**Laserilla toimiva ajanotto laite**



Näyttötyö:

Valtteri Kinnunen K3TVT21C

Savon ammattiopisto

**Johdanto**

Tässä työssä oli tarkoituksena rakentaa ja ohjelmoida ajanotto laite. Aika alkaa, kun laser katkaistaan ja loppuu kun toinen laser katkaistaan. Aika näytetään lcd näytöllä. Työssä on lähtö viiva laser ja loppuviiva laser. Laitteet käyttävät laser moduleja ja ldr vastuksia tunnistamaan laserin valoa.

**Tarvikeluettelo**

|  |  |  |
| --- | --- | --- |
|  | kpl | hinnasto |
| esp32 | 2 |  |
| laser module | 2 |  |
| ldr | 2 |  |
| lcd i2c näyttö | 1 |  |
| nappi | 1 |  |
| akku | 4 |  |
| 10komh vastus | 3 |  |

**Ohjelmointityökalujen käyttöönotto**

Asenna Arduino IDE

<https://www.arduino.cc/en/software>

Asenna CP2014 USB ajuri

<https://learn.adafruit.com/adafruit-huzzah32-esp32-feather/using-with-arduino-ide>

Asenna Adafruit Huzzah with ESP32 kirjastot Arduino Ideen

Lisää tämä rivi Arduino Iden asetuksiin: https://dl.espressif.com/dl/package\_esp32\_index.json

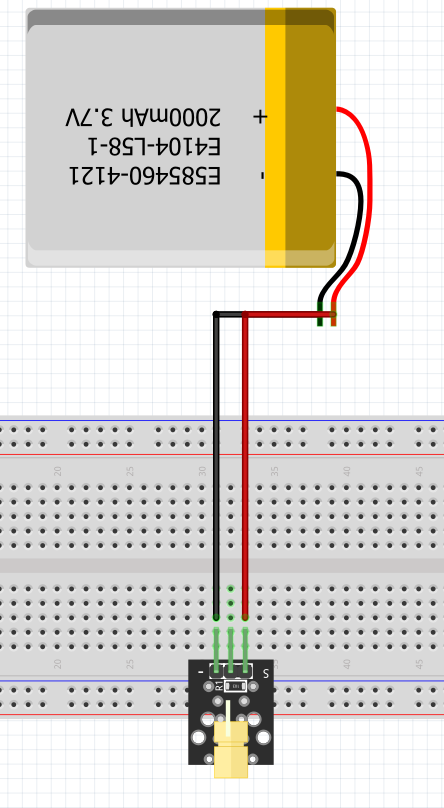
(<https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json>)

Asenna Boards Managerissa esp32 korttipaketti Arduino Ideen

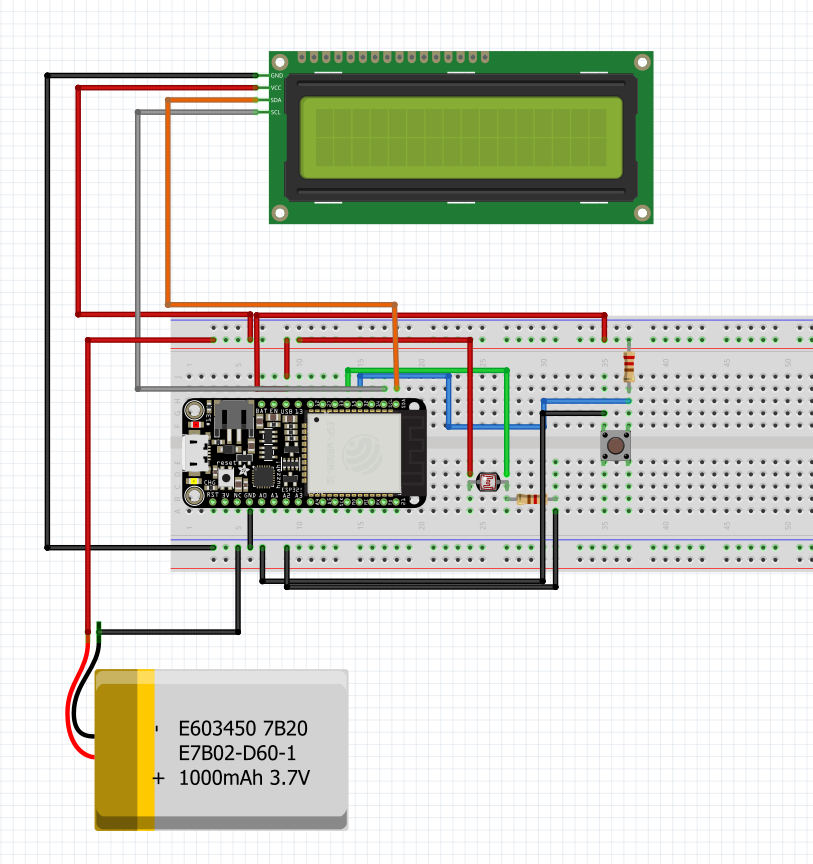
Valitse Arduino Idessä kortti Adafruit ESP32 Feather

**Johdotuskaavio**

Lähettäjä:

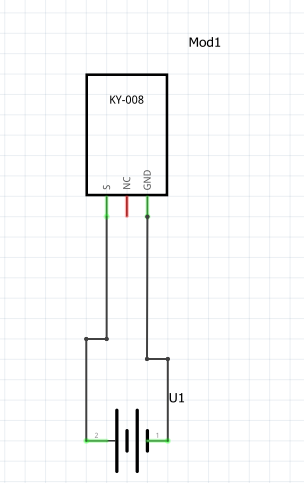
****

Vastaanotin:

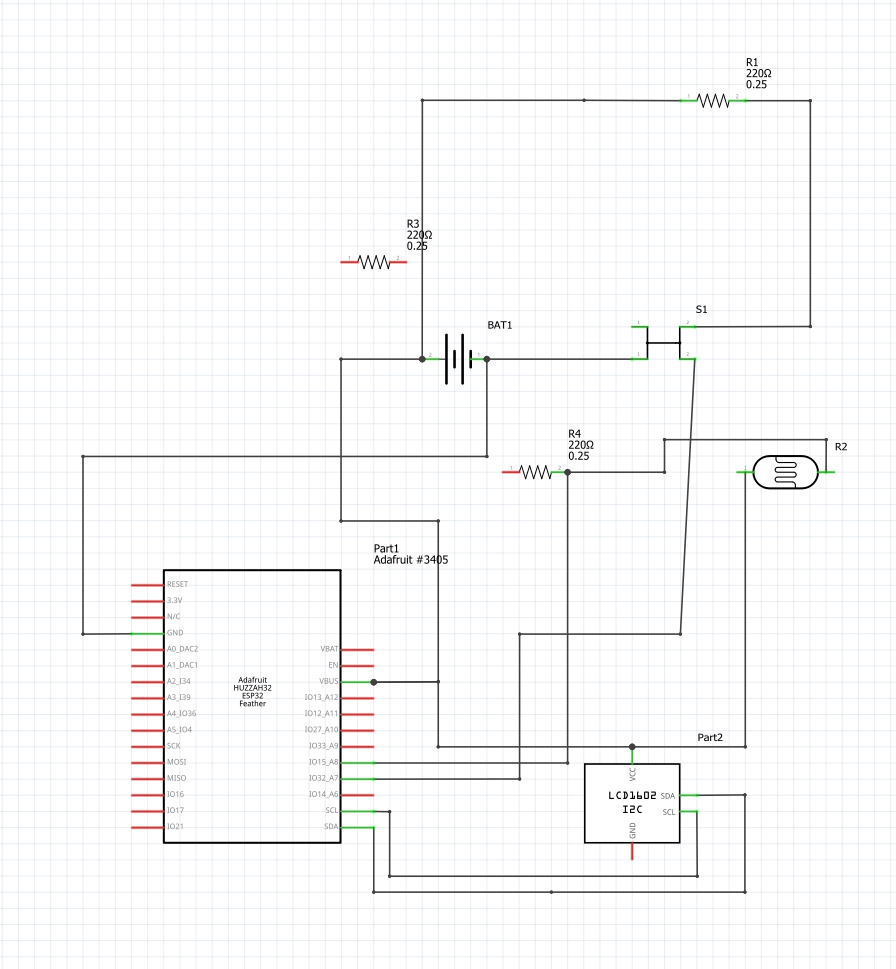
****

**Johdotus kaavio:**

Lähetin

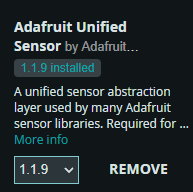
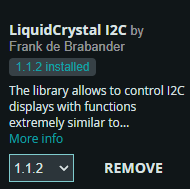


vastaanotin:

****

**Ohjelmointi:**

käytetyt kirjastot



lähettäjä:

#define BLYNK\_TEMPLATE\_ID "TMPL4dw-oY\_P3"

#define BLYNK\_TEMPLATE\_NAME "timer"

#define BLYNK\_AUTH\_TOKEN "46N-ZnxgEH256kKS1vbh0jxBEbkugVzP"

#define BLYNK\_PRINT Serial

#include <WiFi.h>

#include <esp\_now.h>

#include <WiFiClient.h>

#include <BlynkSimpleEsp32.h>

#include <Adafruit\_Sensor.h>

uint8\_t broadcastAddress[] = {0x3C, 0x71, 0xBF, 0x6F, 0x32, 0x70};

// Your WiFi credentials.

// Set password to "" for open networks.

char ssid[] = "TVT-WLAN";

char pass[] = "salasana";

int pin = 33;

int sensorValue;

int StartTimer = 0;

String success;

typedef struct struct\_message{

  float Start;

} struct\_message;

struct\_message SendTimer;

esp\_now\_peer\_info\_t peerInfo;

void OnDataSent(const uint8\_t \*mac\_addr, esp\_now\_send\_status\_t status) {

  Serial.print("\r\nLast Packet Send Status:\t");

  Serial.println(status == ESP\_NOW\_SEND\_SUCCESS ? "Delivery Success" : "Delivery Fail");

  if(status == 0){

    success = "Delivery Success :)";

    Serial.println(StartTimer);

  }

  else{

    success = "Delivery Fail:(";

  }

}

void setup() {

  // put your setup code here, to run once:

  Serial.begin(115200);

  // Set device as a Wi-Fi Station

  WiFi.mode(WIFI\_MODE\_STA);

  // Init ESP-NOW;

  Serial.print(WiFi.macAddress());

  if(esp\_now\_init() != ESP\_OK) {

    Serial.print("Error initializing ESP-NOW");

    return;

  }

  // Once ESPNow is successfully Init, we will register for Send CB to

  // get the status of Trasnmitted packet

  esp\_now\_register\_send\_cb(OnDataSent);

  //register peer

  esp\_now\_peer\_info\_t peerInfo;

  memset(&peerInfo, 0, sizeof(peerInfo));

  memcpy(peerInfo.peer\_addr, broadcastAddress, 6);

  peerInfo.channel = 0;

  peerInfo.encrypt = false;

  // add peer

  if(esp\_now\_add\_peer(&peerInfo) != ESP\_OK){

    Serial.println("failed to add peer");

    return;

  }

}

void loop() {

  // put your main code here, to run repeatedly:

  //reads ldr value

  sensorValue = analogRead(pin);

  Serial.println(sensorValue);

  //if ldr value is less the 3000 if it is starts timer

  if(sensorValue < 3000){

      StartTimer = 1;

      SendTimer.Start = StartTimer;

      Serial.println(StartTimer);

  }

  else{

    StartTimer = 0;

    SendTimer.Start = 0;

  }

  esp\_err\_t result = esp\_now\_send(broadcastAddress, (uint8\_t \*) &SendTimer, sizeof(SendTimer));

  if(result == ESP\_OK) {

    //Serial.println("Sent with success");

  }

  else {

    Serial.println("Error sending the data 1");

  }

  delay(500);

}

vastaanottaja:

#define BLYNK\_TEMPLATE\_ID "TMPL4dw-oY\_P3"

#define BLYNK\_TEMPLATE\_NAME "timer"

#define BLYNK\_AUTH\_TOKEN "46N-ZnxgEH256kKS1vbh0jxBEbkugVzP"

#define BLYNK\_PRINT Serial

#include <WiFi.h>

#include <esp\_now.h>

#include <WiFiClient.h>

#include <BlynkSimpleEsp32.h>

#include <Adafruit\_Sensor.h>

#include <LiquidCrystal\_I2C.h>

uint8\_t broadcastAddress[] = {0xCC, 0x50, 0xE3, 0xBB, 0xE7, 0x84};

// Your WiFi credentials.

// Set password to "" for open networks.

char ssid[] = "TVT-WLAN";

char pass[] = "salasana";

LiquidCrystal\_I2C lcd(0x3F, 16, 2);

BlynkTimer timer;

int pin = 33;

int sensorValue;

int crossed = 0;

int last = 0;

int stop = 1;

int StartTimer = 0;

int TimerStarted = 0;

unsigned long Timer;

unsigned long StartTime;

unsigned long elapsedTimemS;

unsigned long elapsedTimeS;

unsigned long elapsedTimeM;

unsigned long elapsedTimeH;

typedef struct struct\_message{

  float Start;

} struct\_message;

struct\_message ReadStart;

void DataRecieved(const uint8\_t \* mas, const uint8\_t \*incomingData, int len) {

  memcpy(&ReadStart, incomingData, sizeof(ReadStart));

  /\*Serial.print("Recieved data: ");

  Serial.println(len);

  Serial.print("StartTimer: ");

  Serial.println(ReadStart.Start);\*/

  StartTimer = ReadStart.Start;

}

void setup() {

  // put your setup code here, to run once:

pinMode(32, INPUT\_PULLUP);

  Serial.begin(115200);

  WiFi.mode(WIFI\_MODE\_STA);

  Serial.print(WiFi.macAddress());

  elapsedTimeS = 0;

  //initializing lcd

  lcd.init();

  lcd.clear();

  lcd.backlight();

  //Blynk.begin(BLYNK\_AUTH\_TOKEN, ssid, pass);

  //timer.setInterval(1000L, myTimerEvent);

  if(esp\_now\_init() != ESP\_OK){

    Serial.println("Error: connection failed");

    return;

  }

  esp\_now\_peer\_info\_t peerInfo;

  memset(&peerInfo, 0, sizeof(peerInfo));

  memcpy(peerInfo.peer\_addr, broadcastAddress, 6);

  peerInfo.channel = 0;

  peerInfo.encrypt = false;

  if(esp\_now\_add\_peer(&peerInfo) != ESP\_OK){

    Serial.println("Error adding reciever");

    return;

  }

  esp\_now\_register\_recv\_cb(DataRecieved);

}

void loop() {

  // put your main code here, to run repeatedly:

  int button32 = digitalRead(32);

  //reads ldr value

  sensorValue = analogRead(pin);

  //Blynk.run();

  //timer.run();

  last = crossed;

 if(StartTimer == 1){

    Timer = millis();

    TimerStarted = 1;

  }

  //Serial.println(StartTimer);

  if(sensorValue > 0){

    crossed = 0;

  }

  else{

    crossed = 1;

  }

  //start a timer

  StartTime = millis();

  //checks if ldr value is 4000 or over

  if(sensorValue >= 4000){

    stop = 1;

  }

  //if ldr value is less the 3000 if it is starts timer

  else if(sensorValue < 3000){

    //Timer = millis();

    //if timer has started stops the timer when the laser is triggered

   if(TimerStarted == 1 && stop == 1){

     TimerStarted = 0;

     stop = 0;

   }

  }

  //delay(500);

  if(TimerStarted == 1){

    //calculates how long it has been after the laser was triggered in seconds

    //elapsedTimeS = (StartTime - Timer) / 1000;

    //calculates how long it has been after the laser was triggered in tenths of seconds

    elapsedTimemS = (StartTime - Timer) / 100;

    if(elapsedTimemS >= 10){

      elapsedTimemS = 0;

      elapsedTimeS++;

      lcd.clear();

      Timer = millis();

    }

    //changes second to minutes

    if(elapsedTimeS == 60){

      elapsedTimeM++;

      elapsedTimeS = 0;

      lcd.clear();

    }

    //changes minutes to hours

    if(elapsedTimeM == 60){

      elapsedTimeH++;

      elapsedTimeM = 0;

    }

  }

  lcd.setCursor(0, 0);

  lcd.print(elapsedTimeH);

  lcd.setCursor(3, 0);

  lcd.print(elapsedTimeM);

  lcd.setCursor(6, 0);

  lcd.print(elapsedTimeS);

  lcd.setCursor(8, 0);

  lcd.print(".");

  lcd.setCursor(9, 0);

  lcd.print(elapsedTimemS);

  if(button32 == LOW){

    TimerStarted = 0;

    elapsedTimeH = 0;

    elapsedTimeM = 0;

    elapsedTimeS = 0;

    elapsedTimemS = 0;

    lcd.clear();

    lcd.setCursor(0, 0);

    lcd.print(elapsedTimeH);

    lcd.setCursor(3, 0);

    lcd.print(elapsedTimeM);

    lcd.setCursor(6, 0);

    lcd.print(elapsedTimeS);

    lcd.setCursor(7, 0);

    lcd.print(".");

    lcd.setCursor(8, 0);

    lcd.print(elapsedTimemS);

  }

}

**Johtopäätökset ja itsearviointi**

Työ ei ollut erityisen vaikea mutta joiduin kysymään aika paljon apua.

Työhön olisi voinut lisätä ilmoituksen, kun akku oli lopussa ja tiedon olisi voinut jakaa eri tavalla jotta ajan olisi voinut näyttää blynkissä.

Antaisin työstä arvosanan 3