**National Textile University, Faisalabad**



**Department of Computer Science**

**Name**: Ayesha

**Reg\_No:** 23-NTU-CS-1020

**Section:** CS 5th A

**Assignment No**: 01

**Course Name:** Embedded IoT

**Question No 1:**

**Short Questions**

**Question No 1:**

**Why is volatile used for variables shared with ISRs?**

**Answer:**

Volatile is used because the values of the variable can change **suddenly** during program execution.

* It stops the compiler from keeping old values and always compels the variable to be read straightway from memory, securing the main code sees improvement made by the ISR.

**Question No 2:**

**Compare hardware-timer ISR debouncing vs. delay()-based debouncing?**

**Answer:**

**Hardware-timer debouncing delay ()-based debouncing**

* It uses a timer to wait a small It pause using delay() after a

time later a button presses. button press

* The timer rush in the In this wait, the CPU can’t

background, so the CPU can do something else.

keep performing other work.

* It’s precise and doesn’t It’s easy but pause the program

Pause the program. for a short time.

**Question No 3:**

**What does IRAM\_ATTR do, and why is it needed?**

**Answer:**

**What it does:** IRAM\_ATTR keeps a function in fast internal RAM rather than in flash memory.

**Why it’s needed:** It confirms interrupt functions run speedily and securely, even when flash memory is working.

**Question No 4:**

**Define LEDC channels, timers, and duty cycle**

**Answer:**

**LEDC channels:**

Channels check separate LEDs or devices utilizing PWM signals. Each channel can hold its own settings for brightness or speed.

**LEDC timers:**

Timers synchronize the **frequency** and **resolution** for the PWM signals utilized by channels. Numerous channels can allocate one timer.

**Duty cycle:**

It is the **percentage of time** a signal remain ON in one cycle.

Higher duty cycle = LED is brighter

Lower duty cycle = LED is dimmer

**Question No 5:**

**Why should you avoid Serial prints or long code paths inside ISRs?**

**Answer:**

You should eliminate **Serial prints or long code** inside ISRs because ISRs must run steadily.

* Serial prints are slow-moving and can retard the ISR.
* Long code paths retain interrupts disabled longer, leading the system to forget other interrupts or become insecure.

**Question No 6:**

**What are the advantages of timer-based task scheduling?**

**Answer:**

**Advantages of timer-based task scheduling:**

1. **Unobstructed:** Tasks run at set times without using delay(), so the CPU remain idle.
2. **Precise timing:** Timers provide precise control over when tasks run.
3. **Multitasking:** Numerous tasks can run steadily at different intervals.
4. **Lesser power use:** The CPU can sleep between timer events.
5. **More responsive:** The system acts swiftly to inputs or interrupt.

**Question No 7:**

**Describe I²C signals SDA and SCL.**

**Answer:**

**SDA:**  
This transport the real data between devices in the I²C communication.

**SCL:**  
This gives regular timing pulses so the master and slave dispatch and read data at the same interval.

**Question No 8:**

**What is the difference between polling and interrupt-driven input?**

**Answer:**

**Polling:** The CPU keeps examining a device or pin repeatedly to check if anything happened.

* Simple but wastes time because the CPU is repeatedly inspecting.

**Interrupt-driven input:** The device informs the CPU when something occurs utilizing an interruption.

* More efficient because the CPU can do other work until it’s notified.

**Question No 9:**

**What is contact bounce, and why must it be handled?**

**Answer:**

Contact bounce occurs when a button or switch is pressed — its metal touch and free very speedily many times before settling.

This creates numerous fasts on/off signals instead of one clean press.

It must be debounced because those extra signals can make the system suppose that the button was pressed numerous times rather than once.

**Question No 10:**

**How does the LEDC peripheral improve PWM precision?**

**Answer:**

The LEDC makes PWM more precise because it utilizes hardware timers to control the signal.

This means it can remodel the LED’s brightness very smoothly and precisely, without minute errors that occur when the CPU does it by software.

**Question No 11:**

**How many hardware timers are available on the ESP32?**

**Answer:**

The ESP32 has four hardware timers, split into two groups:

* Timer Group 0: Timer 0 and Timer 1
* Timer Group 1: Timer 0 and Timer 1

So, there are 4 general-purpose 64-bit hardware timers available on the ESP32.

**Question No 12:**

**What is a timer prescaler, and why is it used?**

**Answer:**

A timer prescaler is a divider that slows down the input clock going to a timer.

It’s used to make the timer count more slowly, allowing you to measure longer time periods or create lower frequencies without overflow.

**Question No 13:**

**Define duty cycle and frequency in PWM.**

**Answer:**

**Duty cycle:**

It shows how lasting the signal stays ON contrast to the total time of one cycle.  
 Example: If an LED is ON for half the time and OFF for half, the duty cycle is 50%.

**Frequency:**  
It is how many times the PWM signal repeats (ON + OFF) in one second.  
 Example: If the signal replicate 1000 times per second, the frequency is 1 kHz.

**Question No 14:**

**How do you compute duty for a given brightness level?**

**Answer:**

Duty Cycle = Ton \* 100

Ton + Toff

**Question No 15:**

**Contrast non-blocking vs. blocking timing.**

**Answer:**

**Blocking timing:**

The CPU stands by and does not a thing unless the delay is completed.

Example: If you utilize delay the CPU pause for 1 second.

**Non-blocking timing:**

The CPU keeps running other works while examining time in the background.  
 Example: The program can blink an LED and examine a sensor at the same interval.

**Question No 16:**

**What resolution (bits) does LEDC support?**

**Answer:**

LEDC support 1-20 bits of resolution on esp-32

**Question No 17:**

**Compare general-purpose hardware timers and LEDC (PWM) timers.**

**Answer:**

**General-purpose timers:**

* calculating time or produce interrupts.
* Can be utilized for delays or scheduling tasks.
* Elastic, but PWM must be set up directly

**LEDC (PWM) timers:**

* Build to produce PWM signals spontaneously.
* Check LED brightness, motor speed, or buzzer tones.
* Elastic and safe to utilize for PWM applications.

**Question No 18:**

**What is the difference between Adafruit\_SSD1306 and Adafruit\_GFX?**

**Answer:**

**Adafruit\_GFX:**

* A graphics library that gives drawing functions such as lines, circles, text, and shapes.
* Works for numerous types of displays.
* Don’t be in touch with the hardware directly.

**Adafruit\_SSD1306:**

* A display driver library particularly for SSD1306 OLED screens.
* Handles transmission with the OLED hardware.
* Often utilize together with Adafruit\_GFX to draw graphics on the OLED.

**Question No 19:**

**How can you optimize text rendering performance on an OLED?**

**Answer:**

* Update only the part of the screen that changes instead of the whole display.
* Keep away from clearing the screen all the time.
* Utilizing smaller fonts
* Utilize a memory buffer to get prepared the screen prior displaying it.

**Question No 20:**

**Give short specifications of your selected ESP32 board (NodeMCU-32S).**

**Answer:**

**CPU:** Dual core, running up to 240Mhz

**SRAM:** 520Kb

**ROM:** 448Kb

**GPIO pins:** 34

**Flash Memory:** 4Mb

**Wi-Fi:** 2.4Ghz

Question No 2:

Logical Questions

**Question 1:**

**A 10 kHz signal has an ON time of 10 ms. What is the duty cycle? Justify with the formula.**

**Answer:**

**Given:**

F = 10kHz

Ton = 10ms

**Solution:**

F = 1 = 1 = 0.0001s = 0.1ms

T 10,000

Duty Cycle = Ton \*100

Ton + Toff

= 10ms \*100

0.1

= 10,000%

This is not possible because the ON time is much longer in comparison to period

**Question No 2:**

**How many hardware interrupts and timers can be used concurrently? Justify.**

**Answer:**

1 interrupt and 4 timers can work concurrently

Justification:

* **Interrupt:** If two interrupts occur at the same time, the CPU swiftly switches between them based on precedence.
* **Timers:** Each timer possess its own counter and prescaler, so it can calculate time on its own. The 4 hardware timers can run separately at the same time, each counting and producing events without awaiting for the others.

**Question No 3:**

**How many PWM-driven devices can run at distinct frequencies at the same time on ESP32? Explain constraints.**

**Answer:**

On the ESP32, PWM is handled by the LEDC peripheral, which has 16 channels (0–15) and 4 hardware timers (2 per timer group).

**Distinct frequencies**

* Each LEDC timer can generate one frequency.
* Multiple channels can share the same timer, but they must have the same frequency.
* So, the maximum number of distinct frequencies you can have at the same time is 4 (one per timer).

**Constraints:**

1. Channels sharing a timer → same frequency, can only differ in duty cycle.
2. Different frequencies → need separate timers.
3. Maximum of 16 channels total → limited by hardware.

**Question No 4:**

**Compare a 30% duty cycle at 8-bit resolution and 1 kHz to a 30% duty cycle at 10-bit resolution (all else equal).**

**Answer:**

**Case 1:** 2^8 = 256

**Case 2:** 2^10 = 1024

Duty Count = 30% \* 255 = 0.3 \* 255 = 77

Duty Count = 30% \* 1024 = 0.3 \* 1024 = 307

* Both have the same duty cycle, so the LED brightness is same.
* 10-bit PWM provides finer control, because it has 1024 levels

**Question No 5:**

How many characters can be displayed on a 128×64 OLED at once with the minimum font size vs. the maximum font size? State assumptions.

**Answer:**

OLED resolution = 128 \* 64

Assume:

Min\_Font= 6 \* 8

Max\_Font= 24 \* 32

**Minimum Font:**

Column = 128 = 21

6

Row = 64 = 8

8

Maximum Font:

Column= 128 = 5

24

Row = 64 = 2

32

**Assumptions:**

* Using fixed width fonts
* No spacing between characters
* Minimum font = 6×8 , maximum = 24×32