ANEXO 3

Código fuente de la tarjeta de desarrollo HELTEC

1 Declarar las librerías

Creamos un nuevo proyecto en el IDE con el nombre *config.h* en la cual definiremos los pines GPIO que usaremos en la tarjeta HELTEC, las variables que utilizamos son primordiales, las mismas serán llamadas en el código principal

```
1. #pragma once


2. #define LED_BOARD 35
3. #define DATA_DHT11 48
4. #define D_HD38 19
5. #define A_SEN0121 20
6. #define VALVULA_P 26
```

Ahora creamos un nuevo proyecto el cual nombraremos *Nodos.ino,* este será nuestro código fuente que se utilizará en el nodo del proyecto

- Empezamos declarando primero la librería Heltec

```
1. #include "LoRaWan_APP.h"
```

- Ahora declaramos las librerías externas a Heltec, la cual corresponden al sensor de lluvia, humedad, conexión wifi, conexión SPI para comunicarse con otros dispositivos, la librería de comunicación I2C la cual es utilizada para leer y enviar datos entre dispositivos I2C.

```
2. #include <SPI.h>
3. #include <Wire.h>
4. #include <Adafruit_GFX.h>
5. #include <Adafruit_SSD1306.h>
6. #include "DHT.h"
7. #include <WiFi.h>
```

- Configuramos ahora los puertos GPIO, declaramos dos funciones en base al sensor DHT11

```
8. #include <Config.h>
9. uint32_t devAddr =  ( uint32_t )0x007e6ae1;
10.#define DHTPIN DATA_DHT11
11.#define DHTTYPE DHT11
12.DHT dht(DHTPIN, DHTTYPE);
```

- Ahora configuraremos la pantalla OLED para que nos muestre los datos en la misma.

```
#define SCREEN_WIDTH 128 // Pantalla OLED ancho, en pixeles
#define SCREEN_HEIGHT 64 // Pantalla OLED largo, en pixeles
#define SCREEN_ADDRESS 0x3C // Dirección del puerto I2C para la OLED
Adafruit_SSD1306 UIdisplay(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, RST_OLED); // inicializar la OLED
```

- Ahora iniciamos con la Configuración del radio lora node chip SX1262 uno de los apartados más importantes, ya que asignaremos los parámetros del mismo.

PARAMETROS RADIO LORA SX1262	
Frecuencia	915 MHz
Potencia TX	21 dBm
Ancho de banda	125 kHz
Tasa de información para corrección de errores	4/5
Longitud del preámbulo para flujo de datos	8
Tiempo de espera de simbolo (SIMBOL TIMEOUT)	0

Tabla 8. Parámetros del radio LoRa SX1262

```
#define RF_FREQUENCY                                915000000 // Hz
#define TX_OUTPUT_POWER                             21        // dBm
#define LORA_BANDWIDTH                              0         // [0: 125
#define LORA_SPREADING_FACTOR                       7         // [SF7..SF12]
#define LORA_CODINGRATE                             1         // [1: 4/5,
#define LORA_PREAMBLE_LENGTH                        8         // Same for Tx and Rx
#define LORA_SYMBOL_TIMEOUT                         0         // Symbols
#define LORA_FIX_LENGTH_PAYLOAD_ON                  false
#define LORA_IQ_INVERSION_ON                        false
#define RX_TIMEOUT_VALUE                            1000
```

```
#define BUFFER_SIZE                                    30 // Define the
payload size here

char txpacket[BUFFER_SIZE];
char rxpacket[BUFFER_SIZE];

double txNumber;
int16_t rssi,rxSize;
bool lora_idle=true;

static RadioEvents_t RadioEvents;
void OnTxDone( void );
void OnTxTimeout( void );
```

Ahora declaramos en una variable, los parámetros que se visualizaran en la pantalla OLED permanentemente:

```
void ShowData(String sensor_1, String sensor_2, String sensor_3,String
mac,String valvula)
{
UIdisplay.clearDisplay();
UIdisplay.setTextSize(1);
UIdisplay.setTextColor(SSD1306_WHITE);
UIdisplay.setCursor(0, 0);
UIdisplay.println(F("SENSORES"));
UIdisplay.println(("MAC="+mac));
UIdisplay.println(F("_____________________"));
UIdisplay.println(("TEMP:     "+sensor_1));
UIdisplay.println(F("----------------------"));
UIdisplay.println(("HUMEDAD:  "+sensor_2));
UIdisplay.println(F("----------------------"));
UIdisplay.println(("LLUVIA: "+sensor_3+" | VAL:"+valvula));
UIdisplay.display();
}
```

Declaramos la variable de setup

```
void setup() {
```

- Configuración de la Interfaz de depuración serial

```
Serial.begin(115200);
```

- LED Indicador de envió de mensajes

```
pinMode(LED_BOARD,OUTPUT);
```

- Inicia el sensor DHT11

```
dht.begin();
```

- Actuador Valvula

```
pinMode(VALVULA_P,OUTPUT);
digitalWrite(VALVULA_P,0);
```

- Configuración del tipo de board de la marca HELTEC

```
Mcu.begin(HELTEC_BOARD, SLOW_CLK_TPYE);
```

- Inicialización de eventos y configuración del Radio Lora

```
  RadioEvents.TxDone = OnTxDone;
  RadioEvents.TxTimeout = OnTxTimeout;
  Radio.Init( &RadioEvents );
  Radio.SetChannel( RF_FREQUENCY );
  Radio.SetTxConfig( MODEM_LORA, TX_OUTPUT_POWER, 0, LORA_BANDWIDTH,
  LORA_SPREADING_FACTOR, LORA_CODINGRATE,LORA_PREAMBLE_LENGTH,
LORA_FIX_LENGTH_PAYLOAD_ON, true, 0, 0, LORA_IQ_INVERSION_ON, 3000 );

```

- Configuración de la pantalla OLED

```
  Wire.begin(SDA_OLED, SCL_OLED); // Configurar los GPIO para la
Comunicación I2C

  if(!UIdisplay.begin(SSD1306_SWITCHCAPVCC, SCREEN_ADDRESS)) {
    Serial.println(F("SSD1306 allocation failed"));
    for(;;); // Don't proceed, loop forever
  }
```

- Ahora declaramos las funciones con nuestras variables asociadas a los sensores a utilizar.

```
int HD_38_P;
float DHT_TEMP;
int A_LLUVIA;
String E_LLUVIA;
int EBool_LLUVIA;
int VALVULA = 0;
String E_VALVULA;
```

Creamos la función Medición, la cual contiene los parámetros para la captura de datos.

```
79. void Medicion()
80.{
81. HD_38_P = analogRead(D_HD38);
82. HD_38_P = constrain(HD_38_P,0,4095);
83. HD_38_P = map(HD_38_P,0,4095,100,0);
84. Serial.print(F("Humedad del suelo: "));
85. Serial.println(HD_38_P);
86. DHT_TEMP = dht.readTemperature();
87. Serial.print(F("Temperature: "));
88. Serial.println(DHT_TEMP);
89. A_LLUVIA = analogRead(A_SEN0121);
90. Serial.print(F("lluvia: "));
91. Serial.println(A_LLUVIA);
92. E_LLUVIA;
93. EBool_LLUVIA;
94. if(A_LLUVIA > 3000)
95. {
96. E_LLUVIA = "SI";
97. EBool_LLUVIA = 1;
98. }
99. else
100.      {
101.      E_LLUVIA = "NO";
102.      EBool_LLUVIA = 0;
103.      }
104.      }
```

- Creamos la función de actuador, esta nos permitirá activar o desactivar la válvula cuando el índice de humedad del cultivo, se encuentre sobre un nivel bajo, en nuestro proyecto, se ha asignado el 40% el cual será un valor ideal para mantener la humedad del suelo y que el tiempo de riego sea más corto. Está condición está asociada al sensor de lluvia, ya que si el mismo detecta que hay presencia de lluvias y la humedad está por debajo del 40% no activa la valvula, pero si no hay presencia de lluvia, la activara.

```
105.      void Actuador ()
106.      {
107.      if((0 <= HD_38_P && HD_38_P <= 40) && (EBool_LLUVIA == 0) ) //
   0% < humedad < 40% y no hay lluvia entonces se riega
108.      {
109.      VALVULA=1;
```

```
110.      E_VALVULA= "ON";
111.      // activa la valvula para regar el suelo
112.      digitalWrite (VALVULA_P, VALVULA);
113.      }
114.      else
115.      {
116.      VALVULA=0;
117.      E_VALVULA= "OFF";
118.      digitalWrite (VALVULA_P, VALVULA);
119.      }
120.      }
```

Este fragmento del código será también primordial, ya que será parte de la seguridad del enlace, para no permitir que otra tarjeta no registrada sea incluida. Para esto tomaremos la dirección MAC de nuestra tarjeta Heltec, esta será nuestra llave de autenticación con nuestro Gateway. Solo las tarjetas que cuente con este código y con la función habilitada, serán las únicas autorizadas para conectarse con nuestro Gateway. Esta MAC la obtendremos en base al chip wifi que contiene la tarjeta, en la cual será almacenada en una cadena de caracteres.

```
121.      String obtenerMAC() {
```

- Declarar un arreglo para almacenar la dirección MAC

```
122.      byte mac[6];
```

- Obtener la dirección MAC de la interfaz Wi-Fi

```
123.      WiFi.macAddress(mac);
```

- Crear una cadena para almacenar la MAC en formato legible

```
124.      String macString = "";
125.      for (int i = 0; i < 6; i++) {
```

- Convertir cada byte de la MAC a una cadena hexadecimal de dos caracteres

```
126.      macString += String(mac[i], HEX);
```

- Agregar dos puntos entre cada parte de la MAC, excepto al final

```
127.            if (i < 5) {
128.                macString += ":";
129.            }
130.        }
```

```
131.        return macString;
132.      }
```

- Creamos una función tipo loop la cual será la encargada de realizar un recorrido por las variables declaradas asociadas a los sensores y nos recopilará los datos, se mostrará en la pantalla OLED y a su vez los enviará a nuestra tarjeta Gateway.

```
133.      void loop() {
134.      Medicion();
135.      Actuador();
136.      String MAC = obtenerMAC();
137.      ShowData(String(DHT_TEMP)+"C",String(HD_38_P)+"%",E_LLUVIA,MAC,E
   _VALVULA);
138.      TXSender(MAC+"@"+String(DHT_TEMP)+","+String(HD_38_P)+","+String
   (EBool_LLUVIA)+","+String(VALVULA));
```

NOTA: Este apartado del código, se recomienda usar para validar la comunicación con nuestra tarjeta Gateway cuando no hay presencia de ningún sensor instalado. Su función es la de enviar datos fijos, simulando que esta obtieniend los datos.

```
139.      //ShowData("33.50C",String(HD_38_P)+"%",E_LLUVIA,MAC,E_VALVULA);
140.      //TXSender(MAC+"@"+String(33.50)+","+String(HD_38_P)+","+String(
   EBool_LLUVIA)+","+String(VALVULA));
141.      }
```

Por ultimo declararemos la función TXSender, la cual es la encargada de enviar los datos obtenidos atraves de Lora. Este apartado del código es propio de la tarjeta y se recomienda no modificarlo.

```
142.      void TXSender(String datos)
143.      {
144.      if(lora_idle == true)
145.      {
146.      digitalWrite(LED_BOARD,!lora_idle);
147.      delay(5000);
148.      sprintf(txpacket,datos.c_str(),txNumber);  //inicia un paquete
```

```
149.      Serial.printf("\r\nsending packet \"%s\" , length
   %d\r\n",txpacket, strlen(txpacket));

150.      Radio.Send( (uint8_t *)txpacket, strlen(txpacket) ); //enviando
   paquete
151.      lora_idle = false;
152.      digitalWrite(LED_BOARD,!lora_idle);
153.      }
154.      Radio.IrqProcess( );
155.      }

156.      void OnTxDone( void )
157.      {
158.      Serial.println("TX done......");
159.      lora_idle = true;
160.      }

161.      void OnTxTimeout( void )
162.      {
163.      Radio.Sleep( );
164.      Serial.println("TX Timeout......");
165.      lora_idle = true;
166.      }
```

Este sería el seudocódigo que se cargara a nuestra tarjeta de desarrollo Heltec para la estación de se va a implementar.