Science, Engineering & Education, 9, (1), 2024, 13-16

Video Monitoring with Esp32

Teodora Gardeva,
Nikoleta Nikolova, Dinko Dinev*

Vocational High School "Vasil Levski"12 Tsar Ivan Alexander St., Yambol 8600, Bulgaria

Received 05 Septenber 2024, Accepted 11 November 2024
DOI: 10.59957/see.v9.i1.2024.2

ABSTRACT

This paper presents the design and development of a video monitoring system based on the ESP32 microcontroller. The project incorporates various key components including a 1.8-inch colour display for data visualization, an ultrasonic probe for motion detection, audio output through amplifiers, and a power module to ensure continuous operation. The system is programmed using the Arduino IDE and ESP-IDF platform to manage motion detection and audio-visual feedback. This affordable and versatile project aims to demonstrate the capabilities of ESP32 in building smart home security systems and can be expanded with additional sensors in future iterations.

Keywords: ESP32, video monitoring, motion detection, smart home automation, microcontroller systems.

INTRODUCTION

The topic of Video Monitoring with ESP32 was chosen due to its potential for innovative applications. Although there are other similar developments, our goal was to create a functional and affordable video monitor project. The video monitor with ESP32 sparks interest in the development of smart home automation and security systems. Technological advancements allow us to create such systems that are more accessible and easier to use. Previous research and projects, such as signal systems with microcontrollers, inspired us to build a more complex and functional system.

The project includes various components such as a colour display, an ultrasonic probe, amplifiers, a power module, and others [1]. The aim of the project is to create a system that can detect pulse and generate audio-visual signals in response.

EXPERIMENTAL

To implement the project, we used the ESP32 microcontroller, which is extremely powerful and functional, a colour display for visualizing data, an ultrasonic probe for motion detection, amplifiers for audio output, a power module, and other components. We provide the functionality for motion detection and audio-visual signal management programmatically.

**Correspondence to: Dinko Dinev, Vocational High School "Vasil Levski", 12 Tsar Ivan Alexander St., Yambol 8600, Bulgaria, E-mail: dinev_1989@abv.bg*

The key components are the ESP32 microcontroller, the 1.8-inch colour display, the ultrasonic probe, the PAM8403 amplifier with speaker, the LM393 amplifier, the 5V power module, the 10K potentiometer, various electronic components (resistors and capacitors), a 7×9 cm double-layer prototype board, and a 9V/1A network adapter.

The Arduino IDE was utilized for programming the ESP32 microcontroller, taking advantage of the rich library of functions and examples for working with various sensors and peripherals. Additionally, we used the ESP-IDF (ESP32 IoT Development Framework) platform for lower-level programming, which gave us greater flexibility and control over the hardware resources of the device [2].

For the realization of the video monitor, we used the following components (Fig. 1):

- ESP32 Microcontroller: The main component that controls all others.
- 1.8-inch Colour Display: Used for visualizing the video monitor.
- Ultrasonic Probe with a working frequency of 3 MHz: For detecting the distance to objects.
- Amplifier module with speaker - PAM8403: For sound reproduction.
- Amplifier with LM393: For processing signals from the ultrasonic probe.
- 5V Power Module - AMS1117: For powering the components.
- 10K Potentiometer: For adjusting the sound.
- Electronic components - resistors and capacitors: For connecting the components.
- 7×9 cm double-layer prototype board: For mounting the components.
- 9V/1A Network Adapter: For powering the entire project.
- Representative case: To give a finished look to the video monitor.

RESULTS AND DISCUSSION

The project is built on a prototype board where all components are connected (Fig. 2). The ESP32 microcontroller controls the colour display and the ultrasonic probe. The potentiometer is used to adjust the probe's sensitivity. The amplifiers provide sound signal through a speaker connected to the PAM8403 module. The network adapter provides power to the entire project [3].

The process of assembling the entire system was carried out step by step. Initially, we connected the ESP32 microcontroller to the colour display and checked the basic functionality. Then we added the ultrasonic probe and programmed the motion detection algorithm. Once this stage was successfully completed, we integrated the amplifiers and the speaker to play audio signals when motion was detected. An actual measurement involves placing the video

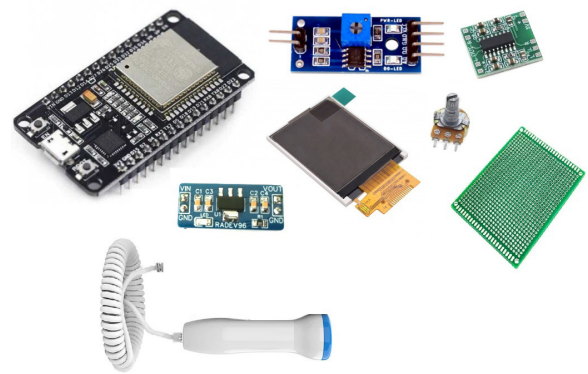

Fig. 1. Components for project implementation.

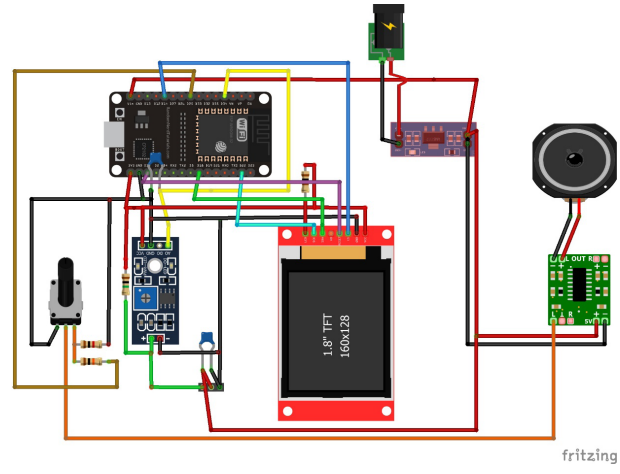

Fig. 2. Connection diagram of all components.

```ino
#include <Adafruit_GFX.h>
#include <Adafruit_ST7735.h>
#include <SPI.h>

#define ST7735_LIGHTGREY   0xC618

#define TFT_CS 14
#define TFT_RST 15
#define TFT_DC 32

Adafruit_ST7735 tft = Adafruit_ST7735(TFT_CS, TFT_DC, TFT_RST);

#define SPEAKER_PIN 25
#define SAMPLING_RATE 8000
#define HEART_RATE 70
#define AMPLITUDE 255
#define PWM_CHANNEL 0

int SENSOR_PIN = 34;
int SENSOR = 0;

#include "Images.h"


void InitDisplay()
{
  tft.fillScreen(ST7735_BLACK);

  tft.setCursor(0, 0);
  tft.setTextColor(ST77XX_CYAN);
  tft.setTextSize(2);
  tft.println("Video zone with ESP32");
  tft.drawLine(0, 33, 128, 33, ST7735_LIGHTGREY);
}


void setup(void)
{
  Serial.begin(115200);

  tft.initR(INITR_BLACKTAB);

  tft.setRotation(0);

  tft.fillScreen(ST7735_BLACK);

  ledcSetup(PWM_CHANNEL, SAMPLING_RATE, 8);
  ledcAttachPin(SPEAKER_PIN, PWM_CHANNEL);

  InitDisplay();
}


void loop()
{
  SENSOR = analogRead(SENSOR_PIN);

  delay(50);

  if(SENSOR >= 2280 && SENSOR <= 3600)
  {
    generateHeartbeatSound();
  }

  if(SENSOR >= 4000)
  {
    displayImageWarning();
  }
}


void generateHeartbeatSound()
{
  int durationBetweenBeats = 60000 / HEART_RATE;
  int durationOfBeat = durationBetweenBeats / 10;

  for (int t = 0; t < durationOfBeat; t++)
  {
    ledcWrite(PWM_CHANNEL, AMPLITUDE);
    delayMicroseconds(1000000 / SAMPLING_RATE / 2);
    ledcWrite(PWM_CHANNEL, 0);
    delayMicroseconds(1000000 / SAMPLING_RATE / 2);
  }

  displayImage();

  delay(durationBetweenBeats - durationOfBeat);
}


void displayImage()
{
  int pulse = map(SENSOR, 2280, 3500, 65, 85);

  InitDisplay();

  for(int i = 0; i <= 2; i++)
  {
    tft.drawRGBBitmap(32, 38, heart_small, 54, 54);
    delay(800);

    tft.fillRoundRect(32, 38, 70, 70, 0, ST77XX_BLACK);
    tft.drawRGBBitmap(32, 38, heart_big, 64, 64);
    delay(800);

    tft.fillRoundRect(32, 38, 70, 70, 0, ST77XX_BLACK);

    tft.setCursor(32, 115);
    tft.setTextColor(ST77XX_RED);
    tft.setTextSize(1);
    tft.print("Pulse: ");
    tft.print(pulse);
  }
  InitDisplay();
}


void displayImageWarning()
{
  tft.fillScreen(ST77XX_WHITE);
  tft.drawRGBBitmap(32, 32, warning, 64, 64);
  delay(1000);
}
```

Fig. 3. Programmable code.

monitoring system in a controlled environment, where the ultrasonic probe detects motion within its range. Once motion is detected, the system triggers the audio-visual feedback, displaying the detection status on the colour screen and emitting a sound through the speaker. The data is processed and visualized in real-time, making the system responsive to environmental changes. This enables users to quickly assess the system's performance through immediate feedback.

After testing the entire system, compiling and uploading the source code (Fig. 3), and ensuring

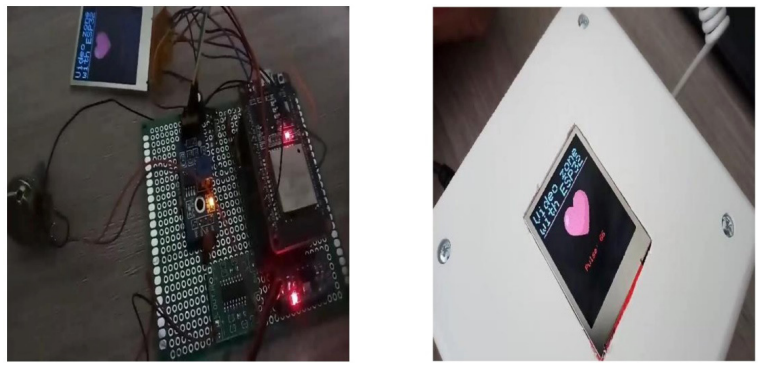

Fig. 4. Video zone with ESP32 in finished form.

that everything functions correctly, we placed all the components in the representative case. This allowed us to present the project to potential users in a more professional manner and as a finished product.

The final design of the video monitor with ESP32 is presented in Fig. 4, where both the component connections and the workstation computer can be observed.

CONCLUSIONS

We created a functional video monitor with ESP32 that can be used for various applications. In the future, we can expand the project by adding additional sensors and functionalities, such as capabilities for connecting to other home automation systems [4]. The developed video monitoring system can be adapted for a range of applications, including smart home security to monitor entrances, baby monitoring systems to detect movement in a child's room, and industrial settings for detecting unauthorized access. Additionally, with further enhancements, it could be integrated into broader home automation systems for environmental monitoring and surveillance.

The video monitor with ESP32 (Fig. 4) represents a successful implementation of a smart system. It demonstrates the capabilities of microcontrollers and various sensors for motion detection. In the future, we can expand the system's functionality by adding additional sensors such as a surveillance camera or environmental monitoring sensors. This will make the system even more comprehensive and functional for users.

REFERENCES

1. N. Kolban, Getting Started with ESP32, Publ. Independently published, 2017.
2. A. Kurniawan, ESP32 Development Workshop, Publ. Independently published, 2018.
3. A. Kurniawan, ESP32 Projects and Applications, Publ. Independently published, 2019.
4. H. Kriegler. ESP32 for Busy People, Publ. Independently published, 2020.