Experiment No 1

Introduction to STM32 Micro-Controller, ARM Architecture and its Programming

Objectives:

Experiment No 1 will give a brief introduction to students about Embedded Systems. Moreover, architecture and programming of STM32 ARM microcontroller will be discussed in detail.

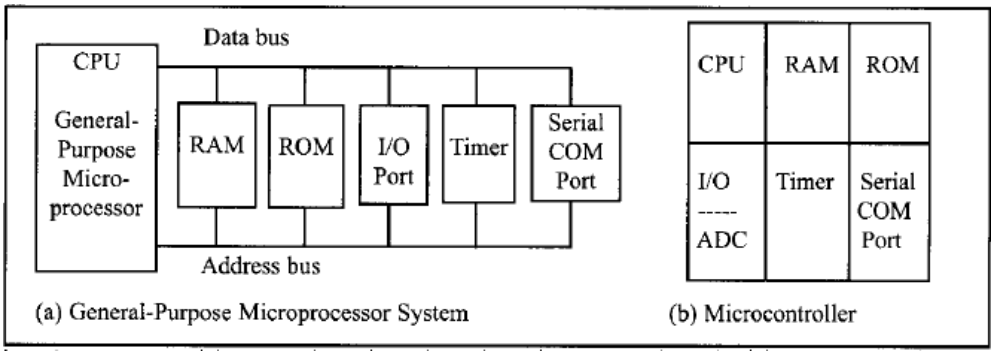
Tasks in this Lab include:

1. Introduction to processor architectures.
2. Architecture of STM32 ARM microcontroller
3. Programming of STM32 ARM using C Language

Introduction to Embedded Systems

An embedded system is a system that has software embedded into computer-hardware, which makes a system dedicated for an application (s) or specific part of an application or product or part of a larger system. Microprocessor and microcontrollers are widely used in embedded system products. An embedded system is controlled by its own internal microprocessor ormicrocontroller. A microcontroller differs from microprocessor, as it has a CPU along with fixed amountof RAM, ROM, I/O Ports, and a timer all embedded on a single chip. This is not the case with microprocessor. It has external RAM, ROM, I/O Ports which makes these systems much bulkier and more expensive, of course with providing more options. Figure 1 is representing the contrast of both microcontroller and microprocessor

Figure 1: Difference between Microprocessor & Microcontroller



There are five major Microcontrollers which are used in various applications as per their capabilities and feasibility to perform desired task such as Motorola 68HC08/68HC11, Intel's 8051, Atmel's AVR, Zilog's Z8 and PIC from Microchip Technology. Most of the activities, in this Lab-Course will be performed on **Keil uVision5** STM32 ARM microcontroller.

Introduction to STM32 ARM Microcontroller

Advanced RISC Machine termed as ARM processor and it is developed by ARM holding. During 1980 Acorn Computers Ltd first developed the Acorn RISC Machine architecture and it used in computers. During 1990 when the Acorn company was incorporated the ARM termed as Advanced RISC Machine.

ARM holding owns the patents of ARM Architectures and licenses the companies like Philips NXP, Atmel, Analog devices, Apple, Broad com, ST Micro Electronics etc… to manufacture chips with ARM core.

ARM Features

ARM processor family has dozens of features some key features are listed.

- A large register file
- A Load/Store Architecture
- Fixed Instruction bit size/Uniform length Instruction field
- Simple addressing mode
- Pipeline execution

- Single cycle Execution
- Hardware Virtualization/ Easy to prototype
- Control over both ALU & Shifter
- Three different Data Sizes & Instruction (32-bit ARM Ins, 16-bit Thumb Ins, 8-bit Jazelle execution)

ARM Controller

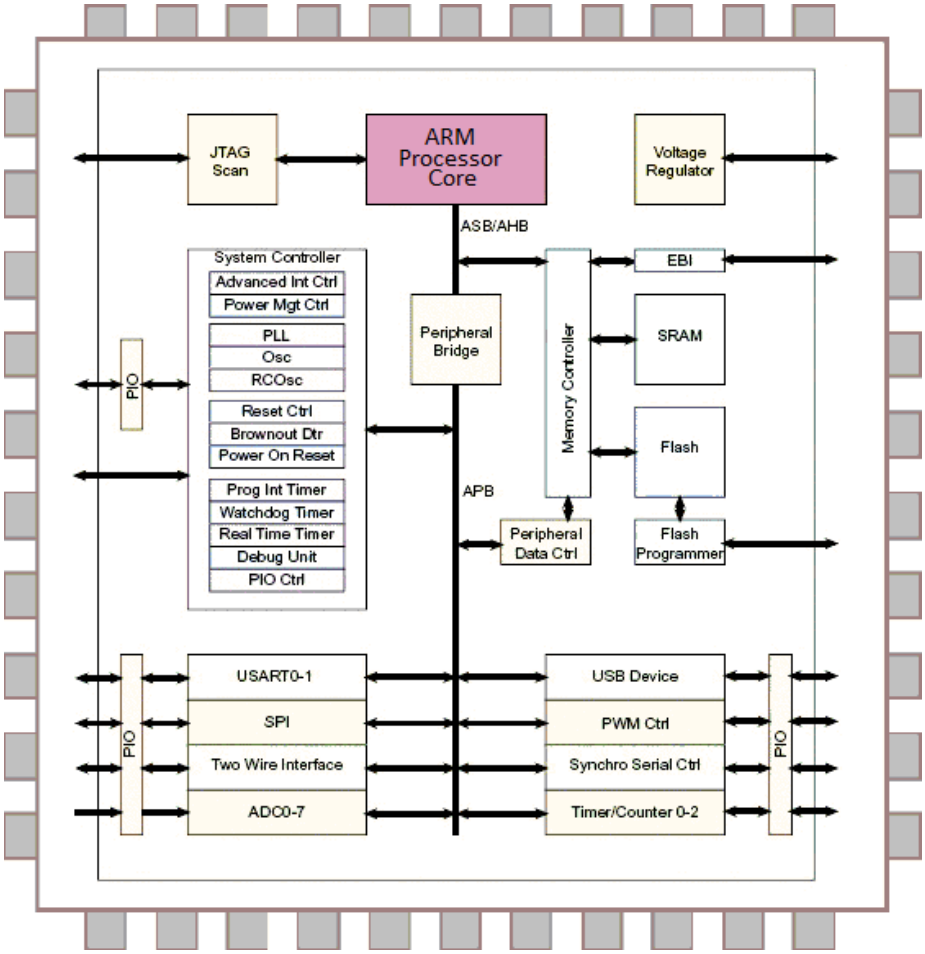


Figure 2: ARM Architecture

ARM Architecture Types

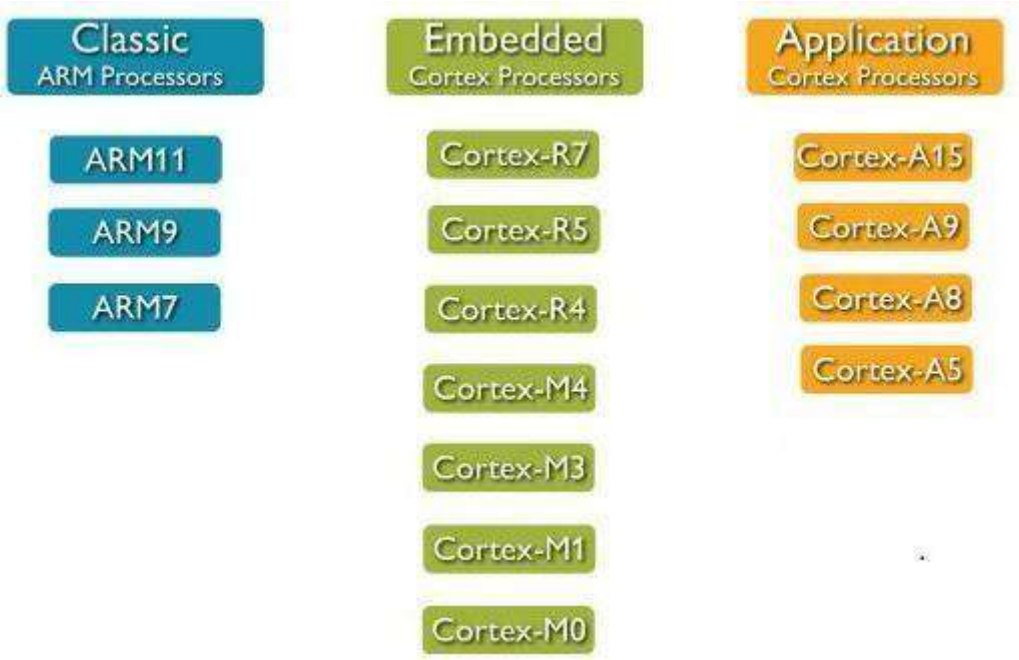


Figure 3: ARM Architecture Types

Difference between Cortex-M0, M0+, M1, M3 and M4

Cortex-M0
The ARM Cortex™-M0 processor is the smallest ARM processor available. The exceptionally small silicon area, low power and minimal code footprint of the processor enables developers to achieve 32-bit performance at an 8-bit price point.

- Supports 3-stage pipeline, thumb2, hardware-single-cycle (32×32) multiply hardware.
- Supports 1 NMI and 32 physical interrupts.
- Has only 56 instructions and has 'C-friendly' architecture.

Cortex-M0+
The ARM Cortex™-M0+ processor is an adaption of Cortex-M0 but with more improved performance and reduced energy footprint.

- Supports 2-stage pipeline, thumb2, hardware-single-cycle (32×32) multiply hardware.

Cortex-M1

The ARM Cortex™-M1 processor is the first ARM processor designed specifically for implementation in FPGAs.

- Supports 3-stage pipeline, thumb2, big-endian & little-endian configuration.

Cortex-M3

The ARM Cortex™-M3 processor is the industry-leading 32-bit processor for highlydeterministic real-time applications, specifically developed to enable partners to develop high- performance low-cost platforms for a broad range of devices including microcontrollers, automotive body systems, industrial control systems and wireless networking and sensors.

- The Cortex-M3 NVIC is highly configurable at design time to deliver up to 240 system interrupts with individual priorities, dynamic re-prioritization and integrated system clock.
- Supports 1 NMI and 240 physical interrupts with 8 to 256 level priorities.
- Supports hardware divide, single cycle-multiply and saturated math support.

Cortex-M4

The ARM Cortex-M4 processor with FPU is the latest generation of ARM processors for embedded systems. It was developed to provide a low-cost platform that meets the needs of MCU implementation, with a reduced pin count and low-power consumption, while delivering outstanding computational performance and an advanced response to interrupts. The ARMCortex- M4 32-bit RISC processor with FPU features exceptional code-efficiency, delivering the high- performance expected from an ARM core in the memory size usually associated with 8- and 16- bit devices. The processor supports a set of DSP instructions which allow efficient signal processing and complex algorithm execution. Its single precision FPU (floating point unit)speeds up software development by using metalanguage development tools, while avoiding saturation.

Features of STM32F1 ARM Microcontroller

1. STM32F103 Microcontrollers use the Cortex-M3 core, with a maximum CPU speed of 72 MHz. The portfolio covers from 16 Kbytes to 1 Mbyte of Flash with motor control peripherals, USB full-speed interface and CAN.
2. 51 fast I/O ports. All I/O ports can be mapped to 16 external interrupts, and almost all ports allow 5V signal input. Each port can be configured by software as output (push-pull or open-drain), input (with or without pull-up or pull-down) or other peripheral function ports.
3. 2 12-bit analog-to-digital converters, 16 external input channels, the conversion rate can reach 1MHz, and the conversion range is 0~36V; With dual sampling and holding function; A temperature sensor is embedded inside, which can conveniently measure the temperature of the processor.

4. The flexible 7-channel general-purpose DMA can manage data transfer from memory to memory, device to memory, and memory to device without any intervention by the CPU. Data can be moved quickly through DMA, and CPU resources can be used for other operations. The DMA controller supports the management of the ring buffer, which avoids the interrupt generated when the controller transfer reaches the end of the buffer. The peripherals it supports include timers, ADCs, SPI, I2C, and USART.

5. Debug mode: Support standard 20-pin JTAG simulation debugging and serial single-wire debugging (SWD) function for Cortex-M3 core. Usually the default debugging interface is the JTAG interface.

6. The STM32F103 series microprocessor contains 7 timers.

7. Containing abundant communication interfaces: three USART asynchronous serial communication interfaces, two I2C interfaces, two SPI interfaces, one CAN interface and one USB interface, which provide a guarantee for data communication.

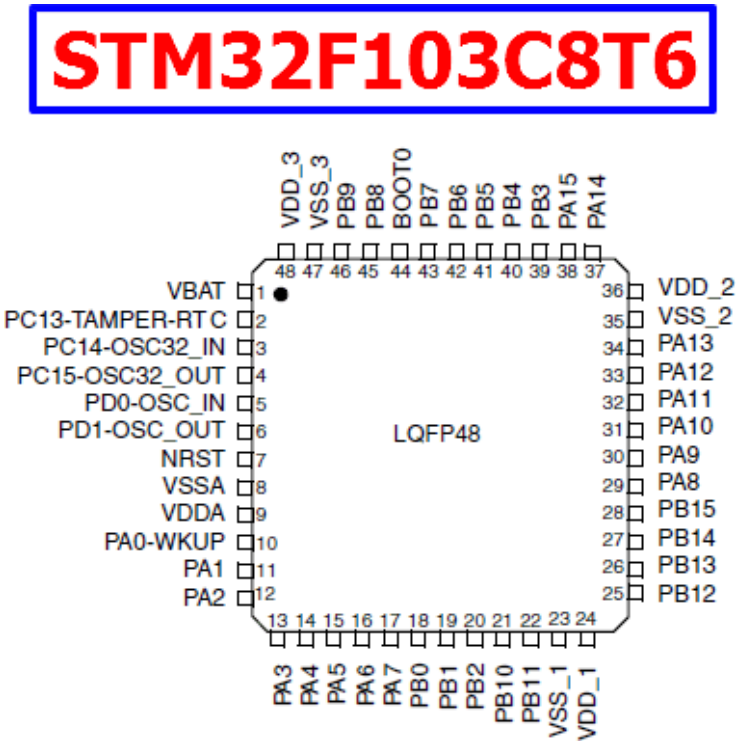


Figure 4: Microcontroller Pin Configuration

In the STM32 there are four GPIO banks GPIOA – GPIOD. When using the library there are four considerations: GPIO Mode, GPIO Output Type GPIO speed – GPIO pull up/pull down resistors

- **STM32F1 Timer:** stm32f103C8 have 4 timers inside. Table below shows their description.

Timer	Counter resolution	Counter type	Prescaler factor	DMA request generation	Capture/compare channels	Complementary outputs
TIM1	16-bit	Up, down, up/down	Any integer between 1 and 65536	Yes	4	Yes
TIM2, TIM3, TIM4	16-bit	Up, down, up/down	Any integer between 1 and 65536	Yes	4	No

Figure 5: STM32F103 Timers

- **ADC:** ADC stands for Analog to Digital Converter. Microcontrollers are digital component, so they only understand discrete/digital signals. Therefore, if you want to read analog voltage that can be from various sensors, you need an ADC. STM32F103 hastwo 12-bit ADCs that can work independently. Every ADC have 10 channels.

- **Communication Options:** STM32f1 has five data transfer modules embedded in it.

  - SPI
  - USART
  - I2C
  - USB
  - CAN

- **External Interrupt:** The external interrupt/event controller consists of up to 20 edge detectors in connectivity line devices, or 19 edge detectors in other devices for generating event/interrupt requests. Each input line can be independently configured to select the type (event or interrupt) and the corresponding trigger event (rising or falling or both). Each line can also masked independently. A pending register maintains the status line of the interrupt requests.

- **DMA controller (DMA):** The flexible 7-channel general-purpose DMA is able to manage memory-to-memory, Peripheral-to-memory and memory-to-peripheral transfers. The DMA controller supports Circular buffer management avoiding the generation of interrupts when the controller reaches the end of the buffer. Each channel is connected to dedicated hardware DMA requests, with support for software trigger on each channel. Configuration is made by software and transfer sizes between source and destination are independent. The DMA can be used with the main peripherals: SPI, I2C, USART,general-purpose and advanced-control timers TIMx and ADC.

Programming STM32F1 ARM:

You can write any Programming algorithm for STM32F1 using assembly or C. Here in Microprocessor and Computer Architecture Lab, we will prefer to program in C language because of following reasons

- it is easier and less time consuming to write a program in C
- C is easier to modify and update
- We can use code available in function libraries
- C code is portable to other microcontrollers with little modification.

IDEs for Programming of Discovery Board

Multiple IDEs can be used for programming the STM32F103. Some include:

1. Keil uVision5
2. Atollic
3. STM32 Cube MX
4. LabVIEW

Here we will use Keil software.



Figure 6: KEIL Software

Installing Keil uVision5

Keil u Vision5 can be downloaded from the following site:

https://www.keil.com/

Before you install your driver 1 LED (PWR LED) will turn on red

stable After you install **st-link_v2_usbdriver**, update your driver.

Device Manager → Other devices → STM32 STLink

Right click and **Update Driver Software…** and choose **Search automatically for updated driver software**

After updating, driver will be shown in **Universal Serial Bus controllers** or **Universal Serial Bus devices** as a **STMicroelectronics STLink dongle**

Now the **COM LED** won't be blink and will be stable red as **PWR LED**.

For next, install Keil uVision5 which you downloaded before. Now you probably need a device pack. Get the latest version from link below (for now **Keil.STM32F1xx_DFP.2.3.0.Pack**).Keil -> STMicroelectronics STM32F4 Series Device Support, Drivers and Examples -> Download

http://www.keil.com/dd2/pack/#/eula-container

After downloading, open Keil uVision5 and install your pack as shown below: Click on **Pack Installer**.

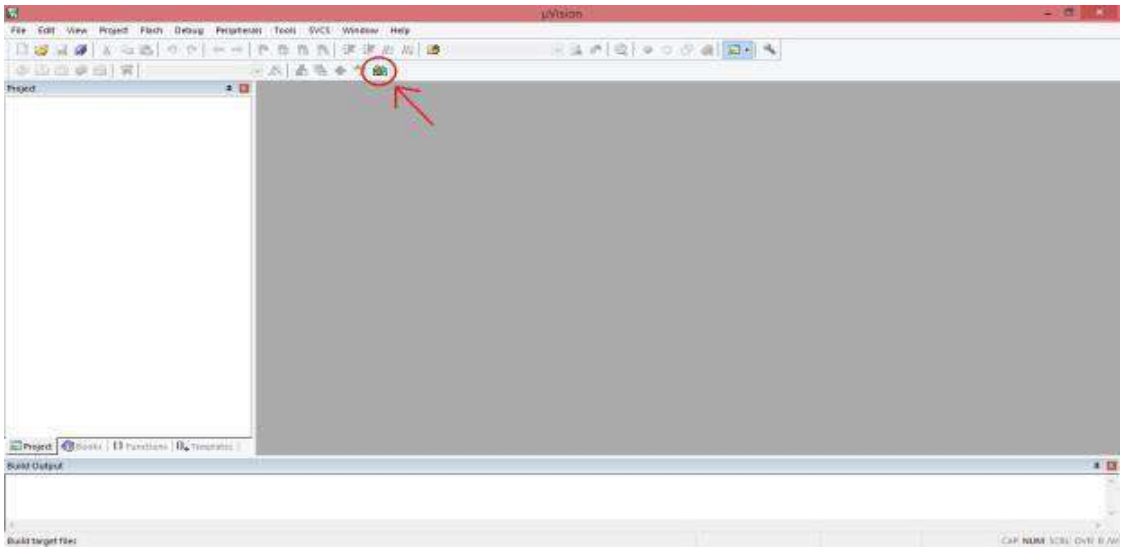


Figure 7: Keil uVision5 main window

Refresh your Packs (File -> Refresh) (To Refresh you must be online). Then Import the pack you downloaded already (File -> Import…). After you imported the pack you can see your pack as a **STM32F103C8T6**.

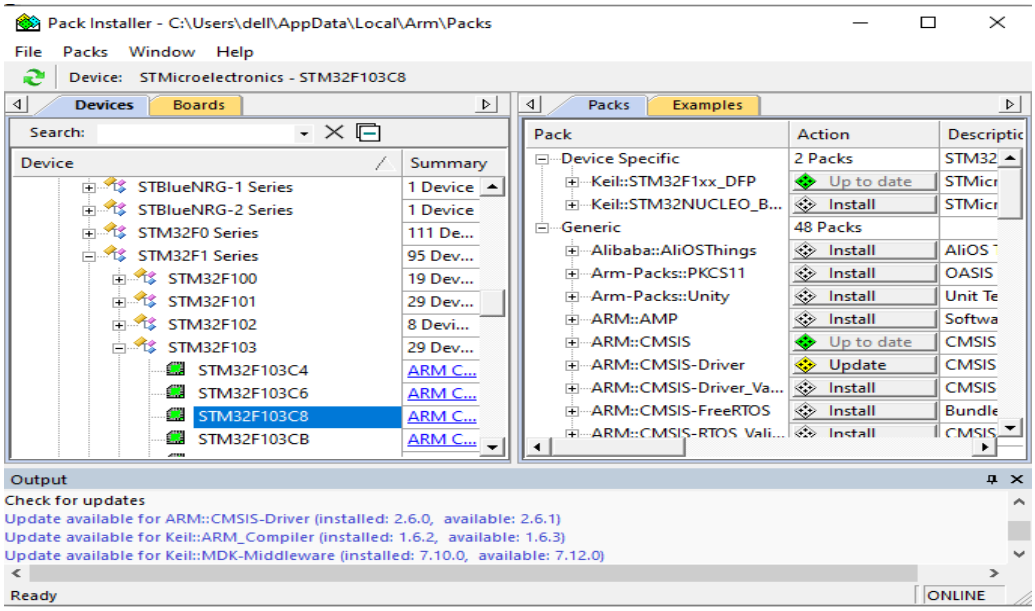


Figure 8: Pack Installer window

LED Blink Program for STM32F1 Discovery Board:

The first step for creating a new project is to go to ‘Project’ and select ‘New uVision Project’.

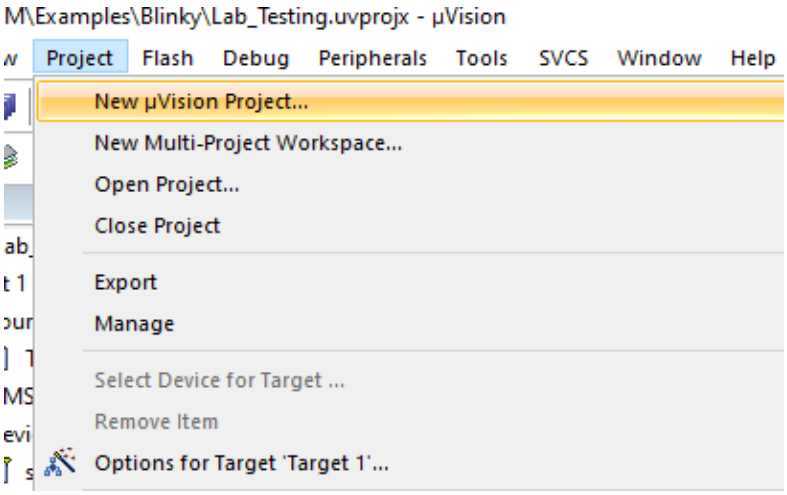


Figure 9: Creating New Project

After naming the project and saving it in a directory, the next step is to select the ‘Target’ device.

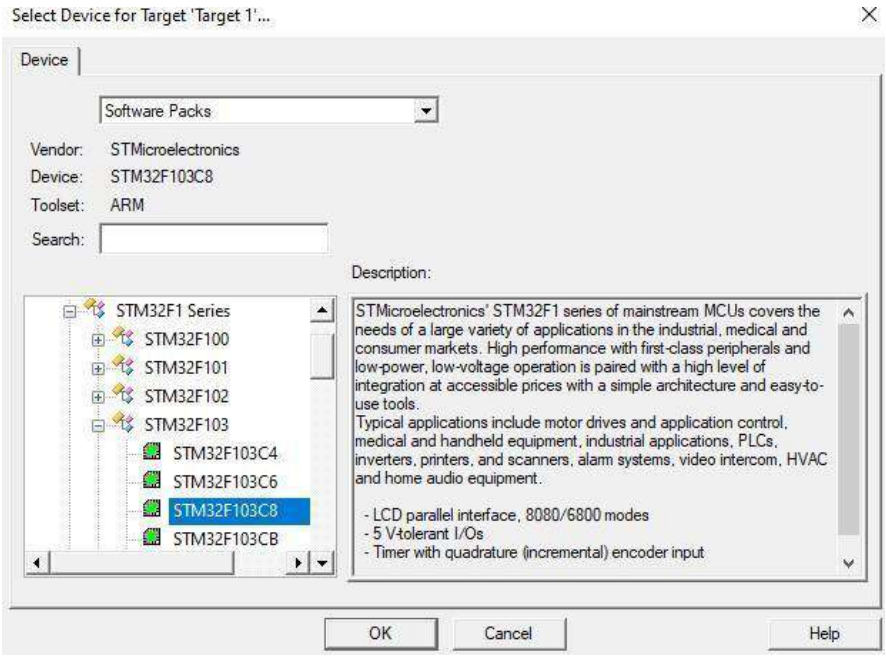


Figure 10: Selecting Device for Target

Then go to STM32F1 series choose STM32F103 and then STM32F103C8

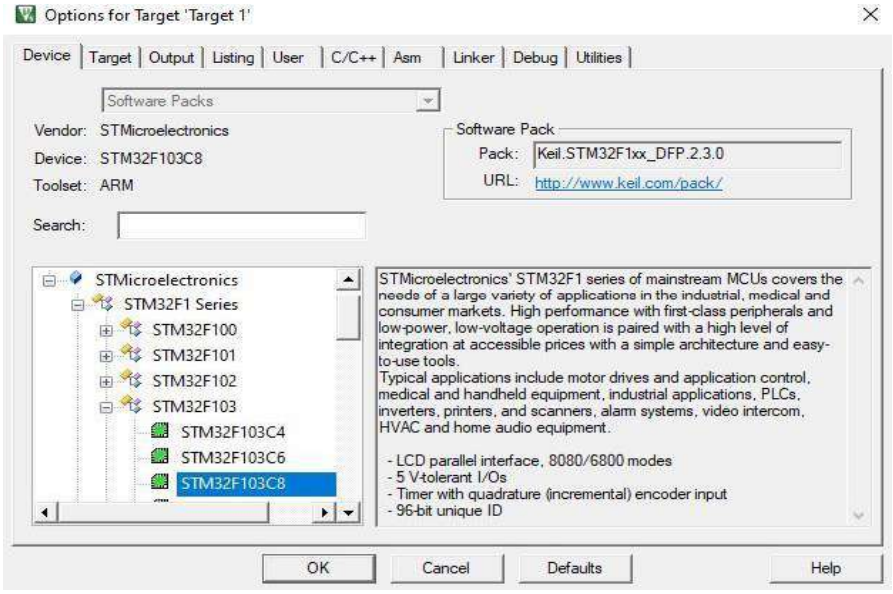


Figure 11: Selecting Device for Target

Next step 'check' the following boxes in Run-Time Environment setting and clicks on Resolve.

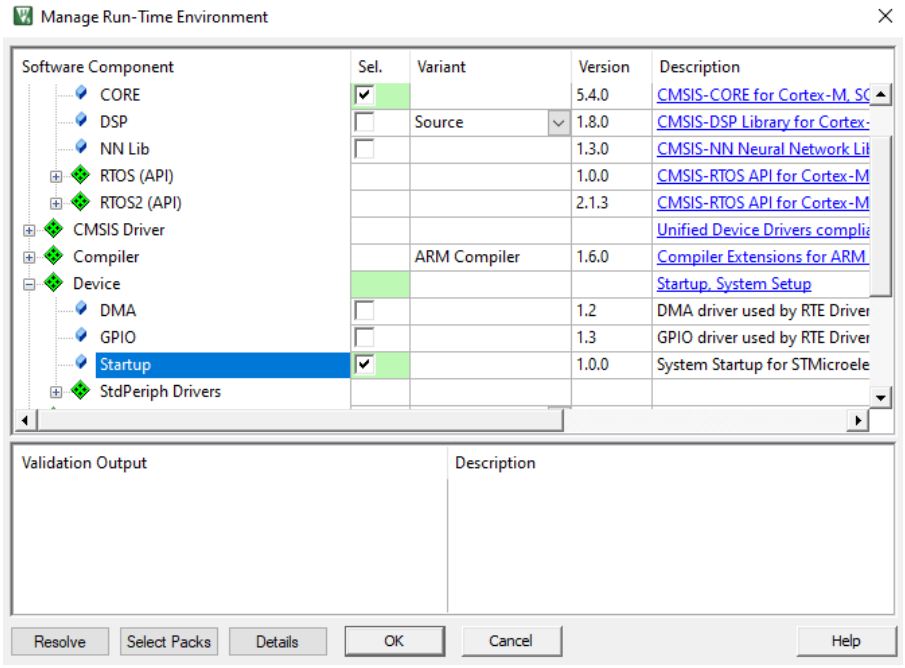


Figure 12: Managing Run-Time environment

Next step is to add a C file to the Target's Source Group 1.

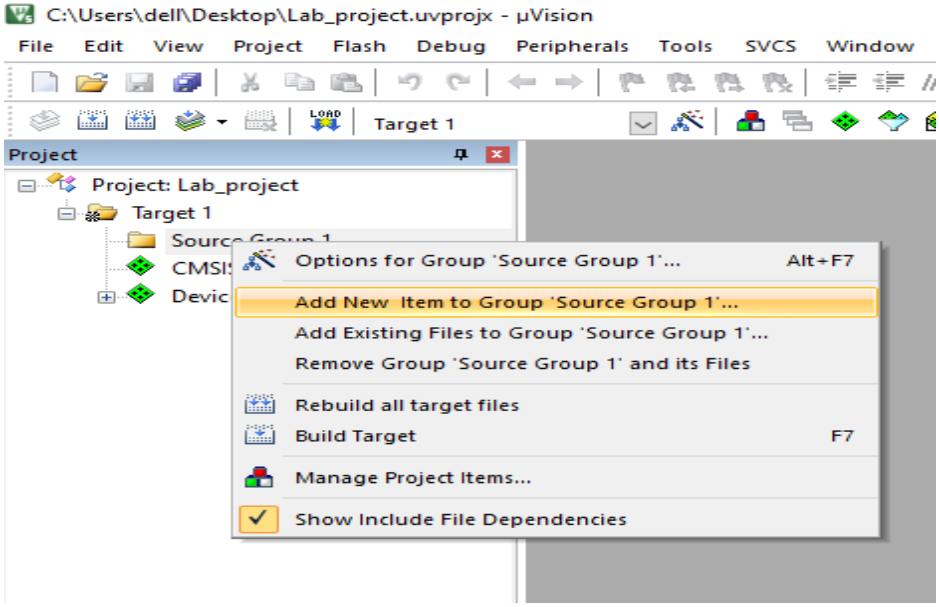


Figure 13: Adding C File

Choose 'C File (.c)', name the file and Add it.

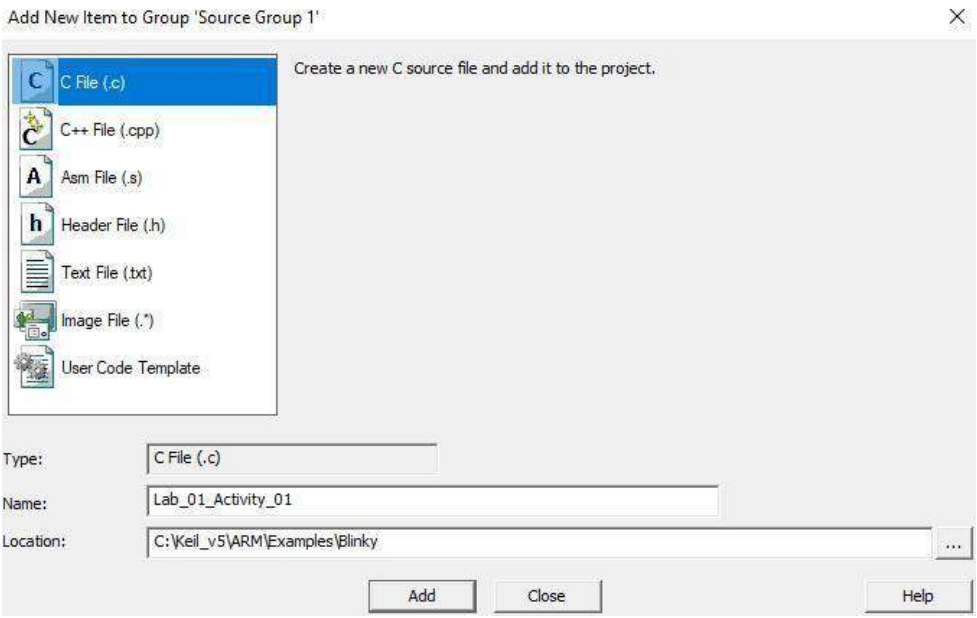


Figure 14: Adding C File

Now set-up the configuration for your board. Click on Options for Target…

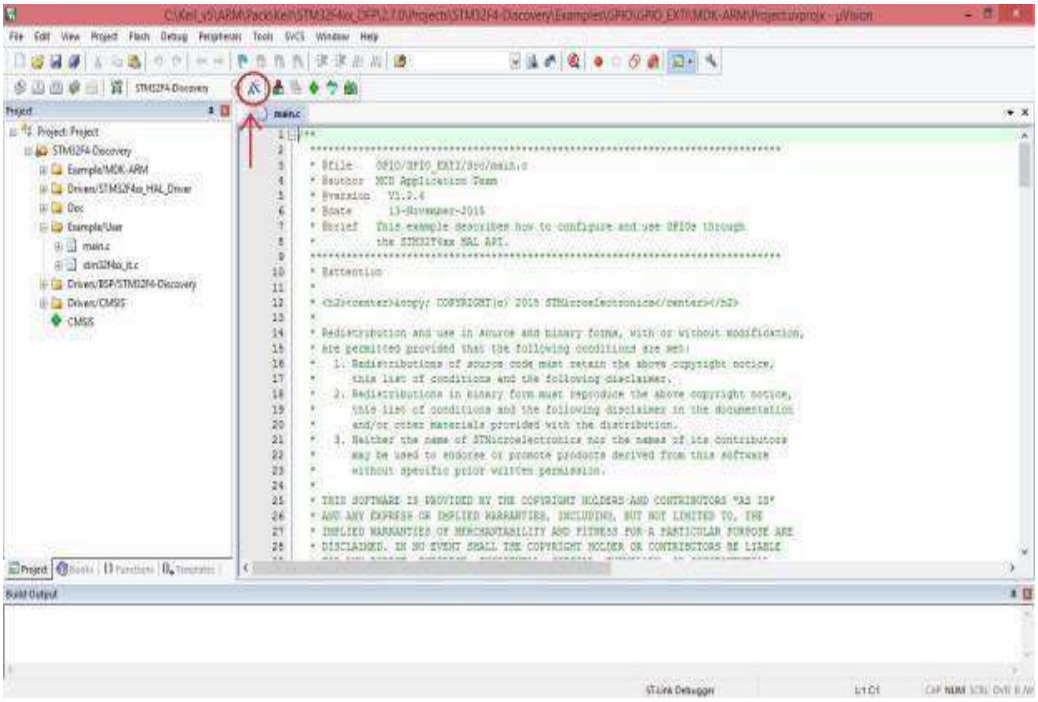


Figure 15: Project window

From **Target** tab set **Xtal (MHz)** up to **8.0**
In **Debug** tab use **ST-Link Debugger**

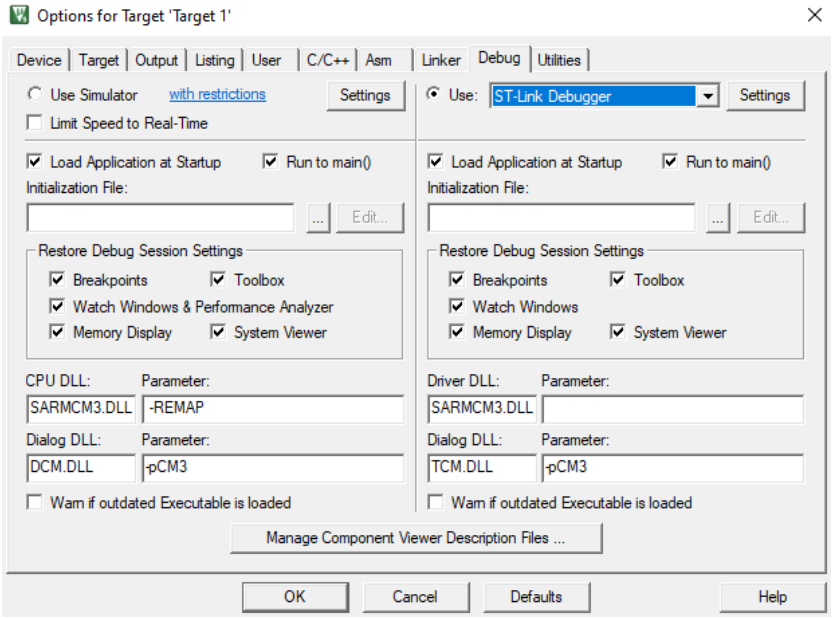


Figure 16: Target Debugger

In **Utilities** tab uncheck **Use Debug Driver** and choose **ST-Link Debugger**

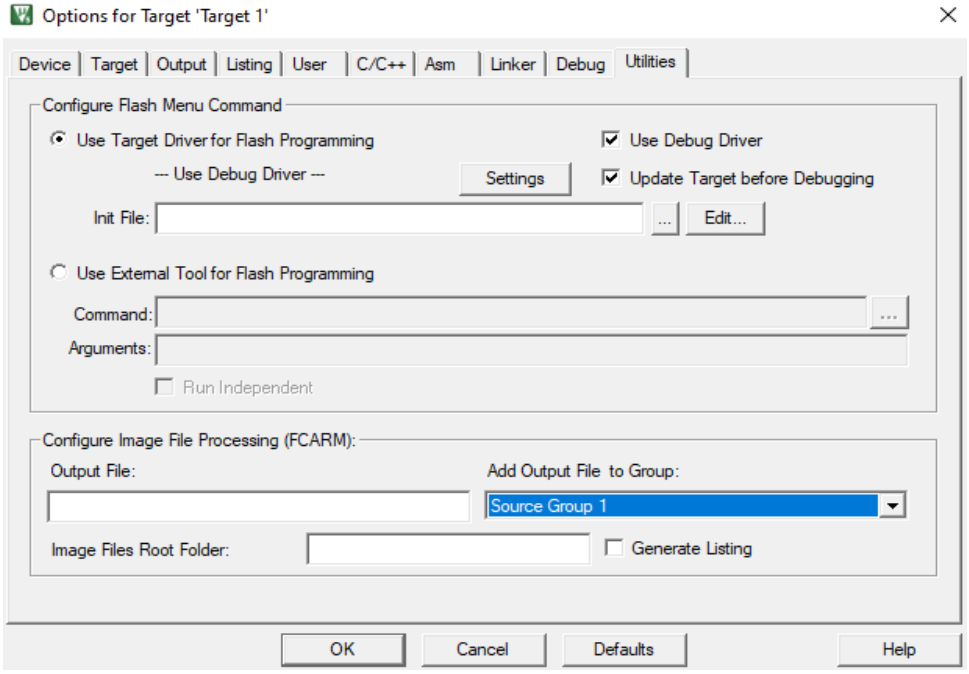


Figure 17: Target Debugger

In **Configure Flash Menu Command** open settings and make sure that in **Flash Download** tab for **Programming Algorithm** you've added **STM32F1xx Flash**.

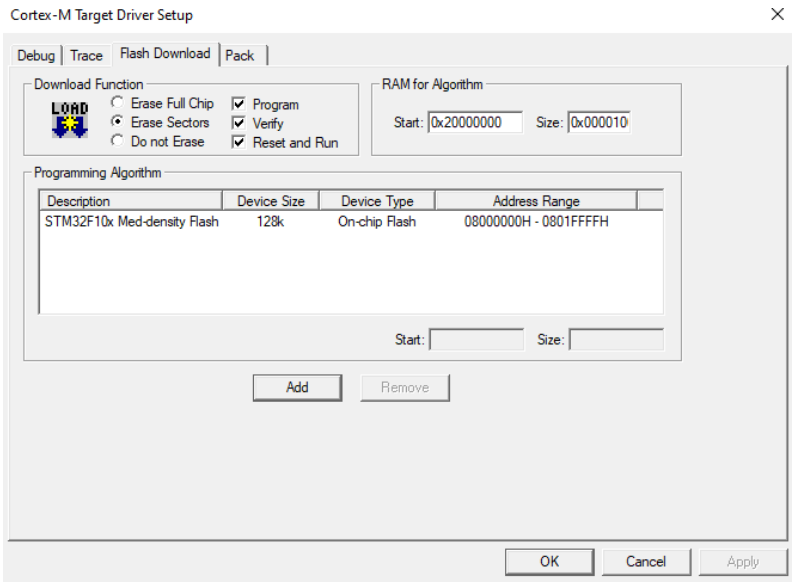


Figure 18: Targeting Device

and in **Debug** tab for **Debug Adapter** unit **ST-Link/V2** is already chosen (when your board is connected to your PC) and the **Port** is **SW**.

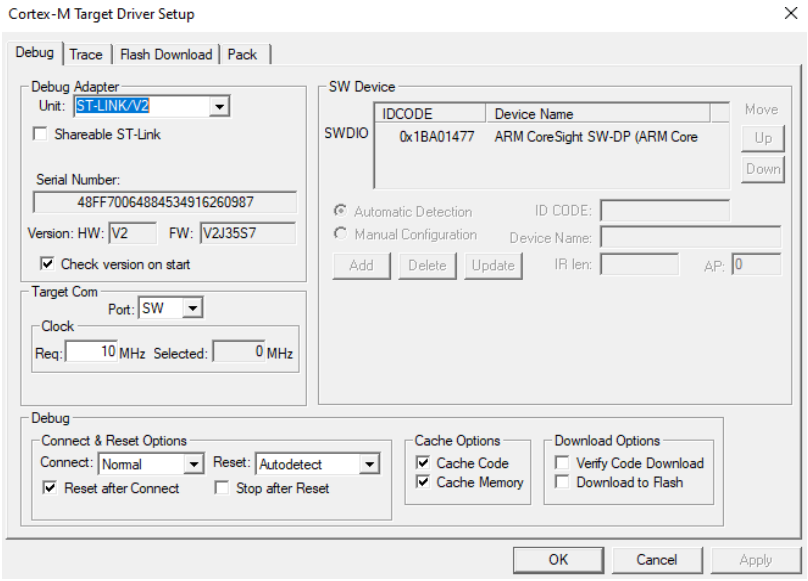


Figure 19: Selecting Debug Adapter

Now open **main.c** from **Example/User** Folder in the **Project** toolbar in the left side. **Build** your project by pressing F7 or the icon. And **Load** it to your board by pressing F8 or the icon

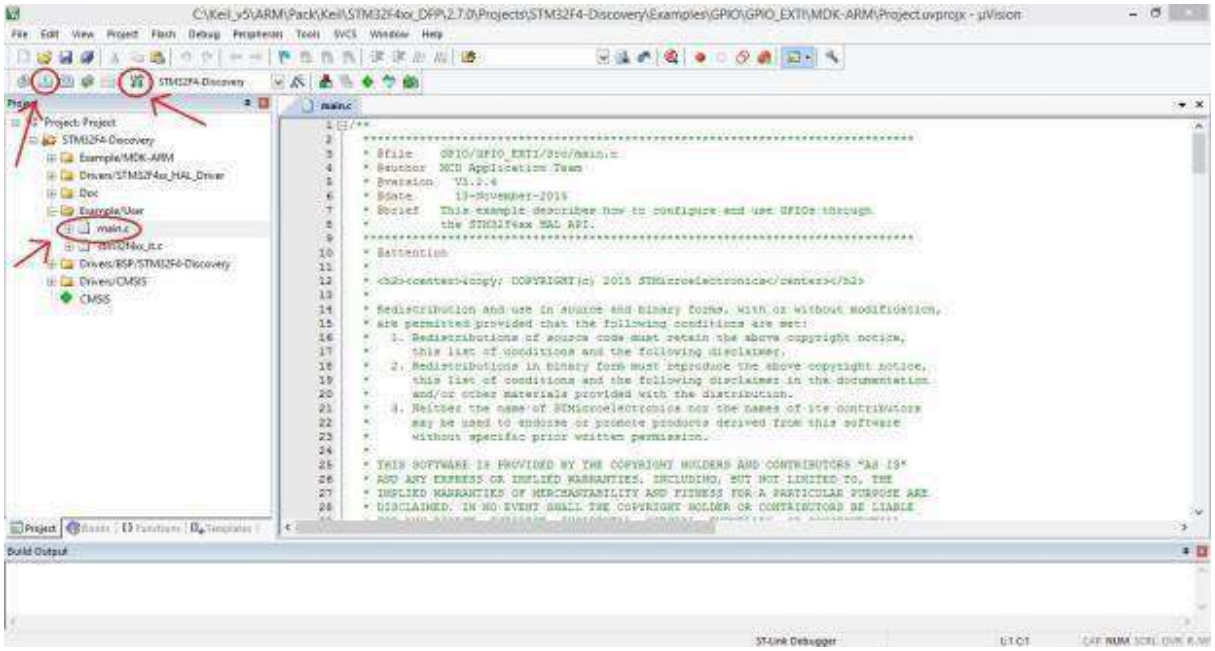


Figure 20: Build and Load

while you are loading your project the LD1 LED on your board will blink red and green repeatedly. It shows that the project is loading to your board.

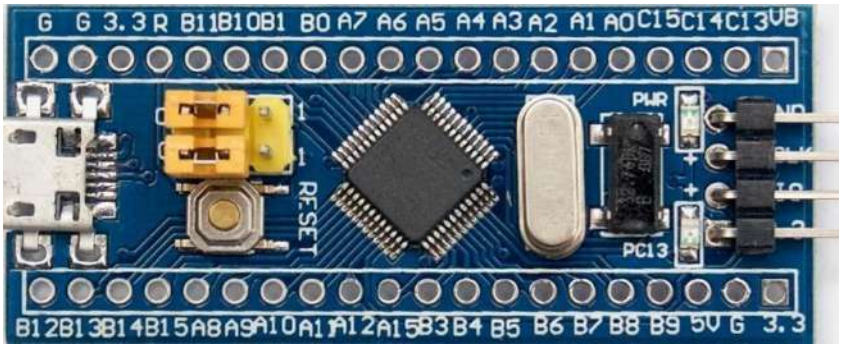


Figure 21: STM32F103C8 board blinks LEDs while loading program

If all the processes gone correctly you will see these messages in **Build Output** window in the bottom.

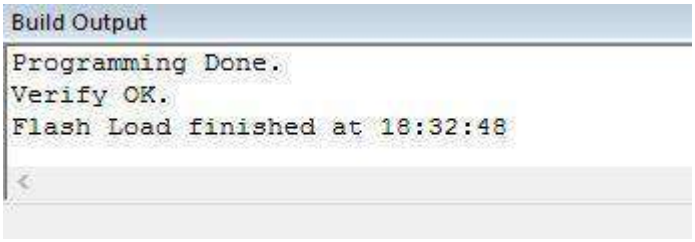


Figure 22: Build Output

By pressing the Reset button on your board, you can see the Green LED turning on and off.