

ARM Trainer Kit



Start Your Embedded System Design Journey Today..!

LPC2148 essential development features a plug and play design that makes it easy for connections and helps Students, hobbyists, enthusiasts, and professionals to focus more on Program / application development. LPC2148 Trainer Kit equipped with on board IO's, communication interfaces & peripherals. It is really easy to design, experiment with, and test circuits without soldering. It's used in many educational institutions and R&D LAB acrossthe world.



Features:

- Plug & Play Interface Connectivity
- Professional EMI/RFI Complaint
 PCB Layout Design
- Modular Block design makes Easy access & quick Prototyping
- FRC connectivity features minimize the connection Error.
- High Quality Grade PCB with wooden Enclosure.
- Stackable daughter board LPC2148.
- On board debugging JTAG Option
- USB onboard programming
- 8 interfacing LED's.
- 1 * 4 interfacing keys.
- 4* 4 interfacing keypad matrix.
- Two channel RS232 port for communication.
- 3 ADC potentiometer input interface.
- 16X2 LCD, OLED interface.

Scope of Learning Experiments:

- LED blinking
- 8 bit LED Left shift, Right shiftand counting operation
- Keypad Interrupt Interface
- 16*2 LCD interface
- Matrix Keypad Interface
- ADC & DAC interface
- Traffic Light Signal Interface
- 8 bit DIP switch interface
- 7 Segment interface
- L298 Driver for DC Motor and Stepper motor interface
- Elevator Interface

- 46 general purpose IO
- On Board Power supply 3.3V, 5V 12V,GND.
- 8 pin DIP switches.
- On board voltage level converter
- Board enabled with the header forstacking the WiFi module.
- Board enabled with the header forstacking the Arduino shields.
- ON board Micro SD card
- ON/OFF slide switch.
- RDL bus.
- External jumper nodes.
- Reset button.
- Power plug-in DC Socket.
- Power supply indicator LED.
- Test led for Tx, Rx.
- 7Seg Multiplexed Display.
- FT232RL USB communication.
- Buzzer, Relay interface
- RS485, RS232 serial communication
- UART Operation
- RTC DS1307 I2C protocol interface
- AT24C04 EEPROM I2C protocol interface
- RF/WiFi Communication
- SPI protocol interface
- Temperature Sensor Interface
- Automatic Number Plate Recognition



ARM Board Narration:



Index

1.	Pre-	Requisites5
	2.1.	How to install Keil uVision 46
	2.2	How to install FTDI driver
	2.3	How to install Flash Magic Utility14
3.	Libra	ary Files (LibFiles)
4.	How	ν to Create a New Project in Keil μ Vision416
5.	How	to Enable Hex File Generation
6.	How	to Upload Hex file Using Flash Magic
7.	How	to Upload Hex file Using Philips Flash Utility
8.	Lab	Programs
	8.1.	Blinking an LED
	8.2.	Liquid Crystal Display
	8.3.	ADC
	8.4.	UART
	8.5.	Real Time Clock
	8.6.	Hex Keypad 49
	8.7.	Stepper Motor
	8.8.	PWM
	8.9.	EEPROM



1. Pre-Requisites

On your system you need to have the following Software installed before executing lab programs.

- 1. Keil uVision 4 software
- 2. FTDI driver
- 3. Flash Magic Utility

Follow the next section to install all these software.



2. Installation

2.1. How to install Keil uVision 4

Follow the steps below to install Keil uVision 4 on your system.

- 1. Run setup file MDK-ARM V4.13a.exe (Double click the file icon).
- 2. Setup Window appears....

Setup MDK-ARM V4.13a	
Welcome to Keil µVision Release 10/2010	
This SETUP program installs:	
MDK-ARM V4.13a	
This SETUP program may be used to update a previous produc However, you should make a backup copy before proceeding.	st installation.
It is recommended that you exit all Windows programs before co	ontinuing with SETUP.
Follow the instructions to complete the product installation.	
— Kell µVision4-Setup —	<< Back Next >> Cancel

3. Click Next.

Setup MDK-ARM V4.13a	×										
License Agreement Please read the following license agreement carefully.											
To continue with SETUP, you must accept the terms of the Lic agreement, click the check box below.	ense Agreement. To accept the										
End-User License Agreement for ARM Keil Software											
THIS END USER LICENCE AGREEMENT ("LICENCE") IS A LEGAL AGREEMENT BETWEEN YOU (EITHER A SINGLE INDIVIDUAL, OR SINGLE LEGAL ENTITY) AND ARM LIMITED ("ARM") FOR THE USE OF THE SOFTWARE ACCOMPANYING THIS LICENCE. ARM IS ONLY WILLING TO LICENSE THE SOFTWARE TO YOU ON CONDITION THAT YOU ACCEPT ALL OF THE TERMS IN THIS LICENCE. BY											
I agree to all the terms of the preceding License Agreement	1										
— Keil µVision4 Setup —	<cancel< th=""></cancel<>										



4. Click Checkbox of agreement in order to proceed and click Next



5. In folder selection window, select suitable directory for the installation and press next (default directory should work fine!)

Setup MDK-ARM V4.13a	X
Folder Selection Select the folder where SETUP will install files.	
SETUP will install $\mu Vision4$ in the following folder.	
To install to this folder, press 'Next'. To install to a different folde folder. ┌ Destination Folder	r, press 'Browse' and select another
C:\Keil	Browse
Update Installation: Create backup tool folder	
— Keil µVision4 Setup	<< Back Next >> Cancel



6. Enter customer information and click next

Setup MDK-ARM V4.	Setup MDK-ARM V4.13a 🛛 🛛 🗙												
Customer Informati Please enter your in	on Iformation.												
Please enter your na	me, the name of the company for whom yo	ou work and your E-mail address.											
First Name:	Pratik												
Last Name:	Deshpande												
Company Name:	Embedded Market												
E-mail:	, pratik@embeddedmarket.com												
— Keil μVision4 Setup	L	<< Back Next >> Cancel											

7. The setup should begin. Setup status window will show installation of different files. Wait until it gets installed completely.

Setup MDK-ARM V4.13a	X
Setup Status	
μVision Setup is performing the requested operations. Install Files Installing dependency.h.	
— Keil µVision4 Setup —	Cancel



8. Once the installation gets over, message should appear with text "File installation Completed", Click Next without making any changes anywhere in the window.

Setup MDK-ARM V4.13a	
File installation completed	
μVision Setup has installed all files successfully.	
Add example projects to the recently used project list.	
Preselect Example Projects for	
Simulated Hardware	-
— Keil μVision4 Setup ———	
	<< Back Next >> Cancel

9. Next message will confirm "Keil uVision 4 Setup completed". Click Finish

Setup MDK-ARM V4.13a	X
Keil µVision4 Setup completed MDK-ARM V4.13a	
µVision Setup has performed all requested operations successfully. I Launch Driver Installation: "ULINK Pro Driver V1.0" I Show Release Notes.	
— Keil μVision4 Setup ————————————————————————————————————	Back Finish Cancel

- **10.** This will lead you to the webpage of Keil Development Suit for ARM. This will provide you release notes for Keil Development kit. This document gives you the brief idea of which microcontrollers are being added in the Keil uVision 4.
- Run "Keil uVision 4" by double clicking the "Keil uVision 4" icon from the desktop. Same can be done from Start menu.



2.2 How to install FTDI driver

1. Visit FTDI's VCP Drivers page (http://www.ftdichip.com/Drivers/VCP.htm) for the latest

download of the Windows FTDI Driver executable and clicking on the Window's **"Available as a setup executable"** link. Make sure to unzip the executable before proceeding to the next step.

FTDI ONP	Future Technol THE USE	ogy Devic II/II/DGING 5	en Inter iou <i>nn</i> ow	national s ser ew	Ltd. ISTA					
Home Products Drivers	 Vistaal COM Por Tris page systems the VCI 	t Dirlyines * diwes current)	e available for	FTQL devices						
	Par DJXX Direct drivers, p Installation guides are ave	leans click <u>hery</u> lable from the <u>b</u>	statistics. Guid	an page of th	e Docum	erfa sector	curtation with	e for estat	lad oparia	log system.
Stupport Andraid EVE MCU Soles Network Web Shop Newsletter Corporate Contact Us	VEEP Drivern Visual COM pert (VCP) dri access a standard COM p This scheme, is publicly using temages (industry, turin menad, attribute user) FTD driver hay be user) If TD driver hay be user) If a course why be fitted If	vera cause the l ort of allow Technolo all an Order the of Instance region of Order the system and the any form and all the any form of another ID or of the Original Technology	USB device to sign Devices () of the scale device comment of the provide scale of the second scale space of the scale place devices the scale place	appear as an remediated bit of or shall be use to definite good any well withing to based on Fi spans weet. It of dia hasa,	n addition missel 1 a schrojog minaento con any w 100 parts pri tempi tempi tempi tempi tempi tempi	er COM port o (s) ent any posities of the episated for watched parentiality o	averatore Arrenatore Artenatore Artenatore Constantion Free prints	n the PC - misled in potential an potential and potential	Applicatio anartisa, 1 Bable far source it at acar it at	e software can access the USB device in the same way as it would exclusing but was testing to the implied treatments of merimination (and any check testing) or command galidal eventuary or the appendial tempton interese can ded and on any theory of National Constitu- tion of the positivity of event demage.
	Connectly Supported VCP I	Detwarte		p	TOCERSI	or Archites	ture.			
	Operating System	Release Date	x04 (32- bit)	x64 (64- bit)	PPC	ARM	MPSI	MIPSIV	584	Commente
Enter, our Customer Burvey for your churice to win!	Windows"	2017-08-30	2 12 28	2,92.28		8	15	8	×.	WHat Centles and the With and proc Available as a setup executable Planar row



2. Choose '*Run*' once it is has finished downloading, or find the file you just

downloaded "CDM21228_Setup.exe" and double-click it to run it.

	COLUMN AND A COLUMN AND A COLUMN AND	CHE LOAD DRIVE T											
Virtual COM Port Drivers									-	• 🛛 • 🕞	👼 + Page+	Safety +	Tools •
night" the Seatt	ansing in any way bu FTDI drivers may be o FTDI drivers may be o If a custom vendor ID WHQL re-certification	t of the use of used only in c listributed in a and/or produc (as a result of of	this software, onjunction with any form as for ct ID or descrip <u>f making these</u> is or CDM20828	even if advi h products t ng as licens tion string i chances Setup eve fin	sed of th based on a informa are used.	e possibility FTDI parts ation is not it is the re hillichip.com	v of such d modified sponsibility	amage y of the pro	duct man	ufacturer to main	tain any chang	es and out	bsequent
	Currently Supported	I VCP Dri	Fie Download - Do you want	Security War to run or sa	miny we this fi	le?		1					
	Operating System	Releas Date		iene: CDM2	0828_Setu	().exe	SH4		Comment	5			
	Windows*	2013-02		For waw.	Rin	- Seve		ancel	-	2 Auqui	08.28 WHQL 0 able as <u>sotup (</u> Release No	Jertified Intecutable	
	Linux	2009-05-	I White	files from the li tally harm your	rtemet car	e n be useful, th if you do not	is file type ca inual the sou	n tos. do not		All FTDI devic Refer to Th	es now support kernel 3 0 0 -101 if you nee VID/PID in Li	ed in Ubun -19 d a custon nux	eu 11.10, n VCP
	Mac OS X	2012-08-1	- 10.04	save this soft	Nale, Wha	La the risk?				Refer to Th	-105 if you nee VID/PID in MA	d a custon C OS	+ VCP
	Windows CE 4.2- 6.2**	2012-01-06	1 1 0 10	1000		1 1.0.14	1,1.0.10	1 1 0 10	1 1 0 10				
	Windows CE 6.0	2012-01-05	110.10	1.41	14	1.1.0.14	1.1.0.10	1.1.0.10	1.1.0.10				
	"Includes the followin Windows Server 2008 "Also, as Windows B	g versions of t I R2 and Wind RT is a close	the Windows o dows 8 d system not a	perating sy allowing for	stem W	ndows XP,	Windows	Server 200 Windows	3. Window 8 driver wil	vs Vista. Windo	vis Server 2008 s variant of the	Windows	7.



3. Choose 'Extract' and continue through the installation until it finishes.





4. If everything was successful, you should see some nice green check marks, indicating success!



Note: You may need administrator privileges on your machine in order for this to run properly.



2.3 How to install Flash Magic Utility

- 1. Visit FlashMagic website <u>http://www.flashmagictool.com/</u> and download the file FlashMagic.exe.
- 2. Execute the downloaded file FlashMagic.exe, and follow the instructions.
- Start Flash Magic by selecting it from the Start Menu. In the Flash Magic windows select Options > Advanced Options ... menu item. In the window that appears enable the check box that says Use DTR and RTS to control RST and P0.14, and click on Ok.

When this option is enabled, during code download, the flashing tool will automatically switch the device into ISP mode. For more information on this, see the board user manual.

Advanced Options	×
Communications Hardware Config Security Just In Time Code Timeouts Misc	
Use DTR and RTS to control RST and PSEN	
Keep RTS asserted while COM Port open	
T1: 50 ms T2: 100 ms	
Assert DTR and RTS while COM Port open	
Cancel OK	



3. Library Files (LibFiles)

Basically "LibFiles" contains "system_lpc21xx.c" and ".h" files.

Before executing lab programs it is important to have this library files in our system as we are going to link our programs to this folder.

Place "LibFiles" folder in a common path in all the computers. (ex: C->Documents->ARM->LibFiles)



4. How to Create a New Project in Keil µVision4

1. Open the Keil software and select "**Project-> New µvision project**" as shown below.

W										P١	WM_Tutori	ial - µVision4	
File Edit View	Proje	ct Flash	Debug	Peripherals	Tools	SVCS	Window	Help					
i 🗋 💕 🛃 🥔		New µVisio	n Project										
		New Multi-	Project Wo	orkspace									i –
Project		Open Proje	ct										-
Torgett 1		Close Proje	ct										-
	3	Export										•	
start	L I	Manage										•	
📄 🔛 mair	1			-1 'Terret 1'									
	P	Demous File	te for large	et larget I									
s	t 🔊	Options for	r File 'main									Alt+F7	
	<u> </u>	options for	The main									AILTI	
		Clean targe	:t										
E syste		Build target	t									F7	
En Syste Rebuild all target files													
		Batch Build											
		Translate F:	\lutorials_	And_Codes\L	pc2148\I	main.c						Ctrl+F/	
		Stop build											
	\checkmark	1 F:\Tutoria	ls_And_Co	des\Lpc2148\	PWM_Tu	itorial.uv	proj						
	1	2 D:\Shared	IFolder\Bo	ardCodes\Lpo	:1768\Ex	ploreLpd	1768_Break	Out\09a	-RTC_UART\	09a-RTC_0	JART.uvproj		
	-	3 D:\Shared	IFolder\Bo	ardCodes\Lpo	:1768\Ex	ploreLpd	1768_Break	Out\06a	UART_Tx_C	har\06a_l	JART_Tx_Char	r.uvproj	
	1	4 D:\Shared	IFolder\Bo	ardCodes\Lpo	:1768\Ex	ploreLpd	1768_Break	Out\05a	-Lcd_4bit_D	isplay\05a	a-Lcd_4bit_Di	splay.uvproj	
	1	5 D:\Shared	IFolder\Bo	ardCodes\Lpo	:1768\Ex	ploreLpd	1768_Break	Out\04a	-Lcd_8bit_D	isplay\04a	a-Lcd_8bit_Di	splay.uvproj	
		6 D:\Shared	JFolder\Bo	ardCodes\Lpo	:1768\Ex	ploreLpc	1768_Break	Out\01a	I-LedBlinkin	g\01a-Led	Blinking.uvp	iroj	
		7 C:\Keil\AR	M\Example	es\Measure\N	leasure.	uvproj							
<		o Ci\Keil\AR		Examples (Irat	nic\iraff	ic.uvproj							
🔚 Р. 🧒 В. {} F.	1	10 CI Kell AR	INI\Example		o.uvproj	ADUmber							
Build Output		TO C:\Kell\A	KWI/D04/0	s (ken (WICD21)	JO\DIINK]	у оппку.	avproj						
Program Size:	Code	e=804 RC)-data=	292 RW-da	ta=0	ZI-da	ta=608						
".\Objects\PW	M Tu	nex fil torial.a	axf" -	0 Error(s	s), O	Warni	ng(s).						
-	-												
Create a new uVisio	n proie	ct											
	M	0											



2. Browse to your project folder (/Create a folder) and provide the project name and click on save.

🕅 Create Ne	w Project	×
	✓ 🖒 Search LedBlink	٩
Organize 🔻 New folder	8== ▼	0
Ibbraries Ibbraries Documents Music Pictures Ibbraries Videos Ibbraries Homegroup Ibbraries Computer Ibbraries Local Disk (C:) Ibbraries Ibbraries Ibbraries Ibbraries Ibbraries	Date modified Type Size	
File name: LedBlink		~
Save as type: Project Files (*.uvproj)		~
➢ Hide Folders	Save	

 Once the project is saved a new pop up "Select Device for Target" opens, Select the controller (NXP:LPC2148) and click on OK.

Se	elect Device for Target 'Target 1'	×
CPU Vendor: NXP (founded by Philips) Device: Toolset: Data base	Description:	
Generic Luminary Micro Nuvoton OKI Semiconductor OKI Semiconductor OKI Semiconductor Sharp OKI Secie Technology Corp. STMicroelectronics Texas Instruments Toshiba OKI Secie Technology Corp.	<	×
	OK Cancel	Help



4. Select the controller (NXP:LPC2148) and click on OK.

🔣 led - µVision4		- 0	\times
<u>File Edit View Project Flash Debug Peripherals</u>	<u>I</u> ools <u>S</u> VCS <u>W</u> indow <u>H</u> elp		
🗄 🗋 😂 💭 🖉 🕹 🕰 🗠 🗠 🔛 😭 🖓	魯教 律 律 准 版 ❷ 💦 🔹 🔹 🔍 🖉 ● ○ 🖉 🏩 🔳 🔍		
🔆 🖉 🏟 🚔 💥 🛛 Target 1 🔹 🔊			
Project • a × Project • a × Brind Pr @Bo () Fu []_[] Te] Build Output	Select Device for Target Target Time X CPU X Vendor: NXP founded by Philps) Device: LPC2148 Toolet: ARM Data base Description: State base State on the Rah ROM with In-System Programming (SP) and In-Apolic Two URD ArDs with Target Time State on the Rah ROM with In-System Programming (SP) and In-Apolic Two URD ArDs with Target Time Two URD ArDs with Target Time (State Programming (SP) and In-Apolic Two URD ArDs with Target Time Two URD ArDs with Target Time Out Device: LPC2142 Out Device: Two URD ArDs with Target Time Ottar Core with Target Time Two URD ArDs with Target Time Out Device: LPC2142 Out Device: Core with target Time Out Device: Core with target Time Out Device: Core with target Time Out Core with target Time Core with target Time Oth Cancel Help		
	Activate Window:	5 Mindows	~
<			>
Type here to search	 「 「 「 「 「 「 「 「 」 「 」 「 」 「 」 」 「 」 」 「 」 」 「 」 」 「 」 」 」 」 」 」 」 」 」 」 」 」 」	4P NUM SCRL O 17:27 G 19-08-2022	VR R/W

5. As LPC2148 needs the startup code, click on Yes option to include the LPC21xx Startup file.

μVision	
Copy NXP LPC17xx Startup Code to Project Folder and Add File to Project ?	
Yes No	



6. Create a new file to write the program.





7. Type the code or Copy paste the below code snippet.





8. After typing the code save the file as main.c.

2	Save As				×
🔄 🏵 🔹 🕇 📕 «	Tutorials_And_Codes → Lpc1768 → LedBlink	~ ¢	Search LedBlink		P
Organize 🔻 New f	older			•== •	0
🁌 Music	^ Name	Date modified	Туре	Size	
Pictures	LedBlink.plg	3/25/2016 10:26 PM	PLG File	0 KB	
Videos	🔣 LedBlink	3/25/2016 10:23 PM	µVision4 Project	0 KB	
	LedBlink_Target 1.dep	3/25/2016 10:26 PM	DEP File	1 KB	
😽 Homegroup	📓 startup_LPC17xx	10/29/2010 12:20	S File	11 KB	
P Computer Local Disk (C:) D Local Disk (D:) New Volume (E:) javaDevelopers (f V desktop (pc)	. .)				
File name: m	ain.c				~
Save as type: Al	l Files (*.*)				~
) Hide Folders			Save	Cancel	



9. Add the recently saved file to the project.

W		LedBlink - µVision4
File Edit View	Proj	ect Flash Debug Peripherals Tools SVCS Window Help
🗋 💕 🛃 🥥	¥	🗈 🛍 🔊 🕫 ሩ → 🖡 幣 幣 幣 幣 🕸 華 揮 //6 //6 🞯 Systeminit 💿 🗟 🌮 🍭 🌢 ○ 🔗 🎎 🖬
I I I I I I I I I I I I I I I I I I I	2	🙀 Target 1 🔍 🔊 👗 🖶
Project		4 🗵 🗋 main.c* 🗵
🖃 🛅 Target 1		01 #include <lpc17xx.h></lpc17xx.h>
🖻 📇 Source Gr	oup	1 02
	۸N	Options for Group Source Group 1 Alt+F7
		Open File 1, j;
		Open List File ++)
		Open Map File 0000; j++);
		Rebuild all target files
		Build target F7
		Translate File program */
		Stop build
	-	//Clock and PLL configuration
		Add Gloup INSEL4 = 0x000000; //Configure the PORT2 Pins as GPIO; Add Fleste Group 'Source Group 1' ODIR = 0xffffffff; //Configure the PORT2 pins as OUTPUT:
		Pemove Group 'Source Group 1' and its Files
	_	
	Ô	Manage Components
	v	Show Include File Dependencies >FIOSET = 0xffffffff; // Make all the Port pins as high
E Project 🛷 Books	()	<pre>24 25 26 LPC_GPI02->FIOCLR = 0xffffffff; // Make all the Port pins as lov 27 delay_ms(100); 28 } 29 } Func 0,Temp</pre>
Build Output		
<		
Add Files to current P	rojeo	ct Group Simulation
🔚 💟	0	



10. Add the **"main.c"** along with system_LPC21xx.c.

		LedBlink - µVision4		
Edit View Project Flash Debug	Peripherals Tools SVCS Window H	lelp		
) 📂 🖬 🗿 🐰 ங 🛍 🔊 (°	←→ 作作作作作	$/\!/_{\mathbb{R}}^{=}$ 2 SystemInit SystemInit	🔹 O 🔗 🍓 🖬 🔹 🔦	
🗟 🍱 🥔 🥘 📙 🙀 Target 1	🖂 🔊 🔁			
ect 🕂 🗸 🛛	📄 main.c* 💌			
🔁 Target 1	01 #include <lpc17xx.h></lpc17xx.h>			
E Group 1 En Source Group 1	02 03 void delay_ms(unsigned	Add Files t	o Group 'Source Group 1'	
	04 { 05 uinsigned int i,j;	Look in: 🚺 LedBlink 💌	← 🗈 📸 🖩 -	
	06	Name	Date modified Type	Size
	08 for (1=0;1 <ms;1++)< td=""><td>📝 main</td><td>3/25/2016 10:39 PM C File</td><td>1 KB</td></ms;1++)<>	📝 main	3/25/2016 10:39 PM C File	1 KB
	09	system_LPC17xx	3/8/2016 4:10 PM C File	23 KB
	10 }			
	<pre>11 12 /* start the main progr 13 void main() 14 { 15 SystemInit(); 16 LPC_PINCON->PINSEL4 17 LPC_GPI02->FIODIR = 18 19 while(1) 20 { 21 22 LPC_GPI02->FIOSE 23 delay_ms(100); 24 25 26 LPC_GPI02->FIOCL 27 delay_ms(100); 28 } 29 } 29 } </pre>	File name: ["main.o" "system_LPC17x.c" Files of type: C Source file (*.c)		Add Close



11. Build the project and fix the compiler errors/warnings if any.

W	LedBlink - µVision4	
File Edit View Project Flash Debug	Peripherals Tools SVCS Window Help	
🗋 💕 🛃 🗿 🐰 🗈 🛍 💆 (*	← → 陀 陰 陰 陵 準 準 //E //長 🖄 SystemInit 🕢 🗟 🎺 🔍 🍳 ♦ ○ 🔗 🚓 🛛	🖬 🔹 🔌
🕸 🍱 📾 🧼 🚉 🏙 Target 1	🖂 🔊 📥 🖷	
Project Build (F7) 4 🗵	🗋 main.c* 💌	
Build target files	01 #include <lpc17xx.h></lpc17xx.h>	
E Source Group 1	02	
startup_LPC1/xx.s	04 {	
system LPC17xx.c	05 uinsigned int i,j;	
	06 07 for/i=0.i(maritt)	
	08 for (j=0; j<500000; j++);	
	09	
	10 }	
	12 /* start the main program */	
	13 void main()	
	14 { 15 SustemThit(): //Clock and BLL configuration	
	16 LPC PINCON->PINSEL4 = 0x000000; //Configure the PORT2 Pins as GPIO;	
	17 LPC_GPI02->FIODIR = 0xffffffff; //Configure the PORT2 pins as OUTPUT;	
	18 19 while(1)	
	20 {	
	21	
	22 LPC_GPI02->FIOSET = 0xffffffff; // Make all the Port pins as hi delay mg(100);	gh
	24 detay_ns(100),	
	25	
	26 LPC_GPI02->FIOCLR = 0xffffffff; // Make all the Port pins as lo 27 delay mg(100);	N.
	28 }	
	29 }	
E Project Books Func D Temp		
Build Output		
Build target files	Cimulat	tion
	Simulat	ION



12. Code is compiled with no errors. The **.hex** file is still not generated. Follow section 4 (<u>how to enable Hex</u> <u>File Generation</u>)

	LedBlink - µVision4
File Edit View Project Flash Debug	Peripherals Tools SVCS Window Help
🗋 💕 🛃 🗿 🐰 🗈 🛍 💙 🥲	← → 隆 陰 陰 律 律 /// // @ SystemInit 🔽 🗟 🌮 @ ● ○ 🔗 🚓 🔤 •
🕸 🕮 🎬 🧼 🔜 🛛 🐙 🛛 Target 1	V 🔊 📥 🔁
Project 📮 🔀	main.c 🗙
Image: Target 1 Source Group 1 Imain.c Imain.com Imain.com	<pre>03 void delay_ms(unsigned int ms) 04 { 05 unsigned int i,j; 06 for(i=0;i<ms;i++) 08 for(j=0;j<500000;j++); 09 } 10 } 11 /* start the main program */ 13 int main() 14 { 15 SystemInit(); //Clock and PLL configuration 16 LPC_PINCON->PINSEL4 = 0x000000; //Configure the PORT2 Pins as GPIO; 17 LPC_GPIO2->FIODIR = 0xfffffff; //Configure the PORT2 pins as OUTPUT; 18 while(1) 20 { 21 LPC_GPIO2->FIOSET = 0xfffffff; // Make all the Port pins as high 23 delay_ms(100); 24 } 25 LPC_GPIO2->FIOCLR = 0xfffffff; // Make all the Port pins as low 27 delay_ms(100); 28 } 29 } 20 }</ms;i++) </pre>
compiling main.c	
linking	
"LedBlink.axf" - 0 Error(s), 0	Warning(s).
<	
	Simulation
🔚 🛛 🧿 🧃	



5. How to Enable Hex File Generation

1. Click on Target Options (or right click on "Target 1" and select "Options for Target 'Target 1'...) to select the option for generating .hex file.





2. Set IROM1 start address as 0x0000.

W	LedBlink - µVision4
File Edit View Project Flash Debug	Peripherals Tools SVCS Window Help
🗋 💕 🛃 🍠 🐇 🗈 🛍 🖉 🐑	← → 隆 魯 豫 津 津 //ミ //ミ 🖄 SystemInit 🕢 🗟 🌮 Q ● ○ 🔗 🅀 💼 - ≪
🛛 🧼 🍱 🥔 🖳 🛛 🙀 🔤 Target 1	
Project 📮 🗵	i main.c 🗙
Forget Target 1 Source Group 1 Startup_LPC17xx.s System_LPC17xx.c System_LPC17x.c Suld Output Compiling main.c compiling main.c compiling main.c compiling system_LPC17xx.c linking Program Size: Code=1100 RO-dat "LedBlink.axf" - 0 Error(s), 0	Options for Target 'Target 1' X 03 void dele Options for Target 'Target 1' X 06 for (i) Device Target Dudput Listing User C/C++ Asm Linker Debug Utilities Code Generation 08 for NXP (founded by Philps) LPC1788 Code Generation 09 11 Visa (MHz): 12.0 Code Generation 12 /* start Operating system: None Use Unk-Time Code Generation 13 int main 14 System None Big Enden 14 fs System Read/Only Memory Areas Big Enden 17 LPC (i) Read/Only Memory Areas Read/Wite Memory Areas Big Enden 18 while (i) C ROM2: C RAM3: III 20 {int ROM2: C RAM3: III III RAM2: III 23 de III IROM2: OK Cancel Defaults Help a=332 RM-data=4 ZI-data=612 Warning (s) . III III III III III III III IIII IIII IIIIII <
<	
	Simulation
🔁 🖸 🧃	



3. Enable the option to generate the .hex file

W		LedBlink - µVision4
File Edit View Project Flash Debug	Peripherals Tools	VCS Window Help
🗋 💕 🖬 🕼 🕺 🕹 🛍 🔊 (°)	← → / /2 /2	1版 講 譚 //E //版 🖄 SystemInit 🔽 🗟 🌾 🔍 🔍 🔸 🔍 🗞
🔮 🔛 🛍 🧼 🔜 🛛 🗱 Target 1	🗸 💉 🚽	
Project 📮 🔀	📓 main.c 💌	
□ Target 1 □ Source Group 1 □ Istartup_LPC17xx.s □ Imain.c Imain.c Imain.c Imai	03 void delá 04 (05 unsign 06 for (i= 07 for (i= 08 for 09 for 10) 11 2 12 /* start 13 int main 14 (15 Syste 16 LPC_C 18 while(1 20 { 21 LR 22 LR 23 de 24 25 26 LR 27 de 28 > 29	Options for Target 'Target 1' Device Target Output Listing User C/C++ Asm Linker Debug Utilities Select Folder for Objects Name of Executable: LedBlink Image: Create Executable: \LedBlink Image: Create Blink Image: Debug Information Image: Create Blink Image: Debug Information Image: Create Blink Image: Debug Information Image: Create HEX File Image: Debug Information Image: Create Ubrary: \LedBlink.LIB
Build Output		OK Cancel Defaults Help
<pre>compiling main.c compiling system_LPC17xx.c linking Program Size: Code=1100 RO-data "LedBlink.axf" - 0 Error(s), 0 <</pre>	a=332 RW-data=4 2 Warning(s).	I-data=612
ai 🛛 🕅		Simulation



4. Hex file is generated after a rebuild.

	LedBlink - µVision4
File Edit View Project Flash Debug	Peripherals Tools SVCS Window Help
🗋 💕 🛃 🍠 🕺 🗈 🛍 💙 (*	← → 陀 🏗 🐘 🔥 ‡ ‡ //= //= //= SystemInit 💿 💽 🕵 🎤 🍳 ♦ ○ 🔗 🅀 💼 🗸
🧇 🍱 🎬 🧼 🔜 🔓 Target 1	🔜 🔊 🛔 🖶
Project 📮 🗴	main.c 🗴
Target 1 Source Group 1 Sutrup_LPC17xx.s Sutrup_LPC17xx.c Project @Books OFunc 0, Temp Build Output Build target 'Target 1'	<pre>03 void delay_ms(unsigned int ms) 04 { 05 unsigned int i,j; 06 07 for(i=0;i<ms;i++) 08 for(j=0;j<500000;j++); 09 10 } 11 12 /* start the main program */ 13 int main() 14 { 15 SystemInit(); //Clock and PLL configuration 16 LPC_PINCON->PINSEL4 = 0x000000; //Configure the PORT2 Pins as GPI0; 17 LPC_GPI02->FIODIR = 0xfffffff; //Configure the PORT2 pins as OUTPUT; 18 19 while(1) 20 { 21 LPC_GPI02->FIOSET = 0xffffffff; // Make all the Port pins as high 23 delay_ms(100); 24 } 25 LPC_GPI02->FIOCLR = 0xffffffff; // Make all the Port pins as low 27 delay_ms(100); 28 } 29 }</ms;i++) </pre>
linking	
Program Size: Code=1100 RO-dat FromELF: creating hex file	a=332 RW-data=4 ZI-data=612
"LedBlink.axf" - 0 Error(s), 0	Warning(s).
<	



5. Check the project folder for the generated .hex file.



RO-data=332 RW-data=4 ZI-data=612		
ile		
r(s), 0 Warning(s).		
	Simulation	
		Local D



6. How to Upload Hex file Using Flash Magic

Once the **".hex"** file is generated in keil, we need to upload the hex file to the hardware. We use Flash Magic Tool for the same.

Open the flash magic software and follow the below steps.

- 1. Select the IC "LPC2148" from Select Menu.
- 2. Select the COM Port. Check the device manager for detected Com port.
- 3. Select Baud rate from 9600-115200
- 4. Select None [ISP] Option for Interface.
- 5. Oscillator Frequency 12.000000(12Mhz).
- 6. Check the Erase blocks used by Hex file option
- 7. Browse and Select the hex file.
- 8. Check the Verify After Programming Option.
- **9.** If DTR and RTS are used then go to Options->Advanced Options-> Hardware Config and select the Use DTR and RTS Option.
- **10.** Hit the Start Button to flash the hex file.
- **11.**Once the hex file is flashed, reset the board. Now the controller should run your application code.

Se Flash Magic - NON PRODUCT	FION USE ONLY 🚽 🗆 👘	
File ISP Options Tools Help		
	1 3 2	If DTR and RTS are used then the below options needs to be
Step 1 - Communications	Step 2 - Erase	selected in Options->Advanced Options->Hardware Config
Select LPC1768 Select the IC	Erase block 0 (0x000000-0x000FFF) Erase block 1 (0x001000-0x001FFF)	Advanced Options
	Erase block 2 (0x002000 0x002FFF) Erase block 3 (0x003000 0x003FFF)	Communications Hardware Config Security Just In Time Code Timeouts Misc
COM PHIC COM 3	Erase block 4 (0x004000-0x004FFF) Erase block 5 (0x005000-0x005FFF)	
Baud Rate: 9600 V	Erase all Flash+Code Rd Prot	See DTR and RTS to control RS1 and ISP pn
Interface: None (ISP)	Ense blocks used by Hex File	The Philasseried write Com Politopen
Oscillato (MHz): 12.000000		11, 00 IIIs 16, 100 IIIs
		Assert DTR and RTS while COM Port open
Step 3 - Hex File		Disable DTR and RTS completely
Hex File: D:\Led_Blinking.hex	Browse	
Modified: Saturday, March 5, 2016, 9:35	22 AM	
Stee 4 - Options	Step 5 - Start	Cancel OK
Verify after programming Fill my def Flash	Start	
Execute Activate Flash Bank	it the Start Button to flash the	e nex file.
CAN Bus Tirring Calculators at		
unum an an adams nom (an Abrans (nale: dates blad		
www.exacademy.com/en/library/calculators.html	0	
1		



7. How to Upload Hex file Using Philips Flash Utility

Once the ".hex" file is generated in keil, we need to upload the hex file to the hardware. We use Flash Magic Tool for the same.

Open the flash magic software and follow the below steps.

- 1. Press the Read Device ID option in the Software. Automatically detects the IC.
- 2. Select the COM Port. Check the device manager for detected Com port.
- 3. Select Baud rate from 9600-115200
- 4. Click Use DTR/RTS for Reset and Boot Loader Selection.
- 5. Check the Erase blocks used by Hex file option
- 6. Click Execute Code after Upload.
- **7.** Browse and Select the hex file.
- 8. Click Upload to Flash.
- **9.** Once the hex file is flashed, reset the board. Now the controller should run your application code.

😻 LPC2000 Flash Utility	- 🗆 X				
LPC2000 Flash Utility V2.2.3					
Flash Programming Filename: Filename: Filename: Filename: Filename: Filename: Filename: F	Communication Connected To Port				
Upload to Flash Image: Execute Code after Upload Compare Flash Manual Reset	Use Baud Rate: 9600				
Device LPC2148 Read Part ID: Use DTR/RTS XTAL Freq. [kHz]: 14745 Device ID Boot Loader ID: Use DTR/RTS					



8. Lab Programs

EXPERIMENT NO 1

Blinking an LED

Aim: Interfacing LED's with ARM LPC2148.

Description:

Turning ON and OFF an LED's after Particular delay.

Hardware Requirement:

ARM LPC2148 Trainer Kit, FRC cables, USB A to B cable and 12V 2A Adapter.





Program:

```
#include<LPC21xx.h>
void delay1(int); //define delay function
 int main()
 {
                          //select pins for blinking led
 PINSEL0=0x0000000;
                          //select I/O pins as output
 IODIR0=0x00ffffff;
 while(1)
{
                          //sets pins high
  IOSET0=0x00ffffff;
  delay1(1000);
                                         //gives 1 SEC delay
  IOCLR0=0x00ffffff;
                          //clears pins
                                         //gives 1 SEC delay
  delay1(1000);
 }
 }
/* Delay routine; gives an approximate delay in milliseconds */
void delay1(int d)
                        // delay routine
{
    int i;
   while(d)
   {
    for(i=0;i<7000;i++){;}
     d--;
   }
}
```



EXPERIMENT NO 2

Liquid Crystal Display

Aim: Interfacing LCD Display with ARM LPC2148

Description:

To display the message on the LCD screen.

Hardware required:

ARM LPC2148 Trainer Kit, FRC cables, USB A to B cable and 12V 2A Power Adapter.



Program:

```
#include <lpc214x.h>
#include <stdint.h>
#include <stdlib.h>
#include <stdio.h>
void delay ms(uint16 t j) /* Function for delay in milliseconds */
{
  uint16 t x,i;
   for(i=0;i<j;i++)
   {
  for(x=0; x<6000; x++); /* loop to generate 1 millisecond delay with Cclk = 60MHz */
   ł
}
void LCD CMD(char command)
{
   IOOPIN = ( (IOOPIN & 0xFF0FFFFF) | ((command & 0xF0)<<16) ); /* Upper nibble of command */
   IO0SET = 0x00020000; /* EN = 1 */
   IOOCLR = 0x00010000; /* RS = 0, RW = 0 */
   delay ms(5);
   IOOCLR = 0x00020000; /* EN = 0, RS and RW unchanged(i.e. RS = RW = 0) */
   delay ms(5);
   IOOPIN = ((IOOPIN & 0xFF0FFFFF)) ((command & 0x0F)<<20)); /* Lower nibble of command */
   IO0SET = 0x00020000; /* EN = 1 */
   IOOCLR = 0x00010000; /* RS = 0, RW = 0 */
   delay ms(5);
   IOOCLR = 0x00020000; /* EN = 0, RS and RW unchanged(i.e. RS = RW = 0) */
   delay ms(5);
}
void LCD INIT(void)
{
   IOODIR = 0x00FFFF0; /* P0.20 to P0.23 LCD Data. P0.16,,17 as RS RW and EN */
   delay ms(20);
   LCD CMD(0x02); /* Initialize lcd in 4-bit mode */
   LCD CMD(0x28); /* 2 lines */
   LCD_CMD(0x0C); /* Display on cursor off */
   LCD CMD(0x06); /* Auto increment cursor */
   LCD_CMD(0x01); /* Display clear */
   LCD CMD(0x80); /* First line first position */
}
```



```
void LCD STRING (char* msg)
{
   uint8_t i=0;
   while(msg[i]!=0)
   {
          IOOPIN = ((IOOPIN \& 0xFF0FFFF) | ((msg[i] \& 0xF0) << 16));
          IOOSET = 0x00030000; /* RS = 1, EN = 1 */
          IOOCLR = 0x0000020; /* RW = 0 */
          delay ms(2);
          IOOCLR = 0x00020000; /* EN = 0, RS and RW unchanged(i.e. RS = 1, RW = 0) */
          delay_ms(5);
          IOOPIN = ((IOOPIN \& 0xFF0FFFF) | ((msg[i] \& 0x0F) << 20));
          IOOSET = 0x00030000; /* RS = 1, EN = 1 */
          IOOCLR = 0x0000020; /* RW = 0 */
          delay ms(2);
          IOOCLR = 0x00020000; /* EN = 0, RS and RW unchanged(i.e. RS = 1, RW = 0) */
          delay_ms(5);
          i++;
   }
}
void LCD CHAR (char msg)
{
          IOOPIN = ((IOOPIN \& 0xFF0FFFF)) | ((msg \& 0xF0) << 16));
          IOOSET = 0x00030000; /* RS = 1, EN = 1 */
          IOOCLR = 0x0000020; /* RW = 0 */
          delay ms(2);
          IOOCLR = 0x00020000; /* EN = 0, RS and RW unchanged(i.e. RS = 1, RW = 0) */
          delay ms(5);
          IOOPIN = ((IOOPIN \& 0xFF0FFFF)) | ((msg \& 0x0F) << 20));
          IOOSET = 0x00030000; /* RS = 1, EN = 1 */
          IOOCLR = 0x0000020; /* RW = 0 */
          delay ms(2);
          IOOCLR = 0x00020000; /* EN = 0, RS and RW unchanged(i.e. RS = 1, RW = 0) */
          delay ms(5);
}
int main(void)
{
   LCD INIT();
   LCD_STRING("--- RDL---");
   LCD CMD(0xC0);
   LCD_STRING(" LCD DISPLAY");
   return 0;
}
```



EXPERIMENT NO 3

ADC

Aim:

Interfacing ADC with ARM LPC2148.

Description:

To learn how to read ADC Values and display the values in the LCD.

Hardware required:

ARM LPC2148 Trainer Kit, FRC cables, USB A to B cable and 12V 2A Power Adapter.





Program

```
#include <lpc214x.h>
#include <stdint.h>
#include <stdlib.h>
#include <stdio.h>
#include <stdint.h>
#include <stdio.h>
#include <string.h>
void delay_ms(uint16_t j) /* Function for delay in milliseconds */
{
  uint16_t x,i;
   for(i=0;i<j;i++)
   {
  for(x=0; x<6000; x++); /* loop to generate 1 millisecond delay with Cclk = 60MHz */
   }
}
void LCD CMD(char command)
{
   IOOPIN = ( (IOOPIN & 0xFF0FFFFF) | ((command & 0xF0)<<16) ); /* Upper nibble of command */
   IOOSET = 0x00020000; /* EN = 1 */
   IOOCLR = 0x00010000; /* RS = 0, RW = 0 */
   delay ms(5);
   IOOCLR = 0x00020000; /* EN = 0, RS and RW unchanged(i.e. RS = RW = 0) */
   delay ms(5);
   IOOPIN = ( (IOOPIN & 0xFF0FFFFF) | ((command & 0x0F)<<20) ); /* Lower nibble of command */
   IO0SET = 0x00020000; /* EN = 1 */
   IOOCLR = 0x00010000; /* RS = 0, RW = 0 */
   delay ms(5);
   IOOCLR = 0x00020000; /* EN = 0, RS and RW unchanged(i.e. RS = RW = 0) */
   delay ms(5);
}
void LCD INIT(void)
{
   IOODIR = 0x00FFFF0; /* P0.12 to P0.15 LCD Data. P0.4,5,6 as RS RW and EN */
   delay ms(20);
   LCD_CMD(0x02); /* Initialize lcd in 4-bit mode */
   LCD CMD(0x28); /* 2 lines */
   LCD_CMD(0x0C); /* Display on cursor off */
   LCD CMD(0x06); /* Auto increment cursor */
```

```
LCD_CMD(0x01); /* Display clear */
   LCD CMD(0x80); /* First line first position */
}
void LCD STRING (char* msg)
{
   uint8 t i=0;
   while(msg[i]!=0)
   {
          IOOPIN = ( (IOOPIN & 0xFF0FFFFF) | ((msg[i] & 0xF0)<<16) );
          IOOSET = 0x00030000; /* RS = 1, EN = 1 */
          IOOCLR = 0x0000020; /* RW = 0 */
          delay ms(2);
          IOOCLR = 0x00020000; /* EN = 0, RS and RW unchanged(i.e. RS = 1, RW = 0) */
          delay ms(5);
          IOOPIN = ((IOOPIN \& 0xFF0FFFF)) | ((msg[i] \& 0x0F) << 20));
          IOOSET = 0x00030000; /* RS = 1, EN = 1 */
          IOOCLR = 0x0000020; /* RW = 0 */
          delay ms(2);
          IOOCLR = 0x00020000; /* EN = 0, RS and RW unchanged(i.e. RS = 1, RW = 0) */
          delay_ms(5);
          i++;
   }
}
void LCD CHAR (char msg)
{
          IOOPIN = ((IOOPIN \& 0xFF0FFFF)) | ((msg \& 0xF0) << 16));
          IOOSET = 0x00030000; /* RS = 1, EN = 1 */
          IOOCLR = 0x0000020; /* RW = 0 */
          delay ms(2);
          IOOCLR = 0x00020000; /* EN = 0, RS and RW unchanged(i.e. RS = 1, RW = 0) */
          delay ms(5);
          IOOPIN = ((IOOPIN \& 0xFF0FFFF)) | ((msg \& 0x0F) << 20));
          IOOSET = 0x00030000; /* RS = 1, EN = 1 */
          IOOCLR = 0x0000020; /* RW = 0 */
          delay ms(2);
          IOOCLR = 0x00020000; /* EN = 0, RS and RW unchanged(i.e. RS = 1, RW = 0) */
          delay ms(5);
}
int main(void)
{
   LCD INIT();
   LCD STRING("--- RDL---");
```



```
LCD CMD(0xC0);
   LCD STRING(" LCD DISPLAY");
   return 0;
}*/
 int main(void)
{
uint32 t result;
// float voltage;
   char volt[18];
// LCD CMD(0xC0);
// LCD_STRING(" LCD DISPLAY");
   LCD_INIT();
          PINSEL1 = 0x01000000; /* P0.28 as AD0.1 */
   AD0CR = 0x00200402; /* ADC operational, 10-bits, 11 clocks for conversion */
   LCD STRING("--- RDL1---");
   while(1)
   {
          AD0CR = AD0CR | (1<<24); /* Start Conversion */
          while (!(AD0DR1 & 0x8000000)); /* Wait till DONE */
          result = AD0DR1;
          result = (result>>6);
          result = (result & 0x00003FF);
   //
          voltage = ( (result/1023.0) * 3.3 ); /* Convert ADC value to equivalent voltage */
          LCD CMD(0xc0);
   //
          sprintf(volt, "Voltage=%.2f V ", voltage);
          sprintf(volt, "ADC=%i ", result);
          LCD_STRING(volt);
          memset(volt, 0, 18);
   }
   }
```

EXPERIMENT NO 4

UART

Aim:

Interfacing UART with ARM LPC2148

Description:

Transmit/Receive Data using UART and display the data's on the terminal Software.

Hardware required:

ARM LPC2148 Trainer Kit, USB A to B Cable and 12V 2A Power Adapter.





Program

```
#include <lpc214x.h>
#include <stdint.h>
#include "UART.h"
int main(void)
{
    char receive;
    UART0_init();
    while(1)
    {
        receive = UART0_RxChar();
        UART0_SendString("Received:");
        UART0_TxChar(receive);
        UART0_SendString("\r\n");
    }
}
```



EXPERIMENT NO 5

RTC (Real Time Clock)

Aim:

Interfacing Real Time Clock with ARM LPC2148

Description:

To transmit Date and Time using UART and display the data's on the terminal Software.

Hardware required:

ARM LPC2148 Trainer Kit, FRC Cables, USB A to B Cable and 12V 2A Power Adapter.



Make the I2C Pins ON



Program

```
#include <lpc214x.h>
#include <stdint.h>
#include <stdio.h>
#include "UARTO.h"
uint8_t alarm, flag;
 _irq void RTC_ISR(void)
{
   if (ILR & 0x01)
   {
          flag = 1;
          ILR = ILR | 0x01;
   }
   if (ILR & 0x02)
   {
          alarm = 1;
          ILR = ILR | 0x02;
   }
   VICVectAddr = 0;
}
typedef struct
{
   uint8_t seconds;
   uint8_t minutes;
   uint8_t hours;
   uint8_t day_of_month;
   uint8_t day_of_week;
   uint16_t day_of_year;
   uint8_t month;
   uint16_t year;
}
RTC_Time;
void RTC_Set_Time( RTC_Time Set_Time)
{
   SEC = Set_Time.seconds;
   MIN = Set_Time.minutes;
   HOUR = Set Time.hours;
   DOM = Set_Time.day_of_month;
   DOW = Set_Time.day_of_week;
   DOY = Set Time.day of year;
   MONTH = Set_Time.month;
   YEAR = Set_Time.year;
```

```
}
```



```
void RTC Set Alarm Time(RTC Time Alarm Time)
{
   ALSEC = Alarm Time.seconds;
   ALMIN = Alarm Time.minutes;
   ALHOUR = Alarm_Time.hours;
   ALDOM = Alarm Time.day of month;
   ALDOW = Alarm_Time.day_of_week;
  ALDOY = Alarm Time.day of year;
  ALMON = Alarm_Time.month;
  ALYEAR = Alarm Time.year;
}
RTC Time RTC Get Time(void)
{
   RTC Time time;
  time.seconds = SEC;
  time.minutes = MIN;
  time.hours = HOUR;
  time.day_of_month = DOM;
  time.day_of_week = DOW;
  time.day of year = DOY;
  time.month = MONTH;
  time.year = YEAR;
   return time;
}
int main(void)
{
  /* Setting Time + Alarm */
   RTC Time set time, alarm time, current time;
   char timestr[30], datestr[30];
   alarm = 0;
  flag = 0;
   IO0DIR = 0x00000010; /* P0.4 as output pin for LED */
   UARTO init();
   PCONP = (PCONP | (1<<9)); /* PCRTC = 1 */
   /* The RTC registers cannot be written to unless we make PCRTC = 1 */
  ILR = 0x0; /* No RTC interrupts */
   CCR = 0x12; /* 32.768kHz clock and Reset Clock Tick Counter */
   CCR = 0x10;
   CIIR = 0x00; /* No interrupts */
   AMR = 0x00; /* Alarm registers not masked */
   VICVectAddr0 = (unsigned) RTC ISR;
   VICVectCntl0 = 0x000002D;
```



VICIntEnable = 0x00002000; VICIntSelect = 0x0000000; set_time.seconds = 00; set time.minutes = 25; set time.hours = 11; set_time.day_of_month = 6; set time.day of week = 5; set_time.day_of_year = 279; set time.month = 10; set_time.year = 2017; RTC Set Time(set time); CIIR = 0x01; /* Seconds value increment interrupt */ alarm time.seconds = 15; alarm time.minutes = 25; alarm time.hours = 11; alarm time.day of month = 6; alarm time.day of week = 5; alarm_time.day_of_year = 279; alarm time.month = 10; alarm time.year = 2017; RTC_Set_Alarm_Time(alarm_time); CCR = 0x11; /* 32.768kHz clock and clock Enable */ ILR = 0x03; /* RTC interrupts enabled */ IOOCLR = 0x0000010;/* Set the Time and Alarm once using above code lines */

/* Once the time and alarm is set, comment out the above code lines and uncomment the code lines for "Only RTC Read" and program the device */

/* If this is not done, the time will be set repeatedly to same value whenever the device is reset or powered */

/* Only RTC Read */

- // RTC_Time current_time;
- // char timestr[30], datestr[30];
- // alarm = 0;
- // flag = 0;
- // IO0DIR = 0x00000010; /* P0.4 as output pin for LED */
- // UART0_init();
- // AMR = 0x00; /* Alarm registers not masked */
- // CCR = 0x10;
- // VICVectAddr0 = (unsigned) RTC_ISR;
- // VICVectCntl0 = 0x000002D;
- // VICIntEnable = 0x00002000;
- // VICIntSelect = 0x00000000;
- // CCR = 0x11; /* 32.768kHz clock and clock enable */

// ILR = 0x03; /* RTC interrupts enabled */ // IOOCLR = 0x0000010; /* Code lines below are common for "Setting time + Alarm" as well as "Only RTC Read" */ while(1) { if(alarm == 1){ current_time = RTC_Get_Time(); sprintf(timestr,"Alarm!!!: %d:%d:%d \r\n",current time.hours,current time.minutes,current time.seconds); UARTO_SendString(timestr); uint8 ti; for(i=0;i<10;i++) { IOOSET = 0x0000010;delay ms(300); IOOCLR = 0x00000010; delay ms(300); } alarm = 0; } if (flag == 1) { current_time = RTC_Get_Time(); sprintf(timestr,"Time: %d:%d:%d ",current_time.hours,current_time.minutes,current_time.seconds); sprintf(datestr,"Date: %d/%d/%d \r\n",current_time.day_of_month,current_time.month,current_time.year); UARTO_SendString(timestr); UARTO_SendString(datestr); flag = 0;} } }



EXPERIMENT NO 6

Hex Keypad

Aim:

To interface 4x4 Hex keypad with ARM LPC2148.

Description:

To display the pressed key on the LCD Display.

Hardware Required:

ARM LPC2148 Trainer Kit, FRC Cables, USB A to B Cable and 12V 2A Power Adapter.



Program

#include <lpc214x.h>
#include <stdint.h>
#include <stdlib.h>
#include <stdlib.h>
#include <stdlib.h>
unsigned int adc_value1,adc_value2,C;
unsigned char buf[16] = {0};
char check_key(void);
unsigned int KeyPort=0x00f00000;

#define sw1 0x00010000 #define sw2 0x00020000 #define sw3 0x00040000 #define sw4 0x00080000

```
void delay_ms(uint16_t j) /* Function for delay in milliseconds */
```

```
uint16 t x,i;
```

{

}

}

```
for(i=0;i<j;i++)
```

```
{
for(x=0; x<6000; x++); /* loop to generate 1 millisecond delay with Cclk = 60MHz */</pre>
```

```
void LCD CMD(char command)
```

```
{
    IOOPIN = ( (IOOPIN & 0xFF0FFFF) | ((command & 0xF0)<<16) ); /* Upper nibble of command */
    IOOSET = 0x00020000; /* EN = 1 */
    IOOCLR = 0x00010000; /* EN = 0, RW = 0 */
    delay_ms(5);
    IOOCLR = 0x00020000; /* EN = 0, RS and RW unchanged(i.e. RS = RW = 0) */
    delay_ms(5);
    IOOPIN = ( (IOOPIN & 0xFF0FFFFF) | ((command & 0x0F)<<20) ); /* Lower nibble of command */
    IOOSET = 0x00020000; /* EN = 1 */
    IOOCLR = 0x00010000; /* EN = 1 */
    IOOCLR = 0x00010000; /* EN = 0, RW = 0 */
    delay_ms(5);
    IOOCLR = 0x00020000; /* EN = 0, RS and RW unchanged(i.e. RS = RW = 0) */
    delay_ms(5);
    IOOCLR = 0x00020000; /* EN = 0, RS and RW unchanged(i.e. RS = RW = 0) */
    delay_ms(5);
</pre>
```

```
}
```



```
void LCD INIT(void)
{
   IOODIR = 0x00FFFF0; /* P0.20 to P0.23 LCD Data. P0.16,,17 as RS RW and EN */
   delay ms(20);
   LCD_CMD(0x02); /* Initialize lcd in 4-bit mode */
   LCD CMD(0x28); /* 2 lines */
   LCD_CMD(0x0C); /* Display on cursor off */
   LCD CMD(0x06); /* Auto increment cursor */
   LCD_CMD(0x01); /* Display clear */
   LCD CMD(0x80); /* First line first position */
}
void LCD STRING (char* msg)
{
   uint8 t i=0;
   while(msg[i]!=0)
   {
          IOOPIN = ((IOOPIN \& 0xFF0FFFF) | ((msg[i] \& 0xF0) << 16));
          IOOSET = 0x00030000; /* RS = 1, EN = 1 */
          IOOCLR = 0x0000020; /* RW = 0 */
          delay ms(2);
          IOOCLR = 0x00020000; /* EN = 0, RS and RW unchanged(i.e. RS = 1, RW = 0) */
          delay ms(5);
          IOOPIN = ((IOOPIN \& 0xFF0FFFF)) | ((msg[i] \& 0x0F) << 20));
          IOOSET = 0x00030000; /* RS = 1, EN = 1 */
          IOOCLR = 0x0000020; /* RW = 0 */
          delay_ms(2);
          IOOCLR = 0x00020000; /* EN = 0, RS and RW unchanged(i.e. RS = 1, RW = 0) */
          delay ms(5);
          i++;
   }
}
void LCD CHAR (char msg)
{
          IOOPIN = ((IOOPIN \& 0xFF0FFFF)) | ((msg \& 0xF0) << 16));
          IOOSET = 0x00030000; /* RS = 1, EN = 1 */
          IOOCLR = 0x0000020; /* RW = 0 */
          delay ms(2);
          IOOCLR = 0x00020000; /* EN = 0, RS and RW unchanged(i.e. RS = 1, RW = 0) */
          delay ms(5);
          IOOPIN = ((IOOPIN \& 0xFF0FFFF)) | ((msg \& 0x0F) << 20));
          IOOSET = 0x00030000; /* RS = 1, EN = 1 */
          IOOCLR = 0x0000020; /* RW = 0 */
```



```
delay_ms(2);
          IOOCLR = 0x00020000; /* EN = 0, RS and RW unchanged(i.e. RS = 1, RW = 0) */
          delay_ms(5);
}
int main(void)
{
  IODIR1=KeyPort;
  LCD INIT();
  LCD_STRING("--- RDL---");
  LCD_CMD(0xC0);
  while(1)
{
delay_ms(5);
LCD_CHAR(check_key());
}
}
  // IODIR0=0X0007ffCF0;
char check_key(void)
   {
  while(1)
  {
    //
  IOSET1=0x00EF0000;
 delay ms(5);
   if((IOPIN1&sw1)==0)
  {
  C='1';
  while((IOPIN1&sw1)==0);
  return(C);
   }
  if((IOPIN1&sw2)==0)
   {
   C='5';
  while((IOPIN1&sw2)==0);
   return(C);
   }
```



```
if((IOPIN1&sw3)==0)
{
C='9';
while((IOPIN1&sw3)==0);
return(C);
}
if((IOPIN1&sw4)==0)
{
C='C';
while((IOPIN1&sw4)==0);
return(C);
}
IOCLR1=0x00EF0000;
delay_ms(5);
IOSET1=0x00DF0000;
delay_ms(5);
if((IOPIN1&sw1)==0)
{
C='2';
while((IOPIN1&sw1)==0);
return(C);
}
if((IOPIN1&sw2)==0)
{
C='6';
while((IOPIN1&sw2)==0);
return(C);
}
if((IOPIN1&sw3)==0)
{
C='0';
while((IOPIN1&sw3)==0);
return(C);
}
if((IOPIN1&sw4)==0)
{
C='D';
while((IOPIN1&sw4)==0);
return(C);
```



}

```
IOCLR1=0x00DF0000;
delay_ms(5);
   IOSET1=0x00BF0000;
delay_ms(5);
   if((IOPIN1&sw1)==0)
  {
  C='3';
  while((IOPIN1&sw1)==0);
   return(C);
   }
  if((IOPIN1&sw2)==0)
  {
  C='7';
  while((IOPIN1&sw2)==0);
   return(C);
  }
  if((IOPIN1&sw3)==0)
  {
   C='A';
   while((IOPIN1&sw3)==0);
   return(C);
   }
   if((IOPIN1&sw4)==0)
   {
   C='E';
   while((IOPIN1&sw4)==0);
   return(C);
   }
  IOCLR1=0x00BF0000;
   delay ms(5);
   IOSET1=0x007F0000;
   delay ms(5);
   if((IOPIN1&sw1)==0)
   {
   C='4';
   while((IOPIN1&sw1)==0);
   return(C);
   }
   if((IOPIN1&sw2)==0)
   {
   C='8';
   while((IOPIN1&sw2)==0);
```

```
return(C);
  }
  if((IOPIN1&sw3)==0)
  {
  C='B';
  while((IOPIN1&sw3)==0);
  return(C);
   }
  if((IOPIN1&sw4)==0)
  {
  C='F';
  while((IOPIN1&sw4)==0);
  return(C);
  }
  IOCLR1=0x007F0000;
delay_ms(5);
```

```
}
```



EXPERIMENT NO 7

Stepper Motor

Aim:

To interface Stepper Motor with ARM LPC2148.

Description:

To rotate Stepper Motor using ARM LPC2148.

Hardware Required:

ARM LPC2148 Trainer Kit, Stepper Motor, FRC Cables, USB A to B Cable and 12V 2A Power Adapter.





Program

```
#include<LPC21xx.h>
void _delay_ms(int j)
{
int x,l;
   for(l=0;l<j;l++)
   {
  for(x=0; x<6000; x++); /* loop to generate 1 milisecond delay with Cclk = 60MHz */
   }
}
int main()
{
int i=0;
PINSEL0=0x0000000;
                        //select pins for blinking led
                        //select I/O pins as output
IODIR0=0x00ff0000;
while(1)
{
 for(i=0;i<100;i++)
          {
                 IOSETO = 0x00EC0000;
                 _delay_ms(4);
                 IOCLR0 = 0x00EC0000;
                 IOSETO = 0x00DC0000;
                 _delay_ms(4);
```



```
IOCLR0=0x00DC0000;
```

IOSET0 = 0x00BC0000;

```
_delay_ms(4);
```

IOCLR0=0x00BC0000;

IOSET0 = 0x007C0000;

_delay_ms(4);

IOCLR0=0x007C0000;

}

```
for(i=0;i<100;i++)
```

{

IOSET0 = 0x007C0000;

```
_delay_ms(4);
```

IOCLR0=0x007C0000;

IOSET0 = 0x00BC0000;

_delay_ms(4);

IOCLR0=0x00BC0000;

```
IOSET0 = 0x00DC0000;
```

_delay_ms(4);

```
IOCLR0=0x00DC0000;
```

```
IOSET0 = 0x00EC0000;
```

```
_delay_ms(4);
```

IOCLR0=0x00EC0000;

}

} }

/* Delay routine; gives an approximate delay in milliseconds */

EXPERIMENT NO 8

PWM

Aim:

To interface PWM with ARM LPC2148.

Description:

To control the Brightness of the LED through PWM using ARM LPC2148.

Hardware Required:

ARM LPC2148 Trainer Kit, FRC Cables, USB A to B Cable and 12V 2A Power Adapter.





Program

```
#include <lpc214x.h>
#define PLOCK 0x00000400
#define PWMPRESCALE 60 //60 PCLK cycles to increment TC by 1 i.e 1 Micro-second
void initPWM(void);
void initClocks(void);
void setupPLL0(void);
void feedSeq(void);
void connectPLL0(void);
void delay ms(int j) /* Function for delay in milliseconds */
{
  int x,i;
   for(i=0;i<j;i++)
   {
  for(x=0; x<6000; x++); /* loop to generate 1 millisecond delay with Cclk = 60MHz */
   }
}
int DUTY=0;
int main(void)
{
   initClocks(); //Initialize CPU and Peripheral Clocks @ 60Mhz
   initPWM(); //Initialize PWM
 //IO0DIR = 0x1; This is not needed!
   //Also by default all pins are configured as Inputs after MCU Reset.
   IOODIR = 0x01;
  while(1)
  {
                  for(DUTY=0;DUTY<10000;DUTY++)</pre>
                  {
                  PWMMR1 = DUTY; //25% Bright
                  PWMLER = (1 << 1);
                  delay_ms(1);
                  }
          /*
                  if( ((IO0PIN) & (1<<2)) ) // Check P0.2
                  {
                         PWMMR1 = 2500; //25% Bright
                         PWMLER = (1<<1);
                  }
```



}

{

```
else if( ((IOOPIN) & (1<<3)) ) // Check P0.3
                 {
                        PWMMR1 = 5000; //50% Bright
                        PWMLER = (1 << 1);
                 }
                 else if( ((IOOPIN) & (1<<4)) ) // Check P0.4
                 {
                        PWMMR1 = 7500; //75% Bright
                        PWMLER = (1<<1);
                 }
                 else if( ((IOOPIN) & (1<<5)) ) // Check P0.5
                 {
                        PWMMR1 = 10000; //T-ON=100% , Hence 25% Bright
                        PWMLER = (1<<1); //Update Latch Enable bit for PWMMR1
                 } */
  }
  //return 0; //normally this wont execute ever
void initPWM(void)
  /*Assuming that PLLO has been setup with CCLK = 60Mhz and PCLK also = 60Mhz.*/
  /*This is a per the Setup & Init Sequence given in the tutorial*/
   PINSEL0 = 0x0000002;
                                      //SELECT PIN Select 0 FOR PWM1(P0.0)
   //PINSEL0 = (1<<1); // Select PWM1 output for Pin0.0</pre>
   PWMPCR = 0x0; //Select Single Edge PWM - by default its single Edged so this line can be removed
   PWMPR = PWMPRESCALE-1; // 1 micro-second resolution
   PWMMR0 = 10000; // 10ms period duration
   PWMMR1 = 2500; // 2.5ms - pulse duration i.e width (Brigtness level)
   PWMMCR = (1<<1); // Reset PWMTC on PWMMR0 match
   PWMLER = (1<<1) | (1<<0); // update MR0 and MR1
   PWMPCR = (1<<9); // enable PWM output
   PWMTCR = (1<<1); //Reset PWM TC & PR
   //Now , the final moment - enable everything
   PWMTCR = (1<<0) | (1<<3); // enable counters and PWM Mode
   //PWM Generation goes active now
  //Now you can get the PWM output at Pin P0.0!
void initClocks(void)
  setupPLLO();
  feedSeq(); //sequence for locking PLL to desired freq.
```

}



```
connectPLL0();
  feedSeg(); //sequence for connecting the PLL as system clock
  //SysClock is now ticking @ 60Mhz!
  VPBDIV = 0x01; // PCLK is same as CCLK i.e 60Mhz
  //PLL0 Now configured!
}
//-----PLL Related Functions :-----
void setupPLLO(void)
{
  //Note : Assuming 12Mhz Xtal is connected to LPC2148.
  PLLOCON = 0x01; // PPLE=1 & PPLC=0 so it will be enabled
           // but not connected after FEED sequence
  PLLOCFG = 0x24; // set the multipler to 5 (i.e actually 4)
          // i.e 12x5 = 60 Mhz (M - 1 = 4)!!!
           // Set P=2 since we want FCCO in range!!!
           // So , Assign PSEL =01 in PLLOCFG as per the table.
}
void feedSeq(void)
{
  PLLOFEED = 0xAA;
  PLLOFEED = 0x55;
}
void connectPLL0(void)
{
  // check whether PLL has locked on to the desired freq by reading the lock bit
  // in the PPLOSTAT register
  while( !( PLLOSTAT & PLOCK ));
  // now enable(again) and connect
  PLLOCON = 0x03;
}
```

EXPERIMENT NO 9

EEPROM

Aim:

To interface EEPROM with ARM LPC2148.

Description:

Transmit EEPROM Data using UART and display the data's on the terminal Software.

Hardware Required:

ARM LPC2148 Trainer Kit, FRC Cables, USB A to B Cable and 12V 2A Power Adapter.



Make the I2C pins ON



Program

```
#include<LPC214x.h>
void I2C_init(void);
void byte write(unsigned char address, unsigned char location, unsigned char data);
void send start(void);
void send_stop(void);
unsigned char byte read(unsigned char address, unsigned char location);
void msdelay(unsigned int time);
void uart0Init(void)
{
  // port 0 tx P0.1 and rx P0.0 selected
  UOLCR=0x83; //8bit data, no parity, 1 stop bit
  U0DLL=97;// 9600 baud rate @15Mhz Pclk
  U0LCR=0x03;// DLAB=0
}
void uart0Putch(unsigned char ch)
{
  UOTHR=ch; // Transmitter holding register
  while(!(UOLSR & 0x20));// wait still THR=0
}
void UART0 Txstring(unsigned char *Str)
{
int i=0;
while(Str[i]!='\0')
{
 uart0Putch(Str[i]);
 i++;
}
}
int main()
{
  int i;
  unsigned char read_data;
  unsigned char
write data[10]="KANWAL";//{0x41,0x42,0x43,0x44,0x45,0x46,0x47,0x48,0x49,0x4A};
  PINSEL0=0x00000055;
  PINSEL1=0x0000000;
  PINSEL2=0x0000000;
uart0Init();
  I2C init();
```

```
// LCD_writestring("Writing from I2C");
```

```
UARTO_Txstring("Writing from I2C");
  for(i=0;i<10;i++)
  {
    byte write(0xA0,i,write data[i]);
 uart0Putch(write data[i]);
    msdelay(100);
    }
  // LCD cmd(0xC0);
 uart0Putch(0x0d);
  UARTO_Txstring("reading from I2C : ");
    for(i=0;i<6;i++)
    {
    read data=byte read(0xA0,i);
    uartOPutch(read data);
    // LCD_data(read_data);
  }
}
void I2C_init(void)
{
  I2COCONCLR=0xFF;
                               //enable I2C
  I2C0CONSET=0x40;
  I2C0SCLH=75; //0x4B
  I2C0SCLL=75; //0x4B
}
void byte write(unsigned char address, unsigned char location, unsigned char data)
{
  I2COCONCLR=0xFF;
  I2C0CONSET=0x40;
  send_start();
                         //send start condition
  while(!(I2C0CONSET&0x08));
                                  //check SI flag
                              //clear SI flag and start
  I2COCONCLR=0x28;
  I2C0DAT=address&0xFE;
                                //selecting address in
  while(!(I2C0CONSET&0x08));
                                  //check SI flag
                              //clear SI flag and start
  I2COCONCLR=0x28;
  I2CODAT=location;
                             //sending memory location
  while(!(I2C0CONSET&0x08));
                                  //check SI flag
                              //clear SI flag and start
  I2C0CONCLR=0x28;
```

I2C0DAT=data; while(!(I2C0CONSET&0x08)); //check SI flag



```
I2COCONCLR=0x28;
                              //clear SI flag and start flag
                         //send stop bit
  send_stop();
}
void send_start()
{
  I2C0CONSET=0x20;
}
void send stop()
{
  I2COCONSET=0x10;
}
unsigned char byte_read(unsigned char address, unsigned char location)
{
  unsigned char data;
  I2C0CONCLR=0xFF;
  I2C0CONSET=0x40;
  send_start();
  while(!(I2C0CONSET&0x08));
                                  //check SI flag
  I2C0CONCLR=0x28;
                              //clear SI flag and start
                                //selecting address in
  I2C0DAT=address&0xFE;
  while(!(I2C0CONSET&0x08));
                                  //check SI flag
  I2C0CONCLR=0x28;
                              //clear SI flag and start
  I2CODAT=location;
                             //sending memory location
  while(!(I2C0CONSET&0x08));
                                  //check SI flag
  I2C0CONCLR=0x28;
                              //clear SI flag and start
  send start();
                         //repeated start
  while(!(I2C0CONSET&0x08));
                                  //check SI flag
  I2C0CONCLR=0x28;
  I2C0DAT=address|0x01;
  while(!(I2C0CONSET&0x08));
                                  //check SI flag
  I2COCONCLR=0x28;
  I2C0CONCLR=0x04;
                              //NACK
  while(!(I2C0CONSET&0x08));
                                  //check SI flag
  I2C0CONCLR=0x28;
  data=I2C0DAT;
  send stop();
  return data;
}
```



void msdelay(unsigned int time)
{


```
int i,j;
for(i=0;i<time;i++)
{
    for(j=0;j<1008;j++)
    {
    }
}</pre>
```