STK-2148
Embedded Lab Manual

SPJ Embedded Technologies Pvt. Ltd.

101, Beaver Grandeur,
Baner Road,
Pune – 411045 (INDIA)
Tel. +91-20-27293002
Fax. +91-20-27293003
http://www.spjsystems.com

DISCLAIMER

Embedded Lab Manual for ARM7TDMI and other documentation:

This manual and the accompanying documentation such as schematic diagrams – in soft or hard form - contains descriptions of copyrighted products, which are not explicitly indicated as such. The absence of trademark symbol does not infer that a Product is not protected.

Additionally, registered patents and trademarks are similarly not expressly indicated in these documents.

The information in these documents has been carefully checked and is believed to be entirely reliable. However, **SPJ Embedded Technologies Pvt. Ltd.** (henceforth "**Company**") assumes no responsibility for any inaccuracies. The Company neither gives any guarantee nor accepts any liability whatsoever for consequential damages resulting from the use of these documents or associated products. The Company reserves the right to alter the information contained herein without prior notification and accepts no responsibility for any damages that might result. Additionally, The Company offers no guarantee nor accepts any liability for damages arising from the improper usage or improper installation of the hardware or software. The Company further reserves the right to alter the layout and/or design of the hardware without prior notification and accepts no liability for doing so.

SPJ Embedded Technologies Pvt. Ltd.
101, Beaver Grandeur,
Baner Road, Baner,
Pune - 411045.
INDIA
Phone: +91 (20) 27293002
Fax: +91 (20) 27293003
Email: spj@spjsystems.com
Web Site: http://www.spjsystems.com

Table of Contents

1. Introduction

This is user's manual for **STK-2148** Trainer Kit based on LPC2148 micro-controller. This trainer board is designed to perform lab practical and has a facility to download the user programs into the on-chip flash memory of the micro-controller.

It is strongly recommended to read this manual carefully before you start using the **STK-2148** board.

CAUTION: This board contains components that are sensitive to Electro-Static Discharge (ESD). The board must be handled carefully, so as not to subject it to ESD. As far as possible, do not touch any conducting part on the board - including any component or connector pins - as this may damage parts of the board permanently. If you want to touch any of the parts, make sure to discharge yourself to earth. Parts damaged due to ESD are not covered under limited warranty.

2. Getting Started

2.1 SCARM Installation:

1. As a part of the SCARM software package, you should have received a CD. Please insert it into the CD-ROM drive. Open **SPJTools** (CD drive), open the folder **SCARM** and run the **SETUP.EXE**
2. If it shows any error then click on Ignore.
3. **SCARM Setup** window will open, click on **NEXT**.
4. Accept the software license agreement by clicking on **YES**.
5. Choose destination location where you want to install the SCARM and click on **NEXT**.
6. Click **NEXT** button and installation will start.
7. Click **FINISH** button to complete the Installation.
8. Now connect the **SCARM SPJ Lock** to **USB port of the computer**.
9. A pop-up will show **"Found New Hardware"**
10. **Found New Hardware Wizard** window will be open. Select **No**, **not this time** option and click on **Next**.
11. Now select **Install from a list or specific location (Advanced)** and click on **Next.**
12. Browse for the folder **drv** from **C:\SCARM\drv** and click on **Next**.
13. Now **Hardware Installation window** will open, click on **Continue Anyway**.
14. After completion of installation, a message will come **The wizard has finished installing the software for SPJ Lock**, now click on **Finish**.
15. After this you can start using **SCARM (*Start > All Programs > SPJ-SCARM > SIDE_ARM*).** However, we recommend going through this user's manual before you actually start using it.

SCARM is C Compiler for ARM. It includes an **IDE** and other tools like **Debugger**, **Visual Code Generator (VCG)** and **Terminal Emulation Utility (SPJTerm)**. This document describes steps to create ARM applications in 'C' using the SCARM.

About “Project”:

What is a project?

A project is a file in which SIDEARM stores all information related to an application. E.g. it stores the name of ‘C’ and/or Assembler source file, memory size to be used and other options for compiler, assembler and linker.

Opening a project:

To open an existing project file, select **Project / Open Project** from the menu.

Creating a new project:

To create a new project, select **Project / New Project** from the menu.

Changing project settings:

To change the project settings (such as adding or removing ‘C’ and/or Assembler source file(s), changing memory settings etc.), select Project / Settings from the menu.

2.2 Quick start to create assembly language applications:

1. Start the **SIDE_ARM** program (i.e. the Integrated Development Environment) from start\Programs\SPJ-SCARM\SIDE_ARM.
2. From **Project** menu, select **Close project** (if any project is open).
3. From **Project** menu, select **New Project**. The Open dialog window will be displayed. Select the desired path where you wish to create this new project. (For example, C:\SPJ). CAUTION: The path and filename must not contain space or other special characters such as tab, comma, semicolon etc. In the “File name” field, type the name of the project, without any extension. For example, you may type “**PROG1**”. Then click on the “**Open**” button.
4. The action in the previous step will display the “**Project Settings**” dialog window. This dialog window has 3 different parts named “**Compiler Options**”, “**Linker Options**”, and “**Source Files**”. Any of these 3 parts can be displayed by clicking on the corresponding name near the top of this dialog window. Currently, the “**Compiler Options**” will be automatically displayed. If the target micro-controller (must be a member of ARM family) is known, you may select the appropriate Manufacturer from the list; and then select the appropriate micro-controller from the device list. If

the target micro-controller is not known or if you cannot find it in the list, then you may simply select “**Philips**” as the manufacturer and “**LPC2148**” as the micro-controller.

5. Click on “**Linker Options**” to display that part of the dialog window. In this window, you will see a list of 8 “Memory Banks”, with names such as “Memory #1”, “Memory #2” and so on. In your target hardware, there may be none or 1 or more number of contiguous memory blocks connected to the ARM micro-controller. Check the appropriate number of memory banks to reflect the target’s memory blocks. For each checked memory bank, specify memory start address (in Hexadecimal) and memory block size (in decimal). Size maybe specified either in number of Kilobytes (KB) or Megabytes (MB). Some of the memory blocks maybe “read-only” (e.g. flash or conventional EPROM). Accordingly, you may check or uncheck the “Read only” box. Based on this information about memory banks, the IDE will automatically create the Linker Script. This auto-generated script is adequate for most users. However, if you wish to use your own script file instead of this auto-generated script, you may check the “Use different linker script” box and further click on the browse button (marked “…”) and select appropriate linker script file.
6. Click on “**Source Files**” to display that part of the dialog window. This window will indicate that IDE has automatically added 2 files in this new project: **PROG1.c** and **startup.asm**. The startup.asm file is automatically created by the IDE and is required for all assembly and C projects. To write program in assembly language remove PROG1.c. To remove PROG1.c, click on it and then click **“Remove File”** button. Now the project contains only startup.asm, click “**OK**” button to create this new project. IDE will be opened and it will show startup.asm file. To write new assembly language program go to **File** option and select **New File**. Write the assembly language program in this file. For example:

```
.global main
.p2align 2
.text
main:
        -----------write program here -----------
        forever:
        -----------write program here -----------
        b forever                @ branch unconditionally to label
.end-main
```

7. To save this file go to **File** option and click on **Save.** Save this file at the same location **(C:\SPJ)**. Give the file name same as project name with .asm extension means **PROG1.asm** and **select All Files (*.*) in the Save as type option**. Now to add this PROG1.asm in the project go to **Project → Settings → Source files**. Click on **Add File** button and **select PROG1.asm**. It will be added in the project. Click on **OK** button. If you wish to add more files in this project, then click on the "**Add file**" button, select the desired filename and then click on "**Open**" button. Now the Project Settings dialog will indicate that selected file has been added into the project. When all necessary files have been added to the project, click "**OK**" button to create this new project.
8. From the **Compile** menu, select **Build**. This will invoke the assembler to assemble the file PROG1.asm; and further (assuming no errors) invoke the linker to create the **.hex** file. If there are any errors or warnings during the process of assembling or linking, then those will be displayed in the output window (below the editor window). If there are errors, then you may correct those by making appropriate changes to the program; select Save from File menu to save the changes and then again select Build from Compile menu. Repeat this until there are no errors.
9. You may inspect contents of the folder where your project files reside. When there are no errors and build has completed successfully and then you will see a filename with same name as the project name and extension .HEX (in above example, **PROG1.hex**). This is the file that you will need to use to program your micro-controller.

2.3 Quick start to create 'C' language applications:

10. Start the **SIDE_ARM** program (i.e. the Integrated Development Environment) from start\Programs\SPJ-SCARM.
11. From Project menu, select **Close project** (if any project is open).
12. From Project menu, select **New Project**. The Open dialog window will be displayed. Select the desired path where you wish to create this new project. (For example, C:\SPJ). CAUTION: The path and filename must not contain space or other special characters such as tab, comma, semicolon etc. In the "File name" field, type the name of the project, without any extension. For example, you may type "**PROG1**". Then click on the "**Open**" button.

13. The action in the previous step will display the "**Project Settings**" dialog window. This dialog window has 3 different parts named "**Compiler Options**", "**Linker Options**", and "**Source Files**". Any of these 3 parts can be displayed by clicking on the corresponding name near the top of this dialog window. Currently, the "**Compiler Options**" will be automatically displayed. If the target micro-controller (must be a member of ARM family) is known, you may select the appropriate Manufacturer from the list; and then select the appropriate micro-controller from the device list. If the target micro-controller is not known or if you cannot find it in the list, then you may simply select "**Philips**" as the manufacturer and "**LPC2148**" as the micro-controller.
14. Click on "**Linker Options**" to display that part of the dialog window. In this window, you will see a list of 8 "Memory Banks", with names such as "Memory #1", "Memory #2" and so on. In your target hardware, there may be none or 1 or more number of contiguous memory blocks connected to the ARM micro-controller. Check the appropriate number of memory banks to reflect the target's memory blocks. For each checked memory bank, specify memory start address (in Hexadecimal) and memory block size (in decimal). Size maybe specified either in number of Kilobytes (KB) or Megabytes (MB). Some of the memory blocks maybe "read-only" (e.g. flash or conventional EPROM). Accordingly, you may check or uncheck the "Read only" box. Based on this information about memory banks, the IDE will automatically create the Linker Script. This auto-generated script is adequate for most users. However, if you wish to use your own script file instead of this auto-generated script, you may check the "Use different linker script" box and further click on the browse button (marked "…") and select appropriate linker script file.
15. Click on "**Source Files**" to display that part of the dialog window. This window will indicate that IDE has automatically added 2 files in this new project: **PROG1.C** and **STARTUP.ASM**. The STARTUP.ASM file is automatically created by the IDE and is required for all C projects. Similarly, the IDE has automatically created an empty C file (PROG1.C). If the file PROG1.C already exists in the same path, then IDE would neither create/overwrite it nor modify it; but it will anyway add it to the project automatically. If you wish to add more files in this project, then click on the "**Add file**" button, select the desired filename and then click on "**Open**" button. Now the Project Settings dialog will indicate that selected file has been added into the project. When all necessary files have been added to the project, click "**OK**" button to create this new project.
16. The PROG1.C file created by the IDE will be an empty file containing only the frame of "main" function. You may write the desired program statements in this file (or other files that you may have

added to the project). When done, select **Save** from **File menu**. If you have modified more than one source files, then select **Save All** from **File menu**.

17. From the **Compile** menu, select **Build**. This will invoke the Compiler to compile the file PROG1.C; and further (assuming no errors) invoke the linker to create the **.HEX** file. If there are any errors or warnings during the process of compiling, assembling or linking, then those will be displayed in the output window (below the editor window). If there are errors, then you may correct those by making appropriate changes to the program; select Save from File menu to save the changes and then again select Build from Compile menu. Repeat this until there are no errors.
18. You may inspect contents of the folder where your project files reside. When there are no errors and build has completed successfully and then you will see a filename with same name as the project name and extension .HEX (in above example, **PROG1.HEX**). This is the file that you will need to use to program your micro-controller.

2.4 Quick start for µC/OS-II (RTOS) applications:

2.4.1 How to Open and modify existing uCOS Project:

1. Copy given uCOS programs on **C drive** (eg. ***C:\Multitasking***)
2. Open **SCARM (Start → All Programs → SPJ – SCARM → SIDE-ARM)**
3. Close the project in SIDE-ARM, if any project is open.
4. **To Open** the existing **project** go to **(SIDE-ARM → Project → Open project → *C:\Multitasking* → main.P51).** uCOS Multitasking project will open.
5. Now you can modify **app.c** (if required bsp.c) file as per your application. If you have added more tasks then their prototypes must be defined in app_cfg.h (***C:\Multitasking\src\Appl\app_cfg.h***)
6. To compile the project click on **Compile →** ***ReBuild All.***
7. After compilation its hex file will be created in build\bin sub-folder. Eg. ***C:\Multitasking\build\bin\LPC2148.hex***
8. Download this hex file and check the result.

2.4.2 Note: If you want to change folder name of the existing project then follow below mentioned procedure:

1. Copy given uCOS programs on C drive.
2. Make copy of one of the uCOS projects, with another name and save it (suppose ***C:\Multi_Task***).
3. As you have changed the name of the folder (means working directory), you must have to do the same change in the **Makefile** present in the same folder viz. ***C:\Multi_Task*** → ***Makefile***
4. Open this Makefile in WordPad. In Makefile find

   WORK_DIR := *C:\Multitasking (any another name may be present)*

   Change this line to

   WORK_DIR := C:\Multi_Task

   Save the Makefile and close it.
5. Open **SCARM (Start → All Programs → SPJ – SCARM → SIDE-ARM)**
6. Close the project in SIDE-ARM, if any project is open.
7. **To Open** the existing **project** (**Multi_Task**) go to **(SIDE-ARM → Project → Open project → *C:\Multi_Task* → main.P51).** Multi_Task project will open.
8. To use the Makefile modified in Steps 3 and 4, go to **Project → Setting → Linker Options**.
9. In Linker Options window last line is **Use this Makefile →**. Enable (Check) it (if not enabled). Browse the proper Makefile ie. **C:\Multi_Task → Makefile. Click on Ok.**

9. Now you can modify **app.c** (if required bsp.c) file as per your application. If you have added more tasks then their prototypes must be defined in app_cfg.h (***C:\Multi_Task\src\Appl\app_cfg.h***)
10. To compile the project click on **Compile →** ***ReBuild All.***
11. After compilation its hex file will be created in build\bin sub-folder. Eg. ***C:\Multi_Task\build\bin\LPC2148.hex***
12. Download this hex file and check the result.

2.4.3 How to Create New Project for µC/OS-II (RTOS):

1. Copy given uCOS programs on C drive.
2. Make copy of one of the uCOS projects, with another name and save it (suppose **C:\Multi_Task**).
3. As you have changed the name of the folder (means working directory), you must have to do the same change in the Makefile present in the same folder viz. **C:\Multi_Task → Makefile**
4. Open this Makefile in WordPad. In Makefile find

   WORK_DIR := *C:\Multitasking (any another name may be present)*

   Change this line to

   WORK_DIR := C:\Multi_Task

   Save the Makefile and close it.
5. Open **SCARM (Start → All Programs → SPJ – SCARM → SIDE-ARM)**
6. Close the project in SIDE-ARM, if any project is open.
7. Go to **Project → New Project** and select **Multi_Task** from **C drive**
8. Open **main.P51 file (C:\Multi_Task → main.P51).** Project Settings window will open. In this window, **Compiler Options** window will be visible.
9. Select Manufacturer as **Phillips** and Select a Micro controller as **LPC2148** and click **OK**.
10. Source Files window will open, click **OK**.
11. Now to select compiler as GCC go to **Edit → Settings → Select Compiler**. Select **GCC** and click **OK**.
12. A warning window will come, showing message to **remove startup.asm** and to **add Startup.s** file. Click **OK**.
13. To add Startup.s in the project go to **Project → Settings → Source Files**.
14. Select **startup.asm** and click **Remove File** button. Now click on **Add File** button, and **Open** window will appear in **Multi_Task** folder.
15. Delete **Startup.s** file from **Multi_Task** folder.
16. Go to **src\system** folder and add **startup.S** and **os_cpu_a.S** files. **(C:\Multi_Task → src → system → startup.S** and **os_cpu_a.S**).
17. Now click on **main.c** file in Workspace window and **Remove it.**
18. Again **Click on Add File** button. Go up in **src** folder (**C:\Multi_Task → src →**)

19. In src folder go to Appl folder and add app.c (**C:\Multi_Task → src → Appl → app.c**). Go up in **src** folder (**C:\Multi_Task → src →**)
20. In src folder go to BSP folder and add bsp.c (**C:\Multi_Task → src → BSP → bsp.c**). Go up in **Multi_Task** folder (**C:\Multi_Task**) and **Close**, **Open window**.
21. Click on **Ok** in **Source Files** window.
22. Close the file **main.c** in **Editor Window**.
23. To use the Makefile modified in Steps 3 and 4, go to **Project → Settings → Linker Options**. In Linker Options window last line is **Use this Makefile** →. Enable (Check) it. Browse the proper Makefile ie. **C:\Multi_Task → Makefile. Click on Ok.**
24. Now you can modify **app.c** (if required bsp.c) file as per your application. If you have added more tasks then their prototypes must be defined in app_cfg.h (***C:\Multi_Task\src\Appl\app_cfg.h***)
25. To compile the project click on **Compile →** ***ReBuild All.***
26. After compilation its hex file will be created in build\bin sub-folder. Eg. ***C:\Multi_Task\build\bin\LPC2148.hex***
27. Download this hex file and check the result.

3. Hardware Description

3.1 STK-2148:

Unpacking:

You will find following items in the package:

- STK-2148 board.
- Serial communication cable (Straight 9-Pin).
- Power adaptor with cable.
- SPJETs' CD-ROM.

Power Supply Requirements:

The power adaptor works with 230Volts AC. It produces approximately 9 Volts DC, and the STK-2148 uses on-board regulators to provide 5 Volts, 3.3 Volts and 1.8 Volts DC to all components on the board.

Connecting the system:

The serial communication cable supplied with the board should be used to connect the board to a PC running Windows95/98/NT/ ME/2000/XP/Vista Operating System. Connect one end of the serial cable to UART0 of STK-2148 board and other end to PCs serial port.

Powering ON:

After connecting the serial communication cable as described above, you may insert the power adaptor output jack into the on-board power socket. Plug the power adaptor into 230VAC mains outlet and turn it on. The power-on indication Green LED will turn on.

CAUTION: Please do not connect or disconnect the serial communication cable while the board is powered ON. Doing so can damage the serial port of the STK-2148 board and/or PC.

4. Connector Details, Jumper and Switch Settings

4.1 STK-2148 Block Diagram:

Below figure shows the locations of different components on the STK-2148 board.

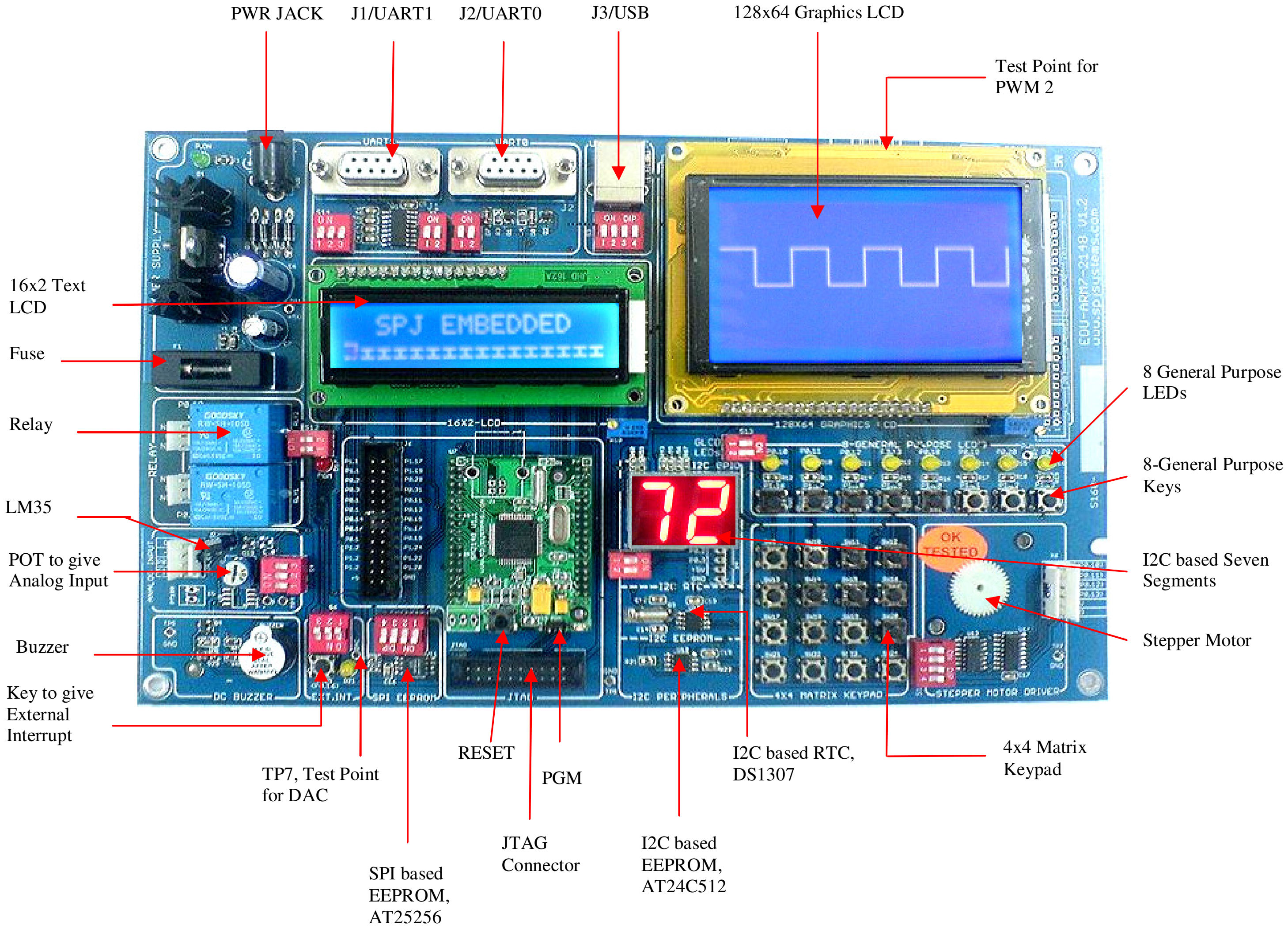
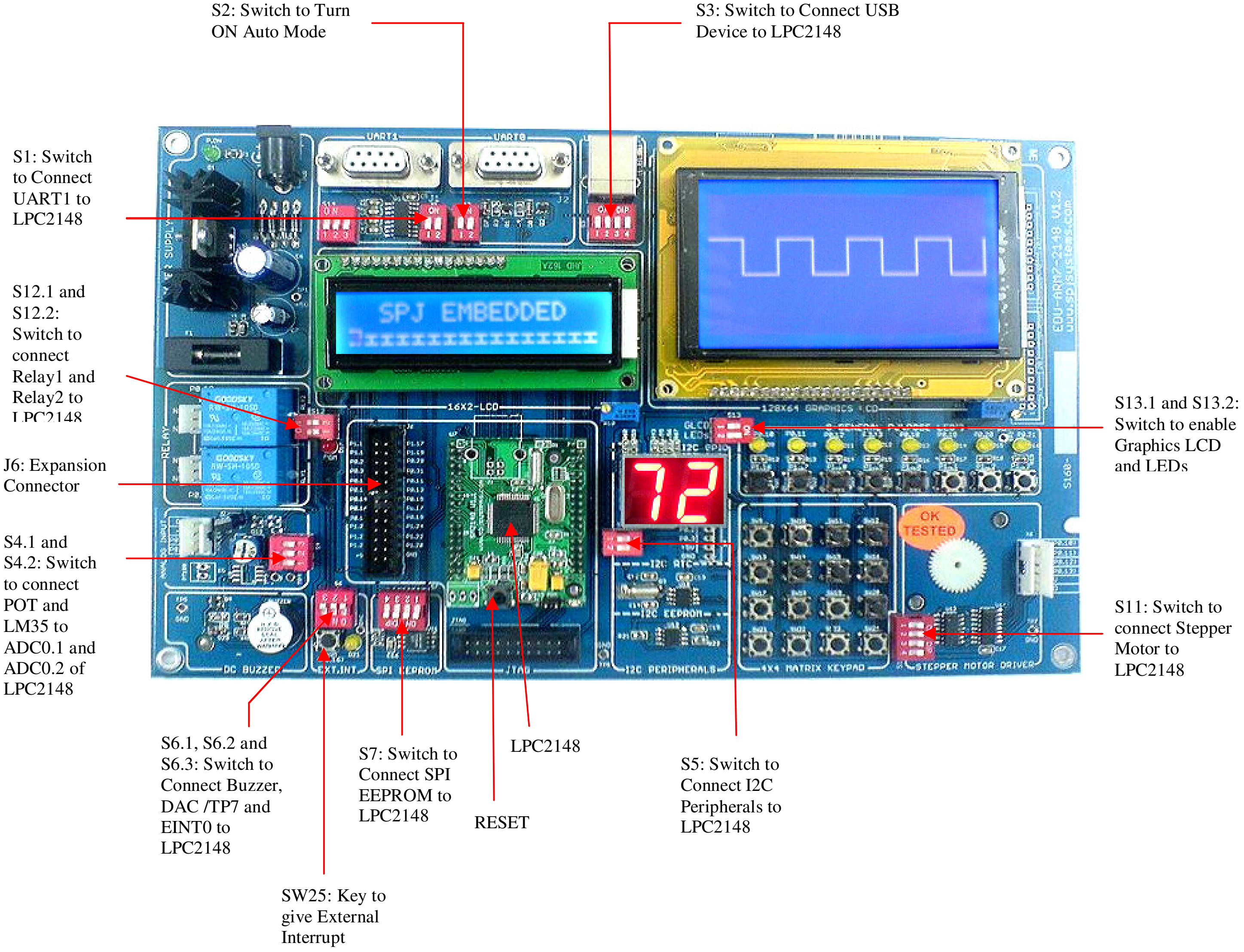


Figure 1

Below figure shows the locations of different switches on the STK-2148 board.



S2: Switch to Turn ON Auto Mode

S3: Switch to Connect USB Device to LPC2148

S1: Switch to Connect UART1 to LPC2148

S12.1 and S12.2: Switch to connect Relay1 and Relay2 to LPC2148

J6: Expansion Connector

S4.1 and S4.2: Switch to connect POT and LM35 to ADC0.1 and ADC0.2 of LPC2148

S13.1 and S13.2: Switch to enable Graphics LCD and LEDs

S11: Switch to connect Stepper Motor to LPC2148

S6.1, S6.2 and S6.3: Switch to Connect Buzzer, DAC /TP7 and EINT0 to LPC2148

S7: Switch to Connect SPI EEPROM to LPC2148

LPC2148

RESET

S5: Switch to Connect I2C Peripherals to LPC2148

SW25: Key to give External Interrupt

Figure 2

4.2 DIP Switches Details:

S1:

Turn ON this switch to connect UART1 connector to UART1 lines (TxD1/P0.8 and RxD1/P0.9) of LPC2148.

S2:

Mode selection switch. The LPC21xx micro-controllers include on-chip flash for storing user program and non-volatile data. The LPC2148 have 512KBytes flash. This flash is In-System-Programmable (ISP). The LPC21xx micro-controllers have a built-in boot-load program. Upon power-on, this boot-load program takes control; it passes control to the user program if pin P0.14 is HIGH and some other conditions are satisfied. Please refer to the LPC21xx data-sheet for further details.

On the STK-2148 board, the P0.14 pin is made available on this S3 switch. Turn ON this switch to control the Mode (ISP mode or Run mode) by Flash Magic.

S3:

Turn ON this switch to connect USB device connector to USB lines of LPC2148.

S4.1:

Turn ON this switch to connect POT (R20) to ADC0.1/P0.28 of LPC2148.

S4.2:

Turn ON this switch to connect LM35 to ADC0.2/P0.29 of LPC2148.

S5:

Turn ON this switch to connect I2C peripherals (Seven Segments, RTC (DS1307) and EEPROM (AT24C512)) to I2C lines (SCL0/P0.2 and SDA0/P0.3) of LPC2148.

S6.1:

Turn ON this switch to connect Buzzer to P0.25 of LPC2148.

S6.2:

Turn ON this switch to connect DAC/TP7 to DACOut/P0.25 of LPC2148.

S6.3:

Turn ON this switch to connect switch SW25 to EINT0/P0.16 of LPC2148. P0.16 is External Interrupt 0 pin of LPC2148.

S7:

Turn ON this switch to connect SPI EEPROM (AT25256) to SPI lines (SCK0/P0.4, MISO0/P0.5, MOSI0/P0.6 and CS/P0.7) of LPC2148.

S11:

Turn ON this switch to connect Stepper Motor to P0.10, P0.11, P0.12 and P0.13 of LPC2148.

S12.1:

Turn ON this switch to connect Relay 1 to P0.18 of LPC2148.

S12.2:

Turn ON this switch to connect Relay 2 to P0.19 of LPC2148.

S13.1:

Turn ON this switch to enable 128x64 Graphics LCD.

S13.2:

Turn ON this switch to enable 8 LEDs connected to P0.10 (D9), P0.11 (D10), P0.12 (D11), P0.13 (D12), P0.18 (D13), P0.19 (D14), P0.20 (D15) and P0.21 (D16) of LPC2148.

4.3 Push-Button Switches Details:

Push-Button	Signal Name
RST/RESET	Reset to LPC2148
SW1	P1.16
SW2	P1.17
SW3	P1.18
SW4	P1.19
SW5	P1.20
SW6	P1.21
SW7	P1.22
SW8	P1.23
SW25	P0.16/ EINT0
SW9 to SW24: Matrix Keypad	P1.16 to P1.23

4.4 LEDs and Buzzer Details:

LED	Signal Name
D1/ P.ON	+5V Power ON/OFF
D9	P0.10
D10	P0.11
D11	P0.12
D12	P0.13
D13	P0.18
D14	P0.19
D15	P0.20
D16	P0.21
D21	SW25

Buzzer	P0.25

4.5 Connector Details:

UART0:

This is a DB9 female connector, used for RS232 serial communication with the PC:

Pin 2 = UART0 RS232 TxD (output of µC)

Pin 3 = UART0 RS232 RxD (input to µC)

Pin 4 = RS232 DTR

Pin 5 = Ground

Pin 7 = RS232 RTS

All other pins of J1/UART0 are unused.

UART1:

This is a DB9 female connector, used for RS232 serial communication with the PC:

Pin 2 = UART1 RS232 TxD (output of µC)

Pin 3 = UART1 RS232 RxD (input to µC)

Pin 5 = Ground

16x2 LCD :

This is a 16 pin, single line connector, designed for connection to standard, text LCD modules. The pin/signal correspondence is designed to be matching with that required by such LCD modules.

Pin 1 = GND

Pin 2 = +5V

Pin 3 = Vlcd

Pin 4 = P1.25 (Used as RS of LCD)

Pin 5 = GND

Pin 6 = P1.24 (Used as EN of LCD)

Pin 7 to 10 = No Connection/GND

Pin 11 = P0.15 (Used as D4 of LCD)

Pin 12 = P0.17 (Used as D5 of LCD)

Pin 13 = P0.22 (Used as D6 of LCD)

Pin 14 = P0.30 (Used as D7 of LCD)

Pin 15 = Backlighting

Pin 16 = GND

128x64 Graphics LCD :

This is a 20 pin, single line connector, designed for connection to standard, 128x64 Monochrome Graphics LCD modules. The pin/signal correspondence is designed to be matching with that required by such LCD modules.

Pin 1 = GND

Pin 2 = +5V

Pin 3 = Vlcd

Pin 4 = P1.25 (Used as RS of GLCD)

Pin 5 = P0.15 (Used as RW of GLCD)

Pin 6 = P1.24 (Used as EN of GLCD)

Pin 7 = P0.10 (Used as D0 of GLCD)

Pin 8 = P0.11 (Used as D1 of GLCD)

Pin 9 = P0.12 (Used as D2 of GLCD)

Pin 10 = P0.13 (Used as D3 of GLCD)

Pin 11 = P0.18 (Used as D4 of GLCD)

Pin 12 = P0.19 (Used as D5 of GLCD)

Pin 13 = P0.20 (Used as D6 of GLCD)

Pin 14 = P0.21 (Used as D7 of GLCD)

Pin 15 = P0.22 (Used as CS1 of GLCD)

Pin 16 = P0.30 (Used as CS2 of GLCD)

Pin 17 =

Pin 18 =

Pin 19 = +5V

Pin 20 = GND

JTAG Connector:

This standard 20 pin JTAG connector provides debugging support for the LPC21xx. This connector is mounted on top side of the board as shown in figure1. **JTAG cables like SJT-S or SJT-U** can be connected to this connector, while other end of the cable can be connected to PC COM port or USB port, respectively. Debugger software (like the **debugger** built into **SCARM**) allows JTAG based debugging. It is also possible to use third party JTAG based emulators /debuggers. The pin-out of JTAG Connector is given below:

Pin	**Signal name**		**Pin**	**Signal name**
1	3.3V		2	3.3V
3	P1.31/NTRST		4	GND
5	P1.28/TDI		6	GND
7	P1.30/TMS		8	GND
9	P1.29/TCK		10	GND
11	P1.26/RTCK		12	GND
13	P1.27/TDO		14	GND
15	NRST		16	GND
17	GND		18	GND
19	GND		20	GND

J6:

This is 26 pin dual line headers. It brings out I/O and most of the pins of the LPC21xx micro-controller. Further, 5V and GND are also made available on these connectors. These connectors are intended for use to connect external peripherals.

The pin/signal details of J6 are as below:

Pin	**Signal name**		**Pin**	**Signal name**
1	P1.16		2	P1.17
3	P1.18		4	P1.19
5	P0.28		6	P0.29
7	P0.30		8	P0.31
9	P0.10		10	P0.11
11	P0.12		12	P0.13
13	P0.14		14	P0.15
15	P0.16		16	P0.17
17	P1.18		18	P1.19
19	P1.25		20	P1.24
21	P1.23		22	P1.22
23	P1.21		24	P1.20
25	+ 5V		26	GND

5. Downloading and Running User Programs

The LPC21xx micro-controllers include on-chip flash for storing user program and non-volatile data. LPC2148 on STK-2148 have 512KBytes flash. This flash is In-System-Programmable (ISP). Therefore it is possible to download user program into on-chip flash of LPC2148, through serial port connected to PC. For doing so, a certain position of S2 switch is required. **S2 Switch should be continuously ON**. This section describes how to use the software Flash Magic to download program into LPC2148.

5.1 How to install Flash Magic:

The CD you have received with this board contains SCARM, C Compiler for ARM. Install it. After installation go to folder **C:\SCARM\Utilities**. This folder contains 5 zip files. Install Flash Magic from FlashMagic3.71.zip. Extract the **FlashMagic3.71.zip** and then run FlashMagic.exe from the extracted files. (If you have wrong version of Flash Magic already installed, then please uninstall it first and then install new version).

5.2 Download and Run program using Flash Magic into LPC2148:

- After installation of Flash Magic, open it.
- In Flash Magic go to Options -> Advanced Options-> Communications. Check High Speed Communications and keep Maximum Baud Rate as 19200. Click on OK.
- Again in Flash Magic go to Options -> Advanced Options-> Hardware Config. “Use DTR and RTS to control RST and P0.14” option should be checked. Click on OK.

  (After doing above mentioned settings, Flash Magic stores it means for the next time just verify if these setting are proper or not. If they are proper then you can directly follow below mentioned procedure)

1. Connect the J2/UART0 connector of STK-2148 board to COM1 or COM2 of a PC, using the serial communication cable (supplied with the board).
2. Keep S2 switch in ON position. (You can keep S2 switch continuously ON) Switch ON power to the STK-2148.

3. Do proper settings in Flash Magic (COM Port: COM1 (if other choose it), Baud Rate: 38400, Device: LPC2148, Interface: None (ISP), Enable “Erase blocks used by Hex File”, Browse the file which you want to download) and click on Start button.
4. Flash Magic will download the program. Wait till Finished comes.
5. After downloading Flash Magic automatically resets the STK-2148 board and program executes. You can see output according to the program.
6. If again you want to Reset the board then press RST switch on SM2148 board. You can see output according to the program.

Note: Flash Magic can be used to download the program into other Philips Microcontrollers also. See the list in Flash Magic itself.

6. JTAG Cable for Debugging

6.1 Type:

SJT-S: Serial JTAG Cable.

6.2 Contents:

The "JTAG Cable" consists of following parts:

- Dongle (a small box with connectors on both ends).
- Cable.

The "dongle" consists of some electronic circuit for interfacing the JTAG port of target processor to the host computer. The cable is a bunch of wires to connect the dongle with the JTAG port of target.

6.3 Power Supply Requirements:

The JTAG cable draws power from the target board. Thus it does not require a separate power source.

6.4 Connecting JTAG Cable:

SJT-S:

There is a DB9 female connector on one end of the dongle. This directly mates with the PC COM port – which has a DB9 male connector (or you can connect yellow color serial cable, supplied with SJT-S or STK-2148 board, between DB9 female connector on one end of the dongle and PC COM port –which has a DB9 male connector). The other end of the dongle has DB25 female connector. There is a DB25 male connector on one end of the cable. These DB25 female and DB25 male connectors are designed to mate with each other directly. The other end of the cable has a 20-pin header. This should be connected to the JTAG connector of the target board.

CAUTION:

The JTAG Cable must not be connected or dis-connected when power is applied to the target board. Turn off power to the target board, connect the JTAG Cable and then you may turn on power to the target board.

Connecting the JTAG Cable with incorrect polarity / orientation may permanently damage the STK-2148 board and/or the JTAG Cable. It will also make the warranty void for both the products.

6.5 Verifying correct cable connection:

When the JTAG Cable is correctly connected to PC as well as the target board, it serves as a link between the JTAG port of target processor and the PC. This link is used by SPJ - SCARM software Tools (e.g. Debugger for ARM microcontrollers).

This software tool, Debugger will work correctly only when the JTAG Cable is connected correctly.

For SJT-S:

There is a crude test to verify SJT-S JTAG Cable connection. You may please follow these steps:

1. Connect JTAG Cable between PC COM port and JTAG connector on STK-2148, as per instructions in this manual.
2. Turn ON power to the target board.
3. On the PC, run SPJTerminal software.
4. In the Port Settings, select appropriate COM port (to which the JTAG cable is connected). Select 115200 baud, no parity, 8 bits per char, 1 stop bit and no flow control.
5. Open the COM Port connection.
6. Type character ‘V’ in the terminal window. I.e. send the character ‘V’ to the PC COM port.
7. If the JTAG Cable connection is correct, it will send version of JTAG cable. As a result, you will see that version appearing in the terminal window. This indicates that JTAG Cable connection is ok.
8. If you don’t see any character in the terminal window, probably the JTAG Cable is not connected appropriately.

6.6 How to Debug Program:

1. Connect SJT-S as mentioned above.
2. Open project in SIDEARM. Rebuild it.
3. Download the same code in the target board.
4. In SIDEARM go to Tools -> Debugger.
5. In Debugger go to Run -> Click on “Not connected to target (click here to connect)”.
6. Device ID starting from 0x4........ will be displayed and program will run.
7. To stop program go to Run -> Stop. Now you can insert breakpoint and say Run.
8. You can use all the functions visible in Run option.
9. In variable watch window you can see only global variables.
10. If you have declared any global variables then find their addresses from .map file.
11. Insert these addresses in variable watch window and you can see global variables also.

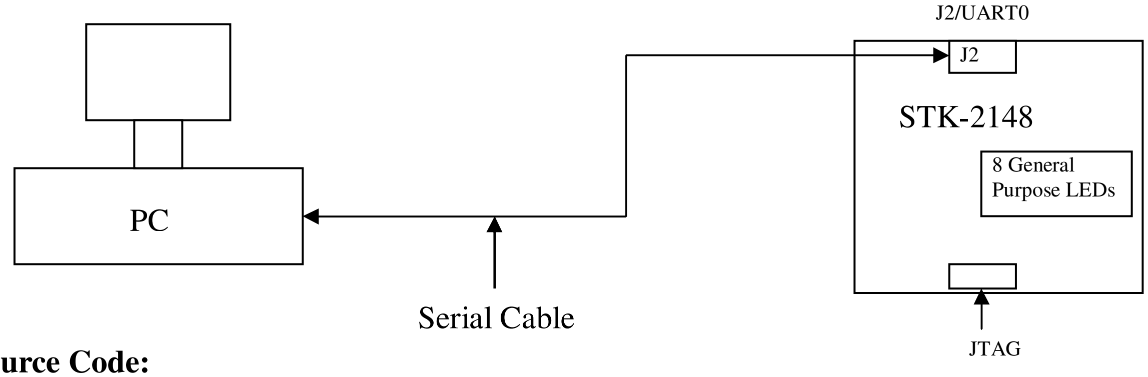
7. Experiments

7.1 Write assembly language program for arithmetic operations (addition, subtraction, multiplication and division).

Aim: Write assembly language program for addition, subtraction, multiplication and division.

Equipments: SCARM, PC, STK-2148, SJT-S (JTAG cable).

Block Diagram:



Source Code:
This program is available in the CD at:
STK-2148\Source\Exp01_Arithmetic

Connections:
To use SJT-S, JTAG cable: Refer Chapter 6

Procedure:
This program is available in the CD at:
STK-2148\Source\Exp01_Arithmetic

To Edit / Compile/ Generate Hex file: Refer Chapter 2
To download and run this program: Refer Chapter 5

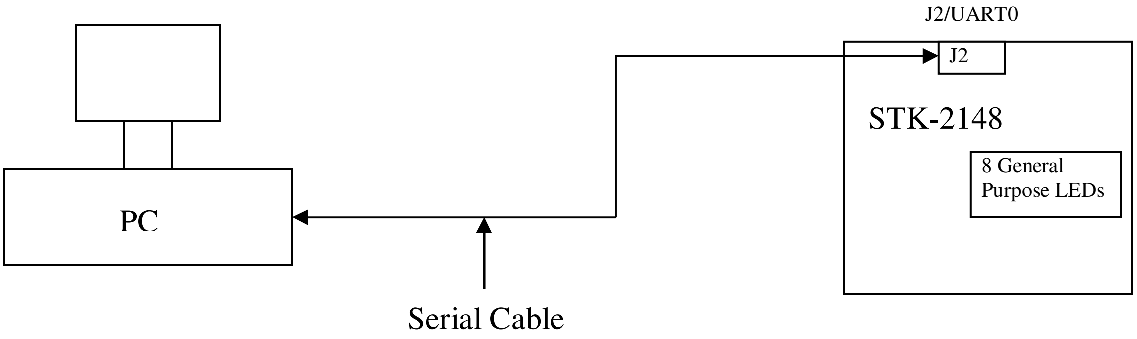
Output:
Arithmetic operations are performed on the ARM core registers. To see the result of operation use Debugger. Refer Chapter 6.

7.2 Write a program for Digital Output.

Aim: Write a Program to Blink LEDs present on STK-2148.

Equipments: SCARM, PC, STK-2148.

Block Diagram:



Source Code:
This program is available in the CD at:
STK-2148\Source\Exp02_Blink

Connections:
Keep S13.2 switch in ON position.
8 LEDs (D9 to D16) present on STK-2148 are connected to P0.10, P0.11, P0.12, P0.13, P0.18, P0.19, P0.20 and P0.21 respectively by Common Anode method.

Procedure:
This program is available in the CD at:
STK-2148\Source\Exp02_Blink

To Edit / Compile/ Generate Hex file: Refer Chapter 2.
To download and run this program: Refer Chapter 5.

Output:
You can see blinking of LEDs.

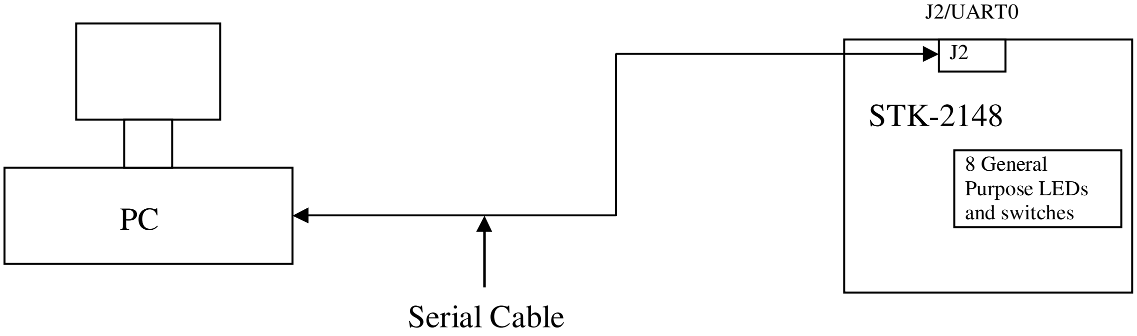
Note: Keep S13.2 switch in OFF position to save power, after execution of program.

7.3 Write a program for Digital Input.

Aim: Write a program to take input from push-button switches SW1 to SW8 (connected from P1.16 to P1.23) and show its status on corresponding LEDs D9 to D16 present on STK-2148.

Equipments: SCARM, PC, STK-2148.

Block Diagram:



Source Code:
This program is available in the CD at:
STK-2148\Source\Exp03_DigitalInput

Connections:
Keep S13.2 switch in ON position.
Push-button switches SW1 to SW8 are connected from P1.16 to P1.23.
LEDs D9 to D12 are connected to P0.10 to P0.13 and D13 to D16 are connected from P0.18 to P0.21.

Procedure:
This program is available in the CD at:
STK-2148\Source\ Exp03_DigitalInput

To Edit / Compile/ Generate Hex file: Refer Chapter 2.
To download and run this program: Refer Chapter 5.

Output:
After pressing any switch from SW1 to SW8, its corresponding LED (D9 to D16) will become ON otherwise it will be OFF.

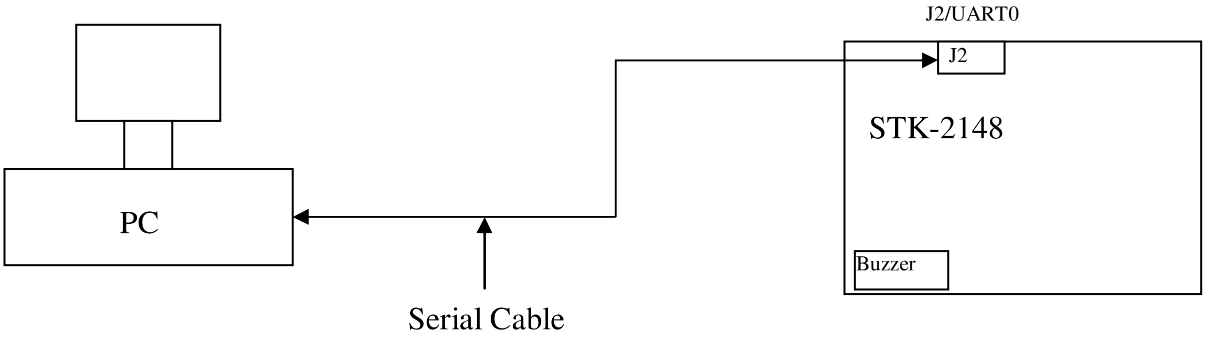
Note: Keep S13.2 switch in OFF position to save power, after execution of program.

7.4 Write a program for Buzzer Interface.

Aim: Write a program for buzzer interface.

Equipments: SCARM, PC, STK-2148.

Block Diagram:



Source Code:

This program is available in the CD at:

STK-2148\Source\Exp04_Buzzer

Connections:

Keep S6.1 switch in ON position.

Buzzer is connected to P0.25.

Procedure:

This program is available in the CD at:

STK-2148\Source\Exp04_Buzzer

To Edit / Compile/ Generate Hex file: Refer Chapter 2.

To download and run this program: Refer Chapter 5.

Output:

Buzzer will turn ON and OFF.

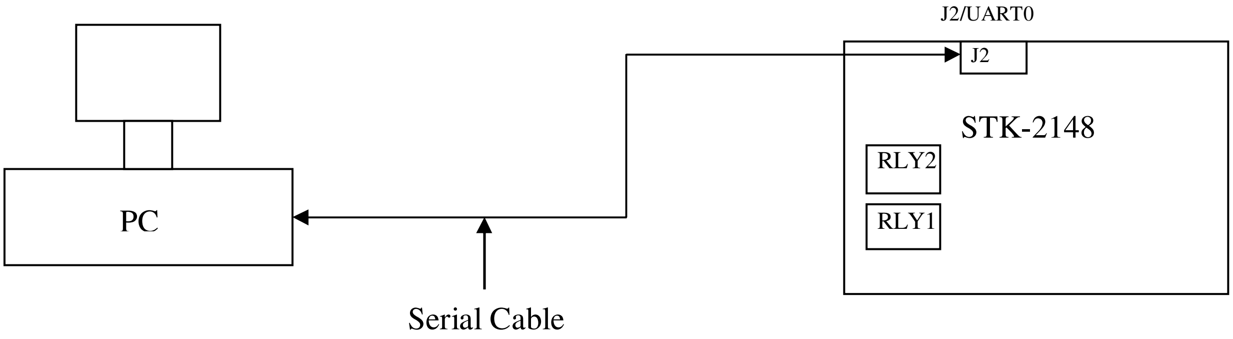
Note: Keep S6.1 switch in OFF position, after execution of program.

7.5 Write a program for Relay Interface.

Aim: Write a program to interface 2 relays with LPC2148.

Equipments: SCARM, PC, STK-2148.

Block Diagram:



Source Code:

This program is available in the CD at:

STK-2148\Source\ Exp05_Relay

Connections:

Keep S13.2 switch in ON position to see relay conditions (ON or OFF) on LEDs (D13 to D14).

Keep S12 switch in ON position. S12.1 is connected to RLY1 and S12.2 is connected to RLY2.

RLY1 and RLY2 are connected to P0.18 and P0.19 respectively through LEDs D13 and D14.

Procedure:

This program is available in the CD at:

STK-2148\Source\ Exp05_Relay

To Edit / Compile/ Generate Hex file: Refer Chapter 2.

To download and run this program: Refer Chapter 5.

Output:

Relays will become ON and OFF continuously.

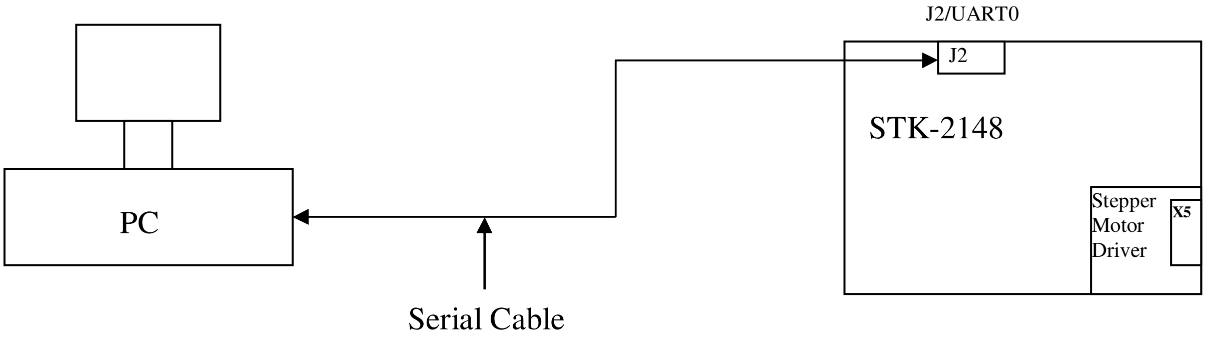
Note: Keep S13.2 and S12 switches in OFF position, after execution of program.

7.6 Write a program for Stepper Motor Interface.

Aim: Write a program to interface stepper motor.

Equipments: SCARM, PC, STK-2148 with Stepper Motor.

Block Diagram:



Source Code:

This program is available in the CD at:

STK-2148\Source\ Exp06_StepperMotor

Connections:

Keep S13.2 switch in ON position to see stepper motor phases on LEDs (D9 to D12).

Keep S11 switch in ON position.

Stepper Motor is connected to P0.10, P0.11, P0.12 and P0.13 through LEDs D9 to D12.

Procedure:

This program is available in the CD at:

STK-2148\Source\ Exp06_StepperMotor

To Edit / Compile/ Generate Hex file: Refer Chapter 2.

To download and run this program: Refer Chapter 5.

Output:

You can see stepper motor moving in a particular direction and corresponding phase changes you can observe on LEDs D9 to D12.

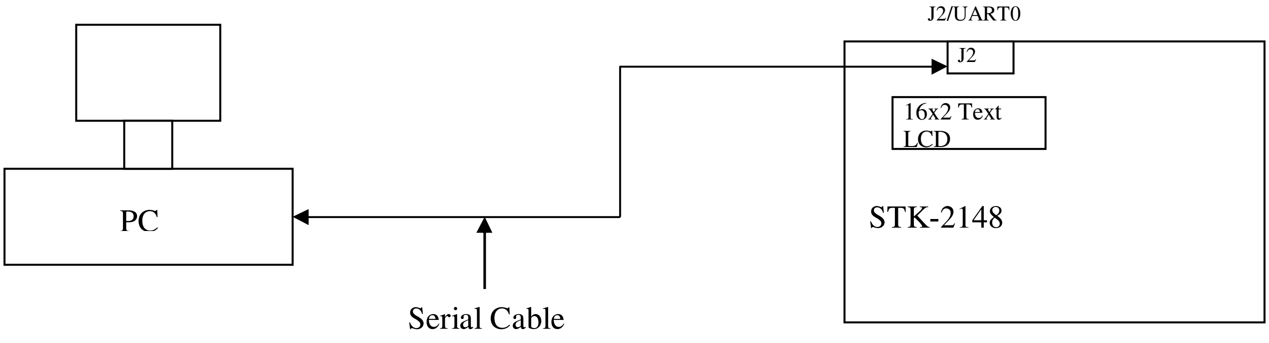
Note: Keep S13.2 and S11 switches in OFF position to save power, after execution of program.

7.7 Interfacing of 16x2 Text LCD.

Aim: Write a program to display message on 16x2 Text LCD.

Equipments: SCARM, PC, STK-2148 with 16x2 Text LCD.

Block Diagram:



Source Code:

This program is available in the CD at:

STK-2148\Source\ Exp07_HelloLCD

Procedure:

This program is available in the CD at:

STK-2148\Source\ Exp07_HelloLCD

To Edit / Compile/ Generate Hex file: Refer Chapter 2.

To download and run this program: Refer Chapter 5.

Output:

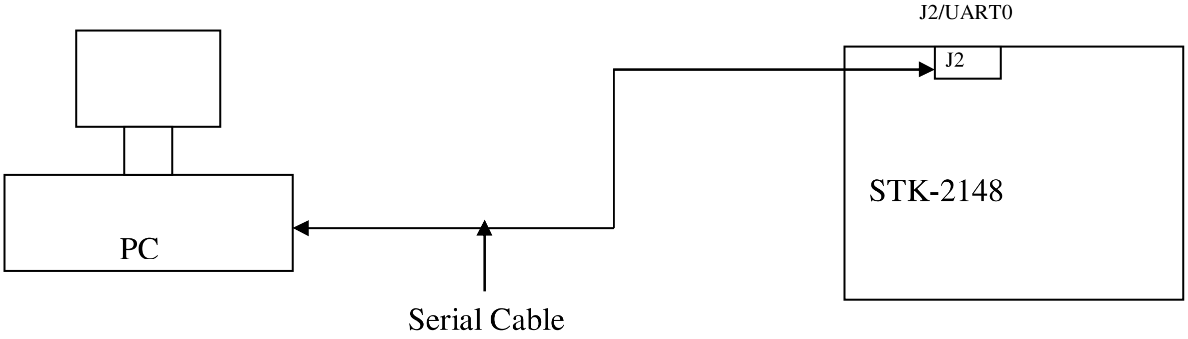
You can see the message **Hello World** on LCD. If required Reset the board.

7.8 Write a program for Serial Communication using UART0.

Aim: Write a Program to transfer massage “Hello world!” serially at 19200-baud rate 8-bit data and 1 stop-bit using UART0.

Equipments: SCARM, PC, STK-2148.

Block Diagram:



Source Code:
This program is available in the CD at:
STK-2148\Source\ Exp08_Hello

Connections:
Connect PCs serial port to J2/UART0 connector on STK-2148 by the cable provided to you with STK-2148.

Procedure:
This program is available in the CD at:
STK-2148\Source\ Exp08_Hello

To Edit / Compile/ Generate Hex file: Refer Chapter 2.
To download and run this program: Refer Chapter 5.

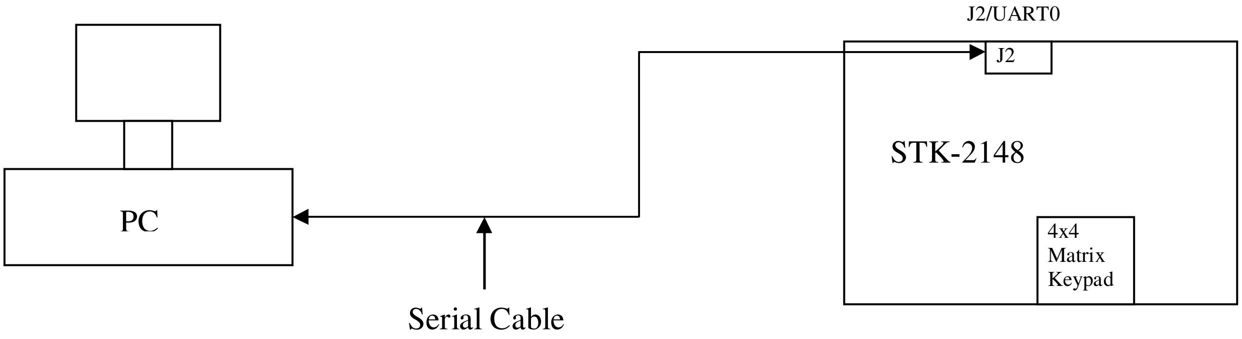
Output:
You can see output on SPJTerminal. Therefore Open SPJTerminal. Go to Port -> Settings. Do proper settings (Baud Rate: 19200, Data Bits: 8, Stop Bits: 1, Echo: Off, Parity: None, Com Port: Com 1 (if other choose it)). Click on OK. Go to Port -> Open. If required Reset the STK-2148 board. It will transmit the message "Hello world!”.

7.9 Write a program for 4*4 Matrix Keypad Interface.

Aim: Write a program to interface 4*4 matrix keypad.

Equipments: SCARM, PC, STK-2148.

Block Diagram:



Source Code:
This program is available in the CD at:
STK-2148\Source\ Exp09_4x4MatrixKeypad

Connections:
16 Keys (SW9 to SW25) present in 4x4 Matrix Keypad region on STK-2148 are connected to P1.16 to P1.23.

Procedure:
This program is available in the CD at:
STK-2148\Source\ Exp09_4x4MatrixKeypad

To Edit / Compile/ Generate Hex file: Refer Chapter 2.
To download and run this program: Refer Chapter 5.

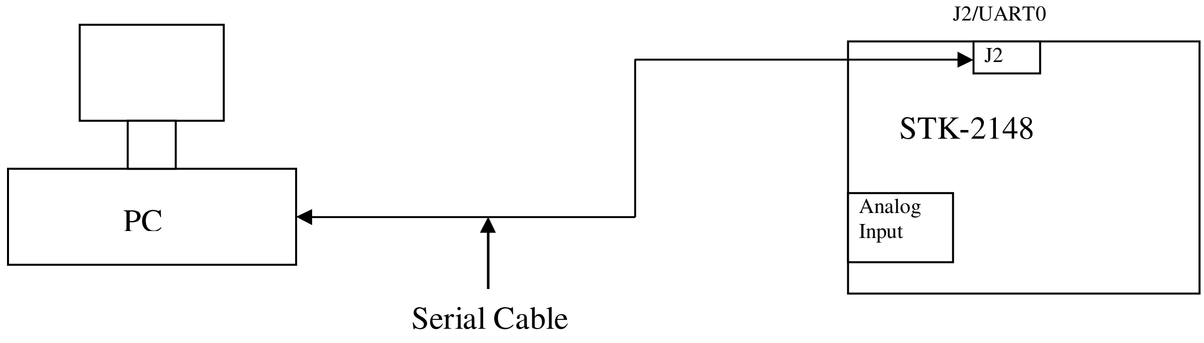
Output:
In this program after pressing any key from SW9 to SW24, its code will be displayed on 16x2 Text LCD.

7.10 Write a program for on-chip Analog to Digital Conversion.

Aim: Write a program for on-chip ADC (ADC 0, Channel 1, P0.28).

Equipments: SCARM, PC, STK-2148.

Block Diagram:



Source Code:
This program is available in the CD at:
STK-2148\Source\ Exp10_ADC

Connections:
To give analog input from Potentiometer R20 present in Analog Input region on STK-2148 keep S4.1 in ON position. To give external analog input, keep S4.1 in OFF position and connect external analog input to pin 1 of AIN_EXT connector. Analog input range is from 0 to 3.3V.

Procedure:
This program is available in the CD at:
STK-2148\Source\ Exp10_ADC

To Edit / Compile/ Generate Hex file: Refer Chapter 2.
To download and run this program: Refer Chapter 5.

Output:
You can see digital reading of the corresponding analog input on 16x2 Text LCD.

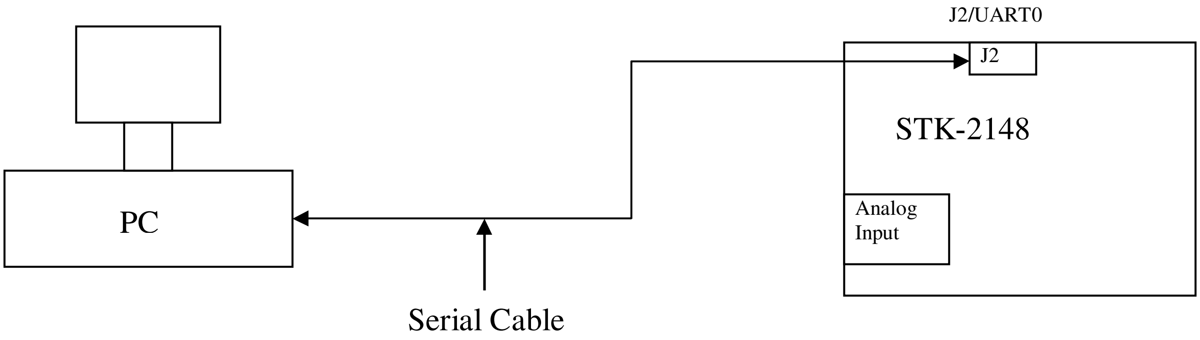
Note: Keep S4.1 switch in OFF position, after execution of program.

7.11 Write a program for LM35 Temperature Sensor Interface.

Aim: Write a program to interface of LM35 temperature sensor to on-chip ADC (ADC 0, Channel 2, P0.29).

Equipments: SCARM, PC, STK-2148.

Block Diagram:



Source Code:
This program is available in the CD at:
STK-2148\Source\ Exp11_LM35_TemperatureSensor

Connections:
To interface LM35 (present in Analog Input region on STK-2148) with AD0.2 keep S4.2 in ON position.

Procedure:
This program is available in the CD at:
STK-2148\Source\Exp11_LM35_TemperatureSensor

To Edit / Compile/ Generate Hex file: Refer Chapter 2.
To download and run this program: Refer Chapter 5.

Output:
You can see the temperature reading on 16x2 Text LCD.

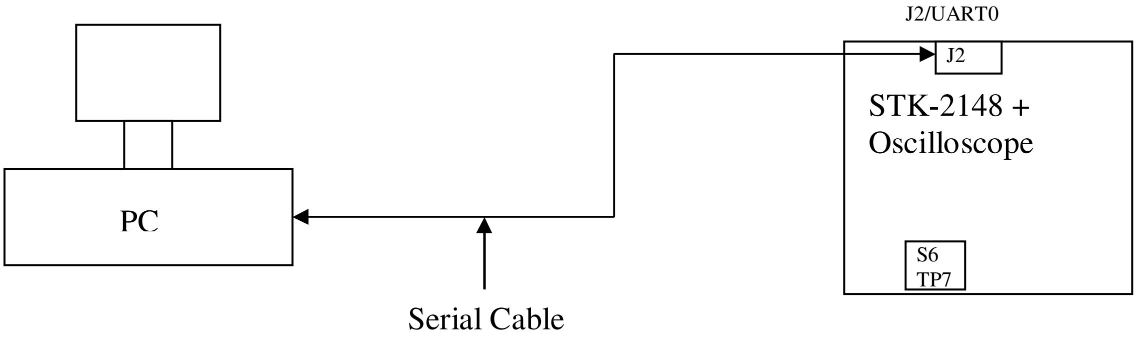
Note: Keep S4.2 switch in OFF position, after execution of program.

7.12 Generation of Ramp wave using on-chip DAC.

Aim: Write a program to generate ramp wave using on-chip DAC (P0.25).

Equipments: SCARM, PC, STK-2148, Oscilloscope.

Block Diagram:



Source Code:

This program is available in the CD at:

STK-2148\Source\DAC

Connections:

Keep S6.2 (present in EXT. INT region) in ON position.

Connect one pin of oscilloscope to TP7/DAC and another to GND.

Procedure:

This program is available in the CD at:

STK-2148\Source\DAC

To Edit / Compile/ Generate Hex file: Refer Chapter 2.

To download and run this program: Refer Chapter 5.

Output:

You can see ramp wave on oscilloscope.

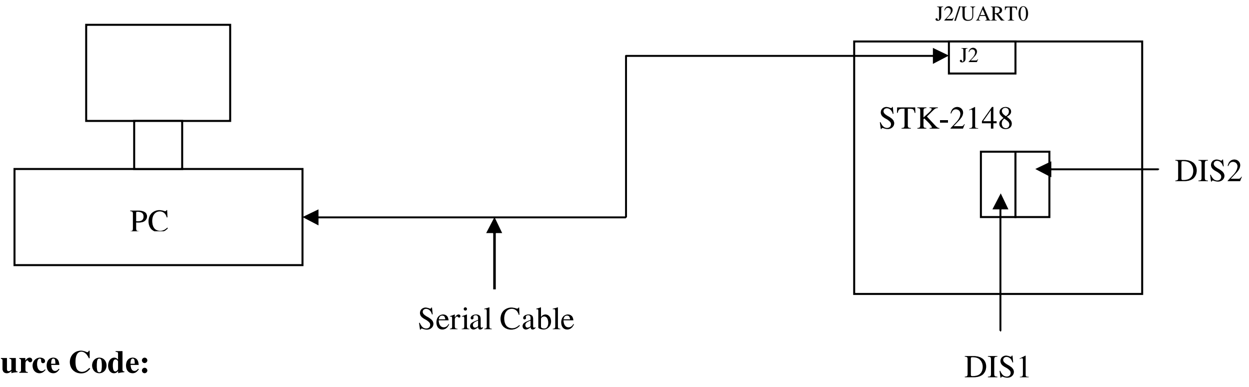
Note: Keep S6.2 switch in OFF position, after execution of program.

7.13 Write a program for I2C based Seven Segment LED Display Interface.

Aim: Write a program to display numbers on 7 segment displays using I2C protocol.

Equipments: SCARM, PC, STK-2148.

Block Diagram:



Source Code:

This program is available in the CD at:

STK-2148\Source\Exp13_I2C_SevenSegments

Connections:

Keep S5 switch (present in I2C Peripherals region) in ON position.

Procedure:

This program is available in the CD at:

STK-2148\Source\ Exp13_I2C_SevenSegments

To Edit / Compile/ Generate Hex file: Refer Chapter 2.

To download and run this program: Refer Chapter 5.

Output:

You can see 7 and 2 numbers on Seven Segments.

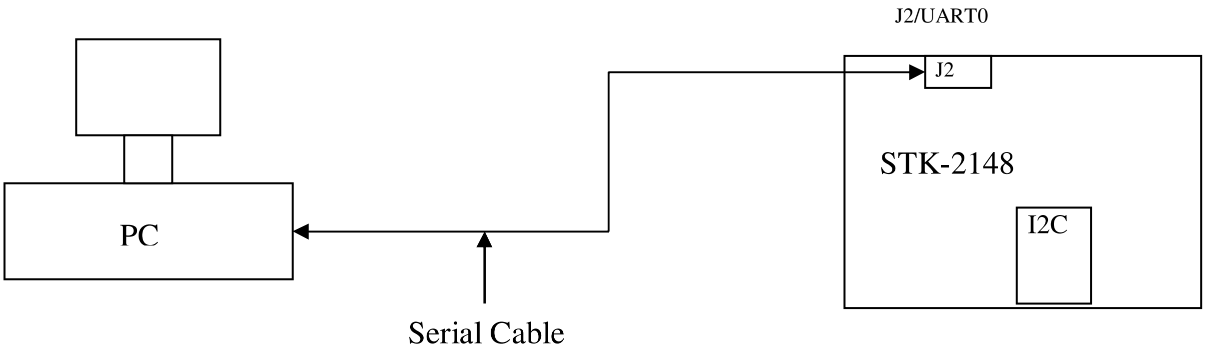
Note: Keep S5 switch in OFF position, after execution of program.

7.14 Write a program for I2C based EEPROM Interface.

Aim: Write a program to interface EEPROM using I2C protocol.

Equipments: SCARM, PC, STK-2148.

Block Diagram:



Source Code:
This program is available in the CD at:
STK-2148\Source\ Exp14_I2C_EEPROM

Connections:
Keep S5 switch (present in I2C Peripherals region) in ON position.

Procedure:
This program is available in the CD at:
STK-2148\Source\ Exp14_I2C_EEPROM

To Edit / Compile/ Generate Hex file: Refer Chapter 2.
To download and run this program: Refer Chapter 5.

Output:
You can see output on SPJTerminal. Therefore Open SPJTerminal. Go to Port -> Settings. Do proper settings (Baud Rate: 19200, Data Bits: 8, Stop Bits: 1, Echo: Off, Parity: None, Com Port: Com 1 (if other choose it)). Click on OK. Go to Port -> Open. If required Reset the STK-2148 board. First data will be written in EEPROM and then it will be read from EEPROM. The same data (0, 1, 2 and 3) will be displayed on SPJTerminal.

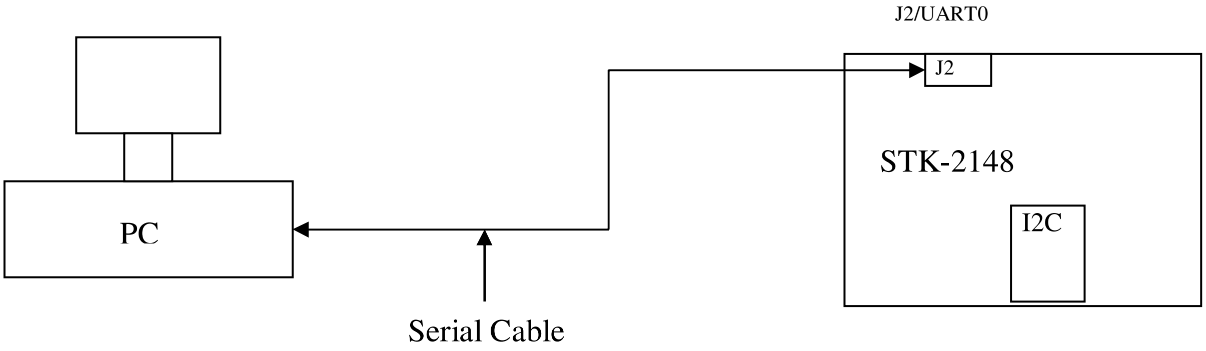
Note: Keep S5 switch in OFF position, after execution of program.

7.15 Write a program for I2C based RTC Interface.

Aim: Write a program to interface DS1307 RTC using I2C protocol.

Equipments: SCARM, PC, STK-2148.

Block Diagram:



Source Code:
This program is available in the CD at:
STK-2148\Source\ Exp15_I2C_RTC

Connections:
Keep S5 switch (present in I2C Peripherals region) in ON position.

Procedure:
This program is available in the CD at:
STK-2148\Source\ Exp15_I2C_RTC

To Edit / Compile/ Generate Hex file: Refer Chapter 2.
To download and run this program: Refer Chapter 5.

Output:
You can see date and time on 16x2 Text LCD.
Also you can see output on SPJTerminal. Therefore Open SPJTerminal. Go to Port -> Settings. Do proper settings (Baud Rate: 19200, Data Bits: 8, Stop Bits: 1, Echo: Off, Parity: None, Com Port: Com 1 (if other choose it)). Click on OK. Go to Port -> Open. If required Reset the STK-2148 board. To SET RTC follow the steps displayed on SPJTerminal otherwise wait for 5 seconds, date and time will be displayed continuously on SPJTerminal.

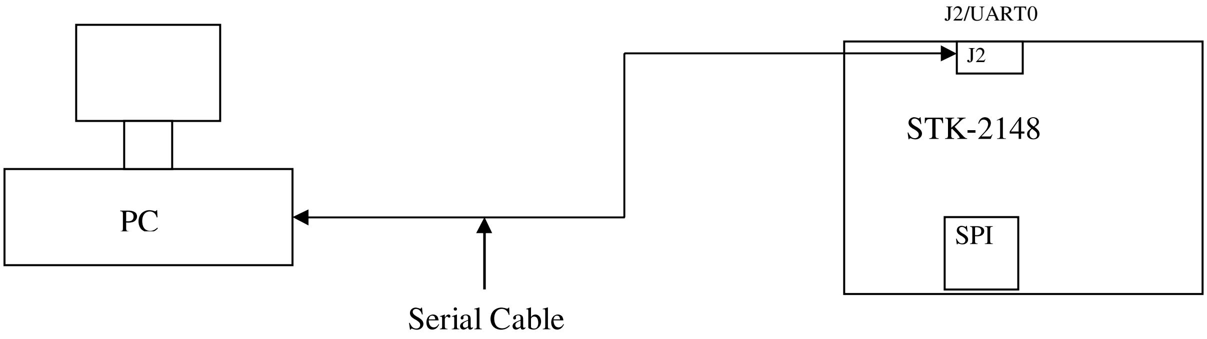
Note: Keep S5 switch in OFF position, after execution of program.

7.16 Write a program for SPI based EEPROM Interface.

Aim: Write a program to interface EEPROM using SPI protocol.

Equipments: SCARM, PC, STK-2148.

Block Diagram:



Source Code:
This program is available in the CD at:
STK-2148\Source\ Exp16_SPI_EEPROM

Connections:
Keep S7 switch (present in SPI EEPROM region) in ON position.

Procedure:
This program is available in the CD at:
STK-2148\Source\ Exp16_SPI_EEPROM

To Edit / Compile/ Generate Hex file: Refer Chapter 2.
To download and run this program: Refer Chapter 5.

Output:
You can see output on SPJTerminal. Therefore Open SPJTerminal. Go to Port -> Settings. Do proper settings (Baud Rate: 19200, Data Bits: 8, Stop Bits: 1, Echo: Off, Parity: None, Com Port: Com 1 (if other choose it)). Click on OK. Go to Port -> Open. If required Reset the STK-2148 board. First data will be written in EEPROM and then it will be read from EEPROM. The same data (Hello) will be displayed on SPJTerminal.

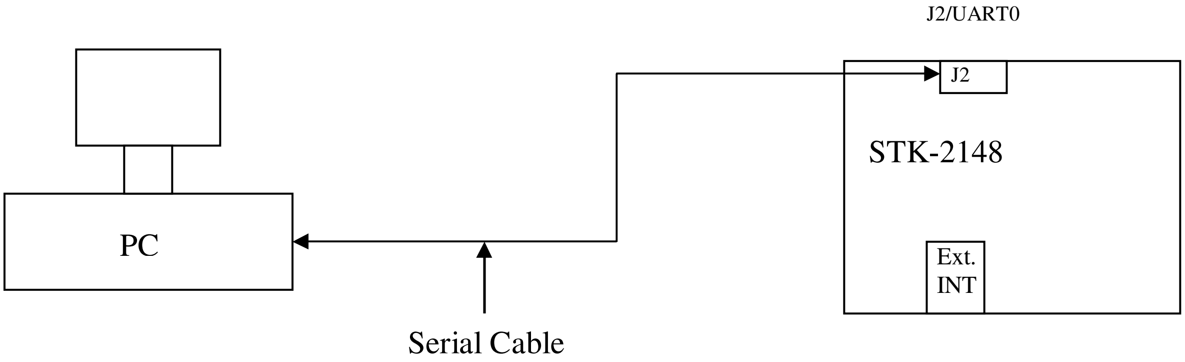
Note: Keep S7 switch in OFF position, after execution of program.

7.17 Write a program for External Interrupt.

Aim: Write a program to operate LED using push-button key connected to external interrupt EINT0 (P0.16) of LPC2148.

Equipments: SCARM, PC, STK-2148.

Block Diagram:



Source Code:

This program is available in the CD at:

STK-2148\Source\ Exp17_EINT0

Connections:

To connect SW25 switch (present in EXT. INT region) to EINT0/P0.16 of LPC2148, keep S6.3 (present in EXT. INT region) in ON position.

Procedure:

This program is available in the CD at:

STK-2148\Source\ Exp17_EINT0

To Edit / Compile/ Generate Hex file: Refer Chapter 2.

To download and run this program: Refer Chapter 5.

Output:

EINT0 is configured as Low-active, Level Sensitive. Therefore if SW25 switch is pressed then D17/PGM LED will glow for that much of time only.

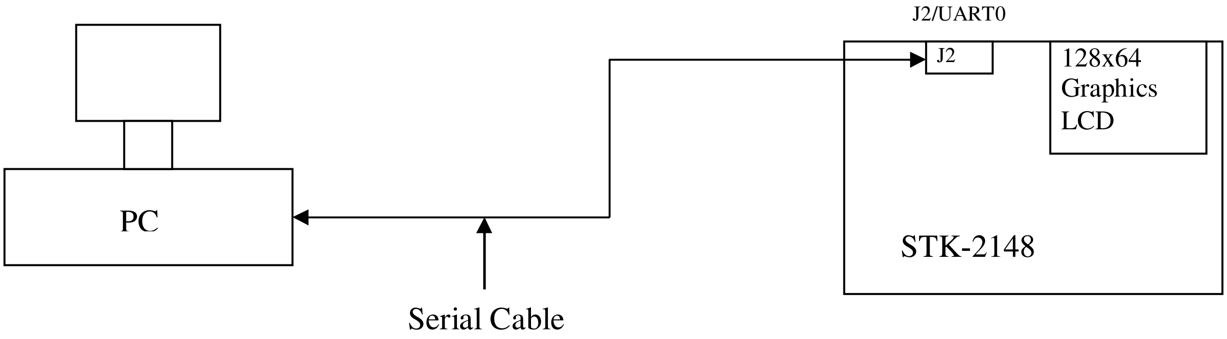
Note: Keep S6.3 switch in OFF position, after execution of program.

7.18 Write a program for 128x64 Graphics LCD Interface.

Aim: Write a program to interface 128x64 Graphics LCD.

Equipments: SCARM, PC, STK-2148.

Block Diagram:



Source Code:

This program is available in the CD at:

STK-2148\Source\ Exp18_GraphicsLCD

Connections:

Keep S13.1 switch in ON position to enable Graphics LCD.

Procedure:

This program is available in the CD at:

STK-2148\Source\ Exp18_GraphicsLCD

To Edit / Compile/ Generate Hex file: Refer Chapter 2.

To download and run this program: Refer Chapter 5.

Output:

For some time, message will be displayed on Graphics LCD. After message square wave will appear on Graphics LCD.

Note: Keep S13.1 switch in OFF position to save power, after execution of program.

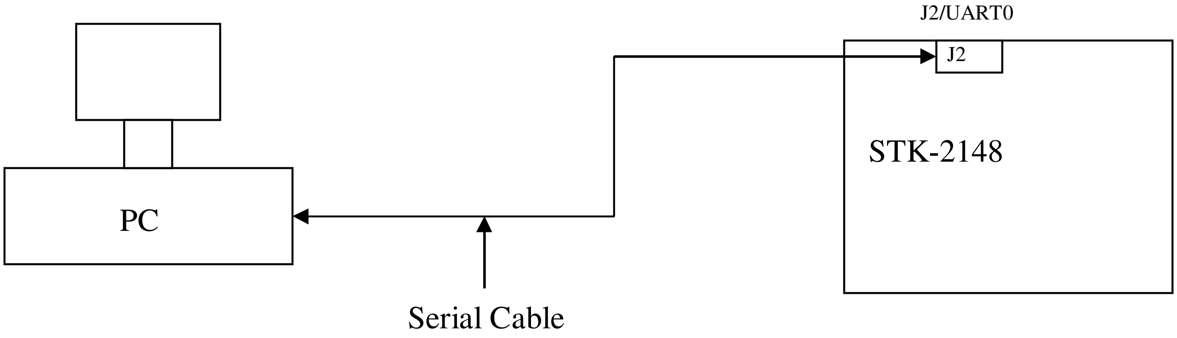
8. Experiments based on µC/OS-II RTOS

8.1 Implement Multitasking with Two separate LED blinking tasks.

Aim: Write a program for multitasking, to blink 2 LEDs (D9 (connected to P0.10) and D10 (connected to P0.11)) in 2 separate tasks.

Equipments: SCARM, PC, STK-2148.

Block Diagram:



Source Code:

uCOS programs are available at STK-2148\Source\uCOS-II

Paste all these programs on C drive.

Now this multitasking program is available at:

C:\Multitasking_LEDs (main.P51)

Connections:

Keep S13.2 switch in ON position.

This program is written to blink 2 LEDs, D9 and D10 connected to P0.10 and P0.11.

Procedure:

Now this program is available at:

C:\Multitasking_LEDs (main.P51)

To Edit / Compile/ Generate Hex file: Refer Section 2.4

To download and run this program: Refer Chapter 5

Output:

You can see LEDs blinking (D9 and D10).

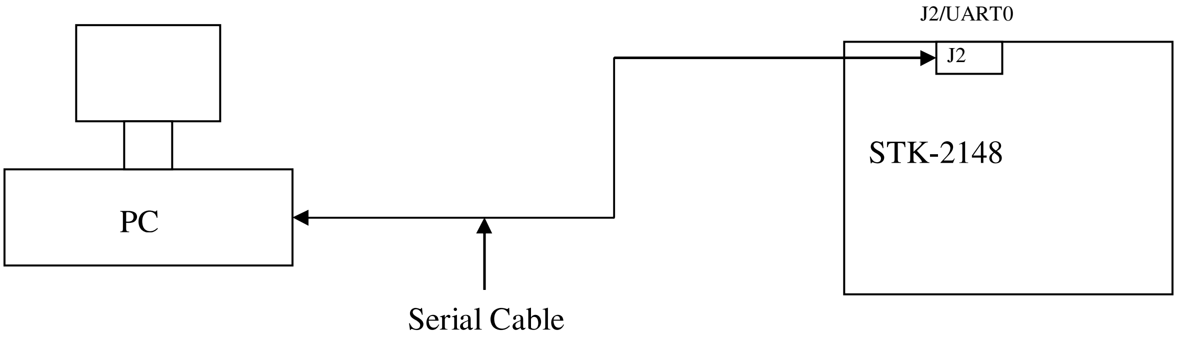
Note: Keep S13.2 switch in OFF position to save power, after execution of program.

8.2 Implement Priority Scheduling and OS Time Delay Functions by writing 3 different UART Transmitting Tasks.

Aim: Write a program to implement Priority Scheduling and OS Time Delay functions for multitasking, by writing 3 different UART Transmitting Tasks.

Equipments: SCARM, PC, STK-2148.

Block Diagram:



Source Code:

uCOS programs are available at STK-2148\Source\uCOS-II

Paste all these programs on C drive.

Now this multitasking program is available at:

C:\ Multitasking_Serial (main.P51)

Procedure:

Now this program is available at:

C:\ Multitasking_Serial (main.P51)

To Edit / Compile/ Generate Hex file: Refer Section 2.4

You can change Priority and value of delay in the tasks and observe the result.

To download and run this program: Refer Chapter 5

Output:

You can see output on SPJTerminal. Therefore Open SPJTerminal. Go to Port -> Settings. Do proper settings (Baud Rate: 19200, Data Bits: 8, Stop Bits: 1, Echo: Off, Parity: None, Com Port: Com 1 (if other choose it)). Click on OK. Go to Port -> Open. If required Reset the STK-2148 board. Observe the result on SPJTerminal.