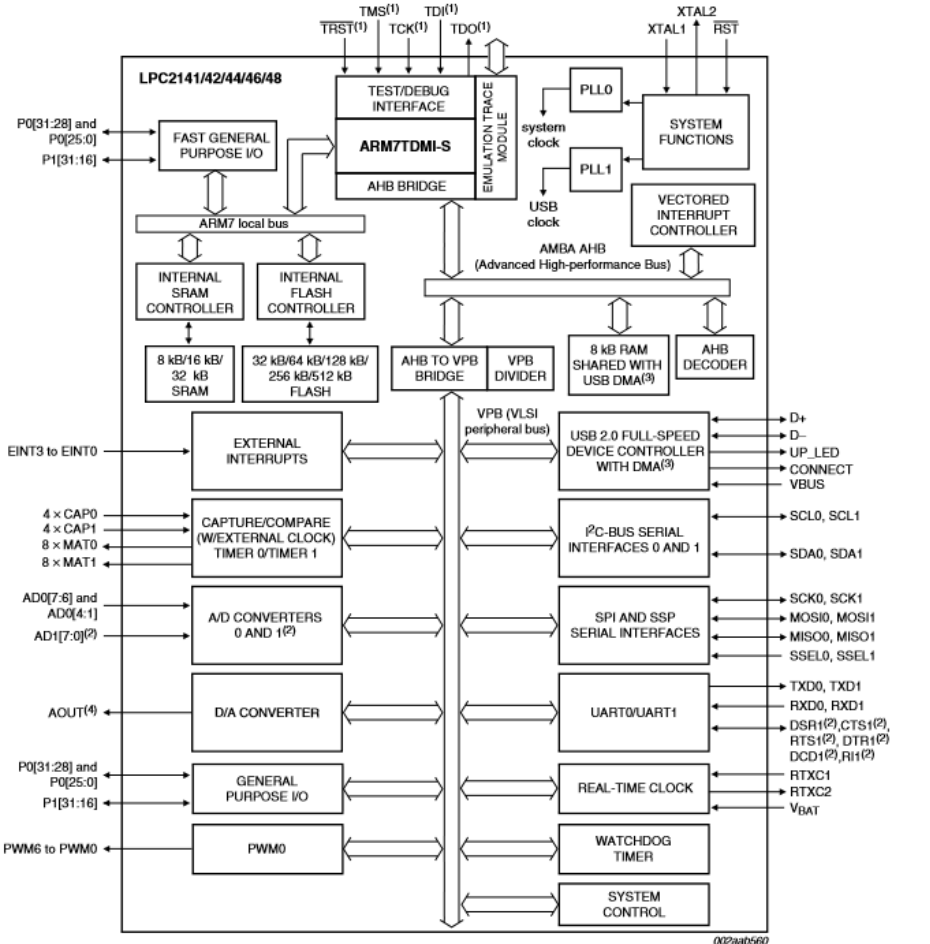
	<p style="text-align: center;">R V College of Engineering Department of Computer Science and Engineering CIE - I: Scheme</p>	
<p>Course: (Code)</p>	<p style="text-align: center;">IOT & Embedded Computing (CS344AI)</p>	<p>Semester : 4th semester</p>

PART B

<p>1</p>	<p>Block diagram – 4 marks</p> <p>Listing peripherals and corresponding Applications – 6 Marks</p> 	<p>10</p>	<p>L2</p>	<p>CO2</p>
<p>2</p>	<p>Each sub question carries 5 marks</p> <p>1.</p>	<p>10</p>	<p>L3</p>	<p>CO2</p>

Criteria	General Purpose Computing System	Embedded System			
Contents	A system which is a combination of a generic hardware and a General Purpose Operating System for executing a variety of applications.	A system which is a combination of special purpose hardware and embedded OS for executing a specific set of applications.			
OS	It contains a general purpose operating system (GPOS).	It may or not contain an operating system for functioning.			
Alterations	Applications are alterable (programmable) by the user. (It is possible for the end user to re-install the OS and also add or remove user applications.)	The firmware of the embedded system is pre-programmed and it is non-alterable by the end-user.			
Key factor	Performance is the key deciding factor in the selection of the system. Faster is better.	Application specific requirements (like performance, power requirements, memory usage, etc.) are key deciding factors.			
Power Consumption	More	Less			
Response Time	Not critical	Critical for some applications			
Execution	Need not be deterministic	Deterministic for certain types of ES like ' Hard Real Time ' systems.			

2. Explain the registers corresponding the ARM modes of operating

User and system

r0					
r1					
r2					
r3					
r4					
r5					
r6	Fast interrupt request				
r7					
r8	r8_fiq				
r9	r9_fiq				
r10	r10_fiq				
r11	r11_fiq	Interrupt request			
r12	r12_fiq		Supervisor	Undefined	Abort
r13 sp	r13_fiq	r13_irq	r13_svc	r13_undef	r13_abt
r14 lr	r14_fiq	r14_irq	r14_svc	r14_undef	r14_abt
r15 pc					
cpsr					
-	spsr_fiq	spsr_irq	spsr_svc	spsr_undef	spsr_abt

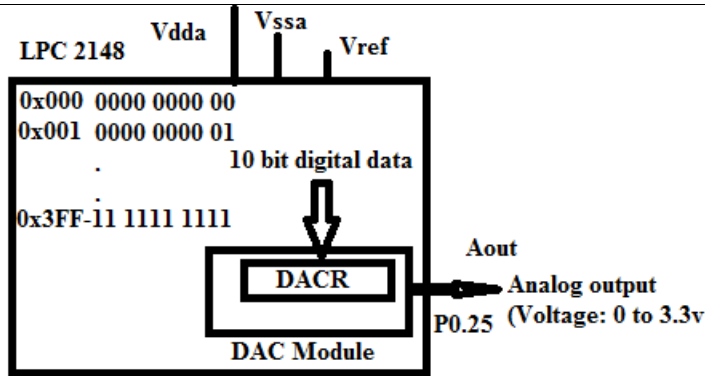
3	<p>Seven Segment Display Program:</p> <pre>//P0.19 Data pin of 1st shift register //P0.20 Clock pin of shift registers, make 1 to 0 //P0.30 Strobe pin of shift registers: 1 to 0 #include <lpc214x.h> #define LED_OFF (IO0SET = 1U << 31) #define LED_ON (IO0CLR = 1U << 31) #define PLOCK 0x00000400 void delay_ms(unsigned int j); void SystemInit(void); unsigned char getAlphaCode(unsigned char alphachar);</pre>	10	L3	CO3
---	---	----	----	-----

<pre> void alphadisp7SEG(char *buf); int main() { IO0DIR = 1U << 31 1U << 19 1U << 20 1U << 30 ; <i>// to set as o/p's</i> LED_ON; <i>// make D7 Led on .. just indicate the program is running</i> while(1) { alphadisp7SEG("fire "); delay_ms(500); alphadisp7SEG("help "); delay_ms(500); } } unsigned char getAlphaCode(unsigned char alphachar) { switch (alphachar) { <i>// dp g f e d c b a - common anode: 0 segment on, 1 segment off</i> case 'I': return 0xf9; case 'O': return 0xc0; case 'T': return 0x93; case 'B':return 0x80; case 'O':return 0xc0; case 'A': return 0xf7; case 'R':return 0xf7; case 'D':return 0xa1; case ' ': return 0xff; <i>//similarly add for other digit/characters</i> default : break; } return 0xff; } void alphadisp7SEG(char *buf) { unsigned char i,j; unsigned char seg7_data,temp=0; for(i=0;i<5;i++) <i>// because only 5 seven segment digits are present</i> { seg7_data = getAlphaCode(*(buf+i)); <i>// instead of this look up table can be used</i> <i>// to shift the segment data(8bits)to the hardware (shift registers) using</i> <i>Data,Clock,Strobe</i> for (j=0 ; j<8; j++) { <i>//get one bit of data for serial sending</i> temp = seg7_data & 0x80; <i>// shift data from Most significan bit (D7)</i> if(temp == 0x80) IOSET0 = 1 << 19; <i>//IOSET0 0x00080000;</i> else IOCLR0 = 1 << 19; <i>//IOCLR0 0x00080000;</i> <i>//send one clock pulse</i> IOSET0 = 1 << 20; <i>//IOSET0 0x00100000;</i> </pre>			
--	--	--	--

	<pre> delay_ms(1); IOCLR0 = 1 << 20; //IOCLR0 0x00100000; seg7_data = seg7_data << 1; // get next bit into D7 position } } // send the strobe signal IOSET0 = 1 << 30; //IOSET0 0x40000000; delay_ms(1); //nop(); IOCLR0 = 1 << 30; //IOCLR0 0x40000000; return; } void delay_ms(unsigned int j) { unsigned int x,i; for(i=0;i<j;i++) { for(x=0; x<10000; x++); } } </pre>			
4	<p>The diagram illustrates the hardware setup for the keypad and stepper motor. The LPC2148 microcontroller is connected to a 4x7 matrix keypad. The keypad's rows are connected to P0.16, P0.17, P0.18, and P0.19, labeled 'selecting Rows'. The columns are connected to P1.19, P1.18, P1.17, and P1.16, labeled 'Columns data'. An 'ENTER' key is also present. The microcontroller's P0.20, P0.21, P0.22, and P0.23 pins are connected to the stepper motor's control pins. The stepper motor is connected to a ULN2803 driver, which has IN1-OUT1 to IN4-OUT4 pins. The ULN2803 is powered by a +12V supply. The keypad is powered by a 3.3V supply. The microcontroller has Vdd, Vss, XTAL1, and XTAL2 pins.</p>	10	L4	CO3
	#include <lpc214x.h>			

<pre> #include <string.h> #define COL0 (IO1PIN & 1<<19) #define COL1 (IO1PIN & 1<<18) #define COL2 (IO1PIN & 1<<17) #define COL3 (IO1PIN & 1<<16) #define LED_ON (IO0CLR = 1U<<31) #define LED_OFF (IO0SET = 1U<<31) #define ENTER 10 void delay_ms(unsigned int); char getKey(void); void open(void); // to open the door void close(void); // to close the door char ch,keys[5],password[5] = "0123"; unsigned char len = 0; unsigned int i = 0; int main () { char ch; IO0DIR = 0x0f<<16 ; do { i = 0; // read the password while (1) { if ((ch = getKey()) == ENTER) break; keys[i++]=ch; } keys[i] = '\0'; // null character, to make it string if (strcmp (keys, password) ==0) { open(); // rotate clockwise for 90 degree, open the door //Wait for a key 'b' to close the door While ((ch = getKey ()) != 'a') { } ; close();// rotate anticlockwise for 90 degree, close the door } }while(1); } void delay_ms(unsigned int ms){ unsigned int x, i; for(x = 0; x < ms; x++) for(i = 0; i < 10000; i++); </pre>			
---	--	--	--

	<pre> } char getKey() { unsigned char lookup_table[4][4]={ {'0', '1', '2','3'}, {'4', '5', '6','7'}, {'8', '9', 'a',10}, {'c', 'd', 'e','f'}}; unsigned char rowsel=0,colsel=0; while(1) { //check for keypress in row0,make row0 '0',row1=row2=row3='1' rowsel=0;IO0SET = 0X000F0000;IO0CLR = 1 << 16; if(COL0==0){colsel=0;break;};if(COL1==0){colsel=1;break;}; if(COL2==0){colsel=2;break;};if(COL3==0){colsel=3;break;}; //check for keypress in other rows delay_ms(50); // debouncing delay // wait for a key release while(COL0==0 COL1==0 COL2==0 COL3==0); delay_ms(50); // debouncing delay return lookup_table[rowsel][colsel]; } void open (){ for (int i = 0; i < 20; i++) { IO0CLR = 0X000F0000; IO0SET = 0X00080000; delay_ms(15); IO0CLR = 0X000F0000; IO0SET = 0X00040000; delay_ms(15); IO0CLR = 0X000F0000; IO0SET = 0X00020000; delay_ms(15); IO0CLR = 0X000F0000; IO0SET = 0X00010000; delay_ms(15); } IO0CLR = 0x00ff0000; } void close () { for (int i = 0; i < 20; i++) { IO0CLR = 0X000F0000; IO0SET = 0X00010000; delay_ms(15); IO0CLR = 0X000F0000; IO0SET = 0X00020000; delay_ms(15); IO0CLR = 0X000F0000; IO0SET = 0X00040000; delay_ms(15); IO0CLR = 0X000F0000; IO0SET = 0X00080000; delay_ms(15); } IO0CLR = 0x00ff0000; } </pre>			
5	DAC Module of LPC 2148: LPC 2148, provides in-built 10-bit Digital to Analog Converter, as shown in the figure below.	10	L3	CO3



DAC module of LPC 2148 is a 10 bit Digital to Analog converter used to convert 10 bit Digital data to corresponding Analog voltage.□

□ Digital I/P : **000 to 3FF (0 to 1023)**, corresponding Analog O/P : **0V to 3.3V**□

□ Resolution = $(3.3/1024) \approx 3.2\text{mili volts}$ □

```
#include <lpc214x.h>
#include <stdio.h>
#define SW2 (IO0PIN & (1 << 14))
#define SW3 (IO0PIN & (1 << 15))
#define SW4 (IO1PIN & (1 << 18))
#define SW5 (IO1PIN & (1 << 19))
#define SW6 (IO1PIN & (1 << 20))
static void delay_ms(unsigned int j); //millisecond delay
short int sine_table[ ] =
{512+0,512+53,512+106,512+158,512+208,512+256,512+300,512+342,512+380,512+413,512+442,512+467,512+486,512+503,512+510,512+511,512+510,512+503,512+486,512+467,512+442,512+413,512+380,512+342,512+300,512+256,512+208,512+158,512+106,512+53,512+0,512-53,512-106,512-158,512-208,512-256,512-300,512-342,512-380,512-413,512-442,512-467,512-486,512-503,512-510,512-511,512-510,512-503,512-486,512-467,512-442,512-413,512-380,512-342,512-300,512-256,512-208,512-158,512-106,512-53};
short int sine_rect_table[ ] =
{512+0,512+53,512+106,512+158,512+208,512+256,512+300,512+342,512+380,512+413,512+442,512+467,512+486,512+503,512+510,512+511,512+510,512+503,512+486,512+467,512+442,512+413,512+380,512+342,512+300,512+256,512+208,512+158,512+106,512+53,512+0};
int main()
{
short int value,i=0;
PINSEL1 |= 0x00080000; /* P0.25 as DAC output :option 3 - 10
While(1){
if ( !SW4) /* If switch for triangular wave is pressed */
{
value = 0;
while ( value != 1023 )
{
DACR = ( (1<<16) | (value<<6) );
```

<pre> value++; } while (value != 0) { DACR = ((1<<16) (value<<6)); value--; } } void delay_ms(unsigned int j) { unsigned int x,i; for(i=0;i<j;i++) { for(x=0; x<10000; x++); } } </pre>			
--	--	--	--

Course Outcomes: After completing the course, the students will be able to:-	
CO 1	Apply Embedded System and IoT fundamentals and formulate sustainable societal relevant cost effective solutions.
CO 2	Demonstrate the development of software programs using Embedded C, using Microcontrollers and different sensors and peripherals to build embedded system applications.
CO3	Design smart systems using various I/O peripherals, Sensors, embedded protocols like UART,I2C,SPI using modern tools like Keil IDE software for various domains like Healthcare, automation, agriculture, smart cities and others.
CO 4	Indulge in developing Novel multi-disciplinary IoT projects using prototype boards, with effective oral & written communication skills and working in teams.
CO 5	Engage in Lifelong Learning by investigating and executing real world societal problems using engineering tools – Cross compilers, debuggers and simulators, emerging processor and controller-based hardware platforms, IOT cloud infrastructure & protocols.

BT LEVELS	L1	L2	L3	L4	L5	L6	COS	CO1	CO2	CO3	CO4
MARKS		10	30	10					20	30	