

```

1  //////////////////////////////////////
2  //CODI PROGRAMA WASPMOTE FINESTRA////////////////////////////////////
3  //Created By Rubén Almansa for EXIT Research group////////////////////////////////
4  //////////////////////////////////////
5  #include <WaspSensorEvent_v20.h>
6  #include <WaspXBeeZB.h>
7  #include <WaspFrame.h>
8  #include <WaspFrameConstants.h>
9
10 // Adreça Meshlium////////////////////////////////
11 char RX_ADDRESS[] = "0013A200408134BA";
12 //////////////////////////////////
13
14 //DEFINICIÓ VARIABLES COMUNS////////////////////////////////
15 uint8_t error,error2;
16 char* macHigh=""          ";
17 char* macLow=""          ";
18 timestamp_t time; // guarda el timestsmp del Meshlium
19 unsigned long epoch; // fa un break del timestamp i ho guarda en aquesta variable
20 int Tx; // registre el flag del frame enviat.
21 int noSend=0; //comptador d'enviaments erronis
22 int errorRTC=0; //registre el flag de la RTC ok.
23 int SumErrorRTC=0; // comptador d'errors de la sincronització deRTC
24 //////////////////////////////////
25
26 //VARIABLES FINESTRA////////////////////////////////
27 float temp;
28 int bat;
29 /* El C1 c2 c3 c4 són els contactes de finestra*/
30 int C1;
31 int C2;
32 int C3;
33 float C31;
34 int C4;
35 float LDR;
36 float LDR2;
37 int PIR=0;
38 int ini=1;
39 int rebo=0;
40 //////////////////////////////////
41 //PROGRAMA D'INICI////////////////////////////////
42 void setup()
43 {
44 /*ACTIVA la RTC i el port USB*/
45
46   RTC.ON();
47   delay(2000);
48   USB.ON();
49   delay(200);
50 /*S'activa la placa de sensor d'esdeveniments*/
51   SensorEventv20.ON();
52   delay(3000);
53   //Activa les interrupcions externes
54   SensorEventv20.setThreshold(SENS_SOCKET7,1);
55   enableInterrupts(SENS_INT);
56
57   IniXBee();
58   checkNetworkParams();
59   ResetNetwork();
60   RTCMeshlium();
61   // Alarma que s'ativa quan esta a punt d'iniciar l'enviament
62   RTC.setAlarm1("00:00:08:00",RTC_OFFSET,RTC_ALM1_MODE4); //
63   delay(1000);
64 }
65
66 //////////////////////////////////
67 //PROGRAMA PRINCIPAL////////////////////////////////
68
69 void loop()
70 {
71   // interrupció externa
72   if ((SensorEventv20.intFlag & SENS_SOCKET7))
73

```

```

74     {
75         intFlag &= ~(SENS_INT);
76         intFlag &= ~(SENS_SOCKET7);
77         USB.println("detecto");
78         PIR=PIR+1; // suma el trafic de persones
79     }
80     // despres de l'alarma s'activa aquest if que llegeix les dades i espera les
ordres d'enviar
81     if(intFlag & RTC_INT)
82     {
83         disableInterrupts(SENS_INT);
84         intFlag &= ~(RTC_INT);
85         RTC.clearAlarmFlag();
86         delay(1000);
87         LlegirDades();
88         while (Tx==0){
89             epoch=RTC.getEpochTime();
90             RTC.breakTimeAbsolute( epoch, &time);
91             USB.println(RTC.getTime());
92             if(((time.minute == 00) || (time.minute == 10) || (time.minute == 20) || (time.minute
== 30) || (time.minute == 40) || (time.minute == 50))&&Tx==0){EnviarDades();}
93             delay(5000);
94         }
95         Tx=0;
96         ini=0;
97         USB.println("enviat");
98         delay(500);
99         PWR.reboot();
100     }
101     //normalment dorm fins que arriba l'hora d'adquirir o la interrupció externa
102     enableInterrupts(SENS_INT);
103     SensorEventv20.attachInt();
104     USB.println("DORMO");
105     PWR.sleep(SOCKET0_OFF);
106     RTC.ON();
107     delay(200);
108     SensorEventv20.detachInt();
109     SensorEventv20.loadInt();
110
111 }
112 ///////////////////////////////////////////////////
113 /*FUNCIO QUE LLEGEIX LES DADES */
114
115 void LlegirDades(){
116
117     temp = SensorEventv20.readValue(SENS_SOCKET5, SENS_TEMPERATURE);
118     bat=PWR.getBatteryLevel();
119     LDR=SensorEventv20.readValue(SENS_SOCKET1); // esta linealitzada amb una resistència.
120     delay(500);
121     // equació de la linealització
122     LDR2=53.125*log(LDR)+35,679;
123     if(LDR2<0.5){LDR2=1;}
124     C1= SensorEventv20.readValue(SENS_SOCKET2);
125     C2= SensorEventv20.readValue(SENS_SOCKET3);
126     // el c31 és perquè l'entrada es analogica i llegeix el contacte de finestra
127     C31= SensorEventv20.readValue(SENS_SOCKET6);
128     C4= SensorEventv20.readValue(SENS_SOCKET8);
129
130     delay(100);
131
132     if (C1>1){C1=1;delay(100);}else {C1=0;}
133     if (C2>1){C2=1; delay(100);}else {C2=0;}
134     if (C31>3.1){C3=1;delay(100);}else {C3=0;}
135     if (C4>1){C4=1;delay(100); }else {C4=0;}
136
137 }
138
139 ///////////////////////////////////////////////////
140 //*****AQUESTA FUNCIO S'ENCARREGA D'ENVIAR LES DADES*//////////
141
142 void EnviarDades(){
143
144     while (Tx==0){

```

```

145
146     frame.setID( macLow ); // és el identificador del WASPMOTE
147     frame.createFrame(ASCII); // indicador per iniciar el frame.
148     //s'afegeix les variables de cada waspmote que s'enviaràn
149     frame.addSensor(SENSOR_STR, temp,LDR2,C1,C2,C3,C4,PIR,bat);
150     delay(100);
151     // envia el paquet
152     error = xbeeZB.send( RX_ADDRESS, frame.buffer, frame.length );
153     if( error == 0 ) // comprova l'enviament
154
155     {
156         USB.println(F("send ok"));
157
158         // blink green LED
159         Utils.blinkGreenLED();
160         noSend=0;
161         Tx=1; // Variable per confirmar qu l'envio ha estat correcte
162         delay(60000);
163     }
164     else
165     {
166         USB.println(F("send error"));
167
168         if (noSend>10){PWR.reboot();}
169         noSend=noSend+1;
170     }
171 }
172 }
173
174 //////////////////////////////////////////////////
175
176 //***** FUNCIO QUE CONNECTA AMB MESHLIUM PER OBTENIR EL TIMESTAMP*////////
177
178 void RTCMeshlium()
179
180 {
181     while (errorRTC ==0){
182         error2 = xbeeZB.setRTCfromMeshlium(RX_ADDRESS);
183         // check flag si la RTC s'ha sincronitzat
184         if( error2 == 0 )
185         {
186             USB.print(F("SET RTC ok. "));
187             errorRTC=1;
188             USB.println(RTC.getTime());
189         }
190         else
191         {
192             USB.print(F("SET RTC error. "));
193             errorRTC=0;
194             SumErrorRTC=SumErrorRTC+1;
195             if (SumErrorRTC>10){PWR.reboot();} // si es produeixen molts errors reseteja el
            sistema.
196         }
197     }
198 }
199
200 //*****FUNCIO ENCARREGADA D'INICIAR ELS XBEE I OBTENIR LA MAC*////////
201
202 void IniXBee()
203
204 {
205     xbeeZB.ON();
206     delay(3000);
207     xbeeZB.getOwnMacLow(); // obté la MAC LOW
208     xbeeZB.getOwnMacHigh(); // obté la MAC HIGH
209     // Guarda les MAC
210     Utils.hex2str(xbeeZB.sourceMacHigh,macHigh,4);
211     Utils.hex2str(xbeeZB.sourceMacLow,macLow,4);
212 }
213
214 //////////////////////////////////////////////////
215 //*****COMPROVA ELS PARAMETRES DE LA XARXA XBEE*////////
216 void checkNetworkParams()

```

```

217 {
218     // Obté la identificació de xarxa
219     xbeeZB.getOperating64PAN();
220
221     // comprova la associació amb meshlium
222     xbeeZB.getAssociationIndication();
223
224     while( xbeeZB.associationIndication != 0 )
225     {
226         delay(2000);
227
228
229         xbeeZB.getOperating64PAN();
230
231         USB.print(F("operating 64-b PAN ID: "));
232         USB.printHex(xbeeZB.operating64PAN[0]);
233         USB.printHex(xbeeZB.operating64PAN[1]);
234         USB.printHex(xbeeZB.operating64PAN[2]);
235         USB.printHex(xbeeZB.operating64PAN[3]);
236         USB.printHex(xbeeZB.operating64PAN[4]);
237         USB.printHex(xbeeZB.operating64PAN[5]);
238         USB.printHex(xbeeZB.operating64PAN[6]);
239         USB.printHex(xbeeZB.operating64PAN[7]);
240         USB.println();
241
242         xbeeZB.getAssociationIndication();
243     }
244
245     USB.println(F("\nJoined a network!"));
246
247
248     xbeeZB.getOperating16PAN();
249     xbeeZB.getOperating64PAN();
250     xbeeZB.getChannel();
251
252     USB.print(F("operating 16-b PAN ID: "));
253     USB.printHex(xbeeZB.operating16PAN[0]);
254     USB.printHex(xbeeZB.operating16PAN[1]);
255     USB.println();
256
257     USB.print(F("operating 64-b PAN ID: "));
258     USB.printHex(xbeeZB.operating64PAN[0]);
259     USB.printHex(xbeeZB.operating64PAN[1]);
260     USB.printHex(xbeeZB.operating64PAN[2]);
261     USB.printHex(xbeeZB.operating64PAN[3]);
262     USB.printHex(xbeeZB.operating64PAN[4]);
263     USB.printHex(xbeeZB.operating64PAN[5]);
264     USB.printHex(xbeeZB.operating64PAN[6]);
265     USB.printHex(xbeeZB.operating64PAN[7]);
266     USB.println();
267
268     USB.print(F("channel: "));
269     USB.printHex(xbeeZB.channel);
270     USB.println();
271
272 }
273 //////////////////////////////////////////////////
274 //*****RESETEJA ELS PAREMETRES DE LA XARXA*////////
275
276 void ResetNetwork(){
277
278     while((xbeeZB.channel) !=0x10){
279         xbeeZB.OFF();
280         delay(1000);
281         xbeeZB.ON();
282         delay(3000);
283         xbeeZB.resetNetwork(0); // reseteja els paràmetres de la xarxa
284         delay(1000);
285         checkNetworkParams(); // comprova aquests paràmetres.
286     }
287 }
288
289

```