	RV College of Engineering® Department of Computer Science and Engineering Improvement Test and Quiz Paper		
Course & Code	IOT and Embedded Computing (CS344AI)	Semester: 4 th Sem BE	
Date : Aug 2024	Duration:120 minutes	Max.Marks:(10+50)=60 Marks	Staff : KB, SDV, MSS, MH
USN :	Name :		Section : A/B/C/D/CD/CY

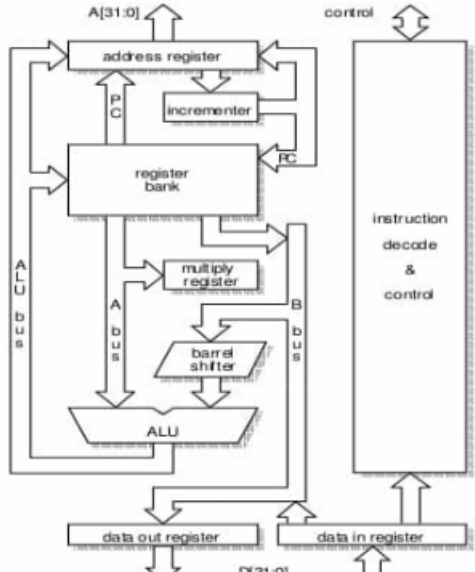
NOTE: Answer all the questions from Part-A (10 M) and Part-B (50 M)

Sl.no	PART - A	Marks
1	Suggest any one application of Level 5 and Level 6 IOT deployment. Refer Reference book for many applications	2
2	Describe an Example of IoT service that uses publish-subscribe communication model. Name the popular application layer protocol for publish-subscribe model used in resource constraint IOT systems. MQTT IS USED EXTENSIVELY FOR UPLOADING SENSOR DATAS TO CLOUD. Weather Monitoring Systems, sensors publish, users/apps subscribe for the sensor data	2
3	Name the pins provided by RaspberryPie to support I2C and SPI interfaces. I2C: SDA,SCL,GND SPI: MOSI,MISO,SCK,SS	2
4	Evaluate the following statements and indicate whether they are true/false. a) Von Neumann Architecture shares common memory for Data and Instructions TRUE: The von Neumann architecture uses a shared bus between program memory and data memory. This means that both program instructions and data are stored in the same memory and are accessed through the same bus. b) Harvard Architecture has separate physical memories for Data and Instructions TRUE: It uses two separate physical addresses for storing and accessing both instructions and data.	2
5	Consider a four-bit ALU which does four bits arithmetic. When the following four-bit numbers are added, what is the status of NZCV flags? 1101 + 1011 ANS: N=1, C=1, Z=0, V=0	ANS

Sl.no	PART - B	Mar
-------	----------	-----

		ks
1	<p>Draw the deployment design of the weather monitoring IOT system. Further, show the mapping of IOT Level to Functional Groups for the weather monitoring IoT system.</p> <p>Refer the reference book</p>	5
2	<p>Write the programs to perform the following: (draw interface diagrams)</p> <ul style="list-style-type: none"> - Interface one LED to GPIO 18, and program for blinking the LED (use RaspberryPie and phython) - Interface one LDR to D36 and LED to D2, and make the LED on/off based on Light Intensity (use ESP32 and embedded C) <div data-bbox="310 640 982 1041" data-label="Diagram"> <p>The diagram shows a rectangular box labeled 'Raspberry Pie'. A line labeled 'GPIO 18' extends from the right side of the box to a resistor. The other end of the resistor is connected to the anode of an LED. The cathode of the LED is connected to a ground symbol. A wavy arrow points towards the LED, indicating light emission.</p> </div> <pre> Python code: Import sleep from time Import RPi.GPIO as GPIO GPIO.setmode(GPIO.BCM) GPIO.setup(18,GPIO.out) Def toggleLED(pin) State = not state GPIO.output(pin,state) While true: Try: toggleLED(pin) sleep(.1) except KeyboardInterrupt: exit() </pre>	5
	<div data-bbox="328 1745 1021 2100" data-label="Diagram"> <p>The diagram shows a rectangular box labeled 'ESP-32 Microcontroller'. A line labeled '(Analog Input) 36' extends from the top of the box to a resistor. The other end of the resistor is connected to a +3.3V supply. The LDR sensor is connected to the bottom of the resistor and to ground.</p> </div>	10

	<p>NOTE:</p> <p>In the above diagram, when the light falls on the LDR, its resistance reduces, hence the voltage read at pin36 will be less (its digital value will be less). More the darkness, more digital value will be read, hence the LED is made ON.</p> <p>Embedded C Code:</p> <pre> #include <esp32.h> #define LEDPIN 2 #define LDRPIN 36 Int ldr_threshold = 800; Void setup() { pinMode(LEDPIN, OUTPUT); pinMode(LDRPIN, INPUT); } Void loop() { Int ldr_value = analogRead(LDRPIN); // give digital value for analog input 0-1023 If(ldr_value > ldr_threshold) digitalWrite(LEDPIN,HIGH); else digitalWrite(LEDPIN,LOW); } </pre>	
3	<p>The purpose of the home intrusion detection system is to detect intrusion using sensors (PIR sensor and Door sensor). Design Home Intrusion Detection system using RPie/ESP32 with PIR motion sensor for motion detection and door sensor for detecting</p>	

	<p>opening / closing of the door (for one room). Draw the following (no explanation required)</p> <ul style="list-style-type: none"> - Process Specification - Domain model - Deployment design - Functional & Operational View specifications <p>Refer reference book</p>	
<p>4</p>	<p>a) With a neat diagram explain the architecture of ARM Microcontroller.</p> <p>The ARM Architecture</p>  <p>Diagram + Explanation: 3m + 2m</p> <p>b) With the neat diagram briefly describe operating modes and register organization of ARM ISA. Mention the use of following Registers: R13,R14,R15,CPSR and SPSR.</p>	<p>5</p> <p>5</p>

User	FIQ	IRQ	SVC	Undef	Abort	
r0						
r1						
r2						
r3						
r4						
r5						
r6						
r7						
r8	r8					
r9	r9					
r10	r10					
r11	r11					
r12	r12					
r13 (sp)	r13 (sp)	r13 (sp)	r13 (sp)	r13 (sp)	r13 (sp)	
r14 (lr)	r14 (lr)	r14 (lr)	r14 (lr)	r14 (lr)	r14 (lr)	
r15 (pc)						
cpsr	spsr	spsr	spsr	spsr	spsr	
						Thumb state Low registers
						Thumb state High registers

Note: System mode uses the User mode register set

Diagram + Explanation: 3m + 2m

5	<p>a) Explain how embedded system are classified.</p> <pre> graph TD ES[Embedded System] --> BPF[Based on Performance and Functional Requirements] ES --> BPM[Based on Performance of the Microcontroller] BPF --> SAE[Stand alone Embedded Systems] BPF --> RTE[Real Time Embedded Systems] BPF --> NE[Networked Embedded Systems] BPF --> ME[Mobile Embedded Systems] BPM --> SSE[Small scale Embedded System] BPM --> MSE[Medium scale Embedded System] BPM --> SOE[Sophisticated Embedded Systems] </pre> <p>Diagram + Explanation: 3m + 2m</p> <p>b) Differentiate between RISC and CISC architecture.</p>	5
		5x1

RISC vs. CISC

CISC	RISC
Emphasis on hardware	Emphasis on software
Multiple instruction sizes and formats	Instructions of same set with few formats
Less registers	Uses more registers
More addressing modes	Fewer addressing modes
Extensive use of microprogramming	Complexity in compiler
Instructions take a varying amount of cycle time	Instructions take one cycle time
Pipelining is difficult	Pipelining is easy