Loppu projektin dokumentointi 26.5.2021

Eemeli Antikainen

K3TTI19A

**Riistakameraverkosto**

(Havaitsijat ovat ultraäänisensoreilla varustettuja Huzzaheita.)

Projekti on 4 huzzah:in verkosto jossa on 3 havaitsijaa, 3 kameraa ja yksi pääasema. Idea on, että kun havaitsijat havaitsevat liikettä kamerat ottavat kuvan ja lähettävät kuvan samassa verkossa olevaan Blynk sovellukseen. Tarkemmin ottaen liikesensorin logiikka toimii siten että kun osoitat sen seinään (3 M maksimi etäisyys) sensori nollaa itsensä ja kun jokin asia liikkuu 8cm sensoria kohti nollapisteestä, sensori laukaisee hälytyksen, ottaa kameralla kuvan ja lähettää tiedon hälytyksestä http pyynnön query parametreissa serverille ja serveri laukaisee ledit. Koko systeemi toimii langattomasti akuilla ja tieto siirretään wifi:n välityksellä. pääasema hostaa wifiä joihin havaitsimet on yhdistetty ja kamerat on yhdistetty luokan verkkoon. Laitteen tarkoitus on olla halpa simppeli valvontajärjestelmä.

HUOM Puhelimen täytyy olla myös samassa verkossa, kuin kamerat jotta kuvan siirto toimii.

**BLYNK**

Käytän projektissani BLYNK sovellusta, jonka testaajan pitää asentaa puhelimelleen, ja jonne testaajan pitää luoda käyttäjä. Kun sovellus on asennettu ja käyttäjä on tehty seuraa alhaalla olevia ohjeita mitä testaajan pitää lisätä BLYNK sovellukseen jotta kokonaisuus toimii.

|  |  |
| --- | --- |
| Ensin luo BLYNK:iin uusi projekti seuraavilla spekseillä. | Kun olet saanut autentikaatio tunnuksen sähköpostiin laitta autentikaatio tunnus kameran koodin auth muuttujaan. |
|  |  |
| Lisää Blynk projektiin 3 Image Gallery komponenttia. | Paina Image Gallery komponenttia ja nimeä ne Kamera 1-3 ja Laita INPUT Pinit V1-V3. |
|  |  |

**Koodit**

Huzzaheita on käytössä 4 ja kameroita 3 joten tässä on havaitsijoiden, kameroiden ja pääaseman koodi. Havaitsijoiden koodeissa ainut ero on get kutsussa [url:in](file:///C:\Users\Topi\AppData\Local\Temp\in) kamera numero. Kameran koodeissa ainut ero on Blynk sovelluksen pin numero. **HUOM DOKUMENTOINNIN TESTAAJAN TÄYTYY ITSE VAIHTAA NÄMÄ KOODISSA RIIPPUEN LAITTEESTA.**

**Havaitsijan koodi:**

//Otetaan tarvittavat paketit

#include <WiFi.h>

#include <HTTPClient.h>

#include <NewPing.h>

//Määritetään yhdistettävän wifin nimi ja salasana

const char\* ssid = "ESP32-Pantera";

const char\* password = "S@lasana";

//Your IP address or domain name with URL path

const char\* serverNameDistance = "http://192.168.4.1/etaisyys";

unsigned long previousMillis = 0;

const long interval = 100;

int trigPin = A1; // Trigger

int echoPin = A12; // Echo

int KameraPin = A11; //Kamera

//Määritetään etäisyyssenorin pinit ja maksimietäisyys

NewPing sonar(trigPin, echoPin,350);

int nykyinenLiikeArvo = 0;

int halytysRaja = 0;

void setup() {

Serial.begin(9600);

//Määritetään kameran laukaisin pin outputiksi

pinMode(KameraPin,OUTPUT);

digitalWrite(KameraPin, LOW);

//Yhdistetään Wifiin

WiFi.begin(ssid, password);

Serial.println("Connecting");

while(WiFi.status() != WL\_CONNECTED) {

delay(500);

Serial.print(".");

}

Serial.print("Connected to WiFi network with IP Address: ");

Serial.println(WiFi.localIP());

}

void loop() {

unsigned long currentMillis = millis();

if(currentMillis - previousMillis >= interval) {

// Check WiFi connection status

if(WiFi.status()== WL\_CONNECTED ){

//Määrittää millä aikavälillä ultraäänisensori ottaa havaintoja

delay(250);

//Otetaan nykyinen etäisyys etaisyys muuttujaan

int etaisyys = sonar.ping\_cm();

nykyinenLiikeArvo = nykyinenLiikeArvo + etaisyys;

Serial.print("Ping: ");

Serial.print(etaisyys);

Serial.println("cm");

//Määritetään hälytysrajaksi nykyinen rajaetäisyys + 8 CM ja -8CM

halytysRaja = halytysRaja + 8;

if(nykyinenLiikeArvo> halytysRaja && nykyinenLiikeArvo != 0){

//Jos kohde liikkuu kauvemmas otetaan nykyinen etäisyys uudeksi nollaarvoksi

halytysRaja = etaisyys;

//Jos liiketunnistimista liikutaan kauemmas ei hälytetäkkään koska tämä aiheuttaa tuplahälyyksen kun kohde kävelee ohi.

//serverNameDistance = "http://192.168.4.1/etaisyys?kamera1=1";

// httpGETRequest(serverNameDistance);

Serial.println("Kauemmas");

}

else{

halytysRaja = halytysRaja - 16;

if(nykyinenLiikeArvo < halytysRaja &&nykyinenLiikeArvo != 0){

//Jos kohde liikkuu lähemmäs otetaan nykyinen etäisyys uudeksi nollaarvoksi

//Myös läheteään pääasemalle http get pyyntö jossa ilmoitetaan että hälytys on lauennut

//Kirjoitetaan kamera pin hetkeksi HIGH:si jotta kamera ottaa kuvan

digitalWrite(KameraPin, HIGH);

delay(50);

digitalWrite(KameraPin, LOW);

halytysRaja = etaisyys;

serverNameDistance = "http://192.168.4.1/etaisyys?kamera1=1";

httpGETRequest(serverNameDistance);

Serial.println("Lähemmäs");

}

else{

halytysRaja = halytysRaja + 8;

}

}

nykyinenLiikeArvo = 0;

// save the last HTTP GET Request

previousMillis = currentMillis;

}

else {

Serial.println("WiFi Disconnected");

}

}

}

String httpGETRequest(const char\* serverName) {

HTTPClient http;

// Your IP address with path or Domain name with URL path

http.begin(serverName);

// Send HTTP POST request

int httpResponseCode = http.GET();

String payload = "--";

if (httpResponseCode>0) {

payload = http.getString();

}

else {

Serial.print("Error code: ");

Serial.println(httpResponseCode);

}

// Free resources

http.end();

return payload;

}

**Pääaseman:**

// Kirjastot piti ottaa suoralla polulla koska sain kirjastokonflikti virhettä

//Joten testaajankin pitää laittaa kirjastoon viittaus suoralla polulla

#include <C:\Users\Topi\AppData\Local\Arduino15\packages\esp32\hardware\esp32\1.0.6\libraries\WiFi\src\Wifi.h>

#include <C:\Users\Topi\Documents\Arduino\libraries\ESPAsyncWebServer-master\src\ESPAsyncWebServer.h>

// Asetetaan wlanin nimi ja salasana

const char\* ssid = "ESP32-Pantera";

const char\* password = "S@lasana";

//Määritetään ledien pinit

int Led1 = A12;

int Led2 = A11;

int Led3 = A10;

//Käynistetään asynkrooninen webserver portille 80

AsyncWebServer server(80);

//Kävijälaskimen muuttuja

int laskinKamera1 = 0;

int laskinKamera2 = 0;

int laskinKamera3 = 0;

void setup(){

Serial.begin(115200);

//Määritetään ledit outputeiksi´

pinMode(Led1,OUTPUT);

pinMode(Led2,OUTPUT);

pinMode(Led3,OUTPUT);

//Tarvitaan äänentuotto vaihtoehoa varten

/\*ledcSetup(0,1E5,12);

ledcAttachPin(Led1,0);\*/

Serial.print("Setting AP (Access Point)…");

//Käynnistetään wifi

WiFi.softAP(ssid, password);

IPAddress IP = WiFi.softAPIP();

Serial.print("AP IP address: ");

Serial.println(IP);

//Luetaan sensoreilta saadut hälytyskutsut jotka lähetetään http get kutsun parametreissa

server.on("/etaisyys", HTTP\_GET, [](AsyncWebServerRequest \*request){

int paramsNr = request->params();

for(int i=0;i<paramsNr;i++){

//Otetaan http pyynnöstä parametrit talteen

AsyncWebParameter\* p = request->getParam(i);

//Tulostetaan http kutsussa saadut parametrien nimet ja arvot testailua varten.

Serial.print("Param name: ");

Serial.println(p->name());

Serial.print("Param value: ");

Serial.println(p->value());

//Pistetään queryssä tullut arvo arvo muuttujaan

int arvo = p->value().toInt();

Serial.println("------");

//Tarkistetaan mikä kamera on kyseessä ja lähetetään tullut arvo tarkistusfunktioon

if(p->name() == "kamera1"){

Tarkista(1, arvo);

}

if(p->name() == "kamera2"){

Tarkista(2, arvo);

}

if(p->name() == "kamera3"){

Tarkista(3, arvo);

}

}

//Lähetetään perus vastaus

request->send(200, "text/plain", "message received");

});

// Käynnistetään serveri

server.begin();

}

void loop(){

}

//Tarkista funktio hälyttää ledejä jos jokin kamera on palauttanut hälytysarvon eli ykkösen.

void Tarkista(int kamera, int arvo){

if(kamera == 1 && arvo == 1){

//Tein huvikseen laskimen joka tulostaa konsoliin kuinka monesti liikettä on havaittu kamera ykköseltä

laskinKamera1 = laskinKamera1 + 1;

Serial.print("Kamera1: ");

Serial.println(laskinKamera1);

//Valinallinen koodi jos haluaa korvata ledin äänibusserilla

/\*ledcWriteTone(0,800);

delay(100);

uint8\_t octave = 1;

ledcWriteNote(0,NOTE\_C,octave); \*/

//For loop pyörii kunnes 5 ei ole suurepi kuin i

for(int i = 0; i < 5; i++){

//For loopin pyöriessä se laittaa 1 ledin päälle ja pois pienellä viiveellä. Tämä saa ledin välkkymään viidesti.

digitalWrite(Led1, HIGH);

delay(50);

digitalWrite(Led1, LOW);

delay(50);

}

}

else{

//Jos serveri ei saa kamera ykköseltä hälytysviestiä laitetaan led pois päältä

digitalWrite(Led1, LOW);

}

//Sama logiikka 2 muulle sensorille

if(kamera == 2 &&arvo == 1){

laskinKamera2 = laskinKamera2 + 1;

Serial.print("Kamera2: ");

Serial.println(laskinKamera2);

for(int i = 0; i < 5; i++){

digitalWrite(Led2, HIGH);

delay(50);

digitalWrite(Led2, LOW);

delay(50);

}

}

else{

digitalWrite(Led2, LOW);

}

if(kamera == 3 && arvo == 1){

laskinKamera3 = laskinKamera3 + 1;

Serial.print("Kamera3: ");

Serial.println(laskinKamera3);

for(int i = 0; i < 10; i++){

digitalWrite(Led3, HIGH);

delay(50);

digitalWrite(Led3, LOW);

delay(50);

}

}

else{

digitalWrite(Led3, LOW);

}

}

**Kameran Koodi:**

//otetaan tarvittavat kirjastot

#include "esp\_camera.h"

#include <WiFi.h>

#include "camera\_pins.h"

#include <BlynkSimpleEsp32.h>

#include "battery.h"

//Määritetään wifin tunnukset ja Blynkin avain

const char\* ssid = "TVT";

const char\* password = "S@lasana";

char auth[] = "evNQURIByTSEMWql4lxJ3xGE3OOGfR7k";

// Set your Static IP address

IPAddress local\_IP(192, 168, 8, 32);

// Set your Gateway IP address

IPAddress gateway(192, 168, 8, 1);

IPAddress subnet(255, 255, 255, 0);

IPAddress primaryDNS(8, 8, 8, 8);

IPAddress secondaryDNS(8, 8, 4, 4);

//Määritetään muuttujat

String my\_Local\_IP;

//Tähän pitää laittaa kameran led pini ja "nappipini" minulla nappipin on kamerassa oleva SDA/G4 pin jos tämä pin saa sähköä kamera ottaa kuvan

#define LED 2

#define BUTTON 4

void startCameraServer();

//Tehdään funktio joka ottaa kuvan ja lähtetään se blynk sovellukseen

void capture()

{

digitalWrite(LED,HIGH);

uint32\_t number = random(40000000);

Blynk.notify("Someone is at the door..");

Serial.println("http://"+my\_Local\_IP+"/capture?\_cb="+ (String)number);

//Riippuen kamerasta pitää vaihtaaBlynkin käytämää Piniä

Blynk.setProperty(V1, "urls", "http://"+my\_Local\_IP+"/capture?\_cb="+(String)number);

delay(100);

digitalWrite(LED,LOW);

}

void setup() {

//Määritetään kameran ledi ja serial portin baudi määrä

Serial.begin(115200);

pinMode(LED,OUTPUT);

Serial.setDebugOutput(true);

bat\_init();

Serial.println();

//Määritetään kameran pinit

camera\_config\_t config;

config.ledc\_channel = LEDC\_CHANNEL\_0;

config.ledc\_timer = LEDC\_TIMER\_0;

config.pin\_d0 = Y2\_GPIO\_NUM;

config.pin\_d1 = Y3\_GPIO\_NUM;

config.pin\_d2 = Y4\_GPIO\_NUM;

config.pin\_d3 = Y5\_GPIO\_NUM;

config.pin\_d4 = Y6\_GPIO\_NUM;

config.pin\_d5 = Y7\_GPIO\_NUM;

config.pin\_d6 = Y8\_GPIO\_NUM;

config.pin\_d7 = Y9\_GPIO\_NUM;

config.pin\_xclk = XCLK\_GPIO\_NUM;

config.pin\_pclk = PCLK\_GPIO\_NUM;

config.pin\_vsync = VSYNC\_GPIO\_NUM;

config.pin\_href = HREF\_GPIO\_NUM;

config.pin\_sscb\_sda = SIOD\_GPIO\_NUM;

config.pin\_sscb\_scl = SIOC\_GPIO\_NUM;

config.pin\_pwdn = PWDN\_GPIO\_NUM;

config.pin\_reset = RESET\_GPIO\_NUM;

config.xclk\_freq\_hz = 20000000;

config.pixel\_format = PIXFORMAT\_JPEG;

config.frame\_size = FRAMESIZE\_UXGA;

config.jpeg\_quality = 10;

config.fb\_count = 2;

// camera init

esp\_err\_t err = esp\_camera\_init(&config);

if (err != ESP\_OK) {

Serial.printf("Camera init failed with error 0x%x", err);

return;

}

sensor\_t \* s = esp\_camera\_sensor\_get();

//initial sensors are flipped vertically and colors are a bit saturated

s->set\_vflip(s, 1);//flip it back

s->set\_brightness(s, 1);//up the blightness just a bit

s->set\_saturation(s, -2);//lower the saturation

//drop down frame size for higher initial frame rate

s->set\_framesize(s, FRAMESIZE\_QVGA);

//Yhdistetään wifiin

// Configures static IP address

if (!WiFi.config(local\_IP, gateway, subnet, primaryDNS, secondaryDNS)) {

Serial.println("STA Failed to configure");

}

Serial.printf("Connect to %s, %s\r\n", ssid, password);

WiFi.begin(ssid, password);

while (WiFi.status() != WL\_CONNECTED) {

delay(500);

Serial.print(".");

}

Serial.println("");

Serial.println("WiFi connected");

//Aloitetaan kamera server

startCameraServer();

Serial.print("Camera Ready! Use 'http://");

Serial.print(WiFi.localIP());

my\_Local\_IP = local\_IP.toString();

Serial.println("' to connect");

Blynk.begin(auth, ssid, password);

}

void loop() {

//Käynnistetään blynk

Blynk.run();

//Jos nappi on high kutsutaan kuvan ottamias funktiota

if(digitalRead(BUTTON) == HIGH)

capture();

}

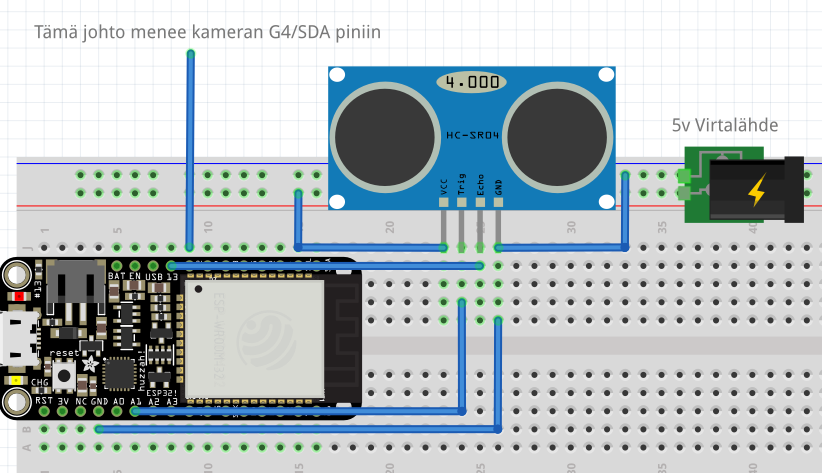
**Kuvat**

Työni kuvaisi parhaiten videolla mutta piti olla kuvat niin esittelen toiminnan sarjakuvamaisesti.

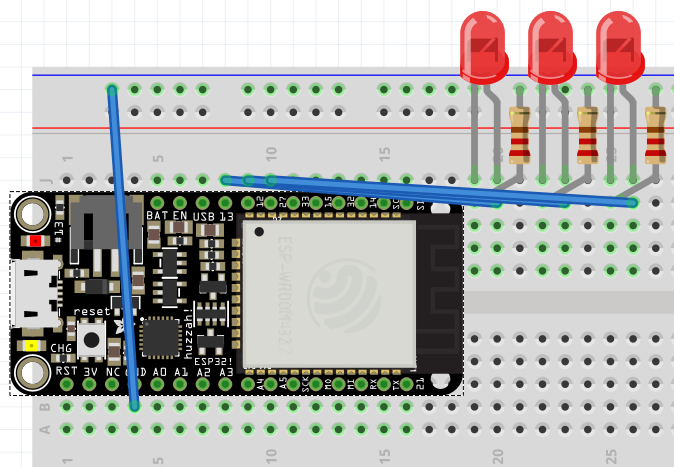
|  |  |
| --- | --- |
| Havaitsija asetetaan minne tahansa (pääaseman ja TVT wifin kantaman sisällä), missä haluat havaita liikettä ja annetaan nollata jotain tasaista pintaa vasten 3 metrin etäisyydellä. (3 metriä on tarkka mitta, koska ultraäänisensorin maksimipituus on 5 metriä, ja etäisyys on rajoitettu koodissa 3 metriin) | Henkilö kävelee havaitsijaan. (Havaitsija ympyröity punaisella) |
|  |  |
| Saatuaan hälytys signaalin langattomasti. Pääasema hälyttää ledejä. | Samanaikaisesti kamera lähettää kuvan Blynkkiin TVT Wifin välityksellä. |
|  |  |

**Kytkentäkaaviot**

Kytkentäkaavio havaitsijalle (Sama kaavio toistuu kaikille 3 havaitsijalle):



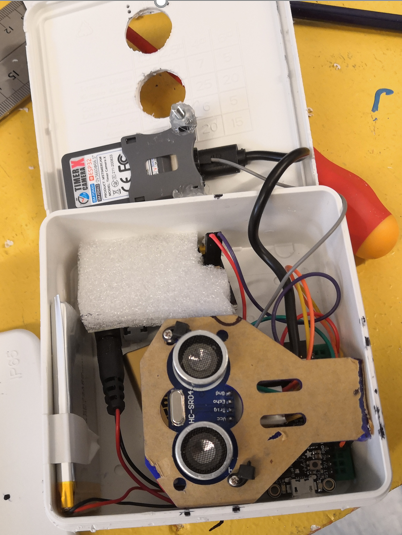
Kytkentäkaavio pääasemalle:



Kuva havaitsijasta koppa kiinni:



Kuva havaitsijasta koppa auki:



**Oma pohdinta**

Työ onnistui minusta hyvin, vaikka oli haastava. Ongelmina olivat kameran käyttöönotto ja ajurien asennus, sillä kamerasta löytyi todella vähän esimerkkejä, myös ensimmäisen kopan tekemisessä mittoja piti siirtää moneen otteeseen. Jos laitteesta haluaisi tehdä oikean turvallisuusjärjestelmän pitäisi saada kuvat tallennettua jonnekin ja laitteen pitäisi saada kuvat lähetettyä puhelimeen ilman että on samassa verkossa. Antaisin itselleni työstä arvosanaksi 5

**Projektissa sovelletut koodiesimerkit:**

**ESP32 Arduino HTTP server: Getting query parameters**

Linkki: <https://techtutorialsx.com/2017/12/17/esp32-arduino-http-server-getting-query-parameters/>

# ESP32 Client-Server Wi-Fi Communication Between Two Boards

Linkki: <https://randomnerdtutorials.com/esp32-client-server-wi-fi/>

# M5stack Timer Camera X ESP32 PSRAM OV3660 Review

# Linkki: [https://www.youtube.com/watch?v=qT9to3wNn3g](#_Hlk71539797)

# Smart Door Bell using ESP32 Cam/ESP-EYE & Blynk

Linkki: <https://www.youtube.com/watch?v=gV3Pr2VrkFg&t=0s>

**M5Stack Community**

Linkki: <https://community.m5stack.com/topic/2424/timer-camera-x/2>

**M5Stack Arudino IDE Developenment**

Linkki: <https://docs.m5stack.com/en/quick_start/timer_cam/quick_start_arduino>

**Projektissa käytetyt resurssit:**

4Kpl esp-32 Feather aka Huzzah

3Kpl HC-SRD04 Ultraäänisensoria

3Kpl Usb paikkaista jännitesovitinta.

3Kpl 9V paristoja

1Kpl Breadboard.

noin 18Kpl johtoja

1Kpl minun tekemä Led piiri (sis 3 220ohm vastusta, 3 punaista lediä,4 johtoa, pieni pala piirilevyä.)

3Kpl nepparista jännitesovittimeen adapteri johtoa

3Kpl M5Stack timercamera x kameraa

3Kpl Usb-C johtoa

4Kpl 3,7V Akkua Huzzaheihin

**Projektissa käytetyt kirjastot:**

**kirjasto**: Wifi

**Löytyy**: arduinon omasta kirjastohakemistosta.

**kirjasto**: HTTPClient

**Löytyy**: arduinon omasta kirjastohakemistosta.

**kirjasto**: Blynk

**Löytyy**: arduinon omasta kirjastohakemistosta.

**kirjasto**: NewPing

**Löytyy**: <https://bitbucket.org/teckel12/arduino-new-ping/downloads/>

**kirjasto**: ESPAsyncWebServer

**Löytyy**: <https://github.com/me-no-dev/ESPAsyncWebServer>

**kirjasto**: AsyncTCP

**Löytyy**: <https://github.com/me-no-dev/AsyncTCP>

**kirjasto**: esp32-camera

**Löytyy**: <https://github.com/espressif/esp32-camera>

**kirjasto**: M5Stack timercam

**Löytyy**: [https://github.com/m5stack/TimerCam-arduino#readme](https://github.com/m5stack/TimerCam-arduino%23readme)

**kirjasto**: CP2014 USB ajuri

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

**kirjasto**: M5Stack

**HUOM:** Arduino iden asetuksiin pitää laittaa seuraava linkki [**https://m5stack.oss-cn-shenzhen.aliyuncs.com/resource/arduino/package\_m5stack\_index.json**](https://m5stack.oss-cn-shenzhen.aliyuncs.com/resource/arduino/package_m5stack_index.json) ja sitten voi hakea board managerista M5Stack board ajuria

**Löytyy**: Arduinon omasta boardmanagerista.

**kirjasto**: Adafruit Huzzah ESP32 Kirjasto

**HUOM:** Arduino iden asetuksiin pitää laittaa seuraava linkki <https://dl.espressif.com/dl/package_esp32_index.json> ja sitten voi hakea board managerista Industruino SAMD Boards (32-Bits ARM Cortex-M0+) board ajuria

**Löytyy**: Arduinon omasta boardmanagerista.