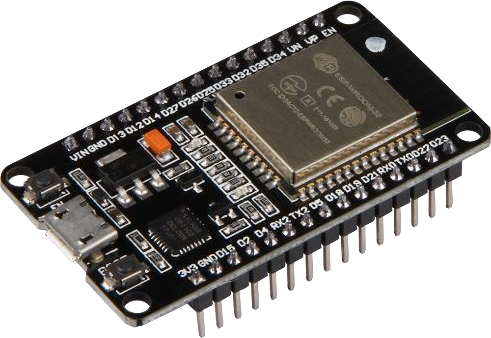
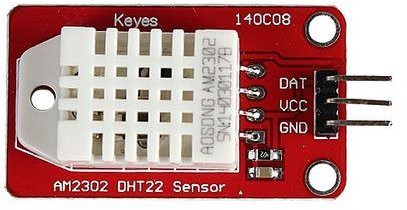
Ceas de birou

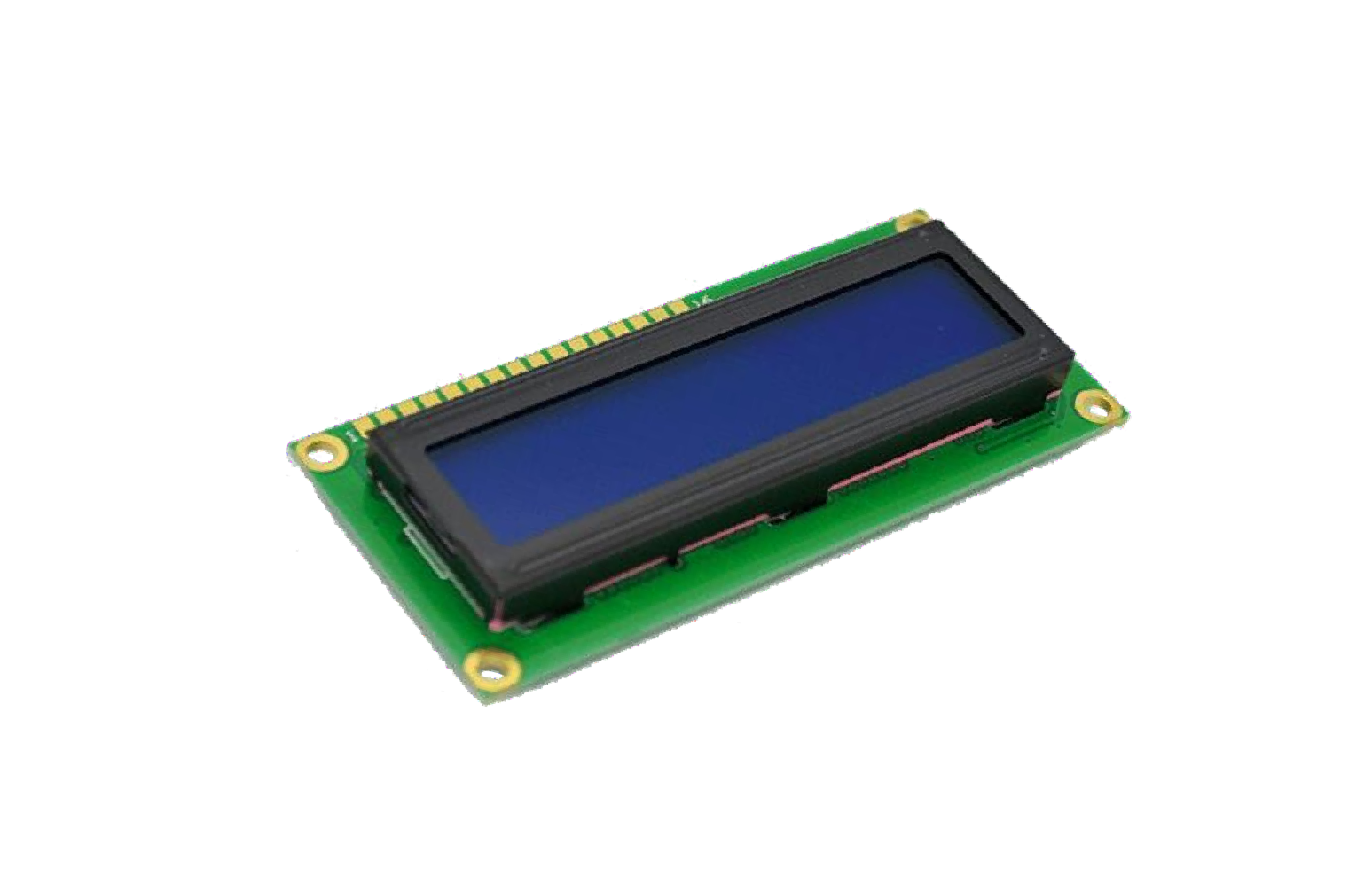
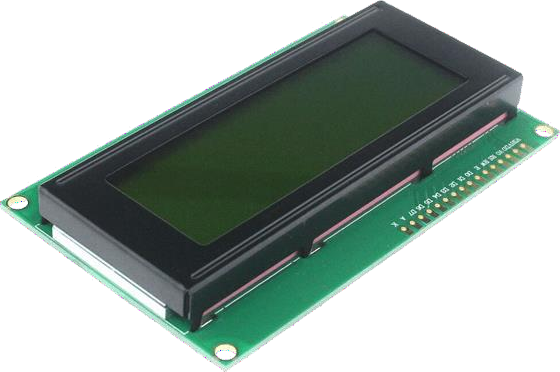
Echipa 1008

1. Componente:

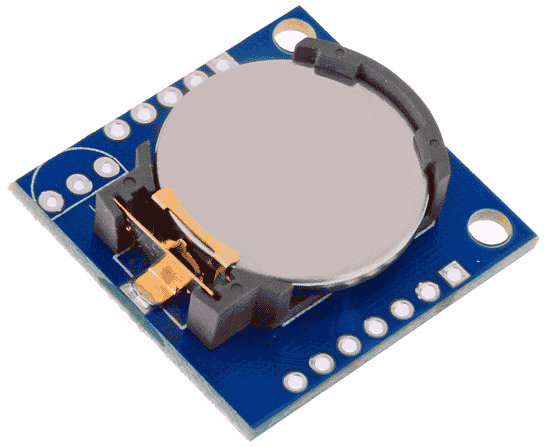
Placuta ESP32 -

Senzor Temperatura DHT22 –

Display LCD 20x4 I2C –



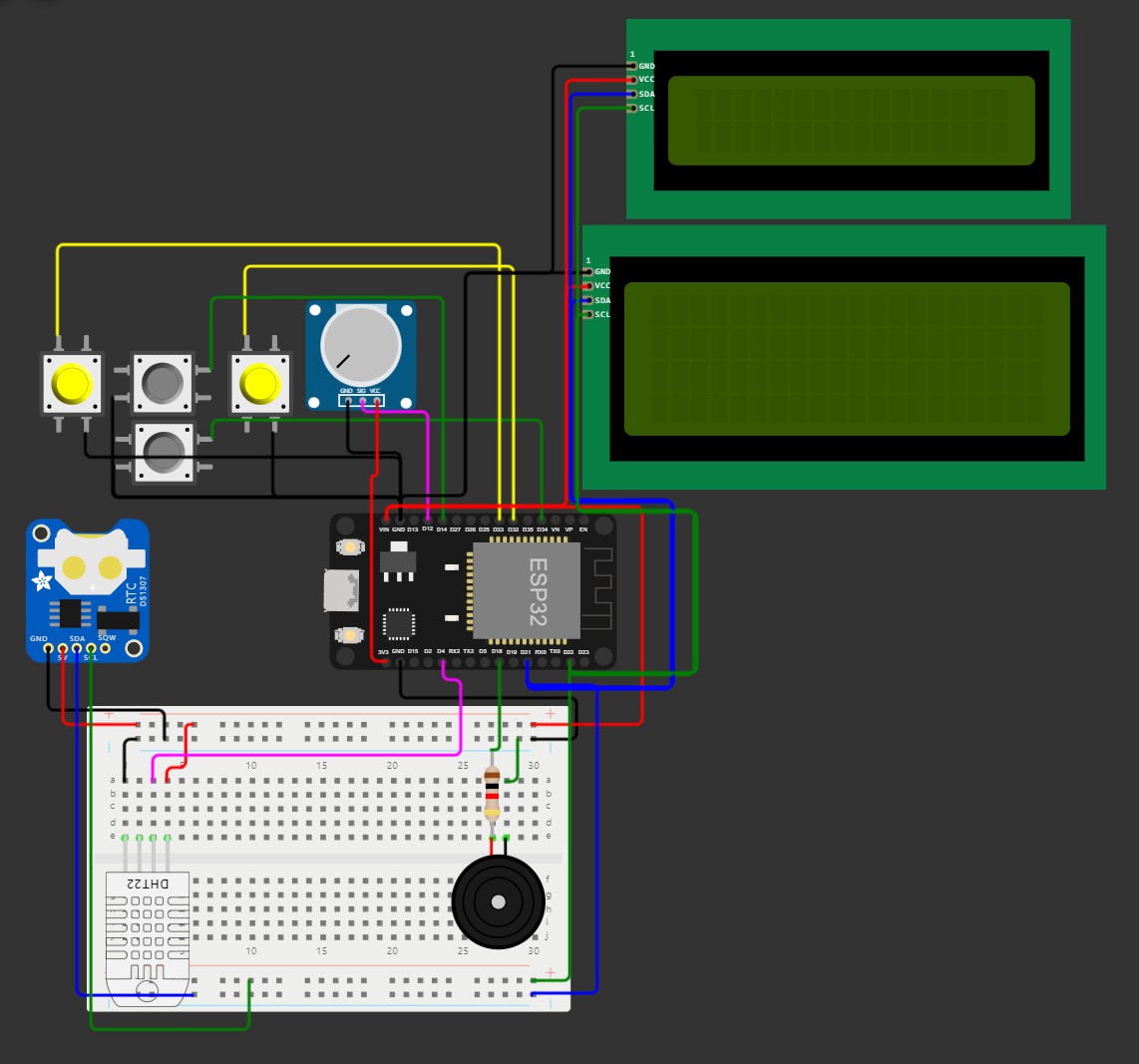
Display LCD 16x2 I2C –

RTC (Real Time Clock) DS1307 –

Potentiometru –

Buzzer -

1. Functii:
   * Mentinerea Datei si a orei cu o precizie perfecta chiar si atunci cand nu este cuplat la o sursa de curent datorita Senzorului RTC DS1307 care dispune de o baterie de ceas de 3V
   * Afisajul temperaturii si a umiditatii din incapere , ambele fiind furnizate de senzorul DHT22
   * Meniu setare ora/minut cu ajutorul potentiometrului
   * Meniu setare alarma cu ajutorul potentiometrului
   * Sunet alarma emis de Buzzer-ul prezent in configuratie
2. Schema Wokwi:



1. Cod:
2. #include <Wire.h>
3. #include <Adafruit\_GFX.h>
4. #include <RTClib.h>
5. #include "DHT.h"
6. #include <LiquidCrystal\_I2C.h>
7. float floatMap(float x, float in\_min, float in\_max, float out\_min, float out\_max) {
8. return (x - in\_min) \* (out\_max - out\_min)

/ (in\_max - in\_min) + out\_min;

13. }

14.

1. LiquidCrystal\_I2C lcd\_i2c(0x26, 20, 4); // I2C address 0x27, 20 column and 4 rows
2. LiquidCrystal\_I2C nume(0x27, 16, 2);
3. RTC\_DS1307 rtc;
4. char daysOfTheWeek[7][12] = {
5. "Duminica",
6. "Luni",
7. "Marti",
8. "Miercuri",
9. "Joi",
10. "Vineri",
11. "Sambata"

28. };

29.

1. #define SCREEN\_WIDTH 128 // OLED width, in pixels
2. #define SCREEN\_HEIGHT 64 // OLED height, in pixels
3. #define DHTPIN 4
4. #define DHTTYPE DHT22
5. int lastState1 = HIGH;
6. int currentState;
7. int oka = 1;
8. int okb = 1;
9. int p = 5;
10. int STOP = HIGH;
11. int Ora\_alarma = 0;
12. int Minut\_alarma = 0;
13. DHT dht(DHTPIN, DHTTYPE);
14. void setup() {
15. Serial.begin(9600);

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 53. | lcd\_i2c.init(); |  | | |
| 54. | lcd\_i2c.backlight(); |
| 55. | nume.init(); |
| 56. | nume.backlight(); |
| 57. |  |
| 58. |  |
| 59. | if (! rtc.begin()) { |
| 60. | Serial.println("Couldn't | find RTC"); | | |
| 61. | Serial.flush(); |  | | |
| 62. | while (1); |  | | |
| 63. | } |  | | |
| 64. |  |  | | |
| 65.  0));  66. | rtc.adjust(DateTime(2023, 5, 29,  dht.begin(); | | 12, | 20, |
| 67. |  | |  |  |
| 68. | pinMode(32, INPUT\_PULLUP); | |  |  |
| 69. | pinMode(33, INPUT\_PULLUP); | |  |  |
| 70. | pinMode(14, INPUT\_PULLUP); | |  |  |
| 71. | pinMode(34, INPUT\_PULLUP); | |  |  |
| 72. | pinMode(18, OUTPUT); | |  |  |
| 73. |  | |  |  |
| 74. | delay(2000); | |  |  |
| 75. |  | |  |  |
| 76. |  | |  |  |
| 77. | nume.setCursor(0,1); | |  |  |
| 78. | nume.print("SERBAN xx GORCEA"); | |  |  |

79. }

80.

81. void loop() { 82.

83.

84.

85.

1. float h = dht.readHumidity();
2. float t = dht.readTemperature();
3. float f = dht.readTemperature(true); 89.
4. if (isnan(h) || isnan(t) || isnan(f)) {
5. Serial.println(F("Failed to read from DHT sensor!"));
6. return;

93. }

94.

95. float hic = dht.computeHeatIndex(t, h, false);

96.

1. DateTime now = rtc.now();
2. if(oka==1 && okb==1)
3. {
4. lcd\_i2c.setCursor(0, 0);
5. lcd\_i2c.print(now.hour(), DEC);
6. lcd\_i2c.print(':');
7. lcd\_i2c.print(now.minute(), DEC);
8. lcd\_i2c.print(':');
9. if(now.second()<10)
10. lcd\_i2c.print(0);
11. lcd\_i2c.print(now.second(), DEC);
12. lcd\_i2c.setCursor(0, 3);
13. lcd\_i2c.print(now.day(), DEC);
14. lcd\_i2c.print('/');
15. lcd\_i2c.print(now.month(), DEC);
16. lcd\_i2c.print('/');
17. lcd\_i2c.print(now.year(), DEC);
18. lcd\_i2c.print("->");
19. lcd\_i2c.print(daysOfTheWeek[now.dayOfTheW eek()]);
20. lcd\_i2c.setCursor(10, 1);
21. lcd\_i2c.print("Temp:");
22. lcd\_i2c.print(t);
23. lcd\_i2c.setCursor(10, 2);
24. lcd\_i2c.print("Umid:");
25. lcd\_i2c.print(h);

129. }

130.

131. else if(oka==0 && okb==1)

|  |  |  |  |
| --- | --- | --- | --- |
| 132. | { |  |  |
| 133. |  |  |
| 134. |  |  |
| 135. |  | lcd\_i2c.setCursor(7, 0); |
| 136. |  | lcd\_i2c.print("ALARMA"); |
| 137. |  | lcd\_i2c.setCursor(5, 1); |
| 138. |  |  |
| 139. |  | lcd\_i2c.print(Ora\_alarma, DEC); |
| 140. |  | lcd\_i2c.print(':'); |
| 141. |  | if(Minut\_alarma<10) |
| 142. |  | lcd\_i2c.print(0); |
| 143. |  | lcd\_i2c.print(Minut\_alarma, DEC); |
| 144. |  |  |
| 145. |  | lcd\_i2c.setCursor(p, 3); |
| 146. |  | lcd\_i2c.print("^"); |
| 147. |  |  |
| 148. |  |  |
| 149. |  | if(p==5) |
| 150. |  | { |
| 151. |  | int analogValue = analogRead(13); |
| 152. |  | Ora\_alarma = floatMap(analogValue, | 0, |
| 4095, | 0, 23); | | |
| 153. |  | | |
| 154. | } | | |
| 155. | else | | |
| 156. | { | | |
| 157. | int analogValue = analogRead(13); | | |

158. Minut\_alarma = floatMap(analogValue, 0, 4095, 0, 59);

159.

160. }

161.

162.

163.

164. currentState = digitalRead(14); 165.

1. if(currentState == LOW)
2. { delay(200);
3. if(p==5)
4. p=8;
5. else p=5;
6. currentState = HIGH;
7. lcd\_i2c.clear();

173. }

174.

175. }

176.

1. else if(oka==1 && okb==0)
2. {
3. lcd\_i2c.setCursor(7, 0);
4. lcd\_i2c.print("SETARE");
5. lcd\_i2c.setCursor(5, 1);
6. lcd\_i2c.print(now.hour(), DEC);
7. lcd\_i2c.print(':');
8. if(now.minute()<10)
9. lcd\_i2c.print(0);
10. lcd\_i2c.print(now.minute(), DEC);
11. lcd\_i2c.setCursor(p, 3);
12. lcd\_i2c.print("^");
13. float Ora;
14. float Minut;
15. if(p==5)
16. {
17. int analogValue = analogRead(13);
18. Ora = floatMap(analogValue, 0, 4095,

0, 23);

200. }

1. else
2. {
3. int analogValue = analogRead(13);
4. Minut = floatMap(analogValue, 0,

4095, 0, 59);

206. }

207.

208. rtc.adjust(DateTime(2023, 5, 29, Ora,

Minut, 9));

209.

|  |  |  |  |
| --- | --- | --- | --- |
| 210. |  | currentState = digitalRead(14); | |
| 211. |  |  | |
| 212. |  | if(currentState == LOW) | |
| 213. | { | delay(200); | |
| 214. |  | if(p==5) | |
| 215. |  | p=8; | |
| 216. |  | else p=5; | |
| 217. |  | lcd\_i2c.clear(); | |
| 218. |  | currentState = HIGH; | |
| 219. | } |  | |
| 220. |  |  | |
| 221. |  |  | |
| 222. |  |  | |
| 223. | } |  | |
| 224. |  |  | |
| 225. | int | | a = digitalRead(32); |
| 226. | int | | b = digitalRead(33); |
| 227. |  | |  |
| 228. | //-----------------PENTRU SET | | |
| 229. | if( b == LOW) | | |
| 230. | { delay(1000); | | |
| 231. | b = HIGH; | | |
| 232. |  | | |
| 233. | if(okb==1) | | |
| 234. | okb=0; | | |

1. else
2. okb=1;
3. lcd\_i2c.clear();

238. }

239. //

240.

1. //----------PENTRU ALARMA
2. if( a == LOW)
3. { delay(1000);
4. a = HIGH;
5. if(oka==1)
6. oka=0;
7. else
8. oka=1;
9. lcd\_i2c.clear();

251. }

252.

253. //

254.

1. nume.setCursor(0,0);
2. nume.print("Alarma: ");
3. nume.print(Ora\_alarma, DEC);
4. nume.print(':');

|  |  |
| --- | --- |
| 260. | nume.print(Minut\_alarma, DEC); |
| 261. |  |
| 262. | if(Ora\_alarma == now.hour() && |
| Minut\_alarma == now.minute()) | |
| 263. | { |
| 264. | ledcWriteTone(0,800); |
| 265. | delay(1000); |
| 266. | uint8\_t octave = 1; |
| 267. | ledcWriteNote(0,NOTE\_C,octave); |
| 268. | delay(1000); |
| 269. | } |
| 270. |  |
| 271. | } |