Designing ESP32 Base Shield Board for IoT Application

Helmi Jamaludin
Politeknik Sultan Idris Shah
helmi_jamaludin@psis.edu.my

Abstract
Nowadays, the ESP32 chip's popularity grows that makes it a perfect solution to build an Internet of Things (IoT) application. During the prototype phase, the ESP32 development board can be plugged on top of the shield board to extend its general-purpose input/output (GPIO) pins. In this paper, an ESP32 base shield board with onboard LEDs have been designed to overcome the coding and circuit troubleshoot problems among beginners. This proposed shield board consists of the male header pin, which extends all the GPIO pins of DOIT ESP32 DevKit V1 with additional header pins for 5V, 3.3V, and GND. The GPIO pins are connected to LEDs, which function as an indicator that shows either the GPIO pins receive signal 5V (logic 1) or 0V (logic 0). Two sample application are illustrated in the paper. The result showed that the ESP32 Base Shield Board provides easy hardware interfacing and flexibility in connecting the peripheral unit and Node MCU. Its onboard LEDs are not as output only but act as an indicator to know what happens on that pin and make it easy to troubleshoot either coding or circuit problems. This product is a beginner-friendly design, low cost, and reliability for IoT testing prototype and educational purposes.

Keywords: ESP32, shield board, troubleshoot

1.0 Introduction

The Internet of Things (IoT) is becoming more and more popular today, with the main advantage that it can provide effective connectivity to ensure reliable remote communication and data transfer in wireless communication. Various IoT applications were developed by electronic hobbyists and professionals using a variety of modules and microcontrollers such as Xbee, WhizFi, and certain Arduino boards. However, the devices are either quite expensive or large in terms of weight and size. Therefore, the ESP32 device becomes a powerful microcontroller with build-in wifi and Bluetooth, designed to be perfect IoT devices (Maier, Sharp & Vagapov, 2017).

Nowadays, the popularity of the ESP32 chip grows, and now both hardware variants of this chip and various branches of its software development are developing (Babiuch, Foltýnek & Smutný, 2019). There are two forms for ESP32, which are the chip form and the module form, which have different sizes and numbers of pins. There are some development boards with an ESP32 chip or ESP32 module, and it can categorize into two models of board-based ESP32. The first board models are the development boards officially manufactured by Espressif. The second models are from their partners or personal makers (Agus, 2019).

The board of ESP32 is a bread-board friendly, ready to use solution for testing and educational purpose (Maier et al., 2017). During the prototype phase, the ESP32 development board can be plugged on top of the shield board to

extend its capabilities, which is easy to access the general-purpose input/output (GPIO) pins on Node MCU. The shield board is easy to mount, no need for bread-boards or jumpers, and cheap to produce.

The DOIT ESP32 DevKit v1 is one of the development boards created by DOIT to evaluate the ESP-WROOM-32 module. It is based on the ESP32 microcontroller with a dual-core system (Xtensa LX6) that boasts wifi, Bluetooth, Ethernet, and low power support all in a single chip. It also has a wide variety of peripherals available, like capacitive touch, ADCs, DACs, UART, SPI, I2C, and much more. It comes with a built-in hall effect sensor and a built-in temperature sensor. Power to the DOIT ESP32 DevKit v1 is supplied via the onboard USB Micro B connector or directly via the "VIN" pin. It comes with two versions of this board, with 30 pins and with 36 pins. The pins are labeled at the top of the board, so it is easy to identify the pins to connect peripherals. It comes with onboard RESET (EN) and BOOT buttons. Additionally, it comes with a USB-to-UART interface, so that it can easily program it using Arduino IDE or other development environments and comes with a voltage regulator circuit. The board can be powered using the micro-USB connector or the VIN or 3.3V pins, and it does not come with the battery connector.

For this purpose, there are a few shield boards designed for different ESP32 development boards and produced by different individuals or firms like Keyestudio ESP32-IO shield, ESP32 IoT Shield Based V1.0 and etc. All the design of shield boards has the same function, which is to bring out all the GPIO pins of the ESP32 development board in the form of pin headers. Besides, to connect other sensors easily, the shield board also has additional pin headers for powering external sensors/modules with DC 3.3V or 5.0V voltage.

Martin (2016) stated that the challenge for learning embedded system was lack of sufficient knowledge and skills from background related field. Embedded system is an interdisciplinary field combining area such as computer engineering, electronic engineering, and automatic control. When a young developer or students lack knowledge of electronic basics, they might have difficulty in troubleshooting the circuit. This is because when the microcontroller does not work, they do not know whether the problem is due to wire connection or coding. They need to spend time identifying the cause of the problem, either the hardware or software part. Besides, from preliminary study, there are no shield board on the market embeds the LEDs as a voltage indicator for GPIO pins of ESP32.

Therefore, this study aims to design and develop the ESP32 base shield board with LED at each of the GPIO pins. The onboard LEDs work as output and as an indicator to show what is happening on that pin. When the pin is used as input, we can see whether the signal is received by just looking at the LEDs' changes. The ESP32 base shield board also provides an easy connection to all the sensors and actuators devices/modules by eliminating the need for bread-board, and it also gives extend the 5V, 3.3V, GND pins.

2.0 Methodology

The methodology of research design consists of three phases. The first phase is designing the prototype of ESP32 Base Shield Board, the second phase is designing the programming of ESP32 controller and third phase is testing the prototype.

2.1 Hardware design of ESP32 base shield board

The design has two main parts: header pin for each GPIO and LED circuit. The ESP32 base shield board's circuit consists of the male header pin, which extends all the 30 GPIO pins of ESP32 DevKit DoIt with eight male header pins for 5V 3.3V, and GND. Each GPIO pin is connected to an Enhancement-mode N-channel MOSFET that acts as a switch circuit for turning on and off an LED. The LED's function as an indicator that shows either the GPIO pins receive signal 5V (logic 1) or 0V (logic 0).

2.2 Fabrication of printed circuit board

The schematic circuit is drawn using Autodesk Eagle software is seen in Figure 1. Then, the next step is designing the printed circuit board (PCB). While drawing the printed circuit, ultimate attention was shown to keep the size as small as possible, and it provides the ease of use. On this design, surface mount components were used, and two-layered PCB. The printed circuit diagram, which has been drawn considering all those points, is seen in Figure 2. In order to avoid possible design flaws while drawing the printed circuit, the 3D form of the set, seen in Figure 3, was examined with the utmost care. After all the executed controls, it was proceeded to fabricate the printed circuit board (Figure 4), and the final form, which is seen in Figure 5, was obtained.

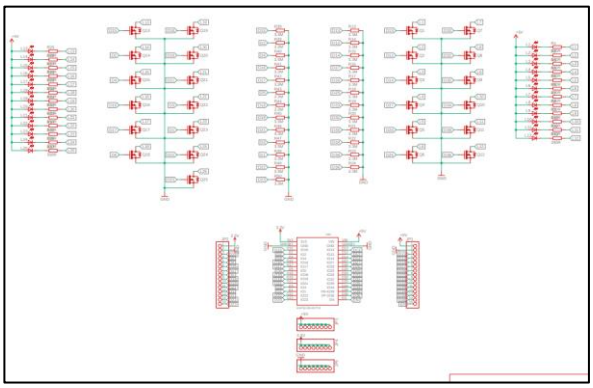

Figure 1: The schematic circuit of ESP32 base shield board

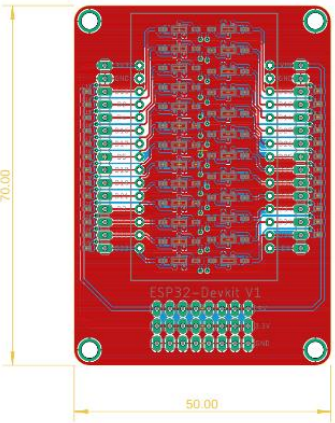

Figure 2: The printed circuit diagram of ESP32 base shield board

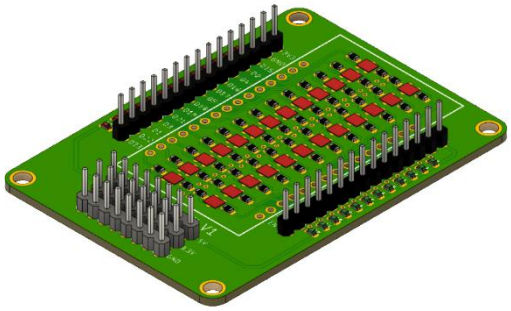

Figure 3: The 3D appearance of ESP32 base shield board

Figure 4: The PCB design of ESP32 base shield board

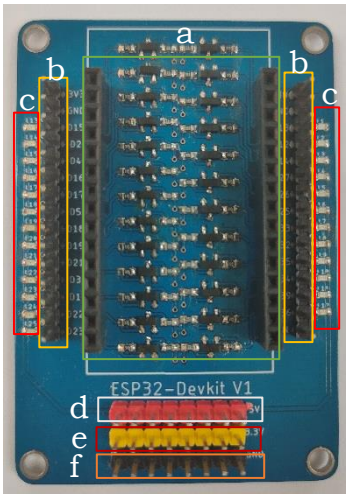

Figure 5: The ESP32 base shield board

Table 1: Design specification

Label	Description
a	Interfaces of ESP32 devkit doit board
b	Pins of ESP32 devkit doit board
c	LED
d	5V
e	3.3V
f	GND

Table 1 show the design specification of ESP32 base shield board from Figure 5. The base shield board is supplied by 5VDC from onboard ESP32 development board via micro-USB connector.

2.3 Software Design of the ESP32 base shield board

Arduino Integrated Development Environment (IDE) is an open source platform which support ESP32 development board (Bhadane & Lal (2018). The Arduino IDE provides editor for typing the program for ESP32 using C or C++. We need to select the microcontroller board types DOIT ESP32 DEVKIT VI from board manager of Arduino IDE. After the compilation of the program, the hex file is uploaded or burned into the flash memory of the ESP32 development board. The execution of program starts immediately on the hardware application system. Each application has two procedures: one to write the programming code by Arduino IDE software, another one to implement the hardware connection. The complete connection between the hardware section and the programming section is shown in Figure 6, via micro-B USB cable.

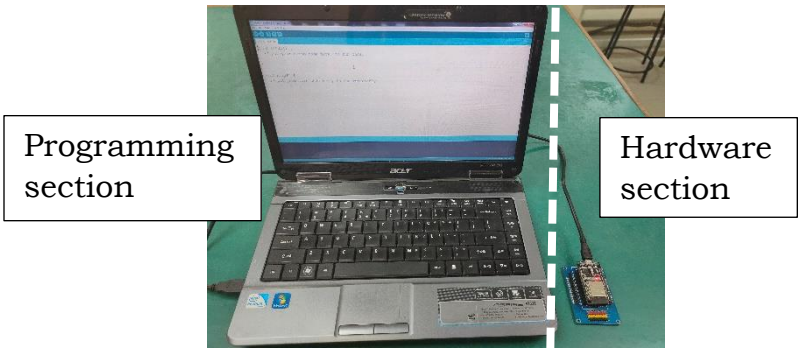

Figure 6: Combination of programming and hardware section via micro-B USB cable

2.4 Testing procedure

In order to show the design of the ESP32 base shield board can facilitate the process of troubleshooting circuits, two sample application was carried out. The ESP32 DevKit DoIt is plugged on top of the ESP32 base shield board. The coding is programmed by Arduino IDE installed on Microsoft Windows 10, 32 bit operating system.

The first executed sample was the LED application with an active high button. The pictures related to those samples are given in Figure 7. The LED will turn on when we press the button then turn it off when we release the button. The button is connected at pin 34 as INPUT, and the onboard LED at pin 16 as OUTPUT. The active high button means when the button is pressed, then the signal voltage sent to NodeMCU will be HIGH (5V), and when the button is not pressed, the signal voltage send to NodeMCU is LOW (0V). Figure 8 showed when we pressed the button, the onboard LED turn on at pin 34 and pin 16. It shows the LED provides very good visual/light as an indicator that the button successfully sends the signal voltage HIGH to NodeMCU at pin 34. At the same time, the LED turn on at pin 16 shows the NodeMCU successfully sends out the signal voltage HIGH to the LED.

The second executed sample used the same programming as the first sample. A relay module was connected at pin 16 as OUTPUT and the button at

pin 34 as INPUT. This sample aims to activate the relay module when the button is pressed. Figure 9 showed when the button is pressed, the onboard LED turns on at pin 34 and pin 16. However, the relay module does not activate, although the OUTPUT pin 16 is HIGH (5V). From observation, the problem comes from the hardware part, not from the programming part. So, we need to troubleshoot the hardware part only. After the continuity test, we found the broken blue wire jumper connected between ESP32 pin16 and relay module input pin. After replacing the blue wire jumper with the new red wire jumper, the relay module successfully activated when the button is pressed, is seen in Figure 10.

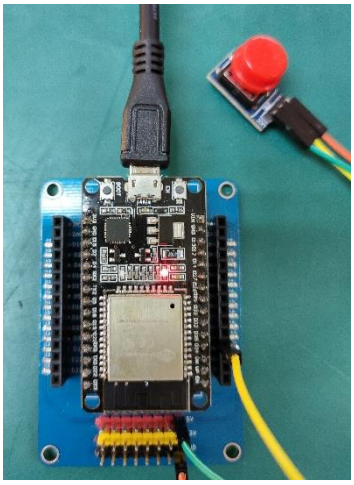

Figure 7: The ESP32 shield board connected with active high button

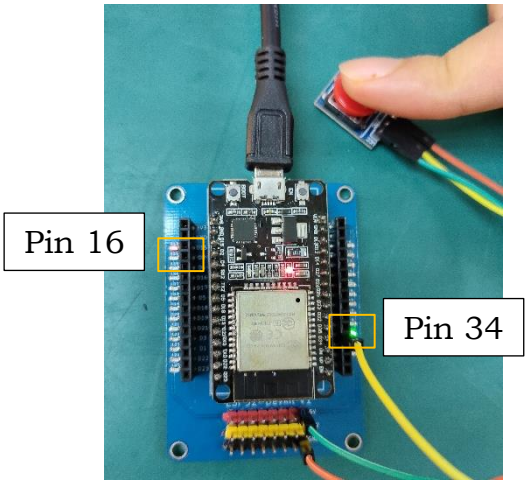

Figure 8: The LEDs at pin 16 and 34 turn on when the button is pressed

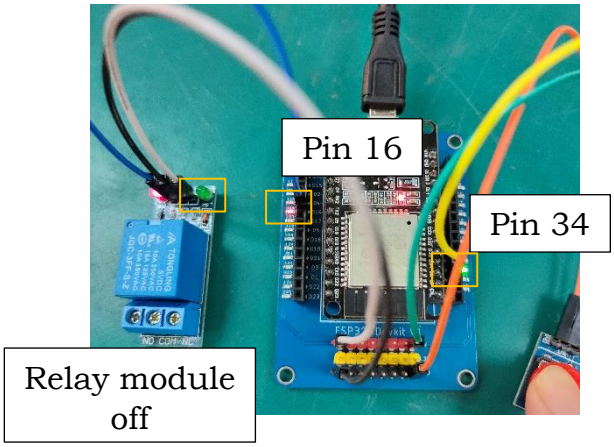

Figure 9: The relay module does not activate when the button is pressed.

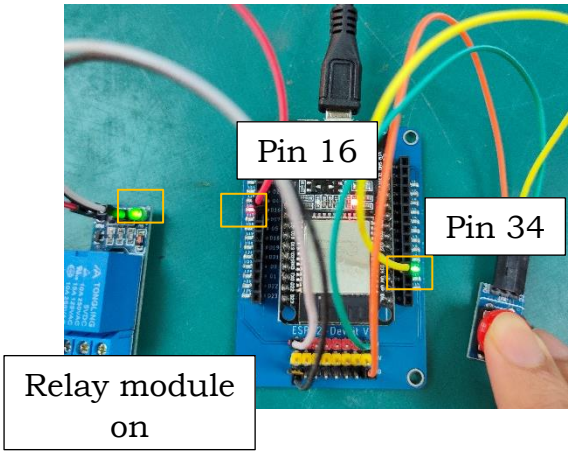

Figure 10: The relay module is activated when the button is pressed.

3.0 Result

The result from the two samples testing are showed in Table 2 and Table 3. On testing the LED as voltage indicator at I/O pins, the test was performed

by viewing the LED light as indicator to show the I/O pin received the digital signal up to 5V as shown in Table 2.

Table 2: LED as voltage indicator test results

Testing	LED indicator	Result	Description
Active high button is pressed	LED at pin 34 (input) and 16 (output) turn on	Successful	Digital signal at pin 34 (input) and pin 16 (output) is 5V (HIGH)
Active high button is not pressed	LED at pin 34 (input) and 16 (output) turn off	Successful	Digital signal at pin 34 (input) and pin 16 (output) is 0V (LOW)

On the troubleshooting test, the test was conducted by identifying the circuit problem is due to wire connection or coding by viewing the LED light at the I/O pin as shown in Table 3.

Table 3: Troubleshooting test results

Testing	LED indicator	Result	Description
Active high button is pressed	LED at pin 34 (input) and 16 (output) turn on but the relay which connected at pin 16, is not triggered.	Not successful	From the observation, the relay is not triggered because of relay circuit connection. Thus, troubleshoot focus to the hardware part only.
Active high button is presses after replacing the wire connected between pin16 (output) to the relay module.	LED at pin 34 (input) and 16 (output) turn on and the relay which connected at pin 16, is triggered.	Successful	Successfully troubleshoot the hardware part.

According to the two samples testing described before, it is observed that "The ESP32 Base Shield Board" provides easy hardware interfacing and flexibility in connecting the peripheral unit and Node MCU. Its onboard LEDs are not as output only but act as an indicator to know what happens on that pin and make it easy to troubleshoot either coding or circuit problems. Thus, it will shorten the time taken during troubleshooting, especially for beginner programmers.

4.0 Conclusion

This study set out to design and develop the ESP32 base shield board with onboard LEDs. The result from testing the ESP32 base shield board show this prototype is a beginner-friendly design because it helps the beginner troubleshoot the coding and circuit also makes all the GPIO pins easily accessible through header pins. The onboard LEDs work as output and also as an indicator to show what is happening on that ESP32 pins. We believe this small size, low cost and reliable product is a solution for IoT testing prototype and educational purposes.

References

Agus, K. (2019). *Internet of things project with ESP32*. Birmingham: Packt Publishing Ltd.

Babiuch, M., Foltýnek, P. & Smutný, P. (2019). Using the ESP32 microcontroller for data processing. *In 20th International Carpathian Control Conference (ICCC) (pp.1-6).* Krakow-Wieliczka, Poland.

Bhadane, P. & Lal, Aparna. (2018). Beginners Approach to the open source programming: case study arduino with ESP32. *In International Journal of Computer Science and Engineering (IJCSE)*, 445-448.

Maier, A., Sharp, A. & Vagapov, Y. (2017). Comparative analysis and practical implementation of the ESP32 microcontroller module for the internet of things. *In Internet Technologies and Applications (ITA) (pp.143-148).* Wrexham.

Martin, T., Martin, G. & Niklas, A. (2007). Experiences from large embedded systems development projects in education, involving industry and research. *Proceeding of the 12th ACM, 4*(1).