LABORATORY MANUAL FOR THE COURSE

MICROCONTROLLERS & EMBEDDED SYSTEMS LABORATORY

(ECE 328)



(Mr. N.Srinivasa Naidu) Signature of the Physical Lab Incharge:

> (Dr. V.Rajya Lakshmi) Signature of the HOD:

ANIL NEERUKONDA INSTITUTE OF TECHNOLOGY & SCIENCES(A)

(Affiliated to AU, Approved by AICTE &Accredited by NBA) Sangivalasa-531162, Bheemunipatnam Mandal, Visakhapatnam Dt. Phone: 08933- 225084,226395

MICROCONTROLLER & EMBEDDED SYSTEMS LABORATORY

ECE328	Credits:2
Instruction: 3 Lab periods	Sessional Marks:50
End Exam: 3 Hours	End Exam Marks:50

Prerequisites:

Microprocessors and Interfacing, Microcontroller & Embedded Systems

COURSE OBJECTIVES

- > To program both 8051 to meet the requirements of the user.
- > To interface various peripherals
- > To handle interrupts
- > To design a microcomputer to meet the requirement of the user

COURSE OUTCOMES

At t	At the end of the course student will be able to				
1.	Program 8051 microcontroller to meet the requirements of the user.				
2.	Interface peripherals like switches, LEDs, stepper motor, Traffic lights controller, etc,				
3.	Handle interrupts				
4.	Design a microcontroller development board to meet the requirements of the user				

Mapping of Course Outcomes with Program Outcomes:

		PO							PSO							
		1	2	3	4	5	6	7	8	9	10	11	12	1	2	3
	1	2	1	2	2	3	-	-	-	-	-		1	2	2	2
	2	3	2	2	2	3	-	-	-	-	-		1	2	3	2
0	3	3	2	2	2	3	-	-	-	-	-		1	2	2	2
	4	3	2	3	3	3	-	-	-	-	-		1	3	3	3

3: high correlation, 2: medium correlation, 1: low correlation

PROGRAM OUTCOMES

- 1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialisation for the solution of complex engineering problems.
- 2. **Problem analysis:** Identify, formulate, research literature, and analyse complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
- 3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs.
- 4. **Conduct investigations of complex problems:** An ability to design and conduct scientific and engineering experiments, as well as to analyze and interpret data to provide valid conclusions
- 5. **Modern tool usage:** Ability to apply appropriate techniques, modern engineering and IT tools, to engineering problems.
- 6. **The engineer and society:** An ability to apply reasoning to assess societal, safety, health and cultural issues and the consequent responsibilities relevant to the professional engineering practice
- 7. **Environment and sustainability:** An ability to understand the impact of professional engineering solutions in societal and environmental contexts
- 8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
- 9. **Individual and team work:** Ability to function effectively as an individual, and as a member or leader in a team, and in multidisciplinary tasks.
- 10. **Communication:** Ability to communicate effectively on engineering activities with the engineering community such as, being able to comprehend and write effective reports and design documentation, make effective presentations.
- 11. **Project management and finance:** An ability to apply knowledge, skills, tools, and techniques to project activities to meet the project requirements with the aim of managing project resources properly and achieving the project's objectives.
- **12.** Life-long learning: Recognise the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

PROGRAM SPECIFIC OUTCOMES

PSO1: Professional Skills: An ability to apply the knowledge of mathematics, science, engineering fundamentals in ECE to various areas, like Analog & Digital Electronic Systems, Signal & Image Processing, VLSI & Embedded systems, Microwave & Antennas, wired & wireless communication systems etc., in the design and implementation of complex systems.

PSO2: Problem-Solving Skills: An ability to solve complex Electronics and communication engineering problems, using latest hardware and software tools, along with significant analytical knowledge in Electronics and Communication Engineering

PSO3: Employability and Successful career: Acquire necessary soft skills, aptitude and technical skills to work in the software industry and/or core sector and able to participate and succeed in competitive examinations.

List of Experiments:

- 1. Study and familiarization of 8051 Microcontroller trainer kit
- 2. Assembly Language Program for addition of 8-bit numbers stored in an array
- 3. Assembly Language Program for Multiplication by successive addition of two 8-bit numbers
- 4. Assembly Language Program for finding largest no. from a given array of 8-bit numbers
- 5. Assembly Language program to arrange 8-bit numbers stored in an array in ascending order
- 6. Stepper motor control by 8051 Microcontroller
- 7. Interfacing of 8-bit ADC 0809 with 8051 Microcontroller
- 8. Interfacing of 8-bit DAC 0800 with 8051 Microcontroller and Waveform generation using DAC
- 9. Implementation of Serial Communication by using 8051 serial ports
- 10. Assembly Language Program for use of Timer/Counter for various applications
- 11. Traffic light controller/Real-time clock display

12. Simple test program using ARM 9 mini 2440 kit (Interfacing LED with ARM 9 mini 2440 kit)

NOTE:

- 1. It is compulsory for each student to create their own Microcontroller Development Board for personal use
- 2. A student has to perform a minimum of 10 experiments.

Scheme of evaluation for MCES Laboratory:

Lab Internal:

I. Observation – 5M

(Successful Wording/Algorithm/flowchart-1M, Successful Program verification – 1M, Successful Program Execution – 1M, Record Initial and Indexing – 2M)

- II. Record 10M (Aim&Apparatus – 1M, Theory – 3M, Algorithm/flowchart – 2M(each experiment should have atleast one flowchart, Calculations, Input/Output observations & Result – 1M, Daily Performance 3M)
- III. Lab Project 10M
 (It is compulsory for each student to create their own Microcontroller Development Board for personal use based on 8051)
- IV. Attendance 5M
- V. Internal End Exam 20M (Aim, Apparatus – 2M, Program – 10M (Mnemonics/code – 5M, Relevant Comments – 2M, Algorithm/flow chart – 3M), Calculations, Input/Output observations & Result – 5M, Performance – 3M)

Lab External:

I. Writeup – 10M (Aim– 2M, Apparatus – 1M, Theory – 2M, Algorithm/flowchart – 5M)

- II. Program 15M (Mnemonics/Code – 10M, Comments – 3M, Optimization– 2M)
- III. Performance 5M (Experimentation skill - Connections, etc.)
- IV. Result 10M (Identifying & Showing the inputs and outputs – 2M and/or theoretical calculations – 2M, Output Verification – 6M (Partial output – 3M, No Output – 0M)
- V. Viva 10M

Experiment-1

Introduction to Keil:-

Embedded system means some combination of computer hardware and programmable software which is specially designed for a particular task like displaying message on LCD. It involves hardware (8051 microcontroller) and software (the code written in assembly language).

Some real life examples of embedded systems may involve ticketing machines, vending machines, temperature controlling unit in air conditioners etc. Microcontrollers are nothing without a Program in it.

One of the important part in making an embedded system is loading the software/program we develop into the microcontroller. Usually it is called "burning software" into the controller. Before "burning a program" into a controller, we must do certain prerequisite operations with the program. This includes writing the program in assembly language or C language in a text editor like notepad, compiling the program in a compiler and finally generating the hex code from the compiled program. Earlier people used different software's /applications for all these 3 tasks. Writing was done in a text editor like notepad/ WordPad, compiling was done using a separate software (probably a dedicated compiler for a particular controller like 8051), converting the assembly code to hex code was done using another software etc. It takes lot of time and work to do all these separately, especially when the task involves lots of error debugging and reworking on the source code.

The μ Vision IDE is the easiest way for most developers to create embedded applications using the Keil development tools. The new Keil μ Vision4 IDE has been designed to enhance developer's productivity, enabling faster, more efficient program development.

Keil MicroVision is a free software which solves many of the main points for an embedded program developer. This software is an integrated development environment (IDE), which integrated a text editor to write programs, a compiler and it will convert your source code to hex files too. μ Vision4 introduces a flexible window management system, enabling us to drag and drop individual windows anywhere on the visual surface including support for **Multiple Monitors**.

Embedded Systems Vs General Computing Systems

General Purpose System	Embedded System
A system which is a combination of generic hardware and General Purpose Operating System for executing a variety of applications	A system which is a combination of special purpose hardware and embedded OS for executing a specific set of applications
Contain a General Purpose Operating System (GPOS)	May or may not contain an operating system for functioning
Applications are alterable (programmable) by user (It is possible for the end user to re- install the Operating System, and add or remove user applications)	The firmware of the embedded system is pre- programmed and it is non-alterable by end-user (There may be exceptions for systems supporting OS kernel image flashing through special hardware settings)
Performance is the key deciding factor on the selection of the system. Always 'Faster is Better'	Application specific requirements (like performance, power requirements, memory usage etc) are the key deciding factors
Less/not at all tailored towards reduced operating power requirements, options for different levels of power management.	Highly tailored to take advantage of the power saving modes supported by hardware and Operating System
Response requirements are not time critical	For certain category of embedded systems like mission critical systems, the response time requirement is highly critical
Need not be deterministic in execution behavior source	Execution behavior is deterministic for certain type of embedded systems like 'Hard Real Time' systems

C51 Development Tools

Keil development tools for the 8051 microcontroller family support every level of developer from the professional applications engineer to the student just learning about embedded software development. The industry-standard Keil C Compilers, Macro Assemblers, Debuggers, Real-time Kernels, and Single-board Computers support ALL 8051-compatible derivatives and help you get your projects completed on schedule.

The following table shows the Keil C51 Product Line (across the top) and the Components that are included (along the left side). You may use this information to find the development tool kit that best fits your needs.

Introduction

The C51 development tool chains are designed for the professional software developer, but any level of programmer can use them to get the most out of the 8051 microcontroller architecture.

With the C51 tools, embedded applications can be generated for virtually every 8051 variant. Refer to the μ Vision Device Database for a list of currently supported microcontrollers.

This introduction includes a brief explanation of the:

- □ Software Development Cycle that describes the steps and tools involved to create a project.
- □ Development Tools that describes the major features of the Keil C51 development tools including the μ Vision IDE and Debugger.
- □ Folder Structure that describes the default location of µVision and the C51 tool chain installation

Development Tools

The Keil C51 development tools offer numerous features and advantages that help you to develop embedded applications quickly and successfully. Find out more about the supported devices and the possible tool combinations available for the different 8051 variants.

The following block diagram shows the components involved in the build process.

The μ Vision IDE is a window-based software development tool that combines project management and a rich-featured editor with interactive error correction, option setup, make facility, and on-line help. Use μ Vision to create source files and organize them into a project that defines your target application.



µVision Integrated Development Environment (IDE)

C Compiler

The Keil Cx51 Compiler is a full ANSI implementation of the C programming language and supports all standard features of the C language. In addition, numerous extensions have been included to directly support the 8051 and extended 8051 architecture.

Macro Assembler

The Keil Ax51 Macro Assembler supports the complete instruction set of the 8051 and all 8051 derivatives.

Library Manager

The LIBx51 Library Manager allows you to create the object library from object files created by the compiler and assembler. Libraries are specially formatted, ordered program collections of object modules that may be used by the linker at a later time. When the linker processes a library, only those object modules necessary to create the program are used.

Linker/Locater

The Lx51 Linker/Locater creates the final executable 8051 program and combines the object files created by the compiler or assembler, resolves external and public references, and assigns absolute addresses. In addition, it selects and includes the appropriate run-time library modules.

µVision Debugger

The μ Vision Debugger is ideally suited for fast and reliable program debugging. The debugger includes a high-speed simulator capable of simulating an entire 8051 system including on-chip peripherals and external hardware.

The µVision Debugger provides several ways to test programs on target hardware:

- □ Use the Keil ULINK USB-JTAG adapter for downloading and testing your program.
- Install a target monitor on your target system and download your program using the built-in monitor interface of the μ Vision Debugger.
- Use the Advanced GDI interface to attach and use the µVision Debugger front end with your target system.

RTOS Kernel

The RTOS Kernel, describes the advantages of using a real-time kernel like the Keil RTX51 Tiny in embedded systems.

Creation of HEX File

Some applications require a HEX file to download the application software into the physical device using a Flash programming utility. μ Vision creates HEX files with each build process when Create HEX File is enabled in the dialog Options for Target Output.

Options for Target 'Target 1'	
Device Target Output Listing User	C51 A51 BL51 Locate BL51 M
Select Folder for Objects	Name of Executable: MyApp
☑ Debug Information	Browse Information
Create HEX File HEX Format:	HEX-80

If code banking is used, then the application has to be converted with the $\underline{OC51}$ Banked Object File Converter prior to using the $\underline{OH51}$ Object/Hex converter. When the extended LX51 Linker is used, it is mandatory to use the <u>OHx51</u> Extended Object-HEX Converter to generate an Intel HEX-386 file that contains the common area and all the code banks.

Start Debugging

 μ Vision provides several ways to invoke debugging commands:

- □ Commands used from the menu Debug or the **Debug Toolbar**.
- □ Commands entered manually in the Command Window.
- Commands available from the **Context Menu** of the **Editor** or **Disassembly** window.
- □ Debug Functions executed from an initialization file.

Start the Debugger

- Use the **Start/Stop Debug Session** button from the **Debug Toolbar** to start or stop a debugging session.
- The current instruction or high-level statement (the one about to execute) is marked with a yellow arrow. For each step-command, the arrow moves to reflect the new current line or instruction.
- Depending on the Options for Target Debug configuration, µVision loads the application program and runs the startup code (Run to main ()).
- µVision saves the editor screen layout and restores the screen layout of the last debug session. When program execution stops, µVision opens an Editor window with the source text or shows MCU instructions in the Disassembly Window.

Execute Commands

- □ Run the program to the next break point, or type **GO** in the **Command Line**.
- Halt the program, or press **Esc** while in the **Command Line**
- □ Click **Reset** from the **Debug Toolbar** or from the **Debug Reset CPU Menu** or type **RESET** in the **Command Line** to reset the CPU.

Single-Stepping Commands

- □ To step through the program and into function calls. Alternatively, you can enter **TSTEP** in the **Command Line**, or press **F11**.
- □ To step over the program and over function calls. Alternatively, you can enter **PSTEP** in the **Command Line**, or press **F10**.
- □ To step out of the current function. Alternatively, you can enter **OSTEP** in the **Command Line**, or press **Ctrl+F11**.

On-Chip Peripherals

There are a number of techniques you must know to create programs that can use the various on-chip peripherals and features of the 8051 family. Use the code examples provided here to get started working with the 8051.

There is no single standard set of on-chip peripherals for the 8051 family. Instead, 8051 chip vendors use a wide variety of on-chip peripherals to distinguish their parts from each other. The code examples demonstrate how to use the peripherals of a particular chip or family. Be aware that there are more configuration options available than are presented in this text.

Follow the links to the on-chip peripherals:

- □ **Header Files** use the include files to define peripheral registers of the device in use.
- □ **Startup Code** initializes the microcontroller and transfers control to the **main** function.
- □ Special Function Registers explains how to use Special Function Registers (SFRs).
- □ **Register Banks -** explains how to use Register Banks.
- □ **Interrupt Service Routines** lists the different interrupt variants on 8051 devices.
- □ **Interrupt Enable Registers** shows how to enable the interrupts.
- □ **Parallel Port I/O** explains how to use standard I/O ports.
- □ **Timers/Counters** explains standard timers and counters.
- □ Serial Interface explains the implementation of serial UART communication.

- □ Watchdog Timer use a watchdog timer to recover from hardware or software failures.
- □ **D/A Converter** convert a digital output voltage to an analog output value.
- □ **A/D Converter** convert an analog input voltage to a digital value.
- □ **Power Reduction Modes** put the device into IDLE or POWER DOWN mode.

Startup Code

Startup Code is executed immediately upon RESET of the target system and performs the following operations:

- Depending on the device variant, device specific features are configured.
- Clears data memory (optionally).
- Initializes the reentrant stack and re-entrant stack pointer (optionally).
- Initializes the 8051 hardware stack pointer.
- \Box Transfers control to the variable initialization code or to the main C function.

Differences Between µVision and C

A number of differences exist between ANSI C and the language subset to support features in user- and signal functions.

 \square µVision does not differentiate between uppercase and lowercase. The names of objects and control statements may be written in either uppercase or lowercase.

 \square µVision has no preprocessor. Preprocessor directives like #define, #include, and #ifdef are not supported.

 \square µVision does not support global declarations. Scalar variables must be declared within a function definition. You may define symbols with the DEFINE command and use them like you would use a global variable.

in μ Vision, variables may not be initialized when they are declared. Explicit assignment statements must be used to initialize variables.

 \square µVision functions only support scalar variable types. Structures, arrays, and pointers are not allowed. This applies to the function return type as well as the function parameters.

 \square µVision functions may only return scalar variable types. Pointers and structures may not be returned.

 \square µVision functions cannot be called recursively. During function execution, µVision recognizes recursive calls and aborts function execution if one is detected.

 \square µVision functions may only be invoked directly using the function name. Indirect function calls via pointers are not supported.

 \square µVision supports only the ANSI style for function declarations with a parameter list. The old K&R format is not supported. For example, the following ANSI style function is acceptable.

High Level Language – 'C' V/s Embedded C

C' is a well-structured, well defined and standardized general purpose programming language with extensive bit manipulation support.

□ 'C' offers a combination of the features of high level language and assembly and help sinhard ware access programming (system level programming) as well as business package developments (Application developments like payroll systems, banking applications etc).

The conventional 'C' language follows ANSI standard and it incorporates various library files for different operating systems.

 \Box A platform (Operating System) specific application, known as, compiler is used for the conversion of programs written in 'C' to the target processor (on which the OS is running) specific binary files.

Embedded C can be considered as a subset of conventional 'C' language.

Embedded C supports all 'C' instructions and incorporates a few target processor specific functions /instructions.

The standard ANSI 'C' library implementation is always tailored to the target processor /controller library files in Embedded C.

 \Box The implementation of target processor /controller specific functions /instructions depends upon the processor /controller as well as the supported cross-compiler for the particular Embedded C language.

 \Box A software program called 'Cross-compiler' is used for the conversion of programs written in Embedded C to target processor /controller specific instructions.

Procedure:-



Create new u vision project

	C:\Keil_v5\C51\Examples\HELLO\Hello.uvproj - µVision						
File Edit View Project Flash Debug Peri	pherals Tools SVCS Window Help						
	→ 作 作 作 作 人 作 作 / に / 後	🖄 🔽 🛛	a 🏘 🔍 🔍 🔹 🖉 🖉 📓	- 4			
🛞 🏥 🕮 🥔 🔜 🞇 Simulator	🗟 🐔 🖶 🔶 🐡 🌚						
Project 🕂 📮 🔟		Create Neu	v Project	×			
E Simulator		create iver	Tiojeer				
in in in in it is a source Group i	(€) ⇒ ↑ <mark>↓</mark> «	in syllabus (F:) → ece lab →	V C Search ece lab	Q			
	Organize 👻 New fo	older		≣ ▼ 🔞			
ABSTRACT.TX	E Pictures	^ Name	Date modified	Туре			
	📕 Videos	🎍 ece expts	29-Nov-14 3:29 PM	File folder			
	Homegroup						
	J Deeps						
	Local Disk (C:)						
	U DVD RW Drive (D						
	in syllabus (E)						
	not a syllabus (G:						
		v «					
	File name:						
	Save as type: Pro	oject Files (*.uvproj; *.uvprojx)		~			
	🔿 Hide Folders		Open	Cancel			
				ati I			
Build Output					a 🖬		
linking	2007 w				^		
<pre>Program Size: data=9.0 xdata=0 code "HELLO" - 0 Error(s), 0 Warning(s)</pre>	e=27						
<					~		
			Simulation		CAP NUM SCRL OVR R/W		
- 6 😭 😒 🛚	1 🔣 📐 🥘	🚝 🚳 📈			- 👬 🌠 🏮 🚍 📲 4 3:29 PM		

Select the folder (newly created) to save the project

1	μVision	- 0 ×
File Edit View Project Flash Debug Peripherals Too	is SVCS Window Help	
□ 🐸 🖬 🕷 🖉 🖉 🖓 🖉 🖓 🖉 🖓 🖉 🖓	18 18 (伊伊///////////////////////////////////	
Project 4 🛛	Create New Project	
	(c) (c) ↑ ↑ (i) × cce lab × ece expts v (c) Search ece expts p	
	Organize ▼ New folder 🛛 🕮 ▼ 🕡	
	Pictures Name Date modified Type Videos No items match your search.	
	n n n n n n n n n n n n n n n n n n n	
	P Deeps Local Disk (C:) ♥ DVD RW Drive (D) □ out of syllabus (E) □ in syllabus (F:)	
	not a syllabus (G;	
	File name expt1 v Save as type Project Files (*.uvproj; *.uvproj;) v	
	Hide Folders Save Cancel	
Bill Pu Strate		
bund Galpar		4 ×
		×
	CAP NUM S	CRL OVR R/W
📑 🥝 🚞 🔇 🖬 💟	- 10 K 🛈 📰 🔦	3:31 PM 29-Nov-14

Save the project

	F:\ece lab\ece expts\expt1.uvproj - µVision	- 🗗 🗡
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help		
□222日間 2 日間 9 0 1 + → 0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2		
🖉 🖾 🥔 🔤 🙀 Target 1 🔍 🔊 🚠 🗟 🖘 🗇 🎰		
Project 4 🖬		
	Select Device for Target 'Target 1'	
CPU Vendor: Armel Device: AT85C51 Toolete: C51 Search:	Use Extended Linker (LXS1) instead of BL51 Use Extended Assembler (AVS1) instead of AS1 Description: 1051 based Fully Static 24MHz CMOS controller with 32 L/O Lines, 12 Timers/Counters, 6 Interrupts/2 Phonty Levels, UART; 12 Timers/Counters, 6 Interrupts/2 Phonty Levels, UART; 12 Bytes On-chip RAM 12 Bytes On-chip RAM OK Cancel Help	
Build Output		
		* *
		~
<		
📑 🥝 🚆 😫 🖬 👿 💆 🛒		▲ 11 K 🔮 🚍 📲 🦿 3:31 PM 29-Nov-14

Select the vendor "Atmel" and device "AT89C51"



Addition of STARTUP.A51 to project folder



STARTUP.A51 is added



The program to print "welcome to ECE department" is written

W			F:\ece lab\ece expts	s\expt1.uvproj - µVisi	on				-	ð ×
File Edit View Project Flas	sh Debug Peripherals Tools	SVCS Window Help								
🗋 🗃 🖬 🖉 🛛 X 😘 🛍	1 1 1 + -> 1 1 12	<u>●●</u> ●律律/////////////////////////////////	<u>ð</u>	🗸 🗟 🍬 🞯 🖉 🖉	୦ 🔗 🚓 🖬	- 2				
🗇 🏔 🏙 🥔 🏭 🕱 🎇 Ta	irget 1 💌 🕺	45 ♦ ♡ @								
Project 🛛 🕂 🔟	Text4*	1		Sava Ac			×			▼ ×
🖃 🔁 Target 1	here after stack i			Dave AS	2					^
Source Group 1	void main (void) {	🔄 🏵 👻 🕇 🕌 « in	syllabus (F:) → ece lab → e	ece expts 🗸 🗸	Search ece expts		P			
	7.	Organize 🔻 New folde	r			8== -	0			
	Setup the serial p	Fictures ^	Name	D	ate modified	Type				
	#ifndef MONITOR51	Videos	event1 build log	20	-Nov-14 3-31 PM	HTM File				
	SCON = 0x50;		expt1	20	-Nov-14 3:31 PM	µVision4 Pro	ject			
	TMOD = 0x20; TH1 = 221;	K Homegroup	STARTUP.A51	29	-Jan-14 12:40 PM	A51 File				
	TR1 = 1;	1 Deens								
	fi = 1; #endif	Local Disk (C:)								
	7.	🕑 DVD RW Drive (D								
	Note that an embed	🥅 out of syllabus (E								
	there is no operat must loop and exec	in syllabus (F:)								
		inot a synabus (G:				_				
	<pre>while (1) { P1 ^= 0x01;</pre>	- ·	•							
	printf ("welco	File name: prg3.c					~			
	}	Save as type: All File	es (*.*)				~			
		Queren			Cauco	Cancel				
< >		Hide Folders			Jave	Cancel				*
E P <3 B {} F 0, T <	-									>
Build Output										Ф 🔀
										^
										3
<										>
					Simulation			L:43 C:39	CAP NUM SCRL	OVR R/W
- 🦲	VI 🛛 🖉		🗐 🐝 🖭	18				- 18 🔀	🕽 🖽 🛋 🖣	3:34 PM 29-Nov-14

The program is saved as prg3.c

W		F:\ece lab\ece expts\expt1.uvproj - µVision	- 🗇 🗙
File Edit Vie	ew Project Flash Debug Peripherals Tools SVCS	Window Help	
🗋 🖬 🖬 🕯	Ø X ⓑ ☎ ♥ ♥ ♥ ₽ ₿	律 律 准 版 🖄 🔍 🔍 🗟 🐢 🔍 🔷 🔿 🔗 🌏 🖬 🗸	
0 🕮 🖽 (🧼 🔜 🗮 Target 1 🛛 🔜 🌋 🛔 🖷	* * @	
Project	# 🔟 prg3.c		▼ ×
🖃 🚞 Target 1	21 here after stack in	itialization.	~
🖻 🔄 Sou	🔊 Options for Group 'Source Group 1' Alt+F7	·*/	
·····	Add New Item to Group 'Source Group 1'		
	Add Existing Files to Group 'Source Group 1'	t for 1200 baud at 16MHz.	
	Add Group	*/	
	Remove Group 'Source Group 1' and its Files	/* SCON: mode 1, 8-bit UART, enable rcvr */	
	Manage Project Items	/* TMOD: timer 1, mode 2, 8-bit reload */	
	Open File	/* TR1: timer 1 run */	
	Open List File	/* TI: set TI to send first char of UART */	
	Open Map File		
	Open Build Log	d program never exits (heraise	
	Rebuild all target files	g system to return to). It	
l i	Build target F7	te forever.	
	Iransiate rile	(i) Endial Car A - District and India 20	
		/* Toggle Pl.0 each time we print */ to ECE department\n"); /* Print "Hello World" */	
	Show include File Dependencies		
	45 / 46		
	47		
	DE DE		, ×
Build Output	er et al anti-		
build output			
0.2			*
Add Existing File	s to current Project Group	Simulation 1/43 C-39 CAP	NUM SCRI OVR RAW
			3:35 PM
			29-Nov-14

Prg3.c is to be added to Source Group1

8	F:\ece lab\ece expts\expt1.uvproj - µVision	- 🗇 🗙
File Edit View Project Flash Debug Peripherals Tools	SVCS Window Help	
D 😂 🗿 🖇 🖓 🖄 🖏 🖉 (↔ →) 🥐 🕅	4 1版 淳 非 川版 20 🖌 🔽 🔛 🔛 🔜 🔹 🔍	
🖉 🏥 🎬 🥔 🧱 🞇 🛛 Target 1 🛛 🗹 🔊 🕯	5 E 🗇 🗇 📾	
Project 🛛 📮 🔛 📄 prg3.c		▼ ×
Image 1 21 here after sta Image 2 22 void main (void Image 2 23 void main (void Image 2 24 Image 2 25 /=	k initialization.) (1 port for 1 port for 1 Add Files to Group 'Source Group 1'	<u>^</u>
29 SCON = 0.01 30 TMOD = 0.01 31 TMI = 2.22 32 TR1 = 1; 33 TI = 1; 34 fendif 35 36 /*	Look in: ↓ ecc expts	
42 P1 ^= 0x01 43 printf ("we 44) 45 } 46 47 48 Build Output	loome to ECE	×
Build Output		* •
3		
	Simulation	L:43 C:39 CAP_NUM_SCRL_OVR_R/Wt: 2:35 DM
		▲ 10 K (2) = ↓11 ▲ 3:35 PM 29-Nov-14

Select the program prg3



Now prg3.c is added to Source Group 1

	F:\ece lab\ece expts\expt1.uvproj - µVision	- 0 ×
File Edit View Project	Flash Debug Peripherals Tools SVCS Window Help	
🗋 😂 🖬 🕔 X 🖙	🛍 🤊 ♡│ ← →│ 🍢 恕 恕 改 (罪 罪 服 🖉 🔍 🔍 🗟 🖋 🔍 🌢 🖉 🔍 🔦	
🖉 🖽 📾 🥔 🔠 💥	🖞 Target 1 🔍 🔊 👗 🖶 🚸 🗇 🚳	
Project 🛛 📮 🗵	H prg3.c	▼ ×
Tropec Target 1 Grand Source Group 1 Grand Source Grand Source	<pre> Project Provide main (void) {</pre>	× ×
linking		*
Program Size: data=	30.1 xdata=0 code=1110	^
"expt1" - 0 Error(s), 0 Warning(s).	
<		>
	Simulation	L:43 C:39 CAP NUM SCRL OVR R/W .:
📑 🤅 [🔋 💽 🖬 🔣 🖉 🥵 🕿 🖎 🖬 😪	▲ 10 🔀 🔕 🚍1 4 3:36 PM 29-Nov-14

Build the target



Debugging the target

8	F:\ece lab\ece expts\expt1.uvproj - µVision	- 8 ×
File Edit View Project	Flash Debug Peripherals Tools SVCS Window Help	
🗋 🖬 🖬 🗿 🕺 🖓	‱ ? ? (← →) ? ? ? ? 推 // // // // // // // // // // // // /	
	Target 1 🔍 🔊 🛔 🗟 🚸 🥎 💩	
Project 🛛 📮 🚺	🖹 prg3.c	▼ ×
B Target 1 B Source Group 1 - B STARTUP.A51 B P Pr93.c	21 here after stack initialization. 22	
	<	, [*]
Build Output	,	Q 🛛
linking		^
Program Size: data=3 "expt1" - 0 Error(s)	0.1 xdata=0 code=1110 , 0 Warning(s).	
 C 		>
	Simulation	L:23 C:19 CAP NUM SCRL OVR R/W
- 🦲 🔚	i 🥵 🖬 🔣 丛 🥹 🐺 🛸 🔤 🚱 🚛 🔛	★ 10 10 10 10 10 10 10 10 10 10 10 10 10

New window evaluation mode appeared. Press ok

F:\ece lab\ece expts\	expt1.uvproj - µVision	- 🗇 🗙
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help		
□ 😂 🛃 🐉 & 印 🚵 🤊 (*) (* →) (*) (*) (*) (*) (*) (#) (#) (#) (#) (#) (#) (#) (#) (#) (#	🗟 🖉 🧕 📀 🔗 🍓 🖬 🗉 🔍	
👬 🖳 🔕 [73 [77 - 17] 🔶 🗖 🔯 📾 🚍 🖓 • 🔲 • 🔯 • 🗰 • 🗰 • 🗮 • 🗩 •		
Registers Run (F5) assembly		д 🚺
Register Start code execution 23: void main (void) {		^
Pregs 24: 25: /*		
rl 0x00 c 26: Setup the serial port for 1200 haud at 16MHz.		× *
		- ×
r5 0x00 25 /*		
r7 0x00 26 Setup the serial port for 1200 baud at 16MHz.		
Sys 28 #ifndef MONITOR51		
a 0x00 29 SCON = 0x50; /* SCON: mode 1, 8-bit UAR	f, enable rcvr */	
sp 0/21 30 IMOD = 0x20; /* IMOD: timer 1, mode 31 TH1 = 221; /* TH1: reload value	2, 8-Dit reload */ for 1200 baud 0 16MHz */	
sp_max 0x21 32 TR1 = 1; /* TR1: timer 1 run	*/	
PC \$ C:0 33 TI = 1; /* TI: set TI to sen	1 first char of UART */	
states 389 35		
sec 0.0 36 /*		
38 there is no operating system to return to). It		
39 must loop and execute forever.		
40*/ 41 while (1) {		0
E Project Registers		• • • • • • • • • • • • • • • • • • •
Command P 🛛	Call Stack + Locals	đ 🔀
Running with Code Size Limit: 2K	Name Location/Value Type	
Load "F:\\ece lab\\ece expts\\expt1"	MAIN C:0x0C11	
x		
>	1	
ASM ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess	Call Stack + Locals Memory 1	
Start code execution	Simulation t1: 0.00019450 sec L:29 C:1	CAP NUM SCRL OVR R/W
📑 🤅 🗮 関 🔣 💆 🐖 🗞 🌍	- 11 🛚 🖉	3:44 PM 29-Nov-14

Run the program

W		F:\ece lab\ece expts\expt1.uvproj - µVision	- 🗇 🗙
File Edit	View Project Flash Debug	Peripherals Tools SVCS Window Help	
i 🗋 💕 🕻	🗸 Status Bar	→ ● ● ● ● ● 注 評 // // 図	
AST EL	Toolbars		
Registers	Project Window		‡ 🔀
Register	🚷 Books Window	d main (void) (^
E Regs	<pre>{} Functions Window</pre>		
- r0	(). Templates Window	up the serial nort for 1200 haud at 16MHz.	*
r2	Source Browser Window		>
r3	Build Output Window	STARTUP.AS1	▼ ×
r4	😹 Error List Window		^
r6	Find In Files Window		
r7	Command Window	*/	
⊟ Sys	Disassambly Window	ndef MONITOR51	
b	Cumb ala Min davu	SCON = 0x50; /* SCON: mode 1, 8-bit UART, enable rovr */	
sp	Symbols window	TH1 = 221; /* TH1; relad value for 1200 baud @ 16MHz */	
sp_r	Registers Window	TR1 = 1; /* TR1: timer 1 run */	
PC	Call Stack Window	TI = 1; /* TI: set TI to send first char of UART */	
state	Watch Windows	> hit	
sec	Memory Windows	P	
⊡ psw	Serial Windows	UART#1 ram never exits (because	
	Analysis Windows	 UART#2 In to return to). It ter. 	
	Trace	• WART#3	
	System Viewer	Debug (printf) Viewer	¥
te Project	🔆 Toolbox Window		>
Command	Periodic Window Update	4 II Call Stack + Locals	P 🔀
Running	WIGH COUC DIEC DIRIC. 21	Name Location/Value Type	
Load "F:	<pre>//ece lab//ece expts//e:</pre>	◆ MAIN C:0x0C11	
ASM ASSI	GN BreakDisable BreakEna	able BreakKill BreakList BreakSet BreakAccess	
Show or hide	e the Serial 1	Simulation t1: 640.86481200 sec L:29 C:1	CAP NUM SCRL OVR R/W
			pe (Not Responding) 3:47 PM
		💘 🚺 🚰 🤍 🐝 💛 🔊 Skype Tools Help 🔹 🛚 🗞	29-Nov-14

Selecting for UART#1 from serial windows

F:\ece lab\ece expts\expt1.uvproj - µVision	– 🗇 🗙
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help	
	a 🕅
Kegistes + Dissembly Ussembly ()	¥ 🕅
regser value 24:	
r0 0x00 25: /*-	~
	>
	▼ ×
-r4 0x0 24	^
- 6 0x00 25/*	
-77 0x00 27*/	
a 0x00 28 \$ifndef MONITORS1	
b 0x00 30 TMOD = 0x20; /* TMOD: timer 1, mode 2, 8-bit reload */	
dpt 0.000 31 TI = 1; //* TI: set TI to send first char of UART */	
FUS CLARU 34 fendif	
8 psw 0x00 37 Note that an embedded program never exits (because	
38 there is no operating system to return to). It 39 must loop and execute forever.	
40*/	
41 while (1) {	×
uurrojet je Registers	
Command # 2 UARt=1	4 🚺
Kaming with order first finite in the second of the second to ECE department Load "Filesce expts/lexpt1"	^
welcome to ECE department	
welcome to ECE department	
	×
2 Dia Strabbieshie Breskfishie Breskfill Breskfist Breskfer Breskfer Breskfere Breskfere Breskfere Bill dark alsole Wildotst at the Breskfish Bres	<i></i>
	M SCRI OVR RAW
	aspondine) 3:48 PM
Skype Tools Help 🔹 🚺 🚰 🥹 🐯 🦃 🖏 🤚 💯 Tools Help 🔹 🖬 🖬 🖬 🖬	29-Nov-14

Check the output at UART#1 window

K F:\ece lab\ece	expts\expt1.uvproj - µVision	- 🗇 🗙
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help		
🗋 😂 🛃 🐉 🛝 🙈 🤌 🐢 📔 Interrupt ! 🛝 🛝 🔅 淳 //////////////////////////////////	🗹 🔜 🏴 📵 🔺 🔿 🔗 🍓 🖬 🗉 🔦	
22 [[2] [] [] [] [] [] [] [] [
Registers A Disassembly		1 X
Popular Volum 23: Timer Port 2		
Progister Value 24: Port 3		
r0 0x00 25: /*		*
r1 0x00 <		>
r3 0x00 I prg3.c I STARTUP.A51		▼ ×
r4 0x00 24		^
r6 0x00 25 /*		
r7 0x00 27*	1	
28 #ifndef MONITOR51		
b 0x00 30 TMOD = 0x20; /* SCON: mode 1, 8-D1 30 TMOD = 0x20; /* TMOD: timer 1.	mode 2. 8-bit reload */	
sp (b21 31 TH1 = 221; /* TH1: reload v	alue for 1200 baud 0 16MHz */	
Spinex Occi 32 TR1 = 1; /* TR1: timer 1 dptr 0x000/ 32 TT = 1; /* TL: cost TL: timer 1	run */	
PC \$ C:0x00 34 #endif	o send filist char of oaki "/	
states 389 35		
B psw 0x00 37 Note that an embedded program never exits (becaus	- e	
38 there is no operating system to return to). It		
39 must loop and execute forever.	1	
41 while (1) {		~
E Project Registers		>
Command	4 🛛 UART#1	4 X
Running with Code Size Limit: 2K	^ welcome to ECE department	^
Load "F://ece lab//ece expts//expt1"	welcome to ECE department	
	welcome to ECE department	
¢	3	~
>		>
ASM ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess	Call Stack + Locals UART #1 Memory 1	
	Simulation t1: 2676.88144500 sec L:29 C:1	CAP NUM SCRL OVR R/W
	- 78 🐹 🖲	
		29-Nov-14

Select Port1 from i/o ports in peripherals

	F:\ece lab\ece expts\expt1.uvproj - µVision	- 8 ×
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help		
🗋 😂 🛃 🐉 👃 🛍 👛 🥙 🐑 🖛 🔶 🥐 🦉 🗑 🖉 🎼 🎼 🥙	🖌 🔜 🥐 🕘 🔶 🔗 🍓 🖬 - 🔦	
🗱 🗏 🔕 (B) (B) (B) (D) (D) (D) (D) (D) (D) (D) (D) (D) (D	■ • 2 2 •	
Registers 4 🔟 Disassembly		÷ 🗵
Register Value 23: void main (void) {		^
Regs 25: /*		
r1 0x00 c 26: Setup the serial port for 1200 haud	at 16MHz.	×
		• X
r4 0x00 24		
15 0x00 25 /*		
r7 0x00 26 Setup the serial port for 1200 baud of 27	at 16MHz.	
Sys 28 #ifndef MONITOR51		
29 SCON = 0x50; /* SCON: b 0x00 30 TMOD = 0x20; /* T	mode 1, 8-bit UART, enable rcvr */ MOD: timer 1, mode 2, 8-bit reload */ Parallel Port 1	
sp 0x21 31 TH1 = 221; /* TH	H1: reload value for 1200 baud @ 16MHz */ Pott 1	
32 TR1 = 1; /* TI dptr 0x0001 22 TT = 1. /* T	R1: timer 1 run */ P1: DxFE / Bits 0	
PC \$ C:0x00 34 #endif		
states 389 35		
psw 0x00 36/*	exits (because	
38 there is no operating system to retu	rn to). It	-
39 must loop and execute forever.	*/	
41 while (1) {		Ŷ
E Project Registers		>
Command	🗭 🔯 UART#1	4 X
Running with Code Size Limit: 2K	<pre>welcome to ECE department</pre>	^
Load "r://ece lab//ece exprs//expt1"	welcome to ECE department welcome to ECE department	
	welcome to ECE department	
C. C	30 ⁻¹	~
>	<	>
ASM ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet Br	reakAccess Call Stack + Locals 📴 UART #1 🛄 Memory 1	
	Simulation t1: 2809.38467500 sec L:29 C:1	CAP NUM SCRL OVR R/W
	- 10 🕅	3:52 PM
		29-1400-14

Port1 window is displayed with D0 as 0

1	F:\ece lab\ece expts\expt1.uvproj - µVision	- 0 ×
File Edit View Project	t Flash Debug Peripherals Tools SVCS Window Help	
🗋 📬 🖬 🎒 🐰 🖻	□ 逸 タ ♡ ← → ♥ 改 改 存 詳 順 版 ● ○ ◇ 会 回・ ≪	
👬 🗉 🥝 🔁 🕅		
Registers 🛛 🛱 🗧	Disassembly	a 🔀
Register Value	23: void main (void) (^
E Regs	25: /*	
r1 0x00	26: Setum the serial nort for 1200 haud at 16MHz.	× *
	Presente Personales	- *
r4 0x00		
r5 0x00	25 /*	^
r6 0x00	26 Setup the serial port for 1200 baud at 16MHz.	
🖃 Sys	27 28 #1fndef MONITOR51	
a 0x00	29 SCON = 0x50; /* SCON: mode 1, 8-bit UART, enable rovr */ Barallal Bort 1 X	
sp 0x21	30 TMOD = 0x20; /* TMOD: timer 1, mode 2, 8-bit reload */ relation of (
sp_max 0x21	32 TR1 = 1; /* TR1: timer 1 run */ P1 her 1 run /* P1 her 1 run /*	
PC \$ C:0x0	33 TI = 1; /* TI: set TI to send first char of UART */	
states 389		
E psw 0x00	36/*	
	38 there is no operating system to return to). It	
	39 must loop and execute forever.	
	40	~
E Project Registers	<	>
Command	₽ 🔯 UART≠1	4
Running with Code S	Size Limit: 2K // welcome to ECE department	^
road .t://ece lab//	Vede expls/vexpli" welcome to ECE department	
	welcome to ECE department	
<	2	-
>	<	>
ASM ASSIGN BreakDis	sable BreakEnable BreakKill BreakList BreakSet BreakAccess 🔯 UART+1 🗐 Memory 1	
	Simulation t1: 2936.28034000 sec 1:29	31 CAP NUM SCRL OVR R/W 35
e_ 1	🚔 🐨 🔣 🚱 🜍	3:52 PM
		29-NOV-14

Port1 window is displayed with D0 as 0



Stop debugging process



After debugging, window appears in this format

Experiment-2

Aim of the Experiment: - Write a C Program to print hello world.

Software required: - Keil u vision 5

Theory: - This experiment aims to print "hello world" using the software keil u vision 5. The program is by default present after installation of this software. Firstly the header files REG52.H, stdio.h are declared for the intended 8051. Programming for debugging with Monitor-51 made. Now the main function starts. The serial port for 1200 baud at 16MHz is set up. An embedded program never exits (because there is no operating system to return to). It must loop and execute forever. So an infinite loop is made that Toggle P1.0 each time we print "Hello World".

Program:-

```
#include <REG52.H>
#include <stdio.h>
#ifdef MONITOR51
char code reserve [3] _at_ 0x23;
#endif
void main (void)
#ifndef MONITOR51
  SCON = 0x50;
  TMOD |= 0x20;
  TH1 = 221;
  TR1 = 1;
  TI = 1;
#endif
 While (1) {
  P1 ^{=} 0x01:
  printf ("Hello World\n");
 }
}
```

Procedure:-

As said the program for hello world is by default present after the installation of the keil u vision software, the program is loaded by following certain steps which is as follows:-



Select open project from project

W		C:\Keil_v5\C51\Example	s\Hello\Hello.uvproj - µVision		- 6	l ×
File Edit View Project Flash Debug	Peripherals Tools SVCS Window Hel	p				
0 00 A B A B B 0 0 0	(⇒) や た た た () 律 津 /// /		🖂 🔊 🖉 🌒 🖉 🖉 🖉	- 3		
🔗 🎬 🥔 🥔 🗒 🙀 Simulator	🗹 🐔 着 着 🧇 🍥			10		
Project 📮 🗵		C-1-	1 D - 1 - 1 51			
🖃 🚵 Simulator	N	Sele	ct Project File			
E Source Group 1	€ ∋ ד ↑ 🚣	▶ Deeps ▶ Local Disk (C:)	✓ C Search Local D)isk (C;) 🔎 🔎		
E Cocumentation	Organize 👻 New	folder)= • 🔟 🕜		
ABSTRACT.TXT	Documents	Name	Date modified	Туре	^	
	🌙 Music	emu8086	12-Nov-14 10:40 A	File folder		
	📔 Pictures	📗 Intel	22-Jan-14 7:19 PM	File folder		
	🛃 Videos	📗 Keil	20-Sep-14 2:03 PM	File folder		
		🍌 Keil_v5	29-Nov-14 2:02 PM	1 File folder		
	🔣 Homegroup	🍌 MSOCache	01-Sep-14 8:08 PM	File folder		
		🍌 PerfLogs	25-Jul-12 11:33 PM	File folder		
	Peeps	🍌 Program Files	16-Oct-14 5:38 AM	File folder		
	Local Disk (C:)	🎉 Program Files (x86)	29-Nov-14 9:52 Alv	1 File folder		
	💓 DVD RW Drive (D 🔋 ProgramData	28-Nov-14 4:45 PN	1 File folder		
	🕞 out of syllabus	(E 🔒 sources	22-Jan-14 6:38 PM	File folder		
	in syllabus (F:)	📕 Update	28-Nov-14 1:10 PN	1 File folder		
	👝 not a syllabus (i	users	01-May-14 10:10 P.	File folder	~	
		~ <		>	-	
		File name:	✓ Project Files (*.uvprojx; *.uvpro v		
			Open	Cancel		
и П Рг 🥎 Во () Fu (), Те						
			Simulatio	on	CAP NUM SCRI	OVR R /W

Select Keil_v5 present in Local Disk (C:)

8	C:\	Keil_v5\C51\Examples\Hello\Hello.uvpro	j - μVision	- 🗗 🗡
File Edit View Project Flash Debug Peripherals Too	ls SVCS Window Help			
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	周氏 津津川川 後 盛	ð 🔽 🗟 🖉 🔍) O 🔗 🍓 🔲 🔍	
🧼 🏭 🖉 🥥 🧱 🙀 Simulator 💽 🔊	📥 🕾 🧇 🃾			
Project 4 2 Simulator Conce Group 1	 (€) → ↑ (↓) > Dee 	Select Project File	Search Keil_v5	•
	Organize 🔻 New folder		iii 🗸 🔟 🔞	
ABSTRACT.TXT	Documents ^	Name	Date modified Type	
	Music Pictures	길 C51	29-Nov-14 2:01 PM File folder	
	Videos	👪 UV4	29-Nov-14 2:01 PM File folder	
	Homegroup Deeps Local Disk (C:) DVD RW Drive (D out of syllabus (E:) in syllabus (F:) not a syllabus (G:	¢		5
	File nar	me:	Project Files (*.uvprojx; *.uvpro v Open Cancel	

Next select the folder C51

W	C:	Keil_v5\C51\Examples\Hello	Hello.uvproj - µVision		- 🗗 🗙
File Edit View Project Flash Debug Peripherals	Tools SVCS Window Help				
9 (+ +) ♥ & 5 @ # 9 € + + ₱	周月月(二年) にほう	9-1 🛛 🔜	🍂 🍳 🔹 🔿 🔗 🍓 💼 🔹	2	
🗇 🏥 🕮 🥔 🔜 🛛 🞇 🛛 Simulator 🔍 🥥	🖇 着 🗣 🗇 🍏				
Project R Simulator R Simulato	Image: Second	Select Project cal Disk (C:) > Keil_v5 > C51 Name ASM BIN Examples FlashMon Hip ISD51 LB MON51 MON51 MON51 MON51 MON51 MON52 M	ct File C Search C51 Date modified 29-Nov-14 2:01 PM 29-Nov-14 2:01 PM	> > <	
Bo. Ofu. O	Filena	<	V Project Files (*.uv Open	> projç *uvpro v Cancel	

Next select the folder examples

W	C	:\Keil_v5\C51\Examples\Hello	\Hello.uvproj - µVision		- 🗗 🗡
File Edit View Project Flash Debug Peripherals T	ools SVCS Window Help				
	医医腹 律律 // // []	🙆 🔽 🔜	🕫 🍭 🔹 o 🔗 🍓 🗖 🔹	2	
🔗 🍱 🍘 🥥 🔜 🕍 Simulator 🔍 🕵	A B & Ø M				
Project 🌐 🖾		Colord Decis	- File	×	
E- 🔁 Simulator		Select Proje	ct File		
E Source Group 1	💽 🏵 👻 🕈 퉬 « Ke	eil_v5 → C51 → Examples	✓ C Search Examples	Q	
Documentation	Organize 🔻 New folde	er	8==	- 11 0	
ABSTRACT.TXT	Documents	Name	Date modified To		
	A Music	Notifie .	Date mounied	ibe -	
	E Pictures	ADI 83x	29-Nov-14 2:01 PM Fil	le folder	
	Videos	ASM	29-Nov-14 2:01 PM Fil	le folder	
		Benchmarks	29-Nov-14 2:01 PM Fil	le folder	
	Homegroup	BLINKY	29-Nov-14 2:01 PM Fil	le folder	
		CodeBanking	29-Nov-14 2:01 PM Fil	le folder	
	Deeps	CSAMPLE	29-Nov-14 2:01 PM Fil	le folder	
	Local Disk (C:)	Dallas 390	29-Nov-14 2:01 PM Fil	le folder	
	DVD RW Drive (D	BarMemory	29-Nov-14 2:01 PM Fil	le folder	
	👝 out of syllabus (E	HELLO	29-Nov-14 2:03 PM Fil	le folder	
	in syllabus (F:)	Infineon C517	29-Nov-14 2:01 PM Fil	le folder	
	not a syllabus (G:	Infineon TLE983x	29-Nov-14 2:01 PM Fil	le folder	
		Infineon XC822	29-Nov-14 2:01 PM Fil	le folder	
		•			
	File n	ame:	 Project Files (*.uvpro 	ojx; *.uvpro ∨	
			Open	Cancel	
wirin wybon Gruniosien			Simulation		CAP NUM SCRL OVR R /W

Next select the folder HELLO



Now select and open uvision4 project file



After selecting the window appears in this format (HELLO.C is added to Source Group 1)



Program of HELLO.C



Build the target



Displaying 0 errors and 0 warnings

W	C:\Keil_v5\C51\Examp	es\HELLO\Hello.uvproj - μVision	- 8 ×
File Edit View Project	Flash Debug Peripherals Tools SVCS Window Help		
🗋 📬 🖬 🗿 🕺 🖓	鑑 ウ や キ ⇒ 作 良 良 後 律 津 准 版 20	🖳 🔊 🔕 🔸 🖉 🖉 🖳 🖳	
	Simulator 🔍 🔊 📥 🖶 🧇 🛞	Start/Stop Debug Session (Ctrl+F5)	
Project 🛛 📮 🗵	HELLO.C	Enter or leave a debug session	▼ ×
Simulator	<pre>min.c.c Tempide 155-000 Extl Software. Inc. "min.c.c Tempide 1550:15. /* special function register delarations '/ finitude 02501.15. /* special function register delarations '/ finitude 02501.15. /* protograps delarations '/ finitude codin.20. /* protograps delarations of '/ finitude codin.20. /* protograps delarations of '/ finitude codin convert [0] _st_ 622: /* Special intercept if '/ for main C function. Program execution marks there after reach initialization. */ for main C function. Program execution marks there after reach initialization. */ for main C function. Program execution marks there after reach initialization. */ for main C function. Program execution marks there after reach initialization. */ for main C function. Program execution marks for marks for marks for marks for main C function. Program execution marks for marks for marks for marks for marks for marks for marks for marks for marks for marks for ma</pre>		×
Build Output			4 ×
linking Program Size: data=	30.1 xdata=0 code=1096		^
"HELLO" - 0 Error(s), 0 Warning(s).		
<			>
Enter or leave a debug sessior		Simulation	L:29 C:1 CAP NUM SCRL OVR R/W:
📑 健 [🗧 💽 🛃 💹 💹 🗐 🛞		▲ 11 K 20 =

Debugging the target



New window evaluation mode appeared. Press ok

W	C:\Keil_v5\C51\Examp	oles\HF	LLO\Hello.uvproj	- µVision			- 🗇 🗙
File Edit View Project	Flash Debug Peripherals Tools SVCS Window Help						
🗋 🖬 🚰 🛃 🖉 🕺 🖓	● 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	~	🗟 🥐 🔯 🧕	ି 🔗 🚓 🖬 🖣			
👫 🗏 🔘 🔁 🖓 🕼	▶ 10 🔶 🗖 👰 🙀 🚍 🖉 🖉 🖉 🖉 🖉 🖉 🖉 🖉 🔶 🕑 🖓 🗸						
Registers 🛛 🗛 🔀	Disassembly						Q 🔀
Register Value	23: void main (void) {						^
- Regs	24: 25: /*	-					
r1 0x00	26: Setup the serial port for 1200 haud at 16MHz.						, ×
	Provide a la construcción de la						
r3 0x00							· · ·
r5 0x00	23 🖸 void main (void) [^
r6 0x00	25 - /*						
E Sys	26 Setup the serial port for 1200 baud at 10MHs. 27*/						
a 0x00	28 [] SINGET MUMITURS1 29 SCON = 0x80; /* SCON: mode 1, S-bit UART, enable rowr */						
sp 0x21	30 TMOD = 0x20; /* TMOD: timer 1, mode 2, 8-bit reload */ 31 TH1 = 221; /* TH1: reload value for 1200 baud 0 16MMs */						
sp_max 0x21	32 TR1 = 1; /* TR1: timer 1 run */ 33 TI = 1; /* TI: set TI to send first char of UART */						
aur1 0x00	34 #endif 35 -						
E dptr 0x000	36 (/*						
states 389	38 there is no operating system to return to). It 39 must loop and execute forever.						
± psw 0x00	40						
22	42 P1 -= Coliz /+ Toggle P1.0 each time we print +/						
	44 -)						
E Project E Registers	48 J						, [*]
Command		д 🗙	Call Stack + Locals				Q 🔀
Running with Code S:	ize Limit: 2K	^	Name	Location/Value	Туре		
Load "C:\\Keil_v5\\(CSI//Examples//HELLO//HELLO"		MAIN	C:0x0C11			
<		>					
>							
ASM ASSIGN BreakDisa	able BreakEnable BreakKill BreakList BreakSet BreakAccess		Call Stack + Local	Memory 1			
				Simulation	t1: 0.00014145 sec	L:29 C:1 CAP NU	M SCRL OVR R/W

Now the window appears in this format

C:\Keil_v5\C51\Exam	ples\H	ELLO\Hello.uvproj -	μVision			- 🗗 🗙
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help		11111 h a				
□ 📽 📓 🗿 👗 □ 10 ← → 🥐 25 15 16 律 律 准 版 20	~	🗟 🍖 🔯 🤞 🔇	े 🔗 🚓 🖬 ।	2		
新 単 ③ P P O 10 + I ● I 風 国 三 会 源 · I · ス · E · E · N · N ·						
Registers 🖳 Run (F5) assembly						ф 🔀
Register Value 23: void main (void) {						^
Regs 24:						
n0 0x00 26: Setup the serial port for 1200 haud at 16MHz.						~
						>
r3 0x00 H HELLO.C						▼ ×
r4 0x00 22 L						
r5 0x00 23 □ void main (void) (
7 0-00 28 ⊕ /*						
Svs 27*/						
a OxOO N 28 B sindef MONITORS1						
b Dx00 30 TMOD = 0x20; /* TMOD: timer 1, mode 2, 8-bit reload */						
Sp (X2) 31 TH1 = 221; /* TH1: reload value for 1200 haud § 16Mfs */ 32 TR1 = 1; /* TR1: timer 1 run */						
PC \$ C:0x0C 33 TI = 1; /* TI: set TI to send first char of UAR */						
auxr1 0x00 35 -						
dptr Ox000/ 36 /*						
states 389 38 there is no operating system to return to). It						
Bec 0.0001 as an execute coreter. Drop and execute coreter. //						
41 🖯 while (1) (42 P1 ~= 0x01; /* Tonule P1.0 such time we print */						
42 printf ("Hello World\n"); /* Print "Hello World" */						10
49 - 1 45 1						~
E Project Registers						>
Command	ą 🔀	Call Stack + Locals				4 X
Running with Code Size Limit: 2K	N	Name	Location/Value	Туре		
Load "C:\\Keil_v5\\C51\\Examples\\HELLO\\HELLO"		MAIN	C:0x0C11			
	1					
C	2					
>			Correct Correc			
ASM ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess		Call Stack + Locals	Memory 1			
Start code execution	_		Simulation	t1: 0.00014145 sec	L:29 C:1	CAP NUM SCRL OVR R/W
					- 10 K 🕻	3:00 PM

Run the program

W				C:\Keil_v5\C51\Examples\HE	LLO\Hello.uvproj	i - μVision			- 🗗 🗙
File Edit	View	w Project Flash Debug	Peripherals Tools SVCS Windo	w Help					
🗋 📬		Status Bar	→ P B B B 律律	= //= //= 🖄 💌	🗟 🅐 🔯 🧕	ି 🔗 🍓 🔳 🖣	8		
		Toolbars		- 🔳 - 🔟 - 📓 - 👷 -					
Registers	III	Project Window							a 🛛
Begister	-	Books Window	d main (void) {						^
- Regs	{}	Functions Window							
rO	0.	Templates Window	un the serial nort for	1200 baud at 16MHz.					*
r2	-	Source Browser Window							>
r3		Build Output Window							▼ ×
r4	×	Error List Window	n (void) (^/					^
r6		Find In Files Window							
r7		Command Window	e serial port for 1200 baud at 1600%						
a	Ø	Disassembly Window	MONITOR51						
ь		Symbols Window	= 0x20; /* SCON: mode 1;	ser 1, mode 2, 8-bit reload */					
sp so	. ==	Registers Window	= 221; /* TH1: rel = 1; /* TR1: tis	er 1 run */					
PC		Call Stack Window	= 1; /* TI: set	TI to send first char of UART */					
au t doi	xr tr	Watch Windows	•						
sta	ste	Memory Windows	t an embedded program never exits (k no operating system to return to).	_It					
sei	c	Serial Windows	• UART #1	*/					
ter par	w	Analysis Windows	UART #2	ab time we wrint hi					
		Trace	UART #3	llo World" */					
		System Viewer	🕨 🛃 Debug (printf) Viewer						×
E Project	2	Toolbox Window							>
Command	V	Periodic Window Update		Ф <mark>Х</mark>	Call Stack + Locals				4 🔀
Running	WIDE	ail w5\\C51\\Evample	AVALUATION HELLO	^	Name	Location/Value	Туре		
Doud C.		(COT/(EXMIPTC			MAIN	C:0x0C11			
				~					
<				>					
>									
ASM ASSI	IGN E	BreakDisable BreakEn	able BreakKill BreakList	BreakSet BreakAccess	Call Stack + Loca	Is Memory 1			
Show or his	de the	Serial 1		The The Second S		Simulation	t1: 27.33349164 sec	L:29 C:1	CAP NUM SCRL OVR R/W "
	6	2 🚞 😒						- 10 🐹	8 🖽 🔳 🔹 3:01 PM
	C.				A Real Property in		And Personnel	1 (1 () () () () () () () () ()	29-Nov-14

Selecting for UART#1 from serial windows

C:\Keil_v5\C51\Examples	HELLO\Hello.uvproj - µVision	- 🗖 🗙
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help		
□ 22 22 22 22 22 22 22 22 22 22 22 22 22	v 🗟 🎢 📵 🗕 🔿 🔗 🍓 💼 - 🔦	
Registers 🛛 📮 🔟 Disassembly		# 🗵
Register Value 23: void main (void) {		^
Regs 25: /*		
r1 0x00 26: Setup the serial port for 1200 baud at 16MHz.		, Y
r2 0x00		*
r3 Dx00		▼ ×
r5 0x00 22 □		^
r6 0x00 24 25 /*		
r7 OxOO 26 Setup the serial port for 1200 haud at 1606fs.		
a Dx00 28 Bifndef MONITOR51		
b 0x00 20 TNOD = 0x20; /* TNOD: timer 1, mode 2, 8-bit reload */		
sp (b/2) 21 THI = 221; /* THI: reload value for 1200 baud @ 16MEs */		_
PC \$ C:DxOC 22 TI = 1; /* TI: set TI to send first char of UART */		
auxr1 0x00 as -		
states 389 27 Note that an embedded program never exits (because		
sec 0.0001 29 must loop and execute forever.		
42 P1 ^= 0x01; /* Toggle P1.0 each time we print */ 43 printf ("Hello World\n"); /* Print "Hello World" */		
44 - 1		U
E Project E Registers		> ⁷¹
Command 4	VART#1	a 🛛
Running with Code Size Limit: 2K	Mello World	^
Load "C://Kell_V5//C51//Examples//HELLO//HELLO"	Hello World	
	Hello World	
>		2
ASM ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess	Call Stack + Locals UART #1 Memory 1	
	Simulation t1: 52.24431600 sec L:29 C:1	CAP NUM SCRL OVR R/W

Check the output at UART#1 window

C:\Keil_v5\C51\Examples	,HELLO\Hello.uvproj - μVision 🛛 🚽 🗖 🔜
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help	
🗋 😂 🛃 🖇 🖏 🧑 🕲 🕴 Interrupt ! 🗽 🤨 注 川浜 🙆	v 🗟 🏘 🕘 🔶 🔗 🍓 🖬 - 🔦
않 및 🙆 관 관 관 관 🚽 🕞 //O-Ports 🔸 Port 0 🔽 👼 - 🖼 - 🗰 - 🐲 -	
Registers # IDisassembly Serial Port 1	a
Register Value 23: Timer Port 2	
Port 3	
r0 0x00 25: /*	
	>
r3 Dx00 HELLO.C	
r4 0x00 22 L	
r7 0x00 26 Setup the serial port for 1200 baud at 16MSHs.	
a 0x00 28 d #ifndef MONITORS1	
b 0x00 29 SCON = 0x50; /* SCON: mode 1, 8-bit UART, enable rovr */ 20 TMOD = 0x20; /* TMOD: timer 1, mode 2, 8-bit reload */	
sp 0x21 21 THI = 221: /* THI: reload value for 1200 haud § 1965: */ 122 TRI = 1; /* TRI: timer 1 run */	
PC \$ C:DxOC 24 and 54 and 54 and 55 and 56 a	
E dptr 0x0001 36 ⊕ /*	
states 369 28 there is no operating system to return to). It	
E psw 0x00 40	
42 P1 ^m OxOls /* Toggle P1.0 each time we print */	
43 printf ("Hello World\n"); /* Print "Hello World" */ 44 -)	
45)	
Command	UADT #1
Running with Code Size Limit: 2K	
Load "C:\\Keil_v5\\C51\\Examples\\HELLO\\HELLO"	Hello World
	Hello World
	Wello World
¢>	
>	>
ASM ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess	Call Stack + Locals UART #1 Memory 1
	Simulation t1: 175.62241418 sec L:29 C:1 CAP NUM SCRL OVR R/W
- & 🛤 🛯 💵 🖉 😓 🧠	- 🛍 🔀 🏮 🖬 📲 🦉 29-Nov-14

Select Port1 from i/o ports in peripherals

	C:\Keil_v5\C51\Examples\H	ELLO\Hello.uvproj - µVision – 🗇	×
File Edit View Project	Flash Debug Peripherals Tools SVCS Window Help		
🗋 📴 🖬 🖉 🕺 X 🖻	22 ウ ひ キ → や な な な 注 注 29 - → や な な な 注 注 29 - → や な な な な 注 注 20 - → - \to - \to	🗟 « 🖉 🍳 🔸 🔿 🚓 💼 📲 🔦	
👫 🖹 🕹 🖪 🗗 C	P 10 👂 🖸 💁 🖷 🖧 🚚 • 🔲 • 👰 • 📾 • 💷 • 📓 • 🔗 •		
Registers 📮 🗵	Disassembly		a 🛛
Register Value	23: void main (void) {		^
E Regs	25: /*		
r1 0x00	26: Setup the serial port for 1200 haud at 16MHz.		, ×
r2 0x00	Purpos		
r4 0x00			
r5 0x00	22 E void main (void) [^
r6 0x00	25 0 /*		
⊟ Sys	27 27 27	Parallel Port 1	
a 0x00	29 SCON = 0x80; /* SCON: mode 1, S-bit UART, enable rovr */	Port 1	
sp 0x21	30 TMOD = 02207 /* TMOD: timer 1, mode 2, 8-bit reload */ 31 TH1 = 2217 /* TH1: reload value for 1200 baud @ 10MDs */		
sp_max 0x21	32 TRI = 1/ /* TRI: timer 1 run */ 33 TI = 1/ /* TI: set TI to send first char of UART */		
aur1 0x00	34 #endif 35 -		
dptr 0x000	(36 白 /* 37 Note that an embedded program never exits (because		
states 389 sec 0.0001	38 there is no operating system to return to). It 39 must loop and execute forever.		
it psw 0x00	40		
	42 P1 ^= 0x01; /* Toggle P1.0 each time we print */ 13 printf ("Hello World's"); /* Print "Hello World" */		
	$\frac{44}{45} - 1$		~
E Project Registers	<		>
Command	д 🗵	UART#1	4 ×
Running with Code S	ize Limit: 2K	Hello World	^
Toad .c://werr_vs//	CSI(/Examples//HELLO//HELLO"	Hello World Hello World	
		Hello World	-
4			~
>		×	8
ASM ASSIGN BreakDis	able BreakEnable BreakKill BreakList BreakSet BreakAccess	Call Stack + Locals UART #1 Memory 1	
Open an existing document		Simulation t1: 198.61602655 sec L:29 C:1 CAP NUM SCRL OVR R/	W

Port1 window is displayed with D0 as 1

W						C:\Keil_v5\C51\Examples\HELLO\Hello.uvproj - µVision -	. 🗇 🗙
File Edir	t View	Project	Flash	Det	pug Peripherals Tools SVCS Window	wHelp	
🗋 🗋 💕		XG	8	Q	Start/Stop Debug Session Ctrl+F5	5 🞼 III 🖄 🖉 🔽 🖉 🄕 🔸 🔿 🔗 🎎 💷 📲 🔦	
RST 🗉	8 7	0 0 0	() ⁺	RST	Reset CPU	■ • 🔟 • 📓 • 🕺 •	
Registers		Ф 🗙	Disas	5	Run F5	5	ů 🗙
Register		Value	-> <mark>C:</mark>	0	Stop	(8.1),C:0C09	^
Regs	1	0~09	c:	(7)	Step F11	1 (x99),A	
r1		0x34	< C:	0	Step Over F10	0	> ×
r2		0x0c		$\{ \}^{2}$	Step Out Ctrl+F11	1	
r3		0x00	4	-{}	Run to Cursor Line Ctrl+F10	0	• *
		0x00		⇔	Show Next Statement		^
r6		0x00			Breakpoints Ctrl+B	8	
E Sys		uk/2			Insert/Remove Breakpoint F9	9 -4/ Parallel Port 1 ×	
а		0x72			Enable/Disable Breakpoint Ctrl+F9	9 pix UART, emable rovr */ Port 1	
SC	,	0x25		8	Disable All Breakpoints	value for 1200 band = 106fs - 4/ P1: OxFE 7 Bits 0	
sp	_max	0x27		æ	Kill All Breakpoints Ctrl+Shift+F9	9 to send first char of UART 4/ Pins: DxFE WWWWWWW	
a	.por1	0x00			OS Support		
⊡ dp	otr	0x0c34			Execution Profiling	and a set	
se	ates IC	269.61			Memory Man		
	SW.	0x00			Inline Assembly	- 4/	
					Function Editor (Open Ini File)	time we print 4/ Merid 4/	
	_				Dahua Callinan		~
E Project	Regis	ters	<	_	Debug Settings		>
Command						4 🔟 UART#1	Q 🔀
Running	with C	ode Si	ze L	imit Evan	: 2K	A Hello World	^
Dodd o				- Auto	pres ((include ((include	Hello World Hello World	
						Hello World	
<						2	~
>							>
ASM ASS	IGN Bre	akDisa	ble H	Brea	kEnable BreakKill BreakList	BreakSet BreakAccess 🚱 Call Stack + Locals 😰 UART#1 💷 Memory 1	
Enter or lea	ave a debu	g session			T H T	Simulation t1: 347.57461491 sec L:29 C:1 CAP NUM SCR	LOVR R/W
	6			S	S 🚺 🔣 🚣 🤇	🕹 🐖 🗞 10 K 🛛 🖬 🗸 🕻	3:06 PM 29-Nov-14

Port1 window is displayed with D0 as 0

W	C:\Keil_v5\C51\Examples\HELLO\Hello.uvproj - µVision	- 🗇 🗙
File Edit View Project	Flash Debug Peripherals Tools SVCS Window Help	
🗋 💕 🖬 🕼 🕺 🖉	🚨 つ で ← → 隆 改 改 律 詳 順 版 🌌 🖉 🔍 🗟 象 Q 🌢 🛇 🔗 🛃 🖬 🔹	
	Simulator 🔍 🔊 📥 🖶 🧇 🍏	
Project 🛛 📮 🔀	HELLO.C	▼ ×
Simulator Simulator	12 0 14 0	×
	<pre>22 TH = 1: //* TH: sime 1 mm */ 33 TI = 1: /* TI: set 1 cosed first char of UNIT */ 34 feedid 35 36 37 37 37 38 feedid 37 39 feedid 38 feedid 39 feedid 39 feedid 39 feedid 39 feedid 39 feedid 39 feedid 30 feedid</pre>	
<u>□□ P.</u> (7 B. () F. U. T.	C	*
Build Output		Q X
Program Size: data=3 "HELLO" - 0 Error(s)	0.1 xdata=0 code=1096 , 0 Warning(s).	, i i i i i i i i i i i i i i i i i i i
≪		
create an empty document	Simulation	1229 CT CAP NUM SCRL OVR R/W

Then stop debugging the process

W	C:\Keil_v5\C51\Examples\HELLO\Hello.uvproj - µVision	- 🗇 🗙
File Edit View Project	Flash Debug Peripherals Tools SVCS Window Help	
🗋 🗃 🖬 🎒 🕺 🖏	臨 り で 年 中 や 忠 忠 律 津 准 版 幽	
0 🕮 🆽 🖉 🖼 🕱	Simulator 🖳 🛣 🐁 🗇 🎰	
Project 2 Simulator		
<pre>< >> Imp.@s. OF O.g.T Build Output linking Program Size: data=3 "HELLO" - 0 Error(s)</pre>	00.1 xdata=0 code=1096 , 0 Warning(s).	* 5 ^ ~
C	Simulation	CAP NUM SCRL OVR R/W

After closing the program, the window appears in this format



Now remove HELLO.C from source group 1

8	C:\Keil_v5\C51\Examples\HELLO\Hello.uvproj - µVision	- 0 ×
File Edit View Project Flash Debug Peripherals Tools SVCS Window	Help	
□2022日間の1001001001001000000000000000000000000	//= //= 🖄 🖉 🖳 🖉 🔍 🖉 🖉 🖉 🖉	
💚 🕮 🕮 📖 🞇 Simulator 🛛 🔽 🐔 📥 🖘 🥎 🙆		
Project 0 Simulator Image: Simulator Image: Simulator Image	µVision × Image: Are you sure to remove File THELLO.C? Yes	
Build Output		a 🖬
linking		^
"HELLO" - 0 Error(s), 0 Warning(s).		
		>
	Simulation	CAP NUM SCRL OVR R/W

Now select yes

Output:-

C:\Keil_v5\C51\Examples\H	ELLO\Hello.uvproj - µVision – 🗇 🗙
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help	
Registers 4 🖬 Disassembly	4 🖬
Register Value 23: void main (void) {	^
E Regs 25: /*	
r1 0x00 26: Setup the serial port for 1200 haud at 16MHz.	、 [×]
r2 0x00	*
	▼ X
r5 0x00 23 🛱 void main (void) (^
r6 0x00 24 25 0 /*	
Sva 27	Devalled Deet 1 X
a 0x00 28 # #indef MONITORS1	Parallel Port I Made
b 0x00 30 TMOD = 0x20; /* TMOD: timer 1, mode 2, 8-bit reload */	Port I
sp Dk21 31 THI = 221s /* THI: reload value for 1200 baud § 16MHs */	
PC \$ C:DxOC 34 gendif	
auxr1 0x00 35 -	
the dptr UKUUU 27 Note that an embedded program never exits (because	
Sec 0.0001 29 must loop and execute forever.	
42 P1 ^= 0x01; /* Toggle P1.0 each time we print */ 43 printf ("Mello Worldha"); /* Print "Mello World" */	
44 - 1	
Project Registers <	, · · · · · · · · · · · · · · · · · · ·
Command 🛛 🗶	UART#1 🛛 🗘 🖬
Running with Code Size Limit: 2K	Hello World
Load "C:\\Keil_V5\\C51\\Examples\\HELLO\\HELLO"	Hello World
	Hello World
×	
5	< > >
ASM ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess	Call Stack + Locals UART #1 🛄 Memory 1
Open an existing document	Simulation t1: 198.61602655 sec L:29 C:1 CAP NUM SCRL OVR R/W

HELLO WORLD with D0=1

File Edit View Project Flash Debug Peripherals Tools SVCS Window Help	
🗋 🚔 🛃 🖓 🖕 🚓 🔞 🔞 Start/Stop Debug Session Ctrl+F5 🗧 //2 🚳 🔍 🗟 🔦 😡 😡 🖉	
新 国 🎱 ① ① ① ① ① (1) 新 Reset CPU 🔤 🗷 🔜 - 🔜 - 🎘 -	
Registers 4 🖬 Disass 🕅 Run F5	4 🔀
Register Value Cri & Stop 18.1), c:0C09	^
- Regs C: (?) Step F11 x39), A	
-ri b34 cri 0 step Over F10	````
12 Duo Distep Out Ctri+F11	
r3 Udt La 10 Run to Cursor Line Ctrl+F10	• ×
-r5 0x00 ψ Show Next Statement	^
- r6 0x00 - r7 0x72 Breakpoints Ctrl+B	
E-Sys Parallel Port 1	
a Uk/Z D Enable/Disable Breakpoint Ctrl+F sis UAIT, enable ever */	
sp 0x25 O Disable All Breakpoints visue 5 100 band § 10865 4/ P1: 0x76 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	
- PC & ChuCi	
aur 0x00 OS Support	
et de 2010 Profiling	
sec 269.51 Memory Man	
e psw 0x00 Inine Assembly	
Function Editor (Open Ini File) Wecklar 4/	
	~
Project Registers c Debug Settings	>
Command a 🖬 UART#1	đ 🗙
Running with Code Size Limit: 2K	^
Hello World	
Hello World	
<	~
×	>
ASM ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess 🛛 🖓 GHI Stack + Locals 💆 UART#1 🔲 Memory 1	
Enter or leave a debug session t1: 347.57461491 sec L23 C:1 CAP NUM	SCRL OVR R/W
- 11 🖉 🚰 💽 📲 💟 💆 🐖 %	3:06 PM 29-Nov-14

HELLO WORLD with D0=0

Observation:-

From the above outputs it is analyzed that the message "hello world" is printed at UART#1 at each time P1.0 is toggled.

Conclusion:-

The experiment to print hello world using keil u vision successfully.

Experiment-3

Aim of the experiment:- Write a C program to store the data in the accumulator.

Software required: - Keil u vision 5

Theory: - This experiment aims to store any data in the Accumulator using the software keil u vision 5. In this program, the data 0x05 is to be stored in accumulator i.e. value 05 in hexadecimal. Firstly the header files REG51.H is declared for the intended 8051. Then the main function starts. In the main function 0x05 is stored in Acc variable.

Program:-

```
#include<reg51.h>
void main()
```

```
{
Acc=0x05;
}
```

Procedure :- The procedure for the program of storing a data in accumulator is as follows.



A new file is is created

C:\Keil_v5\C51\Exampl	es\HELLO\Hello.uvproj - µVision	- 🗇 🗙
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help		
□ 📸 🛃 🖇 🗈 🛍 🧳 🗢 🔶 👄 🗮 🖉 🕼 🖁	🚽 🗟 🥐 🔍 🗉 🗢 🔗 🚓 💷 📲	
🕸 🕮 🚳 🚑 🞇 Simulator 🕞 💰 🛔 🗟 🚸 🕎 🍏		
Project 🛛 🛛 🗋 Text2*		- ×
E Simulator #include <reg51.h></reg51.h>		
Source Group 1 void main ()		
ACC=0x05;		
Build Output		4 🗵
Program Size: data=30.1 xdata=0 code=1096		°
"HELLO" - 0 Error(s), 0 Warning(s).		•
×		*
Rebuild all target files	Simulation	CAP NUM SCRL OVR R/W

The program is written



The program is to be saved



The program is saved in a new folder (created folder ece lab)

		C:\Keil_v5\C51\Exa	mples\HELLO\Hello.uvproj - µVision		- 0 ×
File Edit View Project I	Flash Debug Peripherals Tools	SVCS Window Help			
🗋 🗃 🖬 🗿 🕺 🖓 🛯		四 阪 伊 伊 川 川 🙆	🖳 🍡 🔕 🔹 🖉 🖉 🖉		
@ () () () () () () () () () () () () () () () ()	Simulator	t t			
Project 4 🛛	Text2*	W.	Save As	×	▼ ×
Simulator Source Group 1 Documentation	<pre>#include<reg51.h> void main() {</reg51.h></pre>		s (F:) → ece lab v C Search ece lab	P	
ABSTRACT.TX	ACC=0x05;	Organize 👻 New folder	8≡ ▼	0	
		A Name Name	 Date modified Type No items match your search. 		
		Deeps Local Disk (Cc:) DVD R/W Drive (D out of syllabus (E in syllabus (F:) in syllabus (G:			
		🗣 Network		>	
		File name: prg1.c Save as type: All Files (*.*)		~	
< >>) Hide Folders	Save Can	cel .d	
<u>■</u> P., ③B., {} F., 0, T.,	<				,
Build Output					Q 🛛
linking Program Size: data=30 "HELLO" - 0 Error(s),	0.1 xdata=0 code=1096 , 0 Warning(s).				Î
ć					>
			Simulation		L:4 C:10 CAP NUM SCRL OVR R/W
📑 健 🚞	1 🕄 🖬 🔣		- I Company and the	12	★ 10 1% 10 million 3:15 PM 29-Nov-14

The program is to be saved with the extension .c

W	C:\Keil_v5\C51\Ex	amples\HELLO\Hello.uvproj - µVision	- 🗆 🗙
File Edit View Project Flash Debug Peripherals Tools SVCS V	Vindow Help		
	軍 課 //注 //注 🖄	🖂 🔜 🥐 🔍 🗉 🔹 🔗 🍓 🗖 📼 📲	
🗇 🕮 🥔 📖 🞇 Simulator 🛛 💀 💰 🐁 🧇	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1		
Project # 🖬 🗋 prg1.c			▼ X
Build target F7 Translate File Stop build Show Include File Dependencies			
			>
Build Output			a 💌
<pre>linking Program Size: data=30.1 xdata=0 code=1096 "HELLO" - 0 Error(s), 0 Warning(s). </pre>			× ×
Add Existing Files to current Project Group		Simulation	L:4 C:10 CAP NUM SCRL OVR R/W
📑 Ĉ 🚞 S 🖬 🔣 🖊	🥹 🐖 🗞		- 🕆 🔀 🌒 🖿 📲 ∮ 3:15 PM 29-Nov-14

Addition of prg1.c to the Source Group1

1		C:\Keil_v5\C51\Examples\	HELLO\Hello.uvproj - µVision	- 0' ×
File Edit View Project F	Flash Debug Peripherals Tools SVCS Window	v Help		
🗋 🖬 🖬 🗿 🕺 🗛 🛯	第 11 月 月 9 1 ← → ← ~ 5	//= //m=	🖳 🗟 🥐 🔍 🔍 🔹 🔗 🖉	
	Simulator 🛛 🐼 📥 🖷 🧇 🥎			
Project 🛛 🕂 🚺	prg1.c			▼ ×
Source Group 1	1#includ <req51.h> 2 void main() 3 { 4 ACC=0x05; 5 }</req51.h>	Add Files to Gro	up 'Source Group 1'	
		Look in: 💼 in syllabus (F:)	- 🗢 🖆 📰 -	
		Name	Date modified	
		🎉 antenna	26-Sep-14 1:58 PM	
		C S T	20-Jun-14 5:19 AM	
		lece lab	29-Nov-14 3:15 PM	
		🍌 hfss	Date created: 29-Nov-14 3:14 PM	
		<	Size: 47 bytes Files: pro1	
		File name:		
		Files of type: C Source file (*.c)	Close	
< >				
E P. 8 B. 8 F. 0, T.	<			>
Build Output				ů 🗵
linking	1 vdata=0 gode=1086			^
"HELLO" - 0 Error(s),	0 Warning(s).			
<				>
	N N N		Simulation	L:4 C:10 CAP NUM SCRL OVR R/W
📑 健 🚞	N 🕄 🚺 🔣 🖊	ا الله الله الله الله الله الله الله ال	1 Contraction of the	∽ 🛍 🔀 🚷 🖃 💷 ∮ 3:16 PM 29-Nov-14

The newly created folder ece lab is selected

W.	C:\Keil_v5\C51\Examples\HELLO\Hello.uvproj - µVision	- 0 ×
File Edit View Project Flash Debug Peripherals Tools SVCS Wind	ow Help	
1 😂 🛃 🖇 3 日 🚨 🤊 (*) (* + +) (*) 日 田 田 田 日 (*) (*)	≓ //≣ //ௐ 🙆 🔍 🔜 🔊 🔍 🔍 🔍 🔍	
💿 🏔 📾 🧔 🗮 🕱 Simulator 💿 🔊 📥 🐁 🔶 🕎		
Project # 🔟 prg1.c		▼ X
□ □ \$ finclude <reg51.h> □ □ Source Group 1 2 void main() □ □ □ 0 □ □ ABSTRACT.TX 3 (□ □ ABSTRACT.TX 5)</reg51.h>	Add Files to Group "Source Group 1" ×	
	Look in: ece lab	
	Name Date modified Ij	
	25*N0V*14 3/13 PM	
	File name: non1	
	Hiles of type: C Source file (*.c)	
	and the second se	
		,
Build Output		a 🛛
linking Program Size: data=30.1 xdata=0 code=1096 "HELLO" - 0 Error(s), 0 Warning(s).		Ŷ
<		*
	Simulation	L:4 C:10 CAP NUM SCRL OVR R/W
📑 🧭 🚞 😵 🔛		∽ 🛍 🔀 🔰 🖬 📲 3:16 PM 29-Nov-14

Now prg1.c is selected

R C:	Keil_v5\C51\Examples\HELLO\Hello.uvproj - µVision	- 🗇 🗙
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help		
□ 😂 🖬 🗿 👃 வ 🚨 💙 ♡ ← → 隆 改 改 数 律 準 准 振	🚈 🔍 🖉 🗟 🌾 🔍 🖉 🖉	
🕸 🎬 🖉 🥥 🏭 🞇 Simulator 💿 🔊 🛔 🐁 🚸 🥎 🎰		
Project Build (F7) prg1.c		▼ ×
Build target files 1 #include <reg51.h></reg51.h>		
Source Group 1 2 void main ()		
ACC=0x05;		
ABSTRACT.TX 5 }		
		>
Build Output		n M
linking		-
Program Size: data=30.1 xdata=0 code=1096		
"HELLO" - U Error(s), O Warning(s).		v
<		Figure 1 Second Seco
Build target files	Simulation	L:4 C:10 CAP NUM SCRL OVR R/W
🦰 🥭 🔚 😵 🖬 🔣 🚣 🥹 !		- 🛍 🔀 🚷 🖬 📲 ₹ 3:17 PM 29-Nov-14

Check whether the prg1.c is added in Source Group1 or not. Now bulid(F7) the target.

C:\Keil_v5\C51\Examp	oles\HELLO\Hello.uvproj - µVision	- 🗇 🗙
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help		
□ 😂 🛃 🏈 & 与 🛍 🤊 (*) ← → 隆 隆 隆 段 律 律 ///////////////////////////////	🖌 🔜 象 📀 🧅 🧶 🔍	
🤣 🕮 🖉 🥥 🤮 🎇 Simulator 🛛 🖳 🛠 📥 🐟 🧇 🍏		
Project 0 1 prg1.c Image: Simulator 1 \$include 2 yoid main() Image: Simulator 2 yoid main() 3 (• ×
4 ACC=0x05)		
		2
Build Output		4 🗵
linking Frogram Size: data=9.0 xdata=0 code=18 "HELLO" - 0 Error(s), 0 Warning(s).		, U
21 21	Simulation	L4 C10 CAP NUM SCRL OVR R/W

Check for any errors in program in the build output. Here displaying 0 errors and 0 warnings.

W.	C:\Keil_v5	\C51\Examples\HELLO\Hello.uvproj - µVision	- 0 ×
File Edit View Project	Flash Debug Peripherals Tools SVCS Window Help		
🗋 💕 🖬 🕔 🕺 🖓	🚨 🤊 (*) (*) 🗢 🖗 🕼 🖉 (*) (*) (*) (*) (*) (*) (*) (*) (*) (*)	🚽 🗟 🥐 🔕 🔶 🔿 🍓 🖬 - 🔍	
	Simulator 🛛 🐼 👗 🕾 🧇 🎰	Start/Stop Debug Session (Ctrl+F5)	
Project 🛛 📮 🔀	prg1.c	Enter or leave a debug session	▼ ×
Simulator	1 #includerceg51.h> 2 void main() 3 (4 ACC=0X05; 5)		
□ P.			
linking			4 🖬
Program Size: data=9 "HELLO" - 0 Error(s)	0 xdata=0 code=18 0 Warning(s).		, ,
Enter or leave a debug session		Simulation	L:4 C:10 CAP NUM SCRL OVR R/W
📑 🥝 🔋	I 🕄 🖬 🔣 🖊 🥹 🐖		^ 📆 🔀 🌒 📰 📲 ◀ 3:18 PM 29-Nov-14

Start debugging session

R.	C:\Keil_v5\C51\Examples\HELLO\Hello.uvproj - µVision
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help	
□22 日 23 日 24 → → → → 2 日 24 2 1 2 1 2 2 2 2 2 2 2	i 🖄 🗸 🔊 🥥 🖕 🖉 🖉
🕸 🕮 🕮 🎼 🔛 Simulator 🛛 💀 🛣 📥 🚸 🐡 🎰	
Project # 🛛 📄 prg1.c	
<pre>Simulator Source Group 1 ABSTRACT.TX ABSTRACT.TX P</pre>	Vision EVALUATION MODE Running with Code Size Limit: 2K OK
×.	
	Simulation L:4 C:10

New window evaluation mode appeared. Press ok

	C:\Keil_v5\C51\Examples\HELLO\Hello.uvproj - µVision	- 🗇 🗙
File Edit View Project	t Flash Debug Peripherals Tools SVCS Window Help	
🗋 📬 🖬 🎒 🕺 🗉	2 82 9 0 ← → 1 10 8 18 8 16 76 76 16 16 18 18 18 18 18 18 18 18 18 18 18 18 18	
1 1 1 🖌 🔘 📲		
Registers 🛛 🗛 📘	I Disassembly	Ф 🗵
Register Value	2: void main()	^
E Regs	3: (4: ACC=0x05;	
r1 0x00	SC.0x080C 7405 MOV A.#0x05	Ň
r2 0x00		
r3 0x00		• *
r5 0x00	2 void main()	
r6 0x00	3 (
E Sys	↓ ↓ 4 QCC=0x05; 5)	
a 0x00		
sp 0x07		
sp_max 0x07		
auxr1 0x00		
E dptr 0x000	0(
states 389	11	
Project Registers		>
Command	0 🖬 UART#1	4 🔀
Running with Code S	Size Limit: 2K	~
Load "C:\\Keil_v5\\	<pre>\C51\\Examples\\HELLO\\HELLO"</pre>	
¢	×	~
>	<	>
ASM ASSIGN BreakDis	sable BreakEnable BreakKill BreakList BreakSet BreakAccess	
	Simulation t1: 0.00014145 see 1:4 C:	CAP NUM SCRU OVR RAV

C:\Keil_v5\C51\Examples\HELLO\Hello.uvproj - µVision	- 🗇 🗙
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help	
□ 🖉 🚽 🕹 💩 🛎 🔊 ♥ 🖛 ⇒ 隆 態 態 復 律 定 度 🖉 🖉 🔍 🔍 🔍 🔍 🔍 🔍 🔍	
(新田) ③ 70 70 · 10 · 20 30 = 20 20 · · · · · · · · · · · · · · · · ·	
Register Run (F5) sassembly	a 🖬
Registe Start code execution 2: void main()	^
□ Regs 3: 1 4: ACC=0x05;	
10 aud ₩0 c 10x000 7405 MOV A.±0x05	× *
12 0x00 Flamt	- X
-r4 0x00 1 tinclude(red51.b>	
-r5 0x00 2 void main()	
- 7 0x00 3 (- 7 0x00 Nb 4 bcc=0x05;	
B − 5 }	
sp 0kd7 so max 0kd7	
auri 0x00	
states 389	
sec 0.0001	
m Project = Registers <	3
Command 2 UART=1	4 🖬
Running with Code Size Limit: 2K	X
Load "C:\\Keil_v5\\C51\\Examples\\HELLO\\HELLO"	
×	4
>	2
ASM ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess 🖗 Call Stack + Locals 🐷 UART#1 🔳 Memory 1	
Simulation. 11: 0.00014	145 sec L:4 C:1 CAP NUM SCRL OVR R/W
📑 🥝 🚞 🕄 💷 🔣 丛 🥹 🐖 🗞 💷 💶 💶	▲ 🔁 🔀 🤮 🚍 📲 ◀ 3:20 PM 29-Nov-14

Run the program

W	C:\Keil_v5\C51\Examples\HELLO\Hello.uvproj - µVision	- 🗆 🗙
File Edit View Project Fl	Flash Debug Peripherals Tools SVCS Window Help	
🗋 🗃 🛃 🌒 🕺 🖓 🕻	3 Start/Stop Debug Session Ctrl+F5 //2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2	
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	1) 篩 Reset CPU 國 • 國 • 國 • 父 •	
Registers 🛛 🛱 🖬	Disas 🗟 Run F5	a 🛛
Register Value	Stop 803	<u>^</u>
E Regs	C:(²) Step F11 ::080C)	
r0 0x0a	Step Over F10	×
	Step Out Ctrl+F11	3
r3 0x00	() Run to Cursor Line Ctrl+F10	• ×
r4 0x00	Show Next Statement	
r6 0x00		
r7 0x00	Breakpoints Ctri-B	
a 0x00	Inservemove breakpoint	
ь 0х00	C Enable/Disable Breakpoint Ctrl+F9	
sp 0x05	Disable All Breakpoints	
PC \$ C:Dx0E	Kill All Breakpoints Ctrl+Shift+F9	
auxr1 0x00	OS Support	
t dptr 0x0000	Execution Profiling >	
sec 25.740	Memory Man	
😟 psw Ox00		
	Funding Filler (Deen Ini File)	
Project Registers	C Debug Settings	>
Command	4 🔟 UART#1	1 X
Running with Code Size	ze Limit: 2K	^
Load "C:\\Keil_v5\\C5:	51\\Examples\\HELLO\\HELLO"	
1	×	
		~
ASM ASSIGN BreakDisabl	ble BreakEnable BreakKill BreakList BreakSet BreakAccess	
Step one line	Simulation t1: 25.74084509	sec L:1 C:1 CAP NUM SCRL OVR R/W .::
		3:20 PM
		29-Nov-14

Select step from debug or press F11

W	C:\Keil_v5\C51\Exampl	es\HELLO\Hello.uvproj - µVision	- 8 ×
File Edit View Projec	t Flash Debug Peripherals Tools SVCS Window Help		
🗋 😂 🖬 🌒 🐰	2 ▲ → ← → ● ◎ ◎ ◎ ↓ → → ●	모 🔜 🍬 💿 🔗 🌒 🖬 🖌	
🎼 🕅 🖾 🖉 🖓 🖓	0° 10 🔶 🔽 🕼 📾 😂 🖉 • 💷 • 🔯 • 🔜 • 📓 • 🖉 •		
Registers 4	Disassembly		₽ ⊠
Register Valu			^
E Regs	C:0x080F 00 NOP		
r1 0x00	C:0x0810 00 NOP		ू ¹⁰
r2 0x00			*
r3 0x00	1 prg1.c		• ×
r5 0x00	1 #include <reg51.h></reg51.h>		
r6 0x00	3 (
Sve CxOL	4 ACC=0x05;		
a 0x05	5		
b 0x00			
sp UxU			
PC \$ C:0x	DE		
auxr1 0x00			
t dptr 0x00			
sec 25.7	40		
🖻 psw OxOC			
Project Persisters			,
Command		UART#1	4 X
Running with Code	Size Limit: 2K		~
Load "C:\\Keil_v5\	\C51\\Examples\\HELLO\\HELLO"		
<			
>			2
ASM ASSIGN BreakDi	sable BreakEnable BreakKill BreakList BreakSet BreakAccess	Call Stack + Locals	
		Cimulation +1. 35 74095745 ros 1.67	54 CAD NUM COLD DAM

Check the value of a register.

Output:-

W	C:\Keil_v5\C51\Examples\HELLO\Hello.uvproj - µVision	- 🗆 🗙
File Edit View	Project Flash Debug Peripherals Tools SVCS Window Help	
🗋 🖬 🖉 🖉	Ѯ ဩ 畿 🥑 ୯ ← → ಶ 懲 懲 準 非 // // // // // // //	
👬 🗄 🚳	관 만 안 안 수 도 國, 團 = 글, 월 * 💷 * 📴 * 📾 * 📾 * 🗒 * 💆 *	
Registers	4 🚺 Disassembly	# 🔀
Register	Value 5: }	^
Regs		
r0		~
1		>
r2	0.00 Prot.c	▼ ×
r4	0x00 1 #include <req51.h></req51.h>	
r5	0x00 2 void main()	
10		
E Sys		
а		
b		
sp max	0x07	
PC s	C.0x0E	
aur1	0,00	
dptr et at ac	GRUUUI 70787	
sec	25.74	
	0x00	
63		
🖭 Project 🚟 Re	gisters S	3
Command	4 🖬 UART#1	4 🔀
Running with	Code Size Limit: 2K	^
Load "C://Ke	II_05//C51//Examples//HELLO//HELLO"	
1	*	
		v
-		
ASM ASSIGN B	ceakuisable breakhable	

The value of accumulator is 05

Observation:-

From the above output it is analyzed that the value 05 is stored in accumulator.

Conclusion:-

The experiment to store any data in accumulator using keil u vision successfully.

Experiment-4

Aim of the experiment:- Write a C Program to send values 00-ff to Port 1.

Software required: - Keil u vision 5

Theory: - This experiment aims to send value 00-ff using the software keil u vision 5. In this program, the data 00-ff is to be sent to Port1. Firstly the header files reg51.H is declared for the intended 8051. Then the main function starts. In the main function an unsigned character is initialized followed by a for loop where 0-255(00-ff) is assigned to Port1.

Program:-

```
#include<reg51.h>
void main()
{
unsigned char z;
for(z=0; z<=255; z++)
P1=z;
}</pre>
```

Procedure: - The procedure for the program of storing a data in accumulator is as follows.

File new Program(code) Save Add existing files to source group 1 Build Start Debug Run Peripherals Port1 Debug the process

Output:-



Observation:-

From the above output it is observed that the data 00-ff is sent to Port1.

Conclusion:-

The experiment to send values 00-ff to Port 1 using keil u vision successfully

Experiment-5

Aim of experiment: - Write a C Program to send hex values for ASCII characters 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f to port 1

Software required:- Keil u vision 5

Theory :- This experiment aims to send hex values for ASCII characters 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f to port 1 using the software keil u vision 5. Firstly the header files reg51.H is declared for the intended 8051. Then the main function starts. In the main function an unsigned character array is initialized with"0123456789abcdef" and then unsigned character variable is initialized followed by a for loop. The loop runs for 16 time where each time one character from array is sent to Port1.

Program:-

}

```
#include<reg51.h>
void main()
{
      unsigned char a[] = "0123456789abcdef";
      unsigned char z;
      for(z=0; z<=16; z++)
      P1=a[z];
```

Procedure:- The procedure for the program of storing a data in accumulator is as follows.

 $\begin{array}{c} \overrightarrow{} & \overrightarrow{} & \overrightarrow{} \\ \overrightarrow{} & \overrightarrow{} \\ \overrightarrow{} & \overrightarrow{} \\ \overrightarrow{}$ $\rightarrow \rightarrow$ P1 Debug the process

Output:-

1. For input '0' 2. For input 'a'

C:\Keil_v5\C51\Examples\H	ELLO\Hello.uvproj - µVision – 🗗 🗡
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help	
Registers 4 Disassembly Recister Value 7: P1=mynum[z];	4 14
Regs C:0x090B 7408 MOV A,#0x08 C:0x090D 2E ADD A 87	
r1 0x08 C:0x090E F8 MOV R0.A	×
r2 0x00 r3 0x00 1 abcdtoport5.c	▼ ×
-r4 0x00 1 #include <reg51.h></reg51.h>	
r6 0x01 3 □ (
<pre>-Sys</pre> 4 unsigned char mynum[]= "0123456789abcdef"; 5 unsigned char z;	
a 0:30 6 for (z=0; z<=16; z++) b 0:08 > 7 Pl=mvnum[z]; Parallel Port 1	x
sp 0x18 8 } sp_max 0x1a 0	
PC \$ C:0x05 P1: 0x30	2
dptr 0x092t Pins: 0x30 TVV TV	
sec 0.0002	
Dia pisw Uxau	
Project Registers <	>
Command 4	Memory 1 🕴 🕽
Running with Code Size Limit: 2K Load "C:\\Keil_v5\\C51\\Examples\\HELLO\\HELLO"	Address: V:0
BS \ABCDTOPORT5\7 LA `P1	V:0x0000: 00 00 00 00 00 00 00 00 00 00 00 0
c >	V:0x0030: 00 00 00 00 00 00 00 00 00 00 00 00 0
>	V:0x0048: 00 00 00 00 00 00 00 00 00 00 00 00 00
ASM ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess	Call Stack + Locals Memory 1
	Simulation TI: 0.00022109 Sec E7 C1 CAP NUM SCRE OVR ROW
C:\Keil_v5\C51\Examples\H	ELLO\Hello.uvproj - µVision – 🗗 🗴
C:\Keil_v5\C51\Examples\H File Edit View Project Flash Debug Peripherals Tools SVCS Window Help	ELLO\Hello.uvproj - µVision - ਰੋ
C:\Keil_v5\C51\Examples\H File Edit View Project Flash Debug Peripherals Tools SVCS Window Help	ELLO\Hello.uvproj - µVision - 🗗 🗙
C:\Keil_v5\C51\Examples\H File Edit View Project Flash Debug Peripherals Tools SVCS Window Help ご ご ご ご ご ご ご ご ご S S ご ご ご ご ご ご ご S	
C:\Keil_v5\C51\Examples\H File Edit View Project Flash Debug Peripherals Tools SVCS Window Help Project Flash Debug Peripherals Tools SVCS Project Flash Debug Peripherals Tools SVCS Project Flash Debug Peripherals Tools SVCS Project Flash Debug Peripherals Project Flash P	ELLO\Hello.uvproj - µVision - 🗗 🗙
C:\Keil_v5\C51\Examples\H File Edit View Project Flash Debug Peripherals Tools SVCS Window Help Project Flash Debug Peripherals Tools SVCS Project Flash Debug Peripherals Tools SVCS Project Flash Debug Peripherals Tools SVCS Project Flash Debug Peripherals Project Flash Debu	
C:\Keil_v5\C51\Examples\H File Edit View Project Fissh Debug Peripherals Tools SVCS Window Help Project Fissh Debug Peripherals Tools SVCS Project Fissh Debug Peripherals Project Peripherals Project Fissh Debug Peripheral	ELLO\Hellouvproj - µVision - 5 ×
C:\Keil_v5\C51\Examples\H File Edit View Project Fish Debug Peripherals Tools SVCS Window Heip Peripherals Tools SVCS Peripherals Tools SVCS Peripherals Tools SVCS Peripherals Tools Tools Tools Tools Tools Tools Peripherals Tools Tools Tools Tools Tools Tools Peripherals Tools Tools Tools Tools Peripherals Tools Tools Tools Tools Tools Peripherals Tools Tools Peripherals Tools Tools Tools Peripherals Tools Tools Peripherals Tools Tools Peripherals To	ELLO\Hellouvproj - µVision - C ×
C:\Keil_v5\C51\Examples\H File Edit View Project Fish Debug Peripherals Tools SVCS Window Heip File Edit View Project Fish Debug Peripherals Tools SVCS Window Heip File Edit View Project Fish Debug Peripherals Tools SVCS Window Heip File Edit View Project Fish Debug Peripherals Tools SVCS Window Heip File Edit View Project Fish Debug Peripherals Tools SVCS Window Heip File Edit View Project Fish Debug Peripherals Tools SVCS Window Heip File Edit View Project Fish Debug Peripherals Tools SVCS Window Heip File Edit View Project Fish Debug Peripherals Tools SVCS Window Heip File Edit View Project Fish Debug Peripherals Tools SVCS Window Heip File Edit View Project Fish Debug Peripherals Tools SVCS Window Heip File Edit View Project File Edit View Project File File Edit View Project File Edit View Project File File Edit View Project File Edit View Project File File Edit View Project File Edit View Project File File Edit View Project File Edit View Project File File Edit View Project File Edit View Project File file Edit View Project File Edit View Project File file Edit View Project File Edit View Project File file Edit View Project File Edit View Project File file Edit View Project File Edit View Project File file Edit View Project File Edit View Project File file Edit View Project File Edit View Project File file Edit View Project File Edit View Project File file Edit View Project File Edit View Project File file Project File Edit View Project File file Edit View Project File Edit View Project File file Edit View Project File Edit View Project File file Edit View Project File Edit View Project File file Edit View Project File Edit View Project File file Edit View Project File Edit View Project File file Edit View Project File Edit View Project File file Edit View Project File Edit View Project File file Edit View Project File file Edit Vie	ELLO\Hellouvproj - µVision - C X
C:\Keil_v5\C51\Examples\H File Edit View Project Flash Debug Peripherals Tools SVCS Window Heip File Edit View Project Flash Debug Peripherals Tools SVCS Window Heip File Edit View Project Flash Debug Peripherals Tools SVCS File Edit View Project Flash Debug Peripherals Tools SVCS File Edit View Project Flash Debug Peripherals Tools SVCS File Edit View Project Flash Debug Peripherals File Edit Peripherals Fi	ELLO\Hellouvproj - µVision - C X
C:\Keii_v5\C51\Examples\H File Edit View Project Fissh Debug Peripherals Total Total <thto< td=""><td>ELLO\Hellouvproj - µVision - 5 ×</td></thto<>	ELLO\Hellouvproj - µVision - 5 ×
C:\Keil_v5\C51\Examples\+ File Edit View Project Flash Debug Peripherals Tools SVCS Window Help Image: State of the s	ELLO\Hellouvproj - µVision - 0 ×
C:\Kei_v5\C51\Examples\+ File Edit View Project Flash Debug Peripherals Tools SVCS Window Help Image: State of the st	ELLOVHellouvproj - µVision - 0 ×
C:\Keii_v5\C51\Examples\H File Edit View Project Flash Debug Peripherals Tools SVCS Window Help Image: State St	ELLOVHellouvproj - µVision - 0 ×
C:\Keii_v5\C51\Examples\F File Edit View Project Flash Debug Peripherals Tools SVCS Window Help Image: State St	ELLO\Hellouvproj - µVision - 0 ×
C:\Keii_v5\C51\Examples\F File Edit View Project Flash Debug Peripherals Tools SVCS Window Help Image: State St	ELLO\Hellouvproj - µVision - D X
C:\Keii_v5\C51\Examples\F File Edit View Project Flash Debug Peripherals Tools SVCS Window Heip Image: State St	ELLO\Hellouvproj - µVision - C X
C:\Keii_v5\C51\Examples\F File Edit View Project Fish Debug Peripherais Tools SVCS Window Heip Image: State Sta	ELLO\Hellouvproj - µVision - C X
C:\Keii_v5\C51\Examples\+ File Edit View Project Flash Debug Peripherais Tools SVCS Window Help Image: State St	ELLO\Hellouvproj - µVision - C X
C:XKeil_vS\C51\Examples\+ File Edit View Project Flash Debug Peripherals Tools SVCS Window Help Image: State St	ELLO\Hellouvproj - µVision - C X
C:\Kei_v5\C51\Examples\+ File Edit View Project File Edit View Project File Edit View Project File Edit View Project Registers Project File Edit View Project File Edit View Registers Project File Edit View Project File Edit View Registers Project File Move File File Move File Registers Project File Move File File Move File Registers Project File Move File File File Project Edit File File File File File Project Examples File File File File Project Examples File File File Project Registers File Fil	ELLOVHellouvproj - µVision - C X
C:\Kei_v\$\C51\Examples\+ File Edit View Project Flash Debug Peripherals Tools SVCS Window Help Image: State Sta	ELLO\Hellouvproj - µVision - 0 ×
C:\Kei_v\$\C51\Examples\F File Edit View Project Flash Debug Peripherals Tools SVCS Window Help File Edit View Project Flash Debug Peripherals Tools SVCS Window Help File Edit View Project Flash Debug Peripherals Tools SVCS Window Help File Edit View Project Flash Debug Peripherals Tools SVCS Window Help File Edit View Project Flash Debug Peripherals Tools SVCS Window Help File Edit View Project Flash Debug Peripherals Tools SVCS Window Help File Edit View Project Flash Debug Peripherals Tools SVCS Window Help File Edit View Project Flash Debug Peripherals Tools SVCS Window Help File Edit View Project Flash Debug Peripherals Tools SVCS Window Help File Edit View Project Flash Debug Peripherals Tools SVCS Window Help File Edit View Project Flash Debug Peripherals Tools SVCS Window Help File Edit View Project Flash Debug Peripherals Tools SVCS Window Help File Edit View Project Flash Debug Peripherals Tools SVCS Window Help File Edit View Project Flash Debug Peripherals Tools SVCS Window Help File Edit View Project Flash Debug Peripherals Tools SVCS Window Help File Edit View Project Flash Debug Peripherals Tools SVCS Window Help File Edit View Project Flash Debug Peripherals Tools SVCS Window Help File Edit View Project Flash Debug Peripherals Tools SVCS Window Help File Flash Debug Peripherals Tools Flash Debug Peripherals Flash	ELLOVHellouvproj - µVision - 0 ×
C:\Kei_v5\C51\Examples\H File Edit View Project Fish Debug Peripherals Tools SVCS Window Help File Edit View Project Fish Debug Peripherals Tools SVCS Window Help File Edit View Project Fish Debug Peripherals Tools SVCS Window Help File Edit View Project Fish Debug Peripherals Tools SVCS Window Help File Edit View Project Fish Debug Peripherals Tools SVCS Window Help File Edit View Project Fish Debug Peripherals Tools SVCS Window Help File Edit View Project Fish Debug Peripherals Tools SVCS Window Help File Edit View Project Fish Debug Peripherals Tools SVCS Window Help File Edit View Project Fish File Edit View Project File Edit File File File File File File File File	ELLOL/Hellouvproj - µVision - 0 ×

Observation:- From the above output it is observed that the data 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f is sent to Port1.

Conclusion:- The experiment to send values 00-ff to Port 1 using keil u vision successfully.

Experiment-6

Aim of experiment:- Write a C program to toggle all the bits of P1 continuously.

Software required:- Keil u vision

Theory:- This experiment aims to toggle all the bits of Port1 continuously using the software keil u vision 5. Firstly the header files reg51.H is declared for the intended 8051. Then the main function starts. In the main function an infinite for loop starts where each time 0x55 followed by 0xAA is sent to Port1 and the loop goes on.

Program:-

```
#include <reg51.h>
void main()
{
for (;;)
{
p1=0x55;
p1=0xAA;
}
}
```

Procedure:- The procedure for the program of storing a data in accumulator is as follows.

 $\begin{array}{c} \xrightarrow{} & \xrightarrow{} &$

Output:-

	e expts/expt
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help	
[1] 22 24 29 24 19 25 19 20 1 40 → 1 10 25 25 26 26 27 27 27 26 26 20 20 20 20 20 20 20 20 20 20 20 20 20	🗵 🛤 🏴 🖳 🔶 🔿 🌒 🖬 📲
Registers 4 Disassembly	• 0
Register Value C:0x080F 7590AA MOV P1(0x90),#0xAA	, All and a second s
r0 0x00 8: }	*
	\$
r3 0x00 startUP.AS1 t prg6.c	• x
r5 0x00 2 void main()	
3 (7 0x00 4 fox (;;;)	
Sys 0x00 5 1	
b 0x00 7 P1=0x83;	Parallel Port 1
sp_max 0x07 9)	Pot 1 7 Bits 0
PC \$ C.0x08	
states 399	
ter psw 0x00	
Running with Code Size Limit: 2K	
Load "F:\\ece lab\\ece expts\\expt1"	
š	
>	
ASM ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess	Call Stack + Locals UART #1 Memory 1
	Simulation t1: 0.00019950 sec L8 C1 CAP NUM SCRL OVR R/W
	 * 11 12 12 * 11 12
Etece lablect	
	e expls/expluvproj - µvision – D
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help	
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help □ 😂 🛃 🎒 👗 🖓 🔍 ሩ 🔶 阳 😤 🕄 🖓 🗮 🎉 🖉	
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help □ 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2	
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help Image: Im	
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help Image: State of the state of th	
Pile Edit View Project Flash Debug Peripherals Tools SVCS Window Help Image: State of the	
Pile Edit View Project Flash Debug Peripherais Tools SVCS Window Help Image: State of the state of t	
File Edit View Project Flash Debug Peripherais Tools SVCS Window Help Image: State	
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help Image: State of the state of t	
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help Image: State of the state	
Pile Edit View Project Plash Debug Peripherais Tools SVCS Window Help Image: State of the state	
Pile Edit View Project Plash Debug Perpherais Tools SVCS Window Help Image: State of the	
Pile Edit View Project Flash Debug Peripherals Tools SVCS Window Help Image: State of the state	Parallel Port 1
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help Image: State S	
Pile Edit View Propherals Tools SVCS Window Help Image: State of the	
Pile Edit View Project Plash Debug Perpherait Tools SVCS Window Help Image: State St	Parallel Port 1
Pile Call View Project Plash Debug Perpherals Tools SVCS Window Help Image: State of the	
Pile Edit View Project Plash Debug Perpherais Tools SVCS Window Help Image: State of the state	Parallel Port 1
Pile Call View Project Flash Debug Peripherals Tools SVCS Window Help Image: State of the	Parallel Port 1 Prot
Pile Call View Project Plash Debug Perpherals Tools SVCS Window Help Image: State of the	Parallel Port 1 Price Desting of the price
Pile Call View Project Plash Debug Perpherait Tools SVCS Window Help Image: State Image: State <t< td=""><td></td></t<>	
Pile Edit View Project Plash Debug Perpherais Tools SVCS Window Help Pile Edit View Project Plash Debug Perpherais Tools SVCS Window Help Pile Edit View Project Plash Debug Perpherais Tools SVCS Window Help Pile Edit View Project Plash Debug Perpherais Tools SVCS Window Help Pile Edit View Project Plash Debug Perpherais Tools SVCS Window Help Pile Edit View Project Plash Debug Perpherais Tools SVCS Window Help Perpice Pile Data Plash Plash Debug Perpherais Tools SVCS Window Help Perpice Plash Debug Perpherais Tools SVCS Window Help Pile Data Plash	Parallel Port 1 Parallel Port 1 Philos Post Prot
Pile Edit View Project Plash Debug Peripherals Tools SVCS Window Help Pile Edit View Project Plash Debug Peripherals Tools SVCS Window Help Pile Edit View Project Plash Debug Peripherals Tools SVCS Window Help Pile Edit View Project Plash Debug Peripherals Tools SVCS Window Help Pile Edit View Project Plash Debug Peripherals Tools SVCS Window Help Register Disastemby Register Disastemby Peripherals Dologram MOV Plast Project Dologram Other Plash Debug Project I Statupnast Project Project Project Dologram Statupnast Project Plasters Command Running With Code Size Limit: 2K Command	
Pile Edit View Project Plash Debug Perpherals Tools SVCS Window Help Image: State Sta	
Pile Edit View Project Plash Debug Perpherals Tools SVCS Window Help Image: State Sta	
Pile Edit View Project Plash Debug Perpherals Tools SVCS Window Help Image: State Sta	Parallel Port 1 Port 1 Philos Port 1 Prilos Image: State of the state
File Edit View Project Flash Debug Perpherais Tools SVCS Window Help Image: State Sta	Parallel Port 1 Parallel Port 1 Phil Post Price Post Price Post Price Post Price Post Price Post Price Price Post Price Post Price Post Price Post Post Price Post
Pie Edit View Project Hash Debug Perpherais Tools SVCS Window Help Image: State	

Observation:- From the above output it is observed that all the bits of P1 are toggling continuously.

Conclusion:- The experiment to toggle all the bits of P1 continuously using keil u vision successfully

Experiment-7

Aim of experiment:- Write a C program to toggle bit D0 of port 1 50,000 times.

Software required:- keil u vision

Theory:- This experiment aims to toggle bit D0 of Port1 50,000 times using the software keil u vision 5. Firstly the header files reg51.H is declared for the intended 8051. The sbit keyword is a widely used 8051 C data type designed specifically to access single bits of SFR registers. Here we use sbit toaccess the individual bits of the Port1. Then the main function starts. In the main function a for loop runs 50000 times where each time D0 of Port1 is made 0 and then 1.

Program:-

```
#include <reg51.h>
sbit MYBIT=P1^0;
void main(void)
{
    unsigned int z;
    for (z=0;z<=50000;z++)
    {
    MYBIT=0;
    MYBIT=1;
    }
}</pre>
```

Procedure:- The procedure for the program of storing a data in accumulator is as follows.



Output:-

1	C:\Keil_v5\C51\Examples\HELLO\Hello.uvproj - µVision	- Ø ×
File Edit View Project	Flash Debug Peripherals Tools SVCS Window Help	
	● 図 り (* + + + * 2:2 (2) (注)注 注 注 塗 ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●	
Ref 🔄 😳 (*) (*)		
Registers 4	Disassembly	4
Register Value	7: for(i=0;i<=50000;i++)	^
- Regs	C:0x0805 F583 MOV DPH(0x83),A	
r0 0x00	C:0x0807 2F82 MOV R7.DPL/0x82)	×
r2 0x00		
r3 0x00	j prg7.c	• ×
r5 0x00	1 #include <reg51.h></reg51.h>	
r6 0x00	3 void main()	
r7 0x00	4 戶 (
a 0x00	5 unsigned int 1;	
- ь 0х00	7 for(i=0;i<=50000;i++) Parallel Port 1 ×	
sp 0x07	8 d (
PC S C:0x0	9 mydata = 0; 7 Bits 0	
auxr1 0x00	11 -) Prove [0:00]	
states 391		
sec 0.000	1	
E psw 0x00		
Project Registers	*	>
Command	9 04 E UART#1	4 🖬
Running with Code S	ize Limit: 2K	~
Load "C:\\Keil_v5\\	C51\\Examples\\HELLO\\HELLO"	
c	>	-
>		>
ASM ASSIGN BreakDis	able BreakEnable BreakKill BreakList BreakSet BreakAccess 😺 Call Stack + Locals 🕎 UART #1 🔟 Memory 1	
	Simulation t1: 0.00014218 sec L7 C:1	CAP NUM SCRL OVR R/W

	C:\KeiLv5\C51\Examples\HELLO\Hello.uvproj - µVision	- 0 ×
File Edit View	Project Flash Debug Peripherals Tools SVCS Window Help	
🗋 💕 🖬 🕔	※ 12 (26 (1) 12 (1) (1) (1) (1) (1) (1) (1) (1) (1) (1)	
舒 🗟 🎯 🥐		
Registers	9 Disassembly	a 🛛
Register	Value C:0x51C3 00 NOP	^
E Regs	C:0x5124 00 NOP	
r0	0400 C10X51C6 00 NOP	Ŷ
<mark>r</mark> 2		>
r3	0.60 / 1 prg7.c	▼ ×
rb r7 	0000 0001 0001 0001 0001 0001 0001 000	
Project Regist	ters <	>
Command	4 🖸 UART#1	4 🔀
Running with C Load "C:\\Keil *** error 65:	ode Size Linit: 2K vyS\CS1\Examplea\HELLO\HELLO" access violation at C:0xS1C3 : no 'execute/read' permission	^
		>
ASM ASSIGN Bre	skDisable BreakKnable BreakKill BreakList BreakSet BreakSecess	
Start code execution		CAP NUM SCRL OVR R/W

Observation:- From the above output it is observed that the bit D0 of the Port1 (P1.0) is toggled 50,000 times.

Conclusion:- The experiment to toggle bit D0 of port 1 50,000 times using keil u vision successfully

Experiment-8

Aim of experiment:- Write a C program to generate a square wave for 1ms delay.

Software required:- keil u vision

Theory:- This experiment aims to generate a square wave for 1ms delay using the software keil u vision 5. There are two ways to create a time delay in 8051 C: 1. Using a simple for loop 2. Using the 8051 timers.

Program:-

```
#include<reg51.h>
void delay( void );
void main()
{
for(;;)
{
 P1=0x00;
 delay();
P1=0xFF;
 delay();
 }
}
void delay(void)
{
TMOD=0x01;
TL0=0X18;
TH0=0XFC;
TR0=1;
 while(TF0==0);
```

Procedure:- The procedure for the program of storing a data in accumulator is as follows. \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow

File new Program(code) Save Add existing files to source group 1 Build Start Debug Run Analysis Window Set up New Write P1 Close Run Debug the process

Output:-



Observation:-

From the above output it is observed that the square wave is generated in the analysis window.

Conclusion:-

The experiment to to generate a square wave for 1ms delay using keil u vision is done successfully

Experiment-9

Aim of experiment :- Write a C program to send the data serially.

Software required:- keil u vision

Theory:- This experiment aims to send the data serially using the software keil u vision 5.

Program:-

```
#include<reg51.h>
void main()
       {
                     TMOD=0X20;
                     TH1=0XFA;
                     SCON=0X50;
                     TR1=1;
                     while(1)
                            {
                                                    SBUF='A';
                                                                                                                              while(TI==0)
                                                                                                                            TI=0;
                          }
       }
                                                                                                                                                                                                                                                                                                      \begin{array}{c} \xrightarrow{\text{Program(code)}} \xrightarrow{\text{Save}} \xrightarrow{\text{Add existing files to source group 1}} \xrightarrow{\text{Program(code)}} \xrightarrow{\text{Save}} \xrightarrow{\text{Add existing files to source group 1}} \xrightarrow{\text{Save}} \xrightarrow{\text{Add existing files to source group 1}} \xrightarrow{\text{Check the source group 1}} \xrightarrow{\text{Save}} \xrightarrow{\text{Save}} \xrightarrow{\text{Save}} \xrightarrow{\text{Save}} \xrightarrow{\text{Save}} \xrightarrow{\text{Check the source group 1}} \xrightarrow{\text{Save}} \xrightarrow{\text{Save}} \xrightarrow{\text{Save}} \xrightarrow{\text{Save}} \xrightarrow{\text{Save}} \xrightarrow{\text{Save}} \xrightarrow{\text{Check the source group 1}} \xrightarrow{\text{Save}} \xrightarrow{\text{Sa
Procedure:-
  Build
                                                                                                                                                                Start Debug
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         Check the
output Debug the process
```

Output:-

D:\cutm\cutm.uvp	oroj - µVision		x
File Edit View	Project Flash	Debug Peripherals Tools SVCS Window Help	
🗋 🍋 😂 🕼	8 4 🔼		
👫 🗉 🚳 🤅	0.0.0		
Registers	P 🖬	Disassembly	P 🔀
Register	Value	13:)	^
E Regs r0 r1 r2 r3 r4 r5 r6 r7 Sys ar8 b8 a8 r6 r7 r7 s8 b8 c8 b8 c8	0x00 0x00	<pre>13: 000 R0,#IDATALEN - 1 C:0x0815 7877 MOV R0,#IDATALEN - 1 13: C:0x0815 7877 MOV R0,#Ox77 13: C:0x0817 E4 CLR A 13: IDATALOP: MOV @R0,A 13: DJNZ R0,IDATALOOP C:0x0819 DEFD DJNZ R0,IDATALOOP C:0x0819 DEFD DJNZ R0,IDATALOOP(C:0818) 185: MOV S8,#251CK-1 185: 185: 1 SARUPASI S S S S S S S S S S S S S S S S S S</pre>	• • ×
UART #1			
Алалалалалалалал алалалалалалал алалалал	AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA		A A A A A A A
@ []		Central 1998 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	1

Observation:-

From the above output it is observed that the data is sent serially.

Conclusion:-

The experiment to to generate a square wave for 1ms delay using keil u vision is done successfully

Experiment-10

Aim of experiment:- Write a C program to receive the data serially

Software required:- keil u vision

Theory:- This experiment aims to receive the data serially using the software keil u vision 5.

Program:-

```
#include<reg51.h>
void main()
{
unsigned char mydata;
TMOD=0x20;
TH1=0xFA;
SCON=0X50;
TR1=1;
while(1)
{
while(RI==0);
mydata=SBUF;
P1=mydata;
RI=0;
}
}
```

Procedure:- File new Program(code) Save Add existing files to source group 1 Build Start Debug Run View Serial Window UART#1 Check the output Debug the process

FOR 9 INPUT

D:\cutm\cutm	.uvproj - µVision			-	and the second in	start day 1 kg	1000		
File Edit Vier	w Project Flash	Debug Peripherals Tools	SVCS Wind	ow Help					
D 😂 🖬 🕯	3 5 8	9 e 🗢 🔿 🖗 😤 🤋	、際目達	🗏 //= //= 🔯 ACC		🗟 🥐 🔯 🔍) o 🔗 🍓 🔳 🖬	• •	
Rift 📃 🚳	() () () ()	🔹 🖸 🖸 🗖 🖉 🕹	a • 🔲 • [💽 🖬 • 🗔 • 🔜 •	** •				
Registers	Ŧ 🖸	Disassembly							‡ 🖸
Register Regs r0 r1	0x00 0x00	3: { 4: unsigned c 5: TMOD=0x20; C:0x0800 758920 6: TH1=0xFA;	MOV	1; TMOD(0x89),#0x2	0				
r2 r3 r5 r6 77 ⊡ Sys a b	6x00 0x00 0x00 0x00 0x00 0x41 0x00 0x00 0	C:0x0803 758DFA 7: SCON=0X50; C:0x0806 758050 8: TR1=1; C:0x0809 D28E 9: while (1) 10: { 11: while (RI== C:0x080B 3098FD	MOV MOV SETB	TH1 (0x8D), #0xFA SCON (0x98), #0x5 TR1 (0x88.6) RI (0x98.0), C:08	0			Parallel Port 1 33 Port 1 93 Port 1 91 Port 1 97 Prox 0 1000000000000000000000000000000000000	
sp max	0x07	•							•
dptr PC S states sec	0x0000 C:0x080B 100569401 50.28470050 0x00	STARTUPAS1 Ifincludexr Z void main(3 (4 unsigned c) 5 ATMOD=0x20 6 TH1=0xFA; 7 SCON=0X50; 8 TR1=1; 9 while(1)	10.C* eg51.h> har mydata	13					. ★ X
UART #1									<u> </u>
4									•
							Simulation	t1: 320.68708350 sec L:4 C:22	CAP NUM SCRL OVR R/W
		<i>(</i>) 🚺 🚺		\$	0			Links ^w Desktop ^w	▲ 🛱 💸 ♦) 6:18 PM 11/30/2014

FOR 8 INPUT

Output:-		
D:\cutm\cutm.uvproj - µVision	Contraction Contraction Contraction	
File Edit View Project Flash	sh Debug Peripherals Tools SVCS Window Help	
🗋 🗋 🖨 🖉 🛛 🕹 🛍 🗎	*) ~ (* * *) (* * * //: //:: //:: //:: * * //: //:: * * //: //:	- ~
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1) + D 🔯 🖬 🚍 🖓 • 💷 • 🔯 • 💷 • 🔯 • 🎌 •	
Registers 📮 🖬	Disassembly	a 🖬
Register Value □ Register 0 □ 0 0x00 □ 0 0x00 □ 12 0x00 □ -72 0x00 □ -73 0x00 □ -76 0x00 □ -76 0x00 □ -77 0x41	3: (4: unsigned char mydata; 5: TMOD=0x20; C:0x0800 758920 MOV TMOD(0x89),#0x20 6: TH1=0xFA; C:0x0803 758DFA MOV TH1(0x8D),#0xFA 7: SCON=0X50; C:0x0806 759850 MOV SCON(0x98),#0x50 8: TR1=1; C:0x0809 D28E SETB TR1(0x88.6) 9: wh11e(1)	Parallel Port 1 23 Port 1 7 Bits 0 P1: [0:33 7 7 [0:07/07] - 1 Pre: [0:33 7 [0:07/07] - 1
a 0x00 b 0x00 sp 0x07	10: { 11: while (RI==0); 	*
sp_max 0x07		,
 	<pre>STARUEAS: Disc i fincludecreg51.h> 2 Void main() 3 (4 unsigned ohar mydata; 5 ATMOD=0x20; 6 TH=0xFA; 7 SCON=0X50; 8 TR1=1; 9 while(1)</pre>	××
UART #1		
		*
	Simulation	11: 239.77412300 sec L:4 C:22 CAP NUM SCRL OVR R/W
		Links " Desktop " 🔺 🚰 🕷 🕷 🕪 6:17 PM 11/30/2014

Observation:-

From the above output it is observed that the data as input 8 (in ASCII as 38) and input 9 (in ASCII as 39) is received serially

Conclusion:-

The experiment to receive the data serially using Keil u vision is done successfully

Experiment-11

Aim of the experiment:- Write a C program to convert packed BCD 0x29 to ASCII and Display the bytes on P1and P2.

Software required:- keil u vision

Theory:- This experiment aims to convert packed BCD 0x29 to ASCII. To covert packed BCD to ASCII, it must be converted to unpacked BCD is tagged with 00110000(30H).

Packed BCD	Unpacked BCD	ASCII
0x29	0x02,0x09	0x32,0x39
00101001	00000010, 00001001	00110010,00111001

Program:-

```
#include <reg51.h>
void main(void)
    ł
 unsigned char x,y,z;
unsigned char mybyte=0x29;
 x=mybyte&0x0F;
 P1=x|0x30;
 y=mybyte&0xF0;
 y=y>>4;
P2=y|0x30;
   }
Procedure:- File new Program(code) \rightarrow Save \rightarrow Add existing files to source \rightarrow Deviation of the source \rightarrow De
                                                                                                 Build Start Debug Run Peripherals Port1
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              Peripherals
 group 1
Port2 Stop debug the process
```

Output:-



Observation:-

From the above output it is observed that packed BCD 0x29 is converted to ASCII and Port2 displays 0x32 (00110010) and Port1 displays 0x39 (00111001).

Conclusion:-

The experiment to convert packed BCD 0x29 to ASCII using keil u vision successfully.

Experiment-12

Aim of the experiment:- Write a C program to convert the hex to decimal and display the digits on P0, P1, and P2.

Software required:- keil u vision

Theory:- One of the most widely used conversions is the binary to decimal coversion.in devices such as ADC(Analog to Digital Conversion) chips, the data is provided to the microcontroller in binary. In order to display binary data we need to convert it to decimal and then to ASCII. Since the hexadecimal format is a convenient way of representing binary data we refer to the binary data as hex. This experiment aims to to convert the hex to decimal using the software keil u vision 5. Firstly the header files reg51.H is declared for the intended 8051. Then the main function starts. One way to do this is to divide it by 10 and keep the remainder.

Program:-

```
#include <reg51.h>
void main(void)
{
unsigned char x,binbyte,d1,d2,d3;
binbyte=0xFD;
x=binbyte/10;
d1=binbyte%10;
d2=x%10;
d3=x/10;
P0=d1;
P1=d2;
P2=d3;
}
Procedure:- File new Program(code) Save Add existing files to source
group 1 Build Start Debug Run Peripherals Port0 Peripherals Port1
Peripherals Port2 Debug the process
```

Output:-



Observation:-

From the above output it is observed that the binary data 00-ff H is converted to decimal which will give us 000-255. In this program binary data 11111101 or FD H is converted to decimal 253 and the output is checked at Port0, Port1 and Port2.

Conclusion:-

The experiment to convert the hex to decimal and displaying the digits on P0, P1, and P2 using keil u vision successfully.