY( x0-40+5,y0+10,x0-40,y0+14);

199 GLCD\_SetTextColor(Yellow);

200 KEN\_draw\_straightlineY( x0-35+5,y0+10,x0-35,y0+14);

201 GLCD\_SetTextColor(Red);

202 KEN\_draw\_straightlineY( x0-30+5,y0+10,x0-30,y0+14);

203 GLCD\_SetTextColor(Yellow);

204 KEN\_draw\_straightlineY( x0-25+5,y0+10,x0-25,y0+14);

205 GLCD\_SetTextColor(Red);

206 KEN\_draw\_straightlineY( x0-20+5,y0+10,x0-20,y0+14);

207 GLCD\_SetTextColor(Yellow);

208 KEN\_draw\_straightlineY( x0-15+5,y0+10,x0-15,y0+14);

209 GLCD\_SetTextColor(Red);

210 KEN\_draw\_straightlineY( x0-10+5,y0+10,x0-10,y0+14);

211 GLCD\_SetTextColor(Yellow);

212 KEN\_draw\_straightlineY( x0-5+5,y0+10,x0-5,y0+14);

213 GLCD\_SetTextColor(Red);

214 KEN\_draw\_straightlineY( x0+5,y0+10,x0,y0+14);

215 GLCD\_SetTextColor(Yellow);

216 KEN\_draw\_straightlineY( x0+4+5,y0+10,x0+4,y0+14);

217 GLCD\_SetTextColor(Red);

218 KEN\_draw\_straightlineY( x0+9+5,y0+10,x0+9,y0+14);

219 GLCD\_SetTextColor(Yellow);

220 KEN\_draw\_straightlineY( x0+14+5,y0+10,x0+14,y0+14);

221 GLCD\_SetTextColor(Red);

222 KEN\_draw\_straightlineY( x0+19+5,y0+10,x0+19,y0+14);

223 GLCD\_SetTextColor(Yellow);

224 KEN\_draw\_straightlineY( x0+24+5,y0+10,x0+24,y0+14);

225

226 x0--;

227 k++;

228 }

229 }

230

233

234 **void** KEN\_draw\_grenade( **int** x0, **int** y0){

235 **int** k ;

236

237 **while** ( k < 10){

238 KEN\_draw\_straightlineX( x0 - 7, y0+19, x0 + 7, y0+19); // straight line for base of

239 KEN\_draw\_straightlineX( x0 - 1, y0+9, x0 + 1, y0+9); // staright line for stem of

240 k++;

241 y0--;

242 }

243 }

244

245 **void** KEN\_draw\_grenadeM( **int** x0, **int** y0){

246 **int** k ;

247 **while** ( k < 5){

248 GLCD\_SetTextColor(Black);

249 KEN\_draw\_straightlineX( x0 - 7, y0+19, x0 + 7, y0+19); // straight line for base of

250 GLCD\_SetTextColor(Magenta);

251 KEN\_draw\_straightlineX( x0 - 1, y0+1, x0 + 1, y0+1); // New stem drawn

252 KEN\_draw\_straightlineX( x0 - 7, y0+9, x0 + 7, y0+9); // New base bottom drawn

253 k++;

254 y0--;

255 }

256 }