MIDDLE EAST TECHNICAL UNIVERSITY

ELECTRICAL AND ELECTRONICS ENGINEERING DEPARTMENT

EE447

Introduction to Microprocessors

Laboratory Manual

Course Instructors:
Prof. Dr. Gözde B. Akar
Prof. Dr. İlkay Ulusoy

Laboratory Assistants:
Ece Selin Böncü
Cumhur Çakmak
Yeti Ziya Gürbüz

October 2018

Contents

Course Instructors

Name	e-mail	Room
Prof. Dr. Gözde B. Akar	bozdagi@eee.metu.edu.tr	D-121/1
Prof. Dr. İlkay Ulusoy	ilkay@metu.edu.tr	EA-410

Laboratory Assistants

Name	e-mail	Room
Ece Selin Böncü	boncu@metu.edu.tr	EA-404 / ARC-202 / DB-14
Cumhur Çakmak	ccakmak@metu.edu.tr	C-206
Yeti Ziya Gürbüz	ygurbuz@metu.edu.tr	EA-404

1 Laboratory Regulations

This laboratory is a very important part of *EE447 - Introduction to Microprocessors* course in order to thoroughly understand the concepts given in lectures. By attending experiments and completing all the work, the key concepts and most of the abstract parts of the lectures can be grasped very easily. It also makes up %25 of all grades of the course. Thus, it is important to know the regulating rules of this laboratory work for both a better understanding and for your grades.

There are rules for the regulation of *EE447 Laboratory*. These rules are strict; and by taking *EE447 - Introduction to Microprocessors* course, you will be considered that you have understood and accepted all the rules stated below.

1.1 Rules

The rules for EE447 - Introduction to Microprocessors Laboratory is given below, please read thoroughly:

1. The manuals of the all laboratory works will be available one week before the first laboratory work.
2. A preliminary work which is to be detailed in the experiment manual has to be prepared for each experiment.
3. There will be a quiz for each experiment in the week of each experiment. The quizzes are to be held in the class hours of EE447 course on Tuesdays. **You will NOT be allowed** to attend the relevant laboratory session if you miss the quiz.
4. A quiz includes both the related laboratory work and the course material covered up to the date when the corresponding quiz is held. It is important to note that the laboratory work covers the related topics from the course and those topics are the ones that are exploited during the preliminary work preparation.
5. Depending on the quiz grade of the laboratory work part, the quizzes may penalize the performance grade of the related laboratory work. Students whose quiz grades for the laboratory work part are **less than 4** will get their performance grade from the related experiment multiplied by the ratio (quiz grade)/10. The performance grades of the students whose quiz grades are **greater than or equal to 4 and less than 6** will be **halved**. There will be no penalty for the students whose grades are **more than or equal to 6**. That is to say, assuming that your quiz grade for the laboratory work part is Q and your laboratory performance grade for the corresponding experiment is P, your penalized performance grade, $\hat{P}$, becomes:

$$\hat{P} = \begin{cases} \frac{Q}{10}\,P, & Q < 4 \\ 0.5\,P, & 4 \leqslant Q < 6 \\ P, & Q \geqslant 6 \end{cases} \tag{1}$$

6. The preliminary works are to be collected before the quizzes.
7. Preliminary work is a crucial part of the laboratory work in both preparing for the experiment and understanding the concepts to be covered in the corresponding experiment. Thus, **You will NOT be allowed** to attend the relevant laboratory session without any preliminary work. Partially done preliminary work reports are acceptable if and only if at least the %30 of the total work is completed. Redrawings or rewritings of the given material will not be considered within the aforementioned %30.
8. You are expected to prepare your preliminary work on your own.
9. Experiments are to be performed in pairs which are to be formed randomly each experiment.
10. Sharing any form of information during a laboratory session is strictly forbidden.
11. No extra time will be given to the latecomers and no one is allowed to take the laboratory session after 20 minutes past the beginning of the session.
12. Leaving laboratory room during a session is not allowed except for the emergency situations.

13. Transfer between laboratory sessions (e.g. from *Thursday Afternoon* to *Friday Morning*) will NOT be allowed.
14. Cheating is an important issue regarding ALL steps of the laboratory work that involve preliminary work and laboratory performance. Disciplinary action is to be taken in case of any cheating and cheating attempt. Please read the subsection 1.2 for detailed information about cheating.
15. Your codes have to be well commented. The codes lacking of comments will not be evaluated. Please read the subsection 1.3 for detailed information about coding and commenting.
16. Students with officially documented legal excuses will be allowed to take make-ups. Only academical permissions (given by University), signed confirmation from the instructors for any exam clashes, and METU Health and Counseling Center (MEDIKO) will be regarded as valid documents for the right to take make-ups. You may take at most 3 make-ups even if you have more than 3 officially documented legal excuses.
17. All your course related e-mails should be sent to ee447coordination@gmail.com address. If you have a general question that may interest the other students, please start a discussion in ODTUClass.
18. Students who do not attend **2 experiments** without legal excuses will get **ZERO** from the laboratory. Students who do not attend **greater than or equal to 3** experiments will get **N/A** from the EE447 course. Please note that missing the quiz of an experiment or insufficient preparation of the preliminary work of an experiment is equivalent to not attending the corresponding experiment.
19. This document supersedes any other previous documents.

1.2 Cheating

You are considered to be graduate and become engineers and colleagues in 1 or 2 years time. Hence, you are expected to act according to the professionalism that is required as a METU graduate.

Copying a work from any other resource (web-page, your friend's report, older resources you have found, etc.) or sharing information, know-how or code files during sessions are considered as cheating.

Helping your friends, studying together, or any form of cooperation is encouraged -since it both fosters your relationships with others and helps you learn the topic better- as well as YOU do your own work. Creating only one report is not studying together or is not cooperation, and will NOT be accepted.

1.3 Remarks and Evaluation

Your laboratory grade is to be composed of your preliminary work grade and your laboratory session performance grade. Preliminary work requires implementation of different tasks some of which are to be experimented during the laboratory sessions. The performance grade is according to the evaluation of the tasks that are considered in the laboratory session. The evaluation is based on the functionality of your implementations and comprehensiveness of your knowledge of the related laboratory topic. If a task is not implemented in your preliminary work, then you will get ZERO performance grade from corresponding step. Moreover, implementations without comments will not be considered as valid implementations of the tasks and the corresponding task will not be evaluated. You are expected to write down explanatory comments to your code. That does not mean you should write an explanation next to each code line. What is required is explaining the functionality of the code blocks and the functionality of the representative code lines where necessary.

You can - actually are expected to - practice your implementations before attending your laboratory session so that major bugs that may cost too much time to fix can be eliminated and you are on the safer side to complete the experimental work within the required time slot. You can practice and work on the TM4C123G board either by using the EA-409 laboratory that is to be open to access 7/24 (once you get your student ID card authorized) or by purchasing your own TM4C123G board. In the following sections, more detailed information is to be provided for the former (Section 2) and the latter (Section 3) option.

2 General Information about Laboratory

As it is mentioned before, this laboratory is a very important part of *EE447 - Introduction to Microprocessors* course. The laboratory work helps you understand most of the concepts given in the lectures. Also, materializing abstract parts of the lectures will be much more easy.

The experiments will be carried out in EA-409 which is located on the 4^{th} floor of A Block of our department. EA-409 laboratory will be open to access 7/24 for you to practice. Your student ID card has to be authorized for the entrance. You may get your ID authorized by visiting the staff responsible for that (@ EA-104). Once you are authorized, you may test and debug your preliminary work prior to your laboratory session.

Figure 1: Microprocessor Laboratory, EA-409, A Building

There are different sessions of laboratory and you are assigned to one of the sessions to demonstrate your laboratory work.

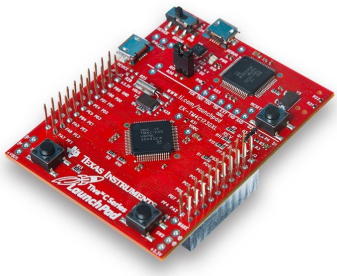
There will be a total of 6 laboratory works. These laboratory works are based on practicing the utilization of the MCU via assembly language and will be in the following subjects:

- Introduction to Laboratory with Assembly Programming
- Subroutines and Stack
- Input/Output
- Interrupts
- Timer System
- Analog-to-Digital Conversion of a Signal

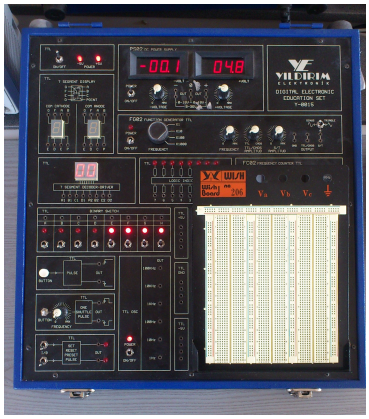
2.1 Experimental Setup

The experimental setup of EE447 Laboratory is composed of the following items:

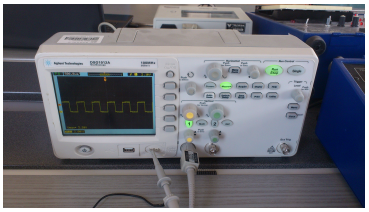
- A notebook PC
  - You may login as `"student"`, which does not require any password
- The TM4C123G LaunchPad Evaluation Kit (TM4C123G) - A board containing the MCU TM4C123GH6PM
  - TM4C123G is connected to the PC via USB data cable.
  - MCU(TM4C123GH6PM) is programmed via Keil MDK installed in the PC (to be explained in Section 3).



(a) TM4C123G Board



(b) DEES



(c) Digital Oscilloscope

Figure 2: Devices used in the laboratory

  - For some user interaction purposes, Termite which is a terminal emulator that provides serial communication between PC and MCU is used (to be explained in Section 3).

- DEES: Digital Electronics Education Set - A set that includes LED displays, switches, and a breadboard in order to construct external circuitry
- Digital Oscilloscope - Oscilloscope for debugging purposes

3 Introducing Keil Microcontroller Development Kit (MDK) and Termite the Terminal Emulator

This section introduces Keil Microcontroller Development Kit (MDK) and Termite the Terminal Emulator, which are software to be used throughout the experiments to program the MCU and to provide user interaction, respectively. The organization of the section is as follows.

First, walkthrough of the installation of Keil MDK is presented for those who consider to purchase their own TM4C123GXL LaunchPad Development Kit and practicing in their own PCs. You may skip reading that section if you are not planning to purchase your own board. However, purchasing a TM4C123GXL board might be beneficial to you in case you consider to use it for your further purposes such as EE493 - EE494 project.

The next subsection explains how to compose code projects in Keil MDK and how to program MCU in TM4C123G board via Keil MDK.

The last subsection introduces Termite terminal emulator that is to be used for some user interaction tasks in some experiments.

3.1 Installation of Keil MDK

The installation is only for Windows operating systems. The MDK contains μVision 5 IDE (Integrated Development Environment) with debugger, flash programmer and the ARM compiler toolchain. The software is free, but limited to compile, assemble, or link programs that are less than 32 Kbytes of code. The installation is to be explained step by step. Before the installation, make sure that your TM4C123GXL board is not connected to your PC since, driver installation is required prior to make a connection.

1. Download Keil MDK-ARM
   - Go to the Keil MDK-ARM product page: `https://www.keil.com/download/product/`
   - Click *MDK-ARM* (Figure 3)

Figure 3: Click *MDK-ARM*

   - Complete the form and proceed (Figure 4)

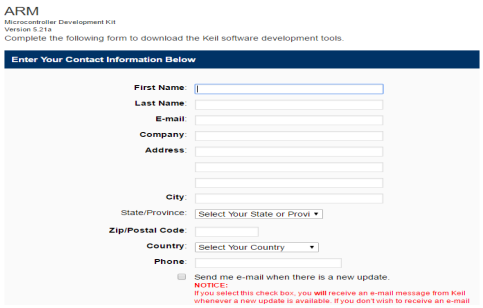


Figure 4: Fill out the form

   - Click the MDK5xx.EXE link, where xx represents the current software version (Figure 5)

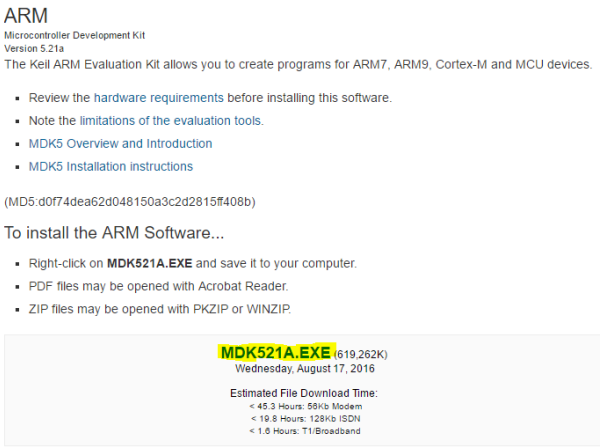


Figure 5: Click the EXE link emphasized by yellow (You might see a different version which is current)

2. Install Keil MDK
   - Run the downloaded setup file
   - Click next on the initial screen
   - Agree to the term of license and proceed
   - Optionally choose where you want to install the software
   - Fill out the required information and start the installation by clicking next
   - If you get a security prompt about installing the device software, click install
   - Installation should be completed
3. Pack Installer
   The Pack Installer is a tool to install the correct support files for the processor / device you are using. Keil μVision can be used to create programs for MANY devices other than the TI TM4C. The Pack Installer is where you choose the packs needed to support your device. **Internet connection is required at this step**.

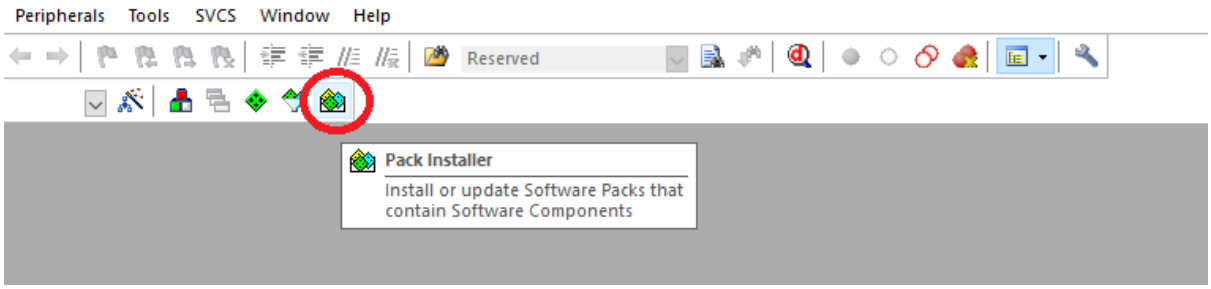


Figure 6: Click the encircled icon to open Pack Installer

You can open Pack Installer from Keil μVision by clicking the corresponding icon (Figure 6. Once it is opened, click *OK* on the first dialog and a large list of packs that support various features on various devices appears. For the first install, it might take a while for the list to fully populate. There exists a button next to each pack. The button might represent different actions to be taken according to the status of the corresponding pack. There are three typical options:

- **Up to Date:** The pack is installed and it is up to date. No action is to be taken upon clicking.

- **Install:** The pack is not installed. Clicking the button will install the pack.
- **Update:** The pack is installed and there is an update available. Clicking the button will fetch and install the required updates for the pack.

Now, you are ready to install the packs required for the TM4C123G board.

- Open Pack Installer and proceed by clicking *OK* on the first dialog
- You should see an interface similar to one provided in Figure 7

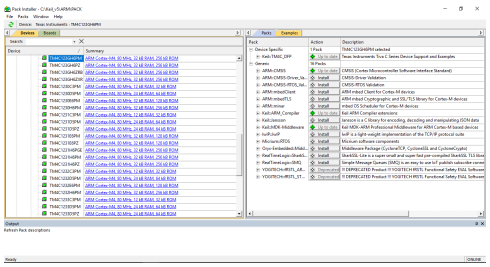


Figure 7: Layout of the Pack Installer interface

- Under *Devices* tab, seek for the TM4C123GH6PM: All Devices > Texas Instruments > Tiva C Series > TM4C123x Series > TM4C123GH6PM (Figure 8)

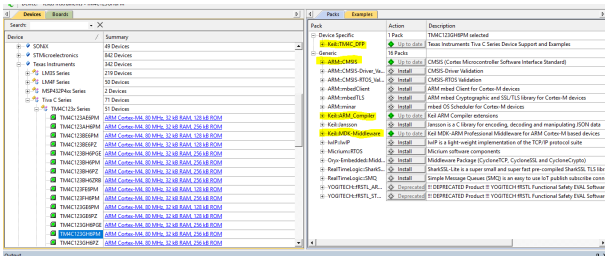


Figure 8: Packs to be installed are emphasized by yellow

- Install the following packs under the *Packs* tab: $Keil :: TM4C\_DFP$, $ARM :: CMSIS$, $Keil :: ARM\_Compiler$, $Keil :: MDK - Middleware$ (Figure 8)
- Under *Devices* tab, seek for the CM4xx (or CM41x) Mixed Signal Control Processors: All Devices > Analog Devices > CM4xx (or CM41x) Mixed Signal Control Processors (Figure 9)

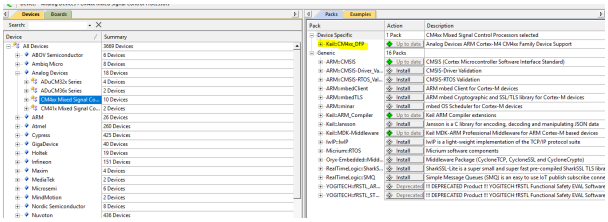


Figure 9: Packs to be installed are emphasized by yellow (Depending on the Keil version selected device under the device tab might be CM41x)

- Install the pack $Keil :: TM4xx\_DFP$ under the *Packs* tab (Figure 9)

4. Download and Install In-Circuit Debug Interface (ICDI) Drivers The TM4C123G development board contains an on-board debugger that can be used to debug programs in Keil MDK. Thus, the PC needs a driver to create a communication link between Keil MDK and the development board.

- Go to TI's Stellaris ICDI drivers page: `http://www.ti.com/tool/stellaris_icdi_drivers`
- Download the latest drivers (Figure 10)

Part Number	Buy from Texas Instruments or Third Party	Alert Me	Status	Current Version	Version Date
SW-ICDI-DRIVERS: Stellaris® ICDI Drivers - Current	Download	Alert Me	ACTIVE	v1.0	21-JAN-2016
SW-ICDI-DRIVERS-AR01: Stellaris® ICDI Drivers - OLD	Download		ACTIVE		

Figure 10: Download the latest drivers

- Extract the downloaded files to a particular location, which is to be browsed during driver installation
- Plug TM4C123G board into a USB port
- Open Device Manager from Control Panel in Windows
- Expand the *Other Devices* branch to observe *In-Circuit Debug Interface devices*
- Right-click on each ICDI device listed and select *Update Driver Software..*
- On the update wizard screen select *Browse my computer for driver software* option
- Provide the path to where the drivers are extracted to and click *Next* by *Including subfolders*
- In case of a security warning about verification of the publisher of the drivers, proceed with *Choose to Install this driver software anyway*
- Once the driver is installed, the individual ICDI device will be removed from the Other Devices branch
- Repeat the driver install for each remaining ICDI device
- At this point, you also possibly have a device under *Ports (COM & LPT)* branch
- Update this port device according to aforementioned steps, as well

Once the aforementioned steps are performed successfully, the installation is complete.

3.2 Composing Code Project with Keil MDK

This subsection introduces creating code projects and programming MCU with Keil MDK. You will be familiar with those concepts once you complete your first laboratory session. However, for a quick reference, a brief documentation is also to be presented.

First of all, you should create a folder for each project you are going to create since, Keil MDK needs a location to place the auxiliary files. When you are ready, run the μVision application, making sure you see the 2 sub-windows: the Project window on the left, the Build Output window along the bottom (Figure 11). The Build Output and Project windows can be made visible or invisible using the View menu at the top of the program.

All files associated with each of your projects, debug, and build settings are organized and saved using Project files. Each project will contain all of the assembly files created for an individual laboratory or project.

3.2.1 Composing a Project

To start a new project:

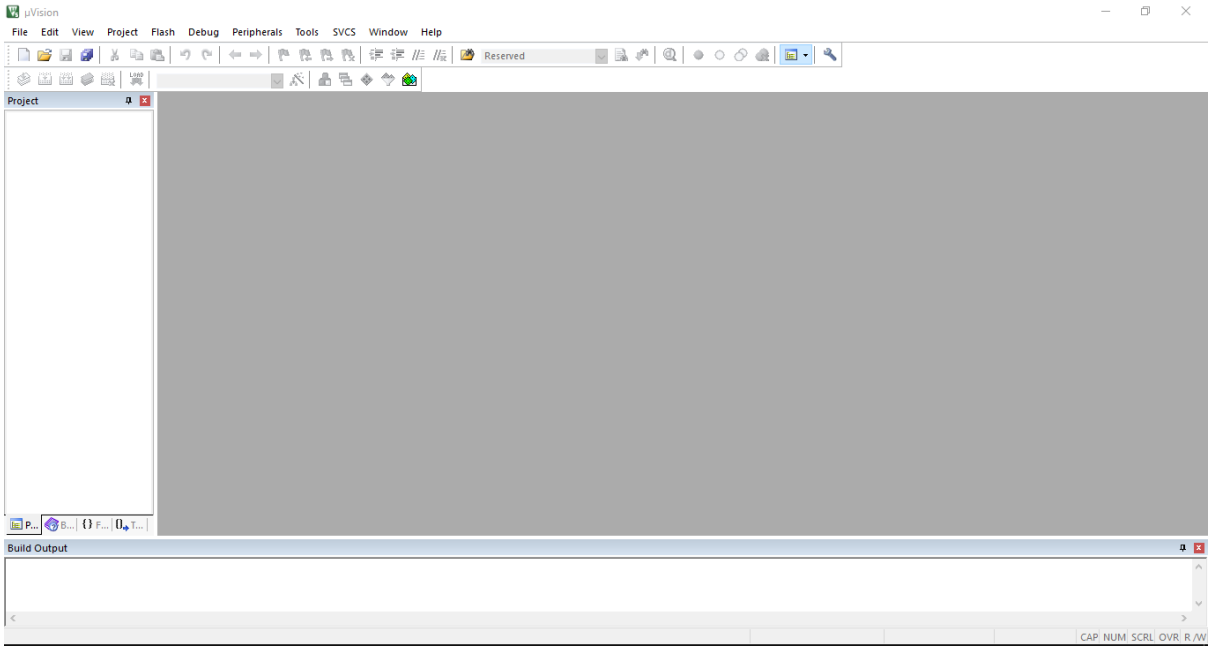


Figure 11: μVision and its sub-windows

- Go to the Project menu > New μVision Project
- Create a project folder, then navigate to the Project Folder you just created and select a name for your project
- Next, select the device you are using: In our case, the TI TM4C123GXL development board uses the TM4C123GH6PM microcontroller
- Expand the tree inside of the Target window to find the correct processor: *Texas Instruments* > *Tiva C Series* > *TM4C123x Series* > Scroll down unti you find the *TM4C123GH6PM*
- Select it and click *OK*
- The next window to pop up is the Manage Run-Time Environment window: Nothing needs to be configured here, so simply click *OK*

You will notice that the Project window now has a tree structure. Expanding the Target1 folder shows the Source Group 1 folder. The Source Group 1 folder is where all of your assembly files will be added. There are two ways to add assembly files to your project. Right-click on the Source Group folder and either select *Add New Item to Group* or *Add Existing Files to Group*.

The Add **Existing** option is for when you already have an existing assembly file that you would like to use in your project.

The Add **New Item** option is used when starting an assembly file from scratch. Selecting this option walks you through creating a new assembly file and opens the new file as a new tab in the Editor.

Any new or old program file can be opened in the Editor (like any word processor) using either the File menu or double clicking it in the Project window. Saving the files can be done by selecting *File* > *Save*, or the save button in the menu.

REMARK: If you want a file to be used in your project it MUST be added to the Source Group folder. Otherwise, μVision does NOT consider it during build time.

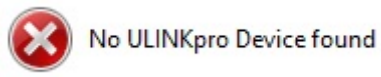
3.2.2 Programming MCU with μVision

TM4C123GH6PM microcontroller requires an initialization procedure in order to run your programs properly. This initialization is also a program that runs before your main program and performs certain tasks for the functionality of MCU. Generally, microcontroller vendors provide users with the required initialization program. You can download yours -***StartUp.s***- from ODTUCLASS. **You should add this file to your project**. For now, you might not follow what is going on with those lines of codes, however you will be able to understand each line of the ***StartUp.s*** file by the end of the semester.

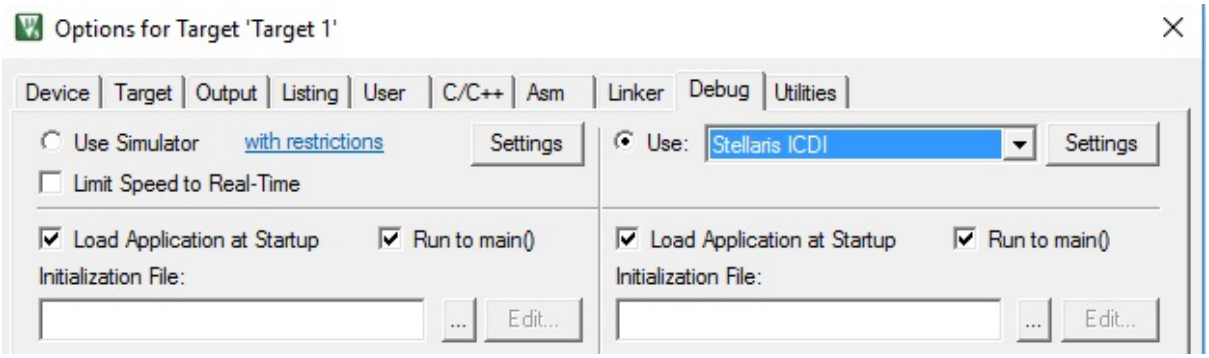
Now, assuming that you have added ***StartUp.s*** file to your project and have written your own assembly code, e.g. ***Lab1.s***; the codes in your project need to be built (assembled) in order to be translated into machine code. This can be done using *Project > Build Target*. When the program is built, other files with the same name, but with other file extensions are created. The .lst file is a common listing file, which provides physical address information with operation and operand information. It also gives the error information above the line having the error. Make sure to edit the main program (*.s) and not the .lst file. Upon a successful build, the Build Output window will display something similar to the following:

```
Build target 'Target 1'
assembling Lab1.s...
linking...
Program Size: Code=40 RO-data=0 RW-data=0 ZI-data=1024
".\Objects\Lab1.axf" - 0 Error(s), 0 Warning(s).
```

Having built your program, the next step in the process is to download it to your board. Enter the drop-down menu *Flash* at the top and click *Download*. This can be done also from the download icon on the toolbar or by pressing F8. The Build Output window shows messages about the download progress. If the pop-up window shown below on left appears and/or Build Output window shows message on right, the driver may not be set up properly.



In order to use the correct driver for TM4C123GH6PM microcontroller select *Project → Options for target > Debug* and choose Stellaris ICDI from drop down menu like shown below.



3.3 Installation of Termite the Terminal Emulator

In several experiments, MCU will interact with the user through serial communication. The terminal emulator is similar to a console screen where output from the TM4C123G board can be displayed in and user input to be sent to the board from keyboard can be given to. Hence, throughout the experiments, Termite from Compuphase is utilized as the terminal emulator.

To install Termite:

- Go to the Compuphase Termite page: `http://www.compuphase.com/software_termite.htm`
- Download the latest program only version (Figure 12

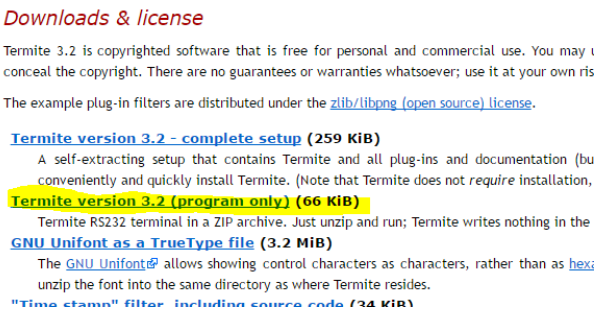


Figure 12: Download the latest program only version

- Extract the downloaded **.zip* file
- Now, you are ready to run the emulator

After the installation, the emulator should be set up to communicate with the development board. To set up the emulator:

- First find the communication port (COM Port) your board is using on your PC (from *Device Manager*)
  - Connect your board to the PC using the supplied USB cable.
  - Go to *Control Panel* and open *Device Manager*
  - Expand Ports (COM & LPT) and find Stellaris Virtual Serial Port (COM x) where x is the number of the communication port
- Open Termite and change the Settings to use the correct COM port
- Click *Settings* and under *Port*, select the number your board is using
- Set Transmitted text to append nothing.
- The rest of the settings should remain default at 9600 baud, 8-bit, 1-stop, no parity

At this point, you are ready to use monitor utility subroutines to have an interaction with the board. Monitor utility subroutines are explained in Section 4.

Once you complete the installation of both Keil MDK and Termite, you can start practicing on your own board and computer.

4 Monitor Utility Subroutines

The monitor utility subroutines are a set of routines that allow communication between the TM4C123G board and a serial window - i.e. Termite 3.3. Those subroutines are to be used in several experiments and the source codes are to be provided to you via ODTUCLASS. You have to add the source codes of the subroutines that you are going to use to your project. Though the explanations of the subroutines are brief in this section, you will understand them better as you proceed in your laboratory sessions. This documentation is for quick reference.

InChar

The InChar subroutine inputs the ASCII character from the keyboard of the PC to register R5. For example, the upper case letter 'A' is loaded into register R5 as 0x41. The subroutine InChar loops until you press a key.

OutChar

The OutChar subroutine can be considered as the reverse of the InChar subroutine. The OutChar subroutine sends the ASCII character value of which is placed in register R5 to the PC through UART0 (serial communication).

OutStr

The other useful subroutine is OutStr. This subroutine outputs a string of ASCII bytes pointed to by the address in index register R5 until the end of transmission character, 0x04, is reached. If you want to display different messages to the terminal screen when certain conditions are met, you can have the string stored in memory with the 0x04 after the last character.

Out1BSP

Out1BSP subroutine takes the hex number located in register R5 and outputs it to the terminal emulator followed by space. Note the difference between calling functions OutChar, OutStr and Out1BSP. OutChar expects, in index register R5, the ASCII value of the character to be printed. OutStr expects, in index register R5, the address of the first character in the string to be printed. Out1BSP expects, in index register R5, the value of number to be printed. Namely, Out1BSP converts a binary byte stored in R5 to two ASCII characters and outputs with space concatenated.