Table of Contents

Note : Remove “Table of Content” before including in CP Book

17ECL67: Embedded Controller Lab

A. LABORATORY INFORMATION

1. Lab Overview

Degree:	BE	*Program:*	EC
Year / Semester :	3/6	*Academic Year:*	2019-20
Course Title:	Embedded controller Lab	*Course Code:*	17ECL67
Credit / L-T-P:	4 / 0-1-2	*SEE Duration:*	180 Minutes
Total Contact Hours:	42 Hrs	*SEE Marks:*	80 Marks
CIA Marks:	20	*Assignment*	
Course Plan Author:	Shilpa Rani P	*Sign*	Dt : 11/01/2019
Checked By:		*Sign*	Dt : 11/01/2019

2. Lab Content

Unit	Title of the Experiments	Lab Hours	Concept	Blooms Level
	Conduct the following Study experiments to learn ALP using ARM Cortex M3 Registers using an Evaluation board and the required software tool.			L4 Analyze
1	ALP to multiply two 16 bit binary numbers.	3	multiplication	L4
2	ALP to find the sum of first 10 integer numbers.	3	addition	L4
	Conduct the following experiments on an ARM CORTEX M3 evaluation board using evaluation version of Embedded 'C' & Keil uVision-4 tool/compiler			L4
3	Display “Hello World” message using Internal UART.	3	UART initialization	
4	Interface and Control a DC Motor.	3	DC motor interface	L4
5	Interface a Stepper motor and rotate it in clockwise and anti-clockwise direction.	3	Stepper motor interface	L4
6	Interface a DAC and generate Triangular and Square waveforms.	3	DAC	L4
7	Interface a 4x4 keyboard and display the key code on an LCD.	3	HEXA Keypad	L4
8	Using the Internal PWM module of ARM controller generate PWM and vary its duty cycle.	3	PWM	L4
9	Demonstrate the use of an external interrupt to toggle an LED On/Off.		External interrupt	L4
10	Display the Hex digits 0 to F on a 7-segment LED interface, with an appropriate delay in between.	3	7 segment interface	L4
11	Interface a simple Switch and display its status through Relay, Buzzer and LED.	3	Buzzer	L4
12	Measure Ambient temperature using a sensor and SPI ADC IC	3	ADC	L4

3. Lab Material

Unit	Details	Available
1	Text books	
		In Lib
2	Reference books	
	Lab manual prepared by Department of E & C Engg, SKIT.	In dept
3	Others (Web, Video, Simulation, Notes etc.)	
		Not Available

4. Lab Prerequisites:

-	-	Base Course:		-	-
SNo	Course Code	Course Name	Topic / Description	Sem	Remarks
1					

Note: If prerequisites are not taught earlier, GAP in curriculum needs to be addressed. Include in Remarks and implement in B.5.

5. General Instructions

SNo	Instructions	Remarks
1	Observation book and Lab record are compulsory.	
2	Students should report to the concerned lab as per the time table.	
3	After completion of the program, certification of the concerned staff in-charge in the observation book is necessary.	
4	Student should bring a notebook of 100 pages and should enter the readings /observations into the notebook while performing the experiment.	
5	The record of observations along with the detailed experimental procedure of the experiment in the Immediate last session should be submitted and certified staff member in-charge.	
6	Should attempt all problems / assignments given in the list session wise.	
7	It is responsibility to create a separate directory to store all the programs, so that nobody else can read or copy.	
8	When the experiment is completed, should disconnect the setup made by them, and should return all the components/instruments taken for the purpose.	
9	Any damage of the equipment or burn-out components will be viewed seriously either by putting penalty or by dismissing the total group of students from the lab for the semester/year	
10	Completed lab assignments should be submitted in the form of a Lab Record in which you have to write the algorithm, program code along with comments and output for various inputs given	

6. Lab Specific Instructions

SNo	Specific Instructions	Remarks
1	Turn on the computer.	

2	Double click on Keil uVision-4 icon.	
3	Select new project in file menu.	
4	Enter the project name and location as shown below and hit Next.	
5	Select the Family, Device, Package and speed as per the requirements and hit Next.	
6	Create a new source by using new source icon or right click on the device/project folder to create new source.	
7	Select the verilog module and enter the file name in New Source Wizard window and hit Next.	
8	Enter the module name – dataflow/behavioral/structural, port name and select the direction. This will create .v source file. Hit Next and finish the initial project creation.	
9	Write complete VHDL/Verilog code for implementation and save.	
10	Click on implementation and check for syntax using “Check syntax” option under synthesize tab. If any error, edit and correct VHDL/Verilog code and repeat check syntax until zero errors.	
11	Double click on ISIM simulator by selecting simulation mode to complete the functional simulation of your design.	
12	Click on user constraints and select pre synthesis/post synthesis for assigning the ports, select the ports and save. It will generate .ucf file to source file.	
13	Click on Implement design for checking Place, Route and Map.	
14	Click generate programming file to generate the .bit file for loading into FPGA kit.	
15	Select the COM port and load the bit file to FPGA kit and check the results. Note down the results in observation book.	

B. OBE PARAMETERS

1. Lab / Course Outcomes

#	COs	Teach. Hours	Concept	Instr Method	Assessment Method	Blooms' Level
17ECL67.1	Display “Hello World” message using Internal UART.	3	UART	Tutorial / Demonstration/ Practical	CIA	L2,L3
17ECL67.2	Interface and Control a DC Motor.	3	DC Motor	Tutorial / Demonstration/ Practical	CIA	L2,L3, L4,L5
17ECL67.3	Interface a Stepper motor and rotate it in clockwise and anti-clockwise direction.	3	Stepper motor	Tutorial / Demonstration/ Practical	CIA	L2,L3, L4,L5
17ECL67.4	Interface a DAC and generate Triangular and Square waveforms.	3	DAC	Tutorial / Demonstration/ Practical	CIA	L2,L3, L4,L5
17ECL67.5	Interface a 4x4 keyboard and display the key code on an LCD.	3	4x4 keyboard	Tutorial / Demonstration/ Practical	CIA	L2,L3, L4,L5
17ECL67.6	Using the Internal PWM module of ARM controller generate	3	PWM	Tutorial / Demonstra	CIA	L2,L3, L4,L5

	PWM and vary its duty cycle.			tipon/ Practical		
17ECL67.7	Demonstrate the use of an external interrupt to toggle an LED On/Off.	3	external interrupt	Tutorial / Demonstratipon/ Practical	CIA	L2,L3, L4,L5
17ECL67.8	Display the Hex digits 0 to F on a 7-segment LED interface, with an appropriate delay in between.	3	7-segment LED interface	Tutorial / Demonstratipon/ Practical	CIA	L2,L3, L4,L5
17ECL67.9	Interface a simple Switch and display its status through Relay, Buzzer and LED.	3	Relay, Buzzer and LED	Tutorial / Demonstratipon/ Practical	CIA	L2,L3, L4,L5
17ECL67.10	Measure Ambient temperature using a sensor and SPI ADC IC	3	SPI ADC IC	Tutorial / Demonstratipon/ Practical	CIA	L2,L3, L4,L5
-	**Total**	**36**	-	-	-	-

Note: Identify a max of 2 Concepts per unit. Write 1 CO per concept.

2. Lab Applications

SNo	Application Area	CO	Level
1	Memory controller.		L3
2	I/O controller		L3
3	Embedded system design for various applications		L3
4			L3
5			L3
6			L3

Note: Write 1 or 2 applications per CO.

3. Articulation Matrix

(CO – PO MAPPING)

-	Course Outcomes	Program Outcomes											
#	COs	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	Level
17ECL67.1	Display “Hello World” message using Internal UART.	3	3	2		2				2			1	L4
17ECL67.2	Interface and Control a DC Motor.	3	3	2		2				2			1	L4
17ECL67.3	Interface a Stepper motor and rotate it in clockwise and anti-clockwise direction.	3	3	2		2				2			1	L4
17ECL67.4	Interface a DAC and generate Triangular and Square waveforms.	3	3	2		2				2			1	L4
17ECL67.5	Interface a 4x4 keyboard and display the key code on an LCD.	3	3	2		2				2			1	L4

	3	3	2													
17ECL67.6	Using the Internal PWM module of ARM controller generate PWM and vary its duty cycle.				3	3	2		2				2			1	L4
17ECL67.7	Demonstrate the use of an external interrupt to toggle an LED On/Off.				3	3	2		2				2			1	L4
17ECL67.8	Display the Hex digits 0 to F on a 7-segment LED interface, with an appropriate delay in between.				3	3	2		2				2			1	L4
17ECL67.9	Interface a simple Switch and display its status through Relay, Buzzer and LED.				3	3	2		2				2			1	L4
17ECL67.10	Measure Ambient temperature using a sensor and SPI ADC IC				3	3	2		2				2			1	L4
17ECL67.	**Average**				**3**	**3**	**2**		**2**				**2**			**1**	**L4**

Note: Mention the mapping strength as 1, 2, or 3

4. Mapping Justification

Mapping		Mapping Level	Justification
CO	**PO**	**-**	**-**
CO1	PO1	L4	The basic engineering knowledge is applied for the ARM programming.
CO1	PO2	L4	Performing experiment allows the easy analysis of problems.
CO1	PO3	L4	Designing a ARM based system to meet the specific needs within the realistic constraints can be done.
CO1	PO5	L4	Modern tools are used for designing and analysis of systems.
CO1	PO9	L4	Experiments are done in teams to develop team work.
CO1	PO12	L4	Practical knowledge inculcates inquisitiveness towards continuous learning.
CO2	PO1	L4	The basic engineering knowledge is applied for the ARM programming.
CO2	PO2	L4	Performing experiment allows the easy analysis of problems.
CO2	PO3	L4	Designing a ARM based system to meet the specific needs within the realistic constraints can be done.
CO2	PO5	L4	Modern tools are used for designing and analysis of systems.
CO2	PO9	L4	Experiments are done in teams to develop team work.
CO2	PO12	L4	Practical knowledge inculcates inquisitiveness towards continuous learning.
CO3	PO1	L4	The basic engineering knowledge is applied for the ARM programming.
CO3	PO2	L4	Performing experiment allows the easy analysis of problems.
CO3	PO3	L4	Designing a ARM based system to meet the specific needs within the realistic constraints can be done.
CO3	PO5	L4	Modern tools are used for designing and analysis of systems.
CO3	PO9	L4	Experiments are done in teams to develop team work.
CO3	PO12	L4	Practical knowledge inculcates inquisitiveness towards continuous learning.
CO4	PO1	L4	The basic engineering knowledge is applied for the ARM programming.
CO4	PO2	L4	Performing experiment allows the easy analysis of problems.
CO4	PO3	L4	Designing a ARM based system to meet the specific needs within the realistic constraints can be done.

CO4	PO5	L4	Modern tools are used for designing and analysis of systems.
CO4	PO9	L4	Experiments are done in teams to develop team work.
CO4	PO12	L4	Practical knowledge inculcates inquisitiveness towards continuous learning.
CO5	PO1	L4	The basic engineering knowledge is applied for the ARM programming.
CO5	PO2	L4	Performing experiment allows the easy analysis of problems.
CO5	PO3	L4	Designing a ARM based system to meet the specific needs within the realistic constraints can be done.
CO5	PO5	L4	Modern tools are used for designing and analysis of systems.
CO5	PO9	L4	Experiments are done in teams to develop team work.
CO5	PO12	L4	Practical knowledge inculcates inquisitiveness towards continuous learning.
CO6	PO1	L4	The basic engineering knowledge is applied for the ARM programming.
CO6	PO2	L4	Performing experiment allows the easy analysis of problems.
CO6	PO3	L4	Designing a ARM based system to meet the specific needs within the realistic constraints can be done.
CO6	PO5	L4	Modern tools are used for designing and analysis of systems.
CO6	PO9	L4	Experiments are done in teams to develop team work.
CO6	PO12	L4	Practical knowledge inculcates inquisitiveness towards continuous learning.
CO7	PO1	L4	The basic engineering knowledge is applied for the ARM programming.
CO7	PO2	L4	Performing experiment allows the easy analysis of problems.
CO7	PO3	L4	Designing a ARM based system to meet the specific needs within the realistic constraints can be done.
CO7	PO5	L4	Modern tools are used for designing and analysis of systems.
CO7	PO9	L4	Experiments are done in teams to develop team work.
CO7	PO12	L4	Practical knowledge inculcates inquisitiveness towards continuous learning.
CO8	PO1	L4	The basic engineering knowledge is applied for the ARM programming.
CO8	PO2	L4	Performing experiment allows the easy analysis of problems.
CO8	PO3	L4	Designing a ARM based system to meet the specific needs within the realistic constraints can be done.
CO8	PO5	L4	Modern tools are used for designing and analysis of systems.
CO8	PO9	L4	Experiments are done in teams to develop team work.
CO8	PO12	L4	Practical knowledge inculcates inquisitiveness towards continuous learning.
CO9	PO1	L4	The basic engineering knowledge is applied for the ARM programming.
CO9	PO2	L4	Performing experiment allows the easy analysis of problems.
CO9	PO3	L4	Designing a ARM based system to meet the specific needs within the realistic constraints can be done.
CO9	PO5	L4	Modern tools are used for designing and analysis of systems.
CO9	PO9	L4	Experiments are done in teams to develop team work.
CO9	PO12	L4	Practical knowledge inculcates inquisitiveness towards continuous learning.
CO10	PO1	L4	The basic engineering knowledge is applied for the ARM programming.
CO10	PO2	L4	Performing experiment allows the easy analysis of problems.
CO10	PO3	L4	Designing a ARM based system to meet the specific needs within the realistic constraints can be done.
CO10	PO5	L4	Modern tools are used for designing and analysis of systems.
CO10	PO9	L4	Experiments are done in teams to develop team work.
CO10	PO12	L4	Practical knowledge inculcates inquisitiveness towards continuous learning.

Note: Write justification for each CO-PO mapping.

5. Curricular Gap and Content

SNo	Gap Topic	Actions Planned	Schedule Planned	Resources Person	PO Mapping
1					
2					
3					
4					
5					

Note: Write Gap topics from A.4 and add others also.

6. Content Beyond Syllabus

SNo	Gap Topic	Actions Planned	Schedule Planned	Resources Person	PO Mapping
1					
2					
3					
4					
5					
6					
7					
8					
9					
10					
11					
12					
13					
14					
15					

Note: Anything not covered above is included here.

C. COURSE ASSESSMENT

1. Course Coverage

Unit	Title	Teaching Hours	No. of question in Exam							CO	Levels
			CIA-1	CIA-2	CIA-3	Asg-1	Asg-2	Asg-3	SEE		
1	ALP to multiply two 16 bit binary numbers.	03	1	-	-	-	-	-	1	CO1	L2
2	ALP to find the sum of first 10 integer numbers.	03	1	-	-	-	-	-	1	CO2	L3
3	Display “Hello World” message using Internal UART.	03	1	-	-	-	-	-	1	CO3	L3
4	Interface and Control a DC Motor.	03	1	-	-	-	-	-	1	CO4	L3
5	Interface a Stepper motor and rotate it in clockwise and anti-clockwise direction.	03	1	-	-	-	-	-	1	CO5	L4

6	Interface a DAC and generate Triangular and Square waveforms.	03	1	-	-	-	-	-	1	CO6	L4
7	Interface a 4x4 keyboard and display the key code on an LCD.	03	1	-	-	-	-	-	1	CO7	L4
8	Using the Internal PWM module of ARM controller generate PWM and vary its duty cycle.	03	-	1	-	-	-	-	1	CO8	L4
9	Demonstrate the use of an external interrupt to toggle an LED On/Off.	03	-	1	-	-	-	-	1	CO9	L4
10	Display the Hex digits 0 to F on a 7-segment LED interface, with an appropriate delay in between.	03	-	1	-	-	-	-	1	CO10	L4
11	Interface a simple Switch and display its status through Relay, Buzzer and LED.	03	-	1	-	-	-	-	1	CO11	L4
12	Measure Ambient temperature using a sensor and SPI ADC IC	03		1							
-	**Total**	**42**	**7**	**8**	**5**	**5**	**5**	**5**	**20**	**-**	**-**

Note: Write CO based on the theory course.

2. Continuous Internal Assessment (CIA)

Evaluation	Weightage in Marks	CO	Levels
CIA Exam – 1	30	CO1, CO2, CO3, CO4	L23, L3
CIA Exam – 2	30	CO5, CO6, CO7,CO8	L1, L2, L3 . .
CIA Exam – 3	30	CO9,CO10,CO11	L1, L2, L3 . .
Assignment - 1	05	CO1, CO2, CO3, CO4	L2, L3, L4 ...
Assignment - 2	05	CO5, CO6, CO7,CO8	L1, L2, L3 ...
Assignment - 3	05	CO9,CO10,CO11	L1, L2, L3 ...
Seminar - 1	05	CO1, CO2, CO3, CO4	L2, L3, L4 . ..
Seminar - 2	05	CO5, CO6, CO7,CO8	L2, L3, L4 . ..
Seminar - 3	05	CO9,CO10,CO11	L2, L3, L4 . ..
Other Activities – define – Slip test		CO1 to CO11	L2, L3, L4 . ..
Final CIA Marks	**40**	**-**	**-**

-

SNo	Description	Marks
1	Observation and Weekly Laboratory Activities	05 Marks
2	Record Writing	10 Marks for each Expt
3	Internal Exam Assessment	20 Marks
4	Internal Assessment	5 Marks
5	SEE	600 Marks
-	**Total**	**100 Marks**

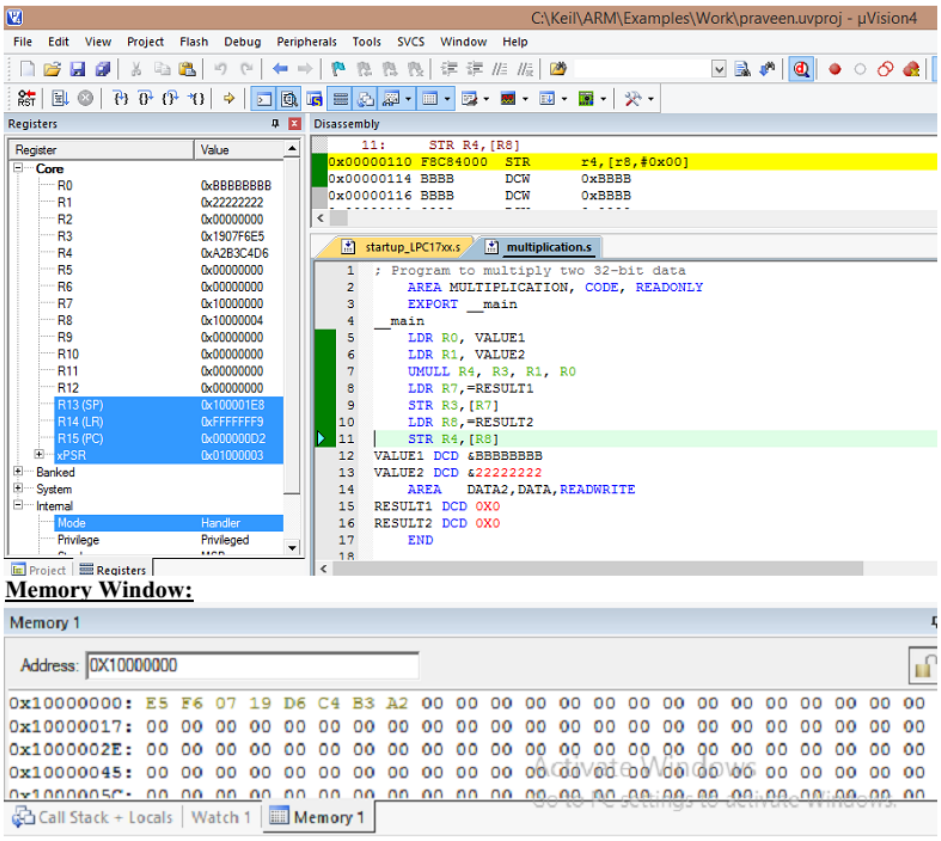
D. EXPERIMENTS

Experiment 01 : ALP to multiply two 16 bit binary numbers.

-	**Experiment No.:**	1	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Write verilog code to realize all the logic gates						
2	Course Outcomes	Create and verify functionality of various gates at the different level of abstractions.						
3	Aim	ALP to multiply two 16 bit binary numbers.						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept							
6	Procedure, Program, Activity, Algorithm, Pseudo Code	• step 1: start • step 2: write programming • step 3: save the program • step 4: check syntax • step 5:if error then correct the errors • step 6:simulate the design • step 7: Hardware implementation • step 8: stop						

7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	**Program:**
		`;//Program to multiply two 32-bit data`
		`AREA MULTIPLICATION, CODE, READONLY`
		`EXPORT __main`
		`__main`
		`LDR R0, VALUE1 ;Read the first data`
		`LDR R1, VALUE2 ;Read the second data`
		`UMULL R4, R3, R1, R0 ;Multiply R0 with R1, store the result in R3 and R4`
		`LDR R7,=RESULT1 ;Read the address`
		`STR R3,[R7] ;Store the contents of R3 into address pointed by R7`
		`LDR R8,=RESULT2 ;Read the address`
		`STR R4,[R8] ;Store the contents of R4 into address pointed by R8`
		`VALUE1 DCD &BBBBBBBB`
		`VALUE2 DCD &22222222`
		`AREA DATA2,DATA,READWRITE`
		`RESULT1 DCD 0X0`
		`RESULT2 DCD 0X0`
		`END`						

8	Observation Table, Look-up Table, Output	•						
9	Sample Calculations	• -						

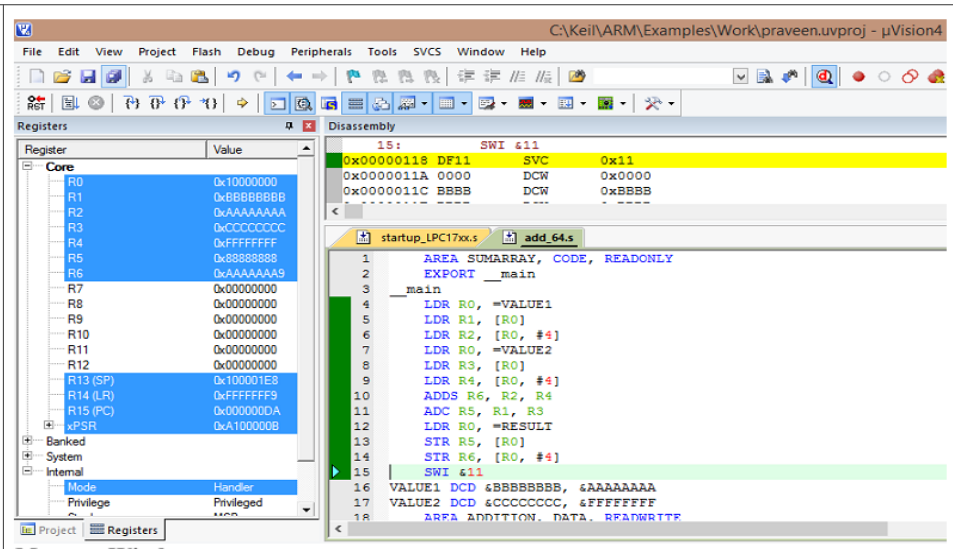
10	Graphs, Outputs	
11	Results & Analysis	•
12	Application Areas	•
13	Remarks	
14	Faculty Signature with Date	

Experiment 02: ALP to find the sum of first 10 integer numbers.

-	**Experiment No.:**	1	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	ALP to find the sum of first 10 integer numbers.						
2	Course Outcomes							
3	Aim	ALP to find the sum of first 10 integer numbers.						
4	Material / Equipment Required	Lab Manual						

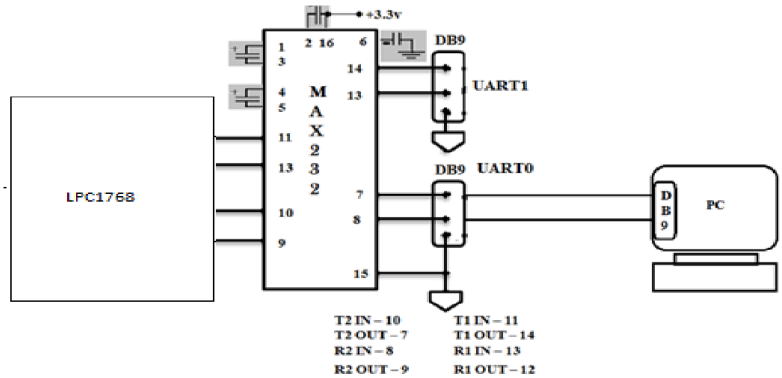
5	Theory, Formula, Principle, Concept	**Program:**
		`;//program to add two 64 bit numbers`
		`AREA SUMARRAY, CODE, READONLY`
		`EXPORT __main`
		`__main`
		`LDR R0, =VALUE1 ;Read the address`
		`LDR R1, [R0] ;Load the data at the address R0 into R1`
		`LDR R2, [R0, #4] ;Load the data at the address into R2`
		`LDR R0, =VALUE2 ;Read the address`
		`LDR R3, [R0] ;Load the data at the address R0 into R3`
		`LDR R4, [R0, #4] ;Load the data at the address into R4`
		`ADDS R6, R2, R4 ;Add the contents of R2 and R4 store the result in R6`
		`ADC R5, R1, R3 ;Add with carry the contents of R1 and R3 store the result in R5`
		`LDR R0, =RESULT ;Read the address`
		`STR R5, [R0] ;Store the contents of R5 to address pointed by R0`
		`STR R6, [R0, #4] ;Store the contents of R6 to address pointed by R0+4`
		`VALUE1 DCD &BBBBBBBB, &AAAAAAAA`
		`VALUE2 DCD &CCCCCCCC, &FFFFFFFF`
		`AREA ADDITION, DATA, READWRITE`
		`RESULT DCD &0`
		`END`

6	Procedure, Program, Activity, Algorithm, Pseudo Code	•
		•
7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	
8	Observation Table, Look-up Table, Output	•
9	Sample Calculations	• -

10	Graphs, Output	 **PSR Status:** R0 = VALUE2, R1 = 0x11111111, R2 = 0xCCCCCCCC, R3 = 0xDDDDDDDD, Z, V, S, C
11	Results & Analysis	• Message “Hello World” appearing for 5 times in hyperterminal..
12	Application Areas	•
13	Remarks	
14	Faculty Signature with Date	

Experiment 03: Display “Hello World” message using Internal UART.

-	**Experiment No.:**	1	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Display “Hello World” message using Internal UART.						
2	Course Outcomes							
3	Aim	Display “Hello World” message using Internal UART.						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept							
6	Procedure, Program, Activity, Algorithm, Pseudo Code	• **Algorithm:** • 1. Create a project and include the above program using Keil tool. • 2. Create a Hex file and build. • 3. Flash the program onto the chip using FlashMagic software. • 4.Check the message transmitted in hyperterminal.						
		• Step 1: Configure the GPIO pin for UART0 function using PINSEL register. • Step 2: Configure the FCR for enabling the FIXO and Reste both the						

		Rx/Tx FIFO. • step 3: Configure LCR for 8-data bits, 1 Stop bit, Disable Parity and Enable DLAB. • step 4: Calculate the DLM,DLL values for required baudrate from PCLK. • step 5:iUpdate the DLM,DLL with the calculated values. 6. . • step 6:Finally clear DLAB to disable the access to DLM,DLL.
	Block Diagram	

7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	`#include<LPC17xx.h>`
		`void UART0_Init(void);`
		
		`unsigned int i, p;`
		`unsigned char *ptr, arr[] = "Hello world\r";`
		`#define THR_EMPTY 0x20`
		`int main(void)`
		`{`
		`UART0_Init();`
		`for (p=0; p<5; p++)`
		`{`
		`ptr = arr;`
		`while ( *ptr != '\0')`
		`{`
		`while((LPC_UART0->LSR & THR_EMPTY) != THR_EMPTY);`
		`LPC_UART0->THR = *ptr++;`
		`for(i=0;i<=2000;i++);`
		
		`}`
		`}`
		`}`
		
		`void UART0_Init(void)`
		`{`
		
		`LPC_SC->PCONP |= 0x00000008;` //UART0 peripheral power and clock enable
		`LPC_PINCON->PINSEL0 = 0x00000050;` //value required for PINSEL0 which //leads to
		//PO.2 and PO.3 pins to operate as TX and RX
		//pins of UART in controller
		
		`LPC_UART0->LCR = 0x00000083;` //enable divisor latch, parity disable,
		//1 stop bit, 8bit word length
		
		`LPC_UART0->DLM = 0X00;`
		`LPC_UART0->DLL = 0x1A;` //select baud rate 9600 bps
		`LPC_UART0->LCR = 0X00000003;` // lock the baud rate
		`LPC_UART0->FCR = 0x07;` // setting FIFO control Register for proper Txn
		`}`

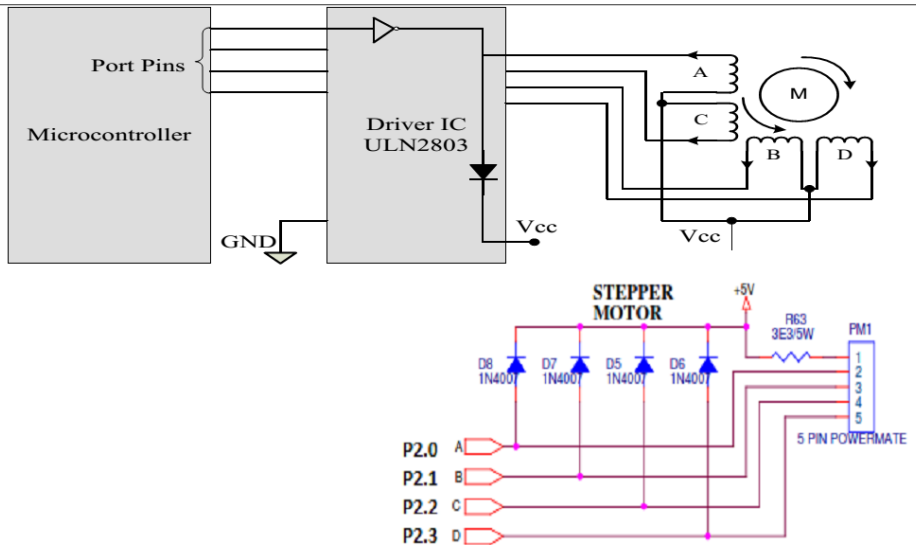
8	Observation Table,	•

	Look-up Table, Output	
9	Sample Calculations	• -
10	Graphs, Output	
1	Results & Analysis	• Message “Hello World” appearing for 5 times in hyperterminal.
12	Application Areas	• Embedded systems.
13	Remarks	
14	Faculty Signature with Date	

Experiment 04: Interface and Control a stepper Motor.

-	**Experiment No.:**	4	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Interface and Control a stepper Motor.						
2	Course Outcomes							
3	Aim	Interface and Control a stepper Motor.						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept	Interface and Control a stepper Motor.						
6	Procedure, Activity,, Pseudo Code	• **Procedure:** • 1. Create a project and include the above program using Keil tool. • 2. Create a Hex file and build. • 3. Flash the program onto the chip using FlashMagic software. • 4. Check the wave in a CRO.						
	Algorithm	• **Algorithm:** • 1. Configure the functionality of p2.0-p2.3 as general IO. • 2. Configure the p2.0-p2.3 as output port. • 3. Call continuously the function to rotate clockwise and anticlockwise with delay. **Clockwise rotation:** • 1. Initialize a variable with initial excitation value 0x00000001. • 2. Clear the four port bits and apply the excitation. • 3. Rotate the pattern left by one-bit position. • 4. Repeat the step 3 and 4 continuously to attain the required angle. • 5. For anticlockwise rotate the pattern right.						

7 Block, Circuit, Model Diagram, Reaction Equation, Expected Graph



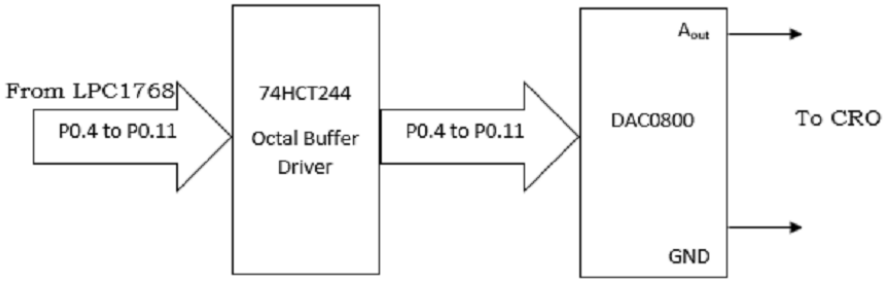
Program

Program:

```
#include <LPC17xx.H>
void clock_wise(void);
void anti_clock_wise(void);
unsigned long int var1,var2;
unsignedinti=0,j=0,k=0;

int main(void)
{
     LPC_PINCON->PINSEL4 = 0x00000000;   //P2.0 to P2.3 GPIO
     LPC_GPIO2->FIODIR = 0x0000000F;     //P2.0 to P2.3 output
     while(1)
     {
          for(j=0;j<50;j++)              //50 times in Clock wise Rotation
          clock_wise();
          for(k=0;k<65000;k++);          //Delay to show  anti_clock Rotation

          for(j=0;j<50;j++)         //50 times in  Anti Clock wise Rotation
          anti_clock_wise();
          for(k=0;k<65000;k++);          //Delay to show clock Rotation
     }                                   //End of while(1)
}                                        //End of main
void clock_wise(void)
{
     var1 = 0x00000001;                       //For Clockwise
for(i=0;i<=3;i++)                        //for A B C D Stepping
     {
        LPC_GPIO2->FIOCLR = 0X0000000F;
        LPC_GPIO2->FIOSET = var1;
     var1 = var1<<1;                     //For Clockwise
for(k=0;k<3000;k++);                     //for step speed variation
    }
}

void anti_clock_wise(void)
{
      var1 = 0x0000008;                                  //For Anticlockwise
for(i=0;i<=3;i++)                                  //for A B C D Stepping
     {
          LPC_GPIO2->FIOCLR = 0X0000000F;
      LPC_GPIO2->FIOSET = var1;
      var1 = var1>>1;                          //For Anticlockwise
for(k=0;k<3000;k++);                       //for step speed variation
     }
}
```

8	Observation Table, Look-up Table, Output	•
9	Sample Calculations	• -
10	Graphs, Outputs	
11	Results & Analysis	•
12	Application Areas	•
13	Remarks	
14	Faculty Signature with Date	

Experiment 05: Interface a DAC and generate Triangular and Square waveforms.

-	**Experiment No.:**	5	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Interface a DAC and generate Triangular and Square waveforms.						
2	Course Outcomes							
3	Aim	Interface a DAC and generate Triangular and Square waveforms.						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept							
6	Procedure, Program, Activity, Algorithm, Pseudo Code	**Algorithm for generating triangular wave:** • 1. Configure the port pins as GPIO. • 2. Make the configured port pins as output. • 3. Define a loop and keep incrementing the variable and send the data out through the port pins to DAC. • 4. Define another loop and keep decrementing the variable and send the data out through the same port pins to DAC. • 5.Repeat from step 4 infinite time, so as to generate the wave continuously. **Algorithm for generating square wave:** • 1. Start • 2. Configure the port pins as GPIO. • 3. Make the configured port pins as output. • 4. Make all the port pin high. 5. Insert a delay • 6. Make all the port pin low. • 7. Insert a delay. 8. Repeat from step 4 infinite time, so as to generate the wave continuously.						
		• **Procedure:** • 1. Create a project and include the above program using Keil tool. • 2. Create a Hex file and build. • 3. Flash the program onto the chip using FlashMagic software. • 4. Check the wave in a CRO.						

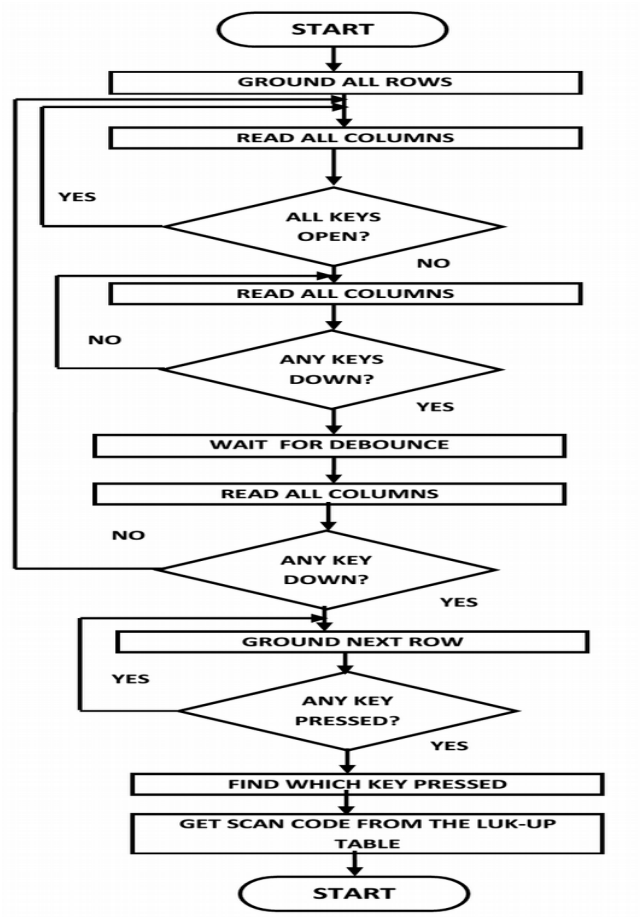
7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	**Program to generate triangular wave:**
		`#include <LPC17xx.H>`
		`int main ()`
		`{unsignedinti=0;` //global declaration
		`LPC_PINCON->PINSEL0=0x00000000;` //Configure port pins P0.4 to P0.11 as GPIO
		`LPC_GPIO0->FIODIR|=0x00000FF0;` //Make port pins P0.4 to P0.11 as output pins
		`while(1)` //Define an infinite loop to generate the triangular wave continuously
		`{`
		`for(i=0;i!=0xFF;i++)` //keep incrementing the value 'i' untill it reaches the max value
		`{`
		`LPC_GPIO0->FIOPIN=(i<<4);` //left shift 'i' 4 times so as to bring the data to port pins P0.4 to P0.11
		`}`
		`for(i=0xFF; i!=0; i--)` //decrement the value 'i' untill it reaches the min value
		`{`
		`LPC_GPIO0->FIOPIN=(i<<4);` //left shift 'i' 4 times to bring the data to second nibble
		`}`
		`}` //End of while(1)
		`}` //End of main

		Program to generate Square wave:
		`#include <LPC17xx.H>`
		`unsigned inti=0;` //global declaration
		`int main ()`
		`{`
		`LPC_PINCON->PINSEL0 = 0x00000000;` //Configure P0.4 to P0.11 as GPIO
		`LPC_GPIO0->FIODIR|=0x00000FF0;` //make port pins P0.4 to P0.11 as output
		`while(1)` //define an infinite loop
		`{`
		`LPC_GPIO0->FIOSET=0x00000FF0;` //make all the port pins P0.4 to P0.11 high
		`for(i=0; i<=9500; i++);` //Delay
		`LPC_GPIO0->FIOCLR=0x00000FF0;` //make all the port pins P0.4 to P0.11 high
		`for(i=0; i<=9500; i++);` //Delay
		`}` //end of while(1)
		`}` //end of main function

8	Observation Table, Look-up Table, Output	•
9	Sample Calculations	• -
10	Graphs, Outputs	
11	Results & Analysis	• Triangular wave displayed on CRO. • Square wave displayed on CRO
12	Application Areas	•
13	Remarks	
14	Faculty Signature with Date	

Experiment 07: Interface a 4x4 keyboard and display the key code on an LCD.

-	**Experiment No.:**	1	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Interface a 4x4 keyboard and display the key code on an LCD.						
2	Course Outcomes	Interface a 4x4 keyboard and display the key code on an LCD.						
3	Aim	Interface a 4x4 keyboard and display the key code on an LCD.						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept							



6	Procedure, Program, Activity, Algorithm, Pseudo Code	**Procedure:** • 1. Create a project and include the above program using Keil tool. • 2. Create a Hex file and build. • 3. Flash the program onto the chip using FlashMagic software. • 4. Check the key pressed on the LCD display.

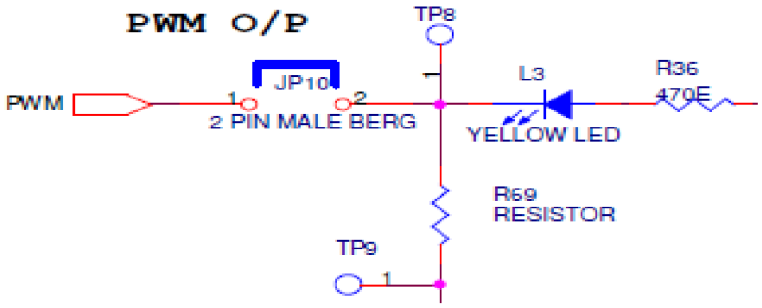
7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	`#include<LPC17xx.h>`
		`#include"lcd.h"`
		`voidscan(void);`
		`unsigned char Msg1[14] = "SKIT BENGALURU";`
		`unsigned char Msg2[13] = "KEY PRESSED=";`
		`unsigned char row, KEY_PRESSED, key;`
		`unsigned long int i, row_val, col_val;`
		`unsigned char SCAN_CODE[16] =`
		`{0x1E,0x1D,0x1B,0x17,`
		`0x2E,0x2D,0x2B,0x27,`
		`0x4E,0x4D,0x4B,0x47,`
		`0x8E,0x8D,0x8B,0x87};`
		`unsigned char ASCII_CODE[16] = {'0','1','2','3',`
		`'4','5','6','7',`

```
'8','9','A','B',
'C','D','E','F'};
int                    main(void)
{
LPC_PINCON->PINSEL3   &=   0xFFFF00FF;   //P1.20   to
P1.23                    MADE                    GPIO
LPC_PINCON->PINSEL0 &= 0x3FFFFFFF; //P0.15 as GPIO
(last     bit     of     first     half     word)
LPC_PINCON->PINSEL1   &=   0xFFFFFFC0;   //P0.16   to
P0.18 made GPIO (first 3 bits of second half word)
LPC_GPIO0->FIODIR   &=   ~0x00078000;   //made   Input
P0.15           to           P0.18           (cols)
LPC_GPIO1->FIODIR   |=   0x00F00000;   //made   output
P1.20           to           P1.23           (rows)
LPC_GPIO1->FIOSET           =           0x00F00000;
lcd_init();
temp1   =   0x80;   //point   to   first   line   of   LCD
lcd_com();
delay_lcd(800);
lcd_puts(&Msg1[0]);   //display   the   message   RNS
BENGALURU
temp1   =   0xC0;   //point   to   second   line   of   LCD
lcd_com();
delay_lcd(800);
lcd_puts(&Msg2[0]);   //display   the   message   KEY
PRESSED=
while(1)
{
while(1)
{
for(row=1;row<5;row++)
{
if(row                      ==                      1)
row_val                 =                0x00100000;
else            if(row             ==             2)
row_val                 =                0x00200000;
else            if(row             ==             3)
row_val                 =                0x00400000;
else            if(row             ==             4)
row_val                 =                0x00800000;
LPC_GPIO1->FIOSET           =           0x00F00000;
LPC_GPIO1->FIOCLR             =             row_val;
KEY_PRESSED                   =                   0;
scan();
if(KEY_PRESSED                  ==                  1)
break;
}                                          //end
for(row=1;row<5;row++)
if(KEY_PRESSED                  ==                  1)
break;
```

```
}                    //2nd                    while(1)
for(i=0;i<16;i++)
{
if(key                 ==                SCAN_CODE[i])
{
key                  =                  ASCII_CODE[i];
break;
}      //end      if(key      ==      SCAN_CODE[i])
}              //end              for(i=0;i<16;i++)
temp1                      =                      0xCC;
lcd_com();
delay_lcd(800);
lcd_puts(&key);
}             //end             while             1
}                    //end                    main
void                                      scan(void)
{
unsigned                long                col_val;
col_val             =             LPC_GPIO0->FIOPIN;
col_val                &=                0x00078000;
if(col_val   !=   0x00078000)   //   if   key   pressed
{
for(i=0;i<500;i++);      //      Debounce      delay
col_val             =             LPC_GPIO0->FIOPIN;
col_val                &=                0x00078000;
if(col_val  !=  0x00078000)  //  confirm  key  pressed
after                debounce                delay
{
KEY_PRESSED   =   1;   //   key   press   is   confirmed
col_val >>= 15; //col_val Shifted to come at lower
nibble           of           first           byte
row_val   >>=   16;   //row_val   shifted   to   come   at
higher       nibble       of       first       byte
key                  =                col_val|row_val;
}      //2nd      if(col_val      !=      0x00000000)
}     //1     st     if(col_val     !=     0x00000000)
} //end scan
```

8	Observation Table, Look-up Table, Output	•
9	Sample Calculations	• -
10	Graphs, Output	
11	Results & Analysis	• All the keys on the hex keypad were displayed on the LCD screen.
12	Application Areas	•
13	Remarks	
14	Faculty Signature with Date	

Experiment 08: Using the Internal PWM module of ARM controller generate PWM and vary its duty cycle.

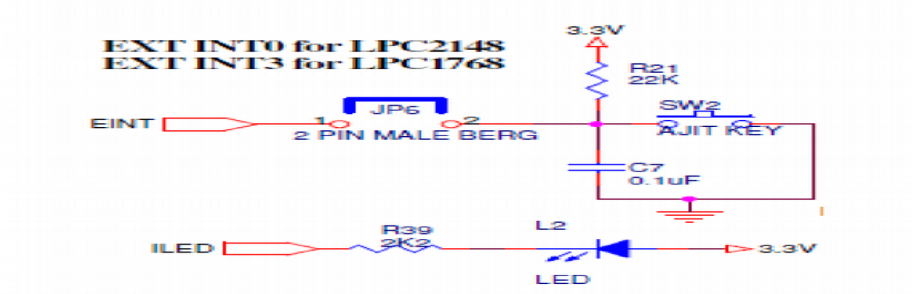
-	**Experiment No.:**	8	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Using the Internal PWM module of ARM controller generate PWM and vary its duty cycle.						
2	Course Outcomes	Using the Internal PWM module of ARM controller generate PWM and vary its duty cycle.						
3	Aim	Using the Internal PWM module of ARM controller generate PWM and vary its duty cycle.						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept							
6	Procedure, Program, Activity, Algorithm, Pseudo Code	• **Step1**: Set bit 6 of PCONP register to set PWM power on. • **Step2**: Configure PINSEL7 to set P3.25 for PWM1.2 • **Step3**: Program PWM Control Register (PCR) to enable single edge PWM and enable output. • **Step4**: Program Match Control Register (MCR) to reset and set the interrupt for the match register PWMMR0. • **Step5**: Set up a match count in MR0(match register 0). This register dictates the upper limit of terminal count. • **Step6**: Initialize MR2 (match register 2). The count value in the register determines the incremental steps (i.e increase in steps of 100,200….etc.). • **Step7**: Program Latch enable register (LER) to enable all the latches. • **Step8**: Program Timer Control Register (TCR) to set/reset counter and enable PWM.						
		• **Algorithm:** • 1. Program the necessary registers for the peripheral. • 2. Initialize the terminal count in a register. • 3. Initialize the incremental count in a register. • 4. Increment the count in accordance wit the value set in Step3. • 5. Go back to step3 if terminal count is reached. • 6. End.						

7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	`#include <LPC17xx.H>`
		`void pwm_init(void);`
		`void PWM1_IRQHandler(void);`
		`int main(void)`
		`{`						

```
pwm_init();
while(1);
     } //end of main
     void pwm_init(void)
     {
LPC_SC->PCONP |= (1<<6); //PWM1 is powered
     LPC_PINCON->PINSEL7 |= 0x000C0000;

LPC_PWM1->PCR = 0x00000400;

LPC_PWM1->MCR = 0x00000003;
LPC_PWM1->MR0 = 30000; //setup match register0
LPC_PWM1->MR2 = 0x00000100;
LPC_PWM1->LER = 0x000000FF; //enable shadow
copy                                    //register
LPC_PWM1->TCR = 0x00000002; //RESET COUNTER
AND                                    //PRESCALER
LPC_PWM1->TCR = 0x00000009; //enable PWM and
counter
NVIC_EnableIRQ(PWM1_IRQn);
return;
}
void                        PWM1_IRQHandler(void)
{
LPC_PWM1->IR = 0xff; //reset the Interrupts
if(LPC_PWM1->MR2<27000)
{
LPC_PWM1->MR2+=100;    //Increases    the    Duty
cycle          //at          every          match
if                          (LPC_PWM1->MR2>=27000)
{
LPC_PWM1->MR2=100; //If mr2 reaches 27000 //the
mr2        is        rolled        to        100
}
}

    LPC_PWM1->LER = 0x000000FF; //Enabling Latch
    register
    } //to copy new MR2 Value.
```

8	Observation Table, Look-up Table, Output	• Truth table verification
9	Sample Calculations	• -
10	Graphs, Outputs	
11	Results & Analysis	•
12	Application Areas	•
13	Remarks	
14	Faculty Signature with Date	

Experiment 09 : Demonstrate the use of an external interrupt to toggle an LED On/Off.

-	**Experiment No.:**	9	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Write verilog code to realize all the logic gates						
2	Course Outcomes	Create and verify functionality of various gates at the different level of abstractions.						
3	Aim	Write Verilog code to realize all the logic gates in behavioural, dataflow and gate level modeling.						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept							
6	Procedure, Program, Activity, Algorithm, Pseudo Code	**Algorithm:** • 1. Configure the pins as external interrupts in PINSELx register. • 2. Clear any pending interrupts in EXTINT. • 3. Configure the EINTx as Edge/Level triggered in EXTMODE register. • 4. Select the polarity (Falling/Rising Edge, Active Low/High) of the interrupt in EXTPOLAR register. • 5. Finally enable the interrupts by calling NVIC_EnableIRQ() with IRQ number.						
7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph							

		Program:
		`#include<LPC17xx.h>`
		`void EINT3_IRQHandler(void);`
		`int main(void)`
		`{`
		`LPC_PINCON->PINSEL4 |= 0x04000000;` //P2.13 as EINT3
		`LPC_PINCON->PINSEL4 &= 0xFCFFFFFF;` //P2.12 GPIO for LED
		`LPC_GPIO2->FIODIR = 0x00001000;` //P2.12 is assigned output
		`LPC_GPIO2->FIOSET = 0x00001000;` //Initiall LED is kept on
		`LPC_SC->EXTINT = 0x00000008;` //writing 1 clr the int, get set if there is int.
		`LPC_SC->EXTMODE = 0x00000008;` //EINT3 is initiated as edge senitive
		`LPC_SC->EXTPOLAR = 0x00000000;` //EINT3 is falling edge sensitive
		`NVIC_EnableIRQ(EINT3_IRQn);` //core_cm3.h
		`while(1);`
		`}`
		`void EINT3_IRQHandler(void)`
		`{`
		`LPC_SC->EXTINT = 0x00000008;` //cleares the interrupt
		`LPC_GPIO2->FIOPIN ^= 0x00001000;`
		`}`						

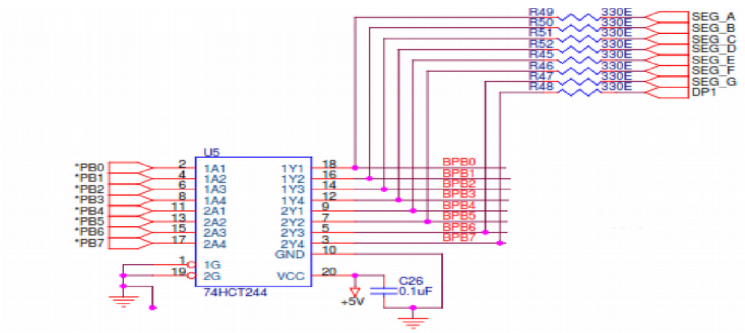
8	Observation Table, Look-up Table, Output	• Truth table verification						
9	Sample Calculations	• -						
10	Graphs, Output							
11	Results & Analysis	•						

EC

Prepared by Checked by Approved

12	Application Areas	•
13	Remarks	
14	Faculty Signature with Date	

Experiment 10: Display the Hex digits 0 to F on a 7-segment LED interface, with an appropriate delay in between.

-	**Experiment No.:**	10	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Display the Hex digits 0 to F on a 7-segment LED interface, with an appropriate delay in between.						
2	Course Outcomes	Display the Hex digits 0 to F on a 7-segment LED interface, with an appropriate delay in between.						
3	Aim	Display the Hex digits 0 to F on a 7-segment LED interface, with an appropriate delay in between.						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept	Basic structure of programming in verilog, Logical expression and Truth table for all the logic gates.						
6	Procedure, Program, Activity, Algorithm, Pseudo Code	• **Algorithm:** • 1. Start • 2. Prepare the lookup table for seven segment display. • 3. Configure the Pinsel0 and pinsel1 for general I/O. • 4. Configure P0.4-p0.11(data lines for seven segment display) and P0.19 and P0.20 (enable pins for seven segment) as output pin. • 5. Enable the two Displays. • 6. Clear the P0.4-P0.11 and send the seven-segment code one by one with delay. • 7. If the count is not equal to 10h(16) repeat step 6. • 8. Otherwise initialize the count to zero and repeat step 5 to 7 continuously. • 9. End.						
7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph							

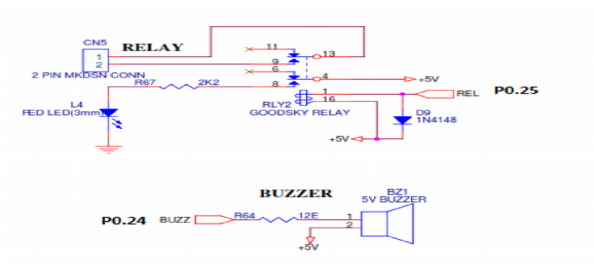
		`#include <LPC17xx.h>`
		`unsigned int delay, count=0, Switchcount=0, j;`
		`unsigned int Disp[16]={0x000003f0, 0x00000060, 0x000005b0,`
		`0x000004f0, 0x00000660,0x000006d0, 0x000007d0, 0x00000070,`
		`0x000007f0, 0x000006f0, 0x00000770,0x000007c0, 0x00000390,`						

```
0x000005e0,        0x00000790,        0x00000710};
#define ALLDISP 0x00180000 //Select all display by
setting          p0.19          and          p0.20
#define DATAPORT 0x00000ff0 //P0.4 to P0.11 : Data
lines      to      drive      Seven      Segments
int                  main                  (void)
{
LPC_PINCON->PINSEL0          =          0x00000000;
LPC_PINCON->PINSEL1          =          0x00000000;
LPC_GPIO0->FIODIR           =          0x00180ff0;
while(1)

{
LPC_GPIO0->FIOSET            |=            ALLDISP;
LPC_GPIO0->FIOCLR  =0x00000ff0;  //clear  the  data
lines         to         7-segment         displays
LPC_GPIO0->FIOSET=Disp[Switchcount]; //get the 7-
seg     display     value     from     the     array
for(delay=0;delay<300000;delay++);   //   1s   delay
Switchcount++;
if(Switchcount  ==  0x10)  //  0  to  F  has  been
displayed      ?      go      back      to      0
{
Switchcount                  =                  0;
LPC_GPIO0->FIOCLR=0x00180ff0;
}
```

8	Observation Table, Look-up Table, Output	•
9	Sample Calculations	•
10	Graphs, Outputs	
11	Results & Analysis	•
12	Application Areas	•
13	Remarks	
14	Faculty Signature with Date	

Experiment 11 : Interface a simple Switch and display its status through Relay, Buzzer and LED.

-	**Experiment No.:**	11	**Marks**		**Date Planned**		**Date Conducted**	
1	Title							
2	Course Outcomes							
3	Aim							
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept							
6	Procedure, Program, Activity, Algorithm,	**Algorithm:** • 1. Start						

	Pseudo Code	• 2. Configure the pin for general IO. • 3. Configure the pin as output port. • 4. Check if the switch is pressed, if yes turn on the buzzer and relay. • 5. Introduce some delay • 6. Turn off the buzzer and relay. • 7. End
7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	

		• `#include <LPC17xx.H>`
		`unsigned int count=0;`
		`int main(void)`
		`{`
		`unsigned int i;`
		`LPC_PINCON->PINSEL1 = 0x00000000; //P0.24 AND P0.25 GPIO`
		`LPC_GPIO0->FIODIR = 0x03000000; //P0.24 AND P0.25 output`
		`while(1)`
		`{`
		`if(!(LPC_GPIO2->FIOPIN & 0x00000800))`
		`{`
		`for(i=0;i<10;i++)`
		`{`
		`LPC_GPIO0->FIOSET = 0x03000000; //relay on`
		`for(i=0;i<10000;i++);`
		`}`
		`else`
		`{`
		`LPC_GPIO0->FIOCLR = 0x03000000; //relay off`
		`for(i=0;i<100000;i++);`
		`} //end int main(void)`
		`}`
		`}`

8	Observation Table, Look-up Table, Output	•
9	Sample Calculations	•
10	Graphs, Outputs	
11	Results & Analysis	•
12	Application Areas	•
13	Remarks	
14	Faculty Signature with Date	