National Textile University, Faisalabad

Department of Computer Science

Name:	**Muhammad Haseeb**
Class:	**BSCS 5th (Section – B)**
Registration No:	**23-NTU-CS-1069**
Lab Report:	**Assignment**
Course Name:	**Embedded IOT Systems**
Submitted To:	**Sir. Nasir Mahmood**
Submission Date:	**19th Oct – 2025**

Question:01

1. Why is volatile used for variables shared with ISRs?

Ans:

Volatile is the opposite of const. As const makes sure the value of a variable remains the same throughout its execution. Therefore, a Volatile variable means the value of the variable can be changed; it does not remain the same.

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

Ans:

Hardware Timers:	Delay() - Band:
• It only pauses a specific part of the code, as desired by the user. • The rest of the code keeps on working.	• It pauses everything.

3. What does IRAM_ATTR do, and why is it needed?

Ans:

- Stores the function in internal RAM, not along the main code.
- So it runs only when required, else stays dormant.
- Used to handle interrupts.

4. Define LEDC channels, timers, and duty cycle.

Ans:

Channel: It controls output via a specific pin.
Timer: It sets the frequency.
Duty Cycle: It is the percentage of the ON signal.

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

Ans:

Our ISR (Interrupt Service Routine) needs to be fast.

- Using Serial.print() means giving info about the current condition, then executing the ISR.
- Also, long code would need more CPU time to execute.

Hence, we keep our ISR as simple as possible.

6. What are the advantages of timer-based task scheduling?

Ans:

It helps in running the code smoothly. It uses the concept of multiprogramming, working on multiple parts at once - unlike delay() that pauses everything.

7. Describe I²C signals SDA and SCL.

Ans:

SDA: Carries the data between devices.
SCL: Carries the clock signal, which controls the timing.

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

Ans:

Polling:	Interrupt:
• Continuously checking the hardware if it needs attention.	• A signal generated by the hardware, telling it needs attention.

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

Ans:

When a button is pressed, the metal plates do not make contact once. The current bounces for a few milliseconds. It rapidly turns on and off for a short time before stabilizing. Therefore, we use debouncing.

10. How does the LEDC peripheral improve PWM precision?

Ans:

LEDC (LED Control) uses the hardware timers. Hence, it works on its own without the need for the CPU.

11. How many hardware timers are available on the ESP32?

Ans:

Two timer groups:

1. Group 0: Timer0 & Timer1
2. Group 1: Timer2 & Timer3

Each has two timers. Hence, total 4 Hardware Timers (64-bit).

12. What is a timer prescaler, and why is it used?

Ans:

It is a clock divider. It divides the main clock speed to create a slower time. Why? To get a desired time for timing/waiting.

13. Define duty cycle and frequency in PWM.

Ans:

Duty Cycle: Percentage of time the signal is ON during a cycle.
Frequency: Number of times a signal cycles (on and off) in a second.

14. How do you compute duty for a given brightness level?

Ans:

$$\text{Duty} = \frac{\text{Brightness Level}}{Max\ Brightness\ Level} \times Max\ Duty\ Value$$

$$\text{Duty Cycle (\%)} = \frac{\mathrm{T}_{ON}}{\mathrm{T}_{ON} + \mathrm{T}_{OFF}} \times 100\%$$

15. Contrast non-blocking vs. blocking timing.

Ans:

Non-Blocking:	Blocking:
• Runs everything in parallel. • Uses millis().	• Stops everything until time passes. • Uses delay().

16. What resolution (bits) does LEDC support?

Ans:

Supports up to 20-bit.

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

Ans:

Hardware Timers:	PWM Timers:
Used for scheduling.	Used for creating frequency.

18. What is the difference between Adafruit_SSD1306 and Adafruit_GFX?

Ans:

SSD1306:	**GFX (Adafruit_GFX Library):**
• Controls the OLED display hardware. • It is the "canvas"	• Provides the drawing and text functions. • It is the "art kit."

19. How can you optimize text rendering performance on an OLED?

Ans:

- Update only what is required.
- Use simple fonts.
- Avoid full screen clears.

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

Ans:

Core: Dual-Core processors
Speed: Up to 240 MHz
Memory: 4MB Flash Storage, 520 KB RAM
Connectivity: Wi-Fi and Bluetooth
Pins: 30
Voltage: 3.3V

Question:02

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

Ans:

Data:

f = 10 KHz = 10000 Hz

T_{ON} = 10 ms = 0.01 ms

$T = \frac{1}{f} = \frac{1}{10000} = 0.0001\ ms$

Sol:

$$\text{Duty Cycle (\%)} = \frac{T_{ON}}{T_{ON} + T_{OFF}} \times 100\%$$

$$\text{Duty Cycle (\%)} = \frac{T_{ON}}{T} \times 100\%$$

$$\text{Duty Cycle (\%)} = \frac{0.01}{0.0001} \times 100\%$$

$$\text{Duty Cycle (\%)} = 10{,}000\ \%$$

Conclusion:

10,000% not possible. Therefore; we take 100%.

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

Ans:

Hardware Timers:	Hardware Interrupts:
Because the ESP-32 itself has only 4 timers. Only 4.	All GPIO pins. Each pin can be configured individually; the limit is the number of pins (GPIO).

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

Ans:

PWM Timers:

High Speed = 4
Low Speed = 4

Hence, **8** different frequencies.

Constraint:

1. Channels that share a timer also share the frequency condition.
2. Duty cycle can be different when the timer is the same.

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).

Ans:

8-bit Resolution	**10-bit Resolution**
Max Value = $2^8 - 1 = 255$ At 30% Duty Cycle (8-bit): $= 255 \times \frac{30}{100}$ $= 76.5 \approx 77$ Find Duty % from Value 77 (8-bit): $\frac{77}{255} \approx 30.2\%$	Max Value = $2^{10} - 1 = 1023$ At 30% Duty Cycle (10-bit): $= 1023 \times \frac{30}{100}$ $= 306.9 \approx 307$ Find Duty % from Value 307 (10-bit): $\frac{307}{255} \approx 30.01\%$

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.

Ans:

$$\text{Total Characters} = \frac{\textbf{Total Bytes}\ \text{Screen}}{\text{Total}\ \textbf{Bytes per Character}}$$

Minimum Font	**Maximum Font**
Font Size: 5x8 pixels **Total Bytes for Screen:** $\frac{Width \times Height}{8} = \frac{128 \times 64}{8} = 1024 \text{ bytes}$ **Total Bytes per Character:** $\frac{Width \times Height}{8} = \frac{5 \times 8}{8} = 5 \text{ bytes per character}$ **Final:** Total Characters (assuming 5 bytes per character): $\frac{\text{Total Bytes}}{\text{Bytes per Character}} = \frac{1024}{5} \approx 204 \text{ characters}$	Font Size**:** 16x32 pixels **Total Bytes for Screen:** $\frac{Width \times Height}{8} = \frac{128 \times 64}{8} = 1024 \text{ bytes}$ **Total Bytes per Character:** $\frac{Width \times Height}{8} = \frac{16 \times 32}{8} = 64 \text{ bytes per character}$ **Final:** Total Characters (assuming 64 bytes per character): $\frac{\text{Total Bytes}}{\text{Bytes per Character}} = \frac{1024}{64} \approx 16 \text{ characters}$