**ภาคผนวก ข**

โปรแกรมอาดูโน ไอดีอี บอร์ดโนห์ด 32 ไลท์

โปรแกรมอาดูโน ไอดีอี บอร์ดโนห์ด 32 ไลท์

|  |  |
| --- | --- |
| 1 | #include <SPI.h> |
| 2 | #include <Wire.h> |
| 3 | #include <Adafruit\_GFX.h> |
| 4 | #include <Adafruit\_SSD1306.h> |
| 5 |  |
| 6 | #include <WiFi.h> |
| 7 | #include <FirebaseESP32.h> |
| 8 | #include <TridentTD\_LineNotify.h> |
| 9 |  |
| 10 | #define WIFI\_SSID "HUB01-HOME\_2.4GHz" |
| 11 | #define WIFI\_PASSWORD "0806550413" |
| 12 | #define FIREBASE\_HOST "https://dust-pm-rmutr.firebaseio.com/" |
| 13 | #define FIREBASE\_AUTH "U8XnaY9JGoGF9ab4OKw1c19iVzE5ItMAYdxHtHeh" |
| 14 | #define LINE\_TOKEN "N68bbKRxsly66VbsBHp1QVfAon8LgqUFkgy4RsmDocm" |
| 15 |  |
| 16 | FirebaseData firebaseData\_Rx; |
| 17 | FirebaseData firebaseData\_Tx; |
| 18 |  |
| 19 | String data\_path = "/Dust\_station\_01/data"; |
| 20 | String log\_path = "/Dust\_station\_01/Log"; |
| 21 |  |
| 22 | #define PIN\_RX\_0 3 |
| 23 | #define PIN\_TX\_0 1 |
| 24 | #define PIN\_RX\_1 16 |
| 25 | #define PIN\_TX\_1 17 |
| 26 |  |
| 27 | char str\_buff[200] = {0}; |
| 28 | unsigned long t\_old = 0; |
| 29 | int tmr\_cnt = 0; |
| 30 |  |
| 31 | int status\_process = 0; |
| 32 | int status\_process\_mon = 0; |
| 33 | int error = 0; |
| 34 |  |
| 35 | int dix = 0; |
| 36 | unsigned char din; |
| 37 | unsigned char dbuff[25] = {0}; |
| 38 |  |
| 39 | unsigned int pm\_1 = 0; |
| 40 | unsigned int pm\_25 = 0; |
| 41 | unsigned int pm\_10 = 0; |
| 42 |  |
| 43 |  |
| 44 | //------------------------------------------------------ |
| 45 | #define SCREEN\_WIDTH 128 |
| 46 | #define SCREEN\_HEIGHT 64 |
| 47 |  |
| 48 | #define OLED\_RESET 4 |
| 49 | Adafruit\_SSD1306 display(SCREEN\_WIDTH, SCREEN\_HEIGHT, &Wire, OLED\_RESET); |
| 50 |  |
| 51 | #define NUMFLAKES 10 |
| 52 |  |
| 53 | #define LOGO\_HEIGHT 16 |
| 54 | #define LOGO\_WIDTH 16 |
| 55 |  |
| 56 | //------------------------------------------------------ |
| 57 | #include <WiFiUdp.h> |
| 58 |  |
| 59 | unsigned int udp\_localPort = 1000; |
| 60 |  |
| 61 | IPAddress timeServerIP; // time.nist.gov NTP server address |
| 62 | const char\* ntpServerName = "pool.ntp.org"; |
| 63 | const int NTP\_PACKET\_SIZE = 48; // NTP time stamp is in the first 48 bytes of the message |
| 64 | byte packetBuffer[ NTP\_PACKET\_SIZE]; // buffer to hold incoming and outgoing packets |
| 65 |  |
| 66 | WiFiUDP udp; |
| 67 | int ntp\_Y, ntp\_M, ntp\_D, ntp\_H, ntp\_m, ntp\_S, ntp\_S\_10; |
| 68 |  |
| 69 | unsigned long sendNTPpacket(IPAddress& address); |
| 70 | void NTP\_Update(); |
| 71 |  |
| 72 |  |
| 73 | //------------------------------------------------------ |
| 74 | void StreamCallback(StreamData data); |
| 75 | void StreamCallback\_Timeout(bool ptimeout); |
| 76 | int Log\_Add(String pdatetime, double pdetail\_0, double pdetail\_1, double pdetail\_2); |
| 77 |  |
| 78 |  |
| 79 |  |
| 80 | //------------------------------------------------------ |
| 81 | void setup() { |
| 82 | Serial.begin (9600, SERIAL\_8N1, PIN\_RX\_0, PIN\_TX\_0); |
| 83 | Serial1.begin(9600, SERIAL\_8N1, PIN\_RX\_1, PIN\_TX\_1); |
| 84 | LINE.setToken(LINE\_TOKEN); |
| 85 | Serial.println(LINE.getVersion()); |
| 86 | LINE.notify("CHECK LINE NOTIFY"); |
| 87 | if (!display.begin(SSD1306\_SWITCHCAPVCC, 0x3C)) { |
| 88 | Serial.println(F("SSD1306 allocation failed")); |
| 89 | for(;;); // Don't proceed, loop forever |
| 90 | } |
| 91 |  |
| 92 | void Disp\_Info(); delay(2000); |
| 93 |  |
| 94 | //------------------------------------------------------ |
| 95 | WiFi.begin(WIFI\_SSID, WIFI\_PASSWORD); |
| 96 |  |
| 97 | Serial.print("\nConnecting to Wi-Fi"); |
| 98 | while (WiFi.status() != WL\_CONNECTED) { |
| 99 | Serial.print("."); |
| 100 | delay(200); |
| 101 | } |
| 102 |  |
| 103 | Serial.print("\nConnected with IP: "); |
| 104 | Serial.println(WiFi.localIP()); |
| 105 |  |
| 106 | //------------------------------------------------------ |
| 107 | udp.begin(udp\_localPort); |
| 108 | ntp\_Y = 0; ntp\_M = 0; ntp\_D = 0; |
| 109 | ntp\_H = 0; ntp\_m = 0; ntp\_S = 0; |
| 110 | ntp\_S\_10 = 0; |
| 111 |  |
| 112 | //------------------------------------------------------ |
| 113 | Firebase.begin(FIREBASE\_HOST, FIREBASE\_AUTH); |
| 114 | Firebase.reconnectWiFi(true); |
| 115 |  |
| 116 | if (!Firebase.beginStream(firebaseData\_Rx, data\_path)) { |
| 117 | Serial.println("Can't begin stream connection..."); |
| 118 | Serial.println("REASON: " + firebaseData\_Rx.errorReason()); |
| 119 | } |
| 120 |  |
| 121 | Firebase.setStreamCallback(firebaseData\_Rx, StreamCallback, StreamCallback\_Timeout); |
| 122 |  |
| 123 | //------------------------------------------------------ |
| 124 | pm\_1 = 0; |
| 125 | pm\_25 = 0; |
| 126 | pm\_10 = 0; |
| 127 |  |
| 128 | } |
| 129 |  |
| 130 | void loop() { |
| 131 |  |
| 132 | if (pm\_25 > 50) { |
| 133 | String LineText; |
| 134 | String string1 = "ค่าฝุ่นละอองเกินมาตรฐาน "; |
| 135 | String string2 = " ug/m3"; |
| 136 | LineText = string1 + pm\_25 + string2; |
| 137 | Serial.print("Line "); |
| 138 | Serial.println(LineText); |
| 139 | LINE.notify(LineText); |
| 140 | delay(10000); |
| 141 | } |
| 142 |  |
| 143 | while (Serial1.available() > 0) { |
| 144 | din = Serial1.read(); |
| 145 |  |
| 146 | if ((din == 0x42) || (din == 0x4D)) { |
| 147 | if (din == 0x42) { dix = 0; } |
| 148 | if (din == 0x4D) { dix = 1; } |
| 149 | } else { |
| 150 | if (dix < 24) { dix++; } |
| 151 | } |
| 152 |  |
| 153 | dbuff[dix] = din; |
| 154 |  |
| 155 | if (dix == 23) { |
| 156 | pm\_1 = ((0x10 \* (unsigned int)dbuff[4]) + (unsigned int)dbuff[5]); |
| 157 | pm\_25 = ((0x10 \* (unsigned int)dbuff[6]) + (unsigned int)dbuff[7]); |
| 158 | pm\_10 = ((0x10 \* (unsigned int)dbuff[8]) + (unsigned int)dbuff[9]); |
| 159 | } |
| 160 | } |
| 161 |  |
| 162 | if (tmr\_cnt == 0) { |
| 163 | NTP\_Update(); |
| 164 |  |
| 165 | display.clearDisplay(); |
| 166 |  |
| 167 | display.setTextSize(1.5); |
| 168 | display.setTextColor(WHITE); |
| 169 |  |
| 170 | display.setCursor(60, 10); |
| 171 | sprintf(str\_buff, "%4d", pm\_1); |
| 172 | display.print(str\_buff); |
| 173 |  |
| 174 | display.setCursor(0, 10); |
| 175 | sprintf(str\_buff, "PM 1.0 = %4d ug/m3 ", pm\_1); |
| 176 | display.print(str\_buff); |
| 177 |  |
| 178 | display.setCursor(0, 25); |
| 179 | sprintf(str\_buff, "PM 2.5 = %4d ug/m3 ", pm\_25); |
| 180 | display.print(str\_buff); |
| 181 |  |
| 182 | display.setCursor(0, 40); |
| 183 | sprintf(str\_buff, "PM 10.0 = %4d ug/m3 ", pm\_10); |
| 184 | display.print(str\_buff); |
| 185 |  |
| 186 | display.display(); |
| 187 |  |
| 188 | sprintf(str\_buff, "%04d-%02d-%02d %02d:%02d:%02d", ntp\_Y, ntp\_M, ntp\_D, ntp\_H, ntp\_m, ntp\_S); |
| 189 | Serial.print(str\_buff); |
| 190 |  |
| 191 | String bdatetime\_str = str\_buff; |
| 192 |  |
| 193 | sprintf(str\_buff, " | %4d | %4d | %4d | ", pm\_1, pm\_25, pm\_10); |
| 194 | Serial.print(str\_buff); |
| 195 |  |
| 196 | if (ntp\_S\_10 != (ntp\_S / 10)) { |
| 197 | ntp\_S\_10 = (ntp\_S / 10); |
| 198 | error = Log\_Add(bdatetime\_str, (double)pm\_1, (double)pm\_25, (double)pm\_10); |
| 199 | Serial.print("Updated "); |
| 200 | } |
| 201 |  |
| 202 | Serial.println(); |
| 203 | } |
| 204 |  |
| 205 | while ((micros() - t\_old) < 100000L); t\_old = micros(); |
| 206 | tmr\_cnt++; if (tmr\_cnt >= 20) { tmr\_cnt = 0; } |
| 207 | } |
| 208 |  |
| 209 |  |
| 210 | //------------------------------------------------------ |
| 211 | void Disp\_Info() { |
| 212 | display.clearDisplay(); |
| 213 |  |
| 214 | display.setTextSize(1); // Draw 2X-scale text |
| 215 | display.setTextColor(WHITE); |
| 216 |  |
| 217 | display.setCursor(40, 5); |
| 218 | display.println(F("PROJECT")); |
| 219 |  |
| 220 | display.setCursor(53, 20); |
| 221 | display.println(F("of")); |
| 222 |  |
| 223 | display.setCursor(30, 35); |
| 224 | display.println(F("THE DUST")); |
| 225 |  |
| 226 | display.setCursor(25, 50); |
| 227 | display.println(F("")); |
| 228 |  |
| 229 | display.display(); |
| 230 | } |
| 231 |  |
| 232 | //------------------------------------------------------ |
| 233 | unsigned long sendNTPpacket(IPAddress& address) { |
| 234 | // set all bytes in the buffer to 0 |
| 235 | memset(packetBuffer, 0, NTP\_PACKET\_SIZE); |
| 236 | // Initialize values needed to form NTP request |
| 237 | // (see URL above for details on the packets) |
| 238 | packetBuffer[0] = 0b11100011; // LI, Version, Mode |
| 239 | packetBuffer[1] = 0; // Stratum, or type of clock |
| 240 | packetBuffer[2] = 6; // Polling Interval |
| 241 | packetBuffer[3] = 0xEC; // Peer Clock Precision |
| 242 | // 8 bytes of zero for Root Delay & Root Dispersion |
| 243 | packetBuffer[12] = 49; |
| 244 | packetBuffer[13] = 0x4E; |
| 245 | packetBuffer[14] = 49; |
| 246 | packetBuffer[15] = 52; |
| 247 |  |
| 248 | // all NTP fields have been given values, now |
| 249 | // you can send a packet requesting a timestamp: |
| 250 | udp.beginPacket(address, 123); //NTP requests are to port 123 |
| 251 | udp.write(packetBuffer, NTP\_PACKET\_SIZE); |
| 252 | udp.endPacket(); |
| 253 | } |
| 254 |  |
| 255 | void NTP\_Update() { |
| 256 | int bbytes = udp.parsePacket(); |
| 257 |  |
| 258 | if (bbytes > 0) { |
| 259 | // Do nothing. |
| 260 | } else { |
| 261 | udp.read(packetBuffer, NTP\_PACKET\_SIZE); |
| 262 |  |
| 263 | unsigned long highWord = word(packetBuffer[40], packetBuffer[41]); |
| 264 | unsigned long lowWord = word(packetBuffer[42], packetBuffer[43]); |
| 265 | unsigned long secsSince1900 = highWord << 16 | lowWord; |
| 266 |  |
| 267 | if (secsSince1900 > 0) { |
| 268 | unsigned long epoch = secsSince1900; |
| 269 | unsigned long gmt\_offset = +7UL; |
| 270 | epoch = (epoch + (gmt\_offset \* 3600UL)); |
| 271 |  |
| 272 | ntp\_Y = (int)2021; |
| 273 | ntp\_M = (int)07;//(epoch / 86400 / 3500); |
| 274 | ntp\_D = (int)29; |
| 275 |  |
| 276 | ntp\_H = (int)((epoch % 86400UL) / 3600UL); |
| 277 | ntp\_m = (int)((epoch % 3600UL) / 60UL); |
| 278 | ntp\_S = (int) (epoch % 60UL); |
| 279 | } |
| 280 | } |
| 281 |  |
| 282 | WiFi.hostByName(ntpServerName, timeServerIP); |
| 283 | sendNTPpacket(timeServerIP); |
| 284 | } |
| 285 |  |
| 286 | //------------------------------------------------------ |
| 287 | void StreamCallback(StreamData data) { |
| 288 | delay(60000); |
| 289 | } |
| 290 |  |
| 291 | void StreamCallback\_Timeout(bool ptimeout) { |
| 292 | if (ptimeout != 0) { Serial.println("Stream timeout, resume streaming..."); } |
| 293 | } |
| 294 |  |
| 295 | int Log\_Add(String pdatetime, double pdetail\_0, double pdetail\_1, double pdetail\_2) { |
| 296 | int lresult\_int = 0; |
| 297 |  |
| 298 | { |
| 299 | FirebaseJson json; |
| 300 | json.add("datetime", pdatetime).add("PM\_1", pdetail\_0).add("PM\_2p5", pdetail\_1).add("PM\_10", pdetail\_2); |
| 301 | if (Firebase.setJSON(firebaseData\_Tx, log\_path + "/" + pdatetime, json) != 0) { |
| 302 | // Do nothing. |
| 303 | } else { |
| 304 | lresult\_int += 1; |
| 305 | Serial.println("Tx FAILED: REASON: " + firebaseData\_Tx.errorReason()); |
| 306 | } |
| 307 | } |
| 308 |  |
| 309 | { |
| 310 | FirebaseJson json; |
| 311 | json.add("datetime", pdatetime).add("PM\_1", pdetail\_0); |
| 312 | if (Firebase.setJSON(firebaseData\_Tx, data\_path + "/PM\_1", json) != 0) { |
| 313 | // Do nothing. |
| 314 | } else { |
| 315 | lresult\_int += 2; |
| 316 | Serial.println("Tx FAILED: REASON: " + firebaseData\_Tx.errorReason()); |
| 317 | } |
|  | } |
| 319 |  |
| 320 | { |
| 321 | FirebaseJson json; |
| 322 | json.add("datetime", pdatetime).add("PM\_2p5", pdetail\_1); |
| 323 | if (Firebase.setJSON(firebaseData\_Tx, data\_path + "/PM\_2p5", json) != 0) { |
| 324 | // Do nothing. |
| 325 | } else { |
| 326 | lresult\_int += 2; |
| 327 | Serial.println("Tx FAILED: REASON: " + firebaseData\_Tx.errorReason()); |
| 328 | } |
| 329 | } |
| 330 |  |
| 331 | { |
| 332 | FirebaseJson json; |
| 333 | json.add("datetime", pdatetime).add("PM\_10", pdetail\_2); |
| 334 | if (Firebase.setJSON(firebaseData\_Tx, data\_path + "/PM\_10", json) != 0) { |
| 335 | // Do nothing. |
| 336 | } else { |
| 337 | lresult\_int += 2; |
| 338 | Serial.println("Tx FAILED: REASON: " + firebaseData\_Tx.errorReason()); |
| 339 | } |
| 340 | } |
| 341 |  |
| 342 | delay(3600000); |
| 343 | return lresult\_int; |
| 344 | } |