REG.NO.: 25MES0023

VIT
Vellore Institute of Technology

SCHOOL OF ELECTRONICS ENGINEERING
CONTINUOUS ASSESSMENT TEST - I
FALL SEMESTER 2025-2026

SLOT: D2+TD2

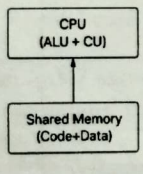
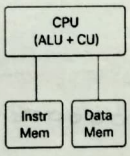
Programme Name & Branch	**: M.Tech Embedded Systems**
Course Code and Course Name	**: MAEDS502 - Microcontroller Architecture and Organization**
Faculty Name(s)	**: Dr. B. Karthikeyan**
Class Number(s)	**: 5667**
Date of Examination	**: 20/08/2025 2:00PM-3:30PM**
Exam Duration	**: 90 minutes** **Maximum Marks: 50**

General instruction(s):

- Answer All Questions
- M - Max mark; CO – Course Outcome; BL – Blooms Taxonomy Level (1 – Remember, 2 – Understand, 3 – Apply, 4 – Analyse, 5 – Evaluate, 6 – Create)

Course Outcomes

1. Analyze basic computer architecture components and differentiate instruction execution mechanisms across RISC, CISC, and ARM architectures
2. Develop ARM and THUMB assembly language programs to perform data processing and control operations, and demonstrate comprehension of their instruction sets

Q. No	Question	M	CO	BL
1.	 Figure A von Neumann Architecture  Figure B Harvard Architecture a) Analyse how the structural differences in Figure A and Figure B influence: i. Instruction fetch rate ii. Data access latency b) Suppose a DSP application requires simultaneous data fetch and instruction execution in every cycle. Which architecture would you recommend, and why? c) State one potential drawback of the chosen architecture for the above DSP application, when applied to general-purpose computing workloads.	4+3+3	01	03
2.	An ARM7 processor uses a **3-stage pipeline** (Fetch, Decode, Execute) to improve instruction throughput. a) Explain how pipelining changes the instruction cycle (execution time) compared to non-pipelined execution. b) For the instruction sequence below, draw the pipeline timing diagram showing what Instruction is in each stage: **{see next page}**	3+3+4	01	03

REG.NO.:

VIT
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

SCHOOL OF ELECTRONICS ENGINEERING
CONTINUOUS ASSESSMENT TEST - I
FALL SEMESTER 2025-2026

SLOT: D2+TD2

```
I1: MOV R0, #5
I2: ADD R1, R0, #3
I3: SUB R2, R1, #2
I4: MUL R3, R0, R2
```

c) If a **branch instruction replaces I3**, draw the pipeline timing diagram showing what Instruction is in each stage assuming there are **I5, I6, I7, I8** instructions present in the program.

3.	In this program, three arrays are defined in memory. The first array, **ARR1**, contains five consecutive words initialized with the values **10, 20, 30, 40, and 50.** The second array, **ARR2**, also contains five words but with larger values: **100, 200, 300, 400, and 500.** Finally, the third array, **ARR3**, is a buffer area reserved for six words, all **initialized to 0**, which will be used to store results or intermediate computations during execution. Take your **own value for Base addresses** of ARR1, ARR2, ARR3.		02	03

```
; Initialize registers R0..R7 with distinct values
MOV   R0, #11     ; R0 = 11
MOV   R1, #21     ; R1 = 21
MOV   R2, #31     ; R2 = 31
MOV   R3, #41     ; R3 = 41
MOV   R4, #51     ; R4 = 51
MOV   R5, #61     ; R5 = 61
MOV   R6, #71     ; R6 = 71
MOV   R7, #81     ; R7 = 81
; Load addresses of arrays into R10, R11, R12
LDR   R10, =ARR1  ; R10 -> base of ARR1   0X20000000
LDR   R11, =ARR2  ; R11 -> base of ARR2
LDR   R12, =ARR3  ; R12 -> base of ARR3

Line1: STMIA  R10!, {R0-R3}   0X20000000 - 0X2000000C
Line2: LDMIB  R11!, {R4-R7}
Line3: STMDA  R12!, {R1-R3
Line4: LDMDB  R10!, {R0-R2}
endloop
    B    endloop
    END
```

3+3+4

a) If STMIB R10!, {R0-R3} is used instead of STMIA in Line1, how would the memory contents and R10 differ? Explain precisely.

b) Imagine the programmer mistakenly includes the base register in the register list in Line 1, for example: STMIA R10!, {R10, R0–R2}.Explain why that is dangerous and what effect it might have.

c) After executing LDMIB R1!, {R7-R9}, what values do R7, R8, and R9 hold?

VIT
Vellore Institute of Technology

REG.NO.:

SCHOOL OF ELECTRONICS ENGINEERING
CONTINUOUS ASSESSMENT TEST - I
FALL SEMESTER 2025-2026

SLOT: D2+TD2

4.	LDR R0, =STACK_TOP ; Initialize SP
	MOV SP, R0
	
	MOV R1, #11
	MOV R2, #22
	MOV R3, #33
	Line1: STMFD SP!, {R1-R3}
	
	MOV R4, #44
	MOV R5, #55
	MOV R6, #66
	Line2: STMEA SP!, {R4-R6}
	
	Line3: LDMFA SP!, {R7-R9}
	Line4: LDMED SP!, {R10-R12}
	
	STOP B STOP
	a) If the stack pointer (SP) initially points to address 0x4000, and the instruction in Line1 STMFD SP!, {R1-R3} is executed, what will be the final value of SP and at which addresses will the registers be stored?
	
	b) After the Line1 & Line2 (STMFD then STMEA) executed, what is the top-to-bottom order of values on the stack?
	
	c) In Line3, Instead of LDMFA, if LDMFD SP!, {R7-R9} is executed, what would R7–R9 hold?
		3+3+4	02	03

5.	Write an ARM7 assembly language program using SWI instructions to implement a simple operations. Two numbers are provided in R0 and R1. Based on the SWI number, perform the following operations: • SWI 0, perform Addition and store Result in R2 • SWI 1 Check Even/Odd for R0 and Store 0 in R2 if even, 1 if odd • SWI 2 find Maximum of R0 and R1 and store Result in R2 • SWI 3 perform Bitwise AND of R0 and R1 & store Result in R2 The program should implement a SWI handler that **decodes the SWI number**, calls the respective routine, and stores the result in R2. Assume the program starts in Supervisor mode with vector table properly initialized.	10	02	03

REG.NO.:

VIT

Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

SCHOOL OF ELECTRONICS ENGINEERING
CONTINUOUS ASSESSMENT TEST - II
FALL SEMESTER 2025-2026

SLOT: D2+TD2

Programme Name & Branch	: M.Tech Embedded Systems
Course Code and Course Name	: MAEDS502 - Microcontroller Architecture and Organization
Faculty Name(s)	: Dr. B. Karthikeyan
Class Number(s)	: 5667
Date of Examination	: 08/10/2025 9:30AM-11:00AM
Exam Duration	: 90 minutes Maximum Marks: 50

General instruction(s):

- Answer All Questions
- M - Max mark; CO – Course Outcome; BL – Blooms Taxonomy Level (1 – Remember, 2 – Understand, 3 – Apply, 4 – Analyse, 5 – Evaluate, 6 – Create)

Course Outcomes

2. Develop ARM and THUMB assembly language programs to perform data processing and control operations, 3. and demonstrate comprehension of their instruction sets

3. Design and implement embedded applications using LPC214x peripherals such as GPIO, timers, PWM, and UART

Q. No	Question	M	CO	BL
1.	Assume you are tasked with optimizing an embedded system application running on ARM7TDMI using the THUMB instruction set for memory-constrained environments. The main program loop currently utilizes several arithmetic and branch instructions. a) Analyse and propose how changing from ARM mode (32-bit instructions) to THUMB mode (16-bit instructions) may impact **code density, execution speed, and interrupt response time.** b) Write a code snippet in assembly language showing the transition from ARM state to THUMB state using appropriate instructions, and explain how you would **ensure proper handling of the Link Register (LR) and Program Counter (PC) during this transition.**	5+5	02	3
2.	The following must be shown clearly in your program. • Configure the appropriate GPIO pins using the PINSEL register for UART0 functionality. • Set the UART0 Line Control Register (U0LCR) for 8 data bits, 1 stop bit, no parity, and enable DLAB for accessing divisor latches. • Calculate and configure the correct values for U0DLL and U0DLM registers for 4800 baud rate, given PCLK = 15 MHz. • Transmit the character 'A' repeatedly, with appropriate checks to ensure the transmission is complete before sending the next character. Now, Write a C program for the following application. A switch is interfaced with P0.5, If the switch is closed(Logic 1) transmit the character "A" at 4800 baud rate Else transmit character "B" at 4800 baud rate.	10	03	4

REG.NO.:

VIT
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

SCHOOL OF ELECTRONICS ENGINEERING
CONTINUOUS ASSESSMENT TEST - II
FALL SEMESTER 2025-2026

SLOT: D2+TD2

3.	#include <lpc214x.h>
	
	void initPLL(void)
	{
	PLL0CFG = 0x24;
	PLL0CON = 0x01;
	PLL0FEED = 0xAA;
	PLL0FEED = 0x55;
	while ((PLL0STAT & 0x0400) == 0);
	PLL0CON = 0x03;
	VPBDIV = 0x02;
	PLL0FEED = 0xAA;
	PLL0FEED = 0x55;
	}
	void UART0_Init(void) {
	PINSEL0 |= 0x00000005;
	U0LCR = 0x83;
	U0DLL = 97;
	U0DLM = 0;
	U0LCR = 0x03;
	}
	void UART0_TxChar(char ch) {
	while ((U0LSR & 0x20) == 0);
	U0THR = ch;
	}
	char UART0_RxChar(void) {
	while ((U0LSR & 0x01) == 0);
	return U0RBR;
	}
	int main(void) {
	char c;
	initPLL();
	UART0_Init();
	while (1) {
	c = UART0_RxChar();
	UART0_TxChar(c);
	}
	}
	
	a) What will happen if the above program receives a continuous stream of characters without delay?
	b) If the main loop is modified to include a delay after each transmit, how will it affect UART communication performance?
	c) What is the effect of not checking the THR empty bit (U0LSR & 0x20) before writing to U0THR in the UART program?
			3+3+4	03	4

4.	#include <lpc214x.h>
	void initPLL(void)
	{
	PLL0CFG = 0x24;
	PLL0CON = 0x01;
	PLL0FEED = 0xAA;
	PLL0FEED = 0x55;
	while ((PLL0STAT & 0x0400) == 0);
	PLL0CON = 0x03;
	VPBDIV = 0x02;
	PLL0FEED = 0xAA;
	PLL0FEED = 0x55;
	}
	void Timer0_Init(void) {
	T0TCR = 0x02;
	T0PR = 14999;
	T0TCR = 0x01;
	}
	void delay_ms(unsigned int ms) {
	unsigned int start_count = T0TC;
	while ((T0TC - start_count) < ms);
	}
	int main(void) {
	PINSEL0 = 0x00000000;
	IODIR0 = 0x01;
	initPLL();
	Timer0_Init();
	while (1) {
	delay_ms(1000);
	IOSET0 = 0x01;
	delay_ms(1000);
	IOCLR0 = 0x01;
	}
	}
	
	a) Show the modification in the program such that it ensures 1 ms timer tick at 20 MHz PCLK frequency
	b) Modify the initPLL() function such that it should generate 36MHZ frequency. Also show the modification in the program such that it ensures 1 ms timer tick at 18 MHz PCLK frequency
		4+6	03	4

5.	Write an embedded C program for the LPC2148 microcontroller to blink an LED connected to port pin P0.0. LED ON time is 1 second and LED OFF time is 2 seconds using Timer0.	10	03	4

Z/K/TY

Reg. No: 25MBS0021

VIT
Vellore Institute of Technology

Final Assessment Test – November 2025

Course: MAEDS502 - Microcontroller Architecture and Organization
Class NBR(s): 5664 Slot: D2+TD2
Time: Three Hours Max. Marks: 100

- KEEPING MOBILE PHONE/ANY ELECTRONIC GADGETS, EVEN IN 'OFF' POSITION IS TREATED AS EXAM MALPRACTICE
- DON'T WRITE ANYTHING ON THE QUESTION PAPER

COs	CO Statements
CO1	Analyze basic computer architecture components and differentiate instruction execution mechanisms across RISC, CISC, and ARM architectures
CO2	Develop ARM and THUMB assembly language programs to perform data processing and control operations, and demonstrate comprehension of their instruction sets
CO3	Design and implement embedded applications using LPC214x peripherals such as GPIO, timers, PWM, and UART
CO4	Apply interfacing techniques for external devices using I2C, SPI, ADC, and DAC, and analyze interrupt handling mechanisms on the ARM7 LPC214x microcontroller
CO5	Develop and analyze embedded programs using ARM/THUMB assembly and LPC2148 peripherals to perform arithmetic, logic, control, and interfacing tasks in real-time applications

BL – Blooms Taxonomy Level (1 – Remember, 2 – Understand, 3 – Apply, 4 – Analyse, 5 – Evaluate, 6 – Create)

Answer ALL Questions
(10 X 10 = 100 Marks)

1. (a) Compare and contrast the von Neumann and Harvard architectures in terms of memory organization, data flow, and execution efficiency. **CO1 BL2**
   (b) Analyze how the Control Unit, ALU, and Register Bank interact during each stage of instruction execution.

2. (a) Evaluate how the peripherals contribute to real-time control applications, citing specific examples such as timers, serial communication ports, or I/O ports. **CO1 BL2**
   (b) Examine the significance of bit and byte addressable memory locations in microcontrollers, and propose scenarios where efficient use of such addressing can optimize program performance and memory utilization.

3. Write an ARM assembly program that performs 32-bit integer division using only integer instructions. Use: register-based parameter passing (r0 = dividend, r1 = divisor), return quotient in r0 and remainder in r1. **CO2 BL3**

4. Write an ARM assembly program that initializes an array of eight 32-bit constants. The program must: **CO2 BL3**
   - Demonstrate at least three different techniques for loading constants into registers.
   - Use different addressing modes to write those constants into memory.

5.	Describe how the ARM processor changes state (including register usage and exception/branch behaviour), how BL/BLX and BX enable interworking, and what software/hardware precautions a systems programmer must take when mixing ARM and THUMB code.	CO3	BL2
6.	Compare and contrast the design goals and trade-offs that led to the creation of the THUMB instruction set versus ARM state. The comparison should focus on code density, performance per cycle, energy efficiency, and typical embedded system constraints. Conclude by evaluating scenarios (two concrete application examples) where selecting THUMB over ARM (or vice-versa) would be justified.	CO3	BL2
7.	Write an embedded C program to configure LPC 2148 Timer 0 in timer mode to generate a 1-second delay. Use polling to monitor the Timer Counter (TC) value. Toggle an LED on P0.10 every 1 second. Derive the delay formula based on PCLK, prescaler, and TC values. Evaluate how changes in system clock frequency affect delay accuracy. Assume your own value wherever necessary.	CO4	BL3
8.	Write an embedded C program to configure LPC2148 UART0 for serial communication at 9600 baud rate. The program should receive a character and transmit the same character at 9600 baudrate.	CO4	BL3
9.a)	Write an embedded C program to configure ADC0 on LPC2148 to read analog voltage from a temperature sensor (LM35) connected to channel AD0.1. Display the converted temperature value on the serial terminal using UART0. Use polling to check ADC conversion completion. Analyze how ADC resolution and reference voltage affect measurement accuracy.	CO5	BL3
	OR		
9.b)	Write an embedded C program to communicate with an I^2C temperature sensor (e.g., LM75 or TMP102) using LPC2148 I^2C0. Read temperature data periodically and send it to a PC terminal via UART0. Use polling to monitor I^2C status flags (do not use interrupts). Evaluate the role of start, stop, and acknowledge signals in ensuring reliable communication. Propose how your program could be extended to interface multiple I^2C sensors on the same bus.	CO5	BL3
10.a)	Write an embedded C program for LPC2148 that controls four LEDs connected to Port 0 pins using switch inputs on Port 1. When a switch is pressed, toggle the corresponding LED. Use bit masking and shifting operations to access I/O pins. Assume your own value wherever necessary.	CO5	BL3
	OR		
10.b)	Write a program to configure PWM1 to vary the brightness of an LED connected to P0.21. Implement a gradual increase and decrease of duty cycle (0% → 100% → 0%) using software-controlled updates to the PWM register values. Do not use interrupt or match-triggered updates.	CO5	BL3

⇔⇔⇔ Z/K/TY⇔⇔⇔