

```

1  //////////////////////////////////////
2  //CODI PROGRAMA WASPMOTE CONSUM
3  GENERAL////////////////////////////////
4  //Created By Rubén Almansa for EXIT Research group////////////////////////////////
5  //////////////////////////////////////
6  #include <WaspXBeeZB.h>
7  #include <WaspFrame.h>
8  #include <WaspFrameConstants.h>
9  // es defineixen els factors de potència per a cada fase
10 #define cosFi1 0.80
11 #define cosFi2 0.72
12 #define cosFi3 0.75
13 #define V 232 // voltatge constant per cada fase
14 // Adreça Meshlium////////////////////////////////
15 char RX_ADDRESS[] = "0013A200408134BA";
16 //////////////////////////////////////
17
18 //DEFINICIÓ VARIABLES COMUNS////////////////////////////////
19 uint8_t error,error2;
20 char* macHigh="";
21 char* macLow="";
22 timestamp_t time; // guarda el timestsmp del Meshlium
23 unsigned long epoch; // fa un break del timestamp i ho guarda en aquesta variable
24 int Tx; // registre el flag del frame enviat.
25 int noSend=0; //comptador d'enviaments erronis
26 int errorRTC=0; //registre el flag de la RTC ok.
27 int SumErrorRTC=0; // comptador d'errors de la sincronització deRTC
28 //////////////////////////////////////
29
30 //VARIABLES CONSUM////////////////////////////////
31 int bat;
32 float Imig1,Imig2,Imig3; // la mitjana del corrent
33 float val,val2,val3,valt,val2t,val3t; // calcula valors parcials i totals
34 float P1,P2,P3; // potències cada fase
35 float C1,C2,C3,CT; // consums
36 float CTS; // suma del conum total
37 float h; // temps en hores dels 10 minuts
38 int i=0; // numero de mostres
39 //////////////////////////////////////
40 //PROGRAMA D'INICI////////////////////////////////
41 void setup()
42 {
43 /*ACTIVA la RTC i el port USB*/
44
45   RTC.ON();
46   delay(2000);
47   USB.ON();
48   delay(200);
49
50   // Activa l'alimentació a les plaques de consum
51   PWR.setSensorPower(SENS_5V,SENS_ON);
52
53   IniXBee();
54   checkNetworkParams();
55   ResetNetwork();
56   RTCMeshlium();
57
58 }
59
60 //////////////////////////////////////
61 //PROGRAMA PRINCIPAL////////////////////////////////
62
63 void loop()
64 {
65
66   USB.println(RTC.getTime());
67   LlegirConsum();
68   epoch=RTC.getEpochTime(); // guarda el time en un format, és necessari per la
69   comanda següent
70   RTC.breakTimeAbsolute( epoch, &time); // trenca el timestamp per obtenir el minut
71   que estem

```

```

71 //Aquest "if" s'encarrega de comprovar el minut d'enviament i a continuació
72 proscessa el càlcul
73 if(((time.minute == 00) || (time.minute == 10) || (time.minute == 20) || (time.minute
74 == 30) || (time.minute == 40) || (time.minute == 50)) && Tx == 0) {
75     bat=PWR.getBatteryLevel();
76     h=0.16;
77     USB.println(h);
78
79     Imig1=valt/i;
80     P1=V*Imig1*cosFi1;
81     C1=P1*h;
82     USB.println(Imig1);
83     USB.println(P1);
84     USB.println(C2);
85
86     Imig2=val2t/i;
87     USB.println(Imig2);
88     P2=V*Imig2*cosFi2;
89     USB.println(P2);
90     C2=P2*h;
91     USB.println(C2);
92
93     Imig3=val3t/i;
94     USB.println(Imig3);
95     P3=V*Imig3*cosFi3;
96     USB.println(P3);
97     C3=P3*h;
98     USB.println(C3);
99
100     CTS=C1+C2+C3;
101     USB.println(CTS);
102
103     EnviarDades();
104     i=0;
105     valt=0;
106     val2t=0;
107     val3t=0;
108     PWR.reboot();
109 }
110
111
112
113
114
115
116 ///////////////////////////////////////////////////
117 /*FUNCIO QUE LLEGEIX LES DADES RESTANTS AL SO */
118
119 void LlegirConsum()
120 {
121     USB.print("Value read from Analog 1: ");
122     val=analogRead(ANALOG1);
123     delay(100);
124     val=val*3.3/1024;
125     USB.println(val);
126     valt=val+valt; // acumula el valor de consum
127
128     USB.print("Value read from Analog 2: ");
129     val2=analogRead(ANALOG3);
130     delay(100);
131     val2=val2*3.3/1024;
132     USB.println(val2);
133     val2t=val2t+val2; // acumula el valor de consum
134
135     USB.print("Value read from Analog 3: ");
136     val3=analogRead(ANALOG5);
137     delay(100);
138     val3=val3*3.3/1024;
139     USB.println(val3);
140     val3t=val3t+val3; // acumula el valor de consum
141     delay(700);

```

```

142 i=i+1; // numero de mostres que s'agafen duran els 10 minuts
143
144 }
145
146
147 ///////////////////////////////////////////////////
148 //*****AQUESTA FUNCIO S'ENCARREGA D'ENVIAR LES DADES*//////////
149
150 void EnviarDades(){
151
152     while(Tx==0){
153
154         frame.setID( macLow ); // és el identificador del WASPMOTE
155         frame.createFrame(ASCII); // indicador per iniciar el frame.
156         //s'afegeix les variables de cada waspmote que s'enviaràn
157         frame.addSensor(SENSOR_STR, CTS,bat);
158         delay(100);
159         // envia el paquet
160         error = xbeeZB.send( RX_ADDRESS, frame.buffer, frame.length );
161         if( error == 0 ) // comprova l'enviament
162
163         {
164             USB.println(F("send ok"));
165
166             // blink green LED
167             Utils.blinkGreenLED();
168             noSend=0;
169             Tx=1; // Variable per confirmar qu l'envio ha estat correcte
170             delay(60000);
171         }
172         else
173         {
174             USB.println(F("send error"));
175
176             if (noSend>10){PWR.reboot();}
177             noSend=noSend+1;
178         }
179     }
180 }
181
182 ///////////////////////////////////////////////////
183
184 //***** FUNCIO QUE CONNECTA AMB MESHLIUM PER OBTENIR EL TIMESTAMP*//////////
185
186 void RTCMeshlium()
187
188 {
189     while (errorRTC ==0){
190         error2 = xbeeZB.setRTCfromMeshlium(RX_ADDRESS);
191         // check flag si la RTC s'ha sincronitzat
192         if( error2 == 0 )
193         {
194             USB.print(F("SET RTC ok. "));
195             errorRTC=1;
196             USB.println(RTC.getTime());
197         }
198         else
199         {
200             USB.print(F("SET RTC error. "));
201             errorRTC=0;
202             SumErrorRTC=SumErrorRTC+1;
203             if (SumErrorRTC>10){PWR.reboot();} // si es produeixen molts errors reseteja el
                sistema.
204         }
205     }
206 }
207
208 //*****FUNCIO ENCARREGADA D'INICIAR ELS XBEE I OBTENIR LA MAC*//////////
209
210 void IniXBee()
211
212 {
213     xbeeZB.ON();

```

```

214     delay(3000);
215     xbeeZB.getOwnMacLow(); // obté la MAC LOW
216     xbeeZB.getOwnMacHigh(); // obté la MAC HIGH
217     // Guarda les MAC
218     Utils.hex2str(xbeeZB.sourceMacHigh,macHigh,4);
219     Utils.hex2str(xbeeZB.sourceMacLow,macLow,4);
220 }
221
222 ///////////////////////////////////////////////////
223 //*****COMPROVA ELS PARAMETRES DE LA XARXA XBEE*//////////
224 void checkNetworkParams()
225 {
226     // Obté la identificació de xarxa
227     xbeeZB.getOperating64PAN();
228
229     // comprova la associació amb meshlium
230     xbeeZB.getAssociationIndication();
231
232     while( xbeeZB.associationIndication != 0 )
233     {
234         delay(2000);
235
236
237         xbeeZB.getOperating64PAN();
238
239         USB.print(F("operating 64-b PAN ID: "));
240         USB.printHex(xbeeZB.operating64PAN[0]);
241         USB.printHex(xbeeZB.operating64PAN[1]);
242         USB.printHex(xbeeZB.operating64PAN[2]);
243         USB.printHex(xbeeZB.operating64PAN[3]);
244         USB.printHex(xbeeZB.operating64PAN[4]);
245         USB.printHex(xbeeZB.operating64PAN[5]);
246         USB.printHex(xbeeZB.operating64PAN[6]);
247         USB.printHex(xbeeZB.operating64PAN[7]);
248         USB.println();
249
250         xbeeZB.getAssociationIndication();
251     }
252
253     USB.println(F("\nJoined a network!"));
254
255
256     xbeeZB.getOperating16PAN();
257     xbeeZB.getOperating64PAN();
258     xbeeZB.getChannel();
259
260     USB.print(F("operating 16-b PAN ID: "));
261     USB.printHex(xbeeZB.operating16PAN[0]);
262     USB.printHex(xbeeZB.operating16PAN[1]);
263     USB.println();
264
265     USB.print(F("operating 64-b PAN ID: "));
266     USB.printHex(xbeeZB.operating64PAN[0]);
267     USB.printHex(xbeeZB.operating64PAN[1]);
268     USB.printHex(xbeeZB.operating64PAN[2]);
269     USB.printHex(xbeeZB.operating64PAN[3]);
270     USB.printHex(xbeeZB.operating64PAN[4]);
271     USB.printHex(xbeeZB.operating64PAN[5]);
272     USB.printHex(xbeeZB.operating64PAN[6]);
273     USB.printHex(xbeeZB.operating64PAN[7]);
274     USB.println();
275
276     USB.print(F("channel: "));
277     USB.printHex(xbeeZB.channel);
278     USB.println();
279
280 }
281 ///////////////////////////////////////////////////
282 //*****RESETEJA ELS PAREMETRES DE LA XARXA*//////////
283
284 void ResetNetwork(){
285
286     while((xbeeZB.channel) !=0x10){

```

```
287     xbeeZB.OFF();
288     delay(1000);
289     xbeeZB.ON();
290     delay(3000);
291     xbeeZB.resetNetwork(0); // reseteja els paràmetres de la xarxa
292     delay(1000);
293     checkNetworkParams(); // comprova aquests paràmetres.
294 }
295 }
296
297
```