Assignment 01

Index

[I. Question 1 2](#_Toc72874100)

[II. Question 2 3](#_Toc72874101)

[III. Question 3 4](#_Toc72874102)

[IV. Question 4 5](#_Toc72874103)

[V. Bibliography 7](#_Toc72874104)

# Question 1

1. **What is the framebuffer?**

A framebuffer is a data structure location set in random-access memory (RAM) that contains a bitmap, which is then used to show graphics on the computer monitor. In the code sample given, the framebuffer is where you can write about the pixel information needed in order to display and play the game. The video controller uses this pixel information by reading the data from the framebuffer 60 times a second, then outputting that information onto the display.

Every computer system and gaming console with a display contains a framebuffer. It is the interface to the video card from the operating system (OS), or in an embedded system, from the firmware. In a gaming console where there are various 3D entities moving around on the screen, such as the player’s character, there will be multiple framebuffers. The video controller will display what is on the front framebuffer. Meanwhile, we will be manipulating the data on the back buffer. When the back buffer is done being updated, the framebuffers are flipped. The video controller will then take the data from the new framebuffer in order to display a new image. Essentially, two arrays are used: one to display to the user, and one to fill up in the meantime. These are then switched.

Flipping the framebuffers mid-stream is not ideal because the video controller will be in the middle of changing the display, based on the values it had received in the original framebuffer. When running the Pacman game in QEMU, this is what causes the screen to continuously flash, as the game display is constantly being redrawn.

1. **How does the *for* loop above make the screen black?**

480\*640 refers to the resolution of the screen. This means that there are 640 pixels in the x-direction, i.e. horizontally across, and 480 pixels in the y-direction, i.e. vertically up-down. In total, the screen resolution of the display will be 307,200 pixels. This is the size of the framebuffer used to play the game.

The framebuffer fb is an integer pointer. fb[i] = 0 will write a zero into the specified location. This will write four byte values, each of which represent the primary colours that are used to display images: red, green, and blue (RGB). Each pixel of the display will be set to an RGB colour intensity (0-255), which, when viewed as a whole, will create an image on the screen for the user to understand. Setting all of the colour values in the framebuffer to zero will create a black output on the screen.

# Question 2

1. **What is the esp8266 WiFi module? Give a brief overview of the IoT and smart home devices based on the esp8266 WiFi module. Refer to firmware for such devices e.g. tasmota, esphome, mongoose OS in your answer.**

The ESP8266 Wi-Fi module is a Wi-Fi chip this is commonly used in embedded devices that are wirelessly connected to the Internet. These devices are generally inexpensive, making them suitable for smart home applications. The devices can be tinkered by the user quite easily, to be adopted to their automated household needs. ESP8266 will allow for any microcontroller to access your Wi-Fi network with its TCP/IP protocol stack included in the module.

ESPhome is a system to control your ESP8266/ESP32 by simple yet powerful configuration files and control them remotely through Home Automation systems. [1] ESPHome supports all devices that use the ESP8266 Wi-Fi module, using the native ESPHome API to connect compatible devices directly to Home Assistant. It allows the user to create firmware for their ESP8266 board. ESPHome works by reading in a YAML configuration file, this is then used to create the custom firmware binary to suit the user’s needs. The setup wizard helps the user to choose their correct board type.

Tasmota is an open-source firmware that is used for smart home integrations. Using its Home Assistant automation solution, it prioritises local control and privacy. It is available for ESP8266 based devices, enabling these devices to connect to the Web, controlled via Hypertext Transfer Protocol (HTTP), Message Queuing Telemetry Transport (MQTT), and KNX. MQTT uses the TCP/IP protocol. During the initial setup, the user must ensure that MQTT is correctly configured for Home Assistant and Tasmota. Within Home Assistant, devices using Tasmota can be added via the preferred method of utilising the official Tasmota integration. Alternatively, the devices can be added by manually editing the configuration file, or by means of MQTT discovery.

Mongoose OS is an open-source operating system designed for low-power microcontrollers, such as the ESP8266. The Internet of things (IoT) development framework is partnered with AWS IoT, Google Cloud IoT, and Microsoft Azure IoT, meaning that is offers support and cloud integration with all of these systems. These cloud systems require Transport Layer Security (TLS). The OS provides over-the-air updates and remote management.

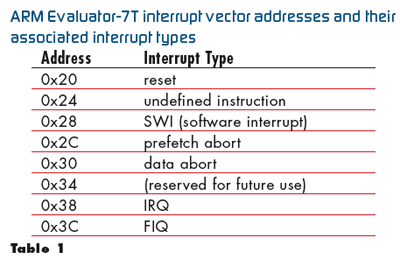
# Question 3

See t.c and vid.c

# Question 4

1. **What is a vector table and give an example?**

In regards to microcontrollers, a vector table is a data structure that is used to store information about interrupt handlers and interrupt requests. The vector table stores the reset value of the stack pointer, and the start addresses for all exception handlers. Each line of the vector table is called an interrupt vector. A vector is the address of an interrupt handler.



Example vector table from Embedded.com [7]

1. **What is an interrupt handler and give an example?**

An interrupt handler is a message from the hardware device to its CPU, alerting the CPU to stop what it is currently doing, and instead to respond to the interrupt message. If the CPU was not busy giving its resources to a higher priority task, it will suspend its current thread, and will manage the interrupt. When the interrupt has been dealt with, the handler informs the CPU that it can continue with the task that it had been doing. eg: pressing a key on the keyboard.

1. **What is the PrimeCell Vectored Interrupt Controller (PL190) and how is it used to control interrupts from several different devices? Use the UART device as an example in your explanation.**

The PrimeCell Vectored Interrupt Controller (VIC) is an Advanced Microcontroller Bus Architecture (AMBA) compliant, System-on-Chip (SoC) peripheral that is developed, tested, and licensed by ARM. [9] It helps in reducing interrupt latency, such as by providing vectored interrupt support for all interrupt sources. The PrimeCell VIC is a software interface to the hardware device’s interrupt system. This interface is used to determine where the interrupt is coming from. It provides the vector address of the service routine.

In ARM systems, there are two different levels of interrupts: fast interrupt request (FIQ) for fast, lower latency interrupt handling; and interrupt request (IRQ) for general interrupt. The PrimeCell VIC supports 32 vectored IRQ interrupts. The user can program the priority of each of the vectored interrupts, meaning that the order in which each interrupt is to be dealt with can be adjusted based on the need of the user. To do this, the values in the vector priority registers are modified. In the case that there is more than one interrupt set with the same priority level, the fixed hardware priority levels will be used to determine which interrupt will be tended to first. This is also the case for when the priority registers have not been programmed.

1. **Explain how you would write a driver for an UART device which makes data available in a circular buffer -** [**https://en.wikipedia.org/wiki/Circular\_buffer**](https://en.wikipedia.org/wiki/Circular_buffer)**.**

A circular buffer is a data structure that earns its name from its shape: it uses a single, fixed-size buffer, connected end-to-end, much like a circle. It is particularly useful when using First In, First Out (FIFO) as it does not need to move any of its stored elements around when they are removed from the buffer.

# Bibliography

1. ESPHome  
   <https://esphome.io/>
2. Home Assistant

<https://www.home-assistant.io/>

1. Tasmota

<https://tasmota.github.io/docs/About/>

1. What is Tasmota and What Can it do for You?

<https://iotrant.com/2018/10/18/what-is-tasmota-and-what-can-it-do-for-you/>

1. Mongoose OS

<https://mongoose-os.com/>

1. Cortex-M3 Devices Generic User Guide

<https://developer.arm.com/documentation/dui0552/a/the-cortex-m3-processor/exception-model/vector-table>

1. Modeling interrupt vectors

<https://www.embedded.com/modeling-interrupt-vectors/>

1. ARM PrimeCell Vectored Interrupt Controller (PL192) Technical Reference Manual

<https://developer.arm.com/documentation/ddi0273/latest>