Digital Control in Switched Mode Power Converters and FPGA-based Prototyping

Prof. Santanu Kapat
Department of Electrical Engineering
Indian Institute of Technology, Kharagpur

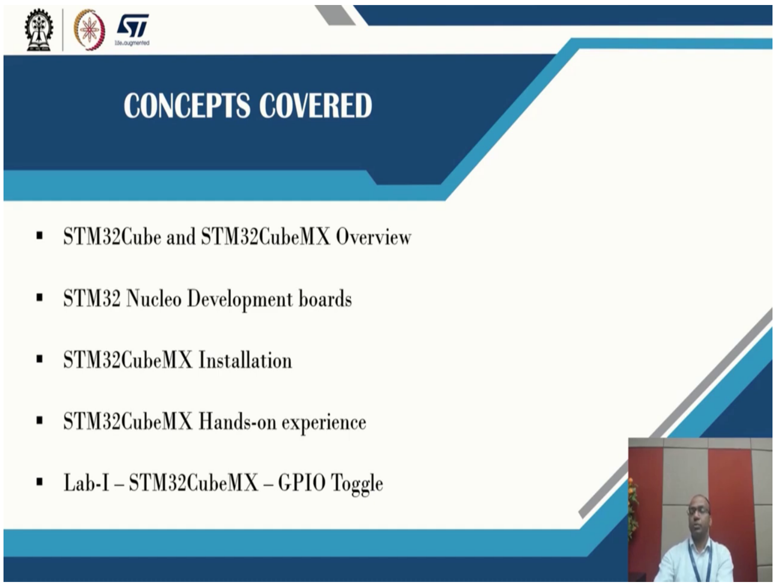
Module - 09
Digital Control Implementation using Microcontroller
Lecture - 82
Getting started with STM32CubeMX (Part - I)

Hello, all welcome to the next lecture on Digital Control and Switch Mode Power Converters and FPGA-based Prototyping course. Before we move forwards let us have a quick recap of the last lecture. In the last lecture, we started the discussion with STM12 Microcontrollers' various options available.

Then we discuss the difference between analog and digital control in particular digital control or we can see a digital power that what all features or peripherals required within the microcontroller to be the best fit for any digital power applications. Then we focused our discussion on STM12G4Microcontroller which is mainly meant for any digital power applications then we discussed various features and the complete ecosystem built around it.

We also discussed the various hardware and software tools available for STM32G4. Particularly for the hardware we mentioned the Nucleo boards, discovery boards, and several evaluation boards. While for the software we discussed the STM32CubeMX, various IDEs, and the digital power workbench.

(Refer Slide Time: 01:31)



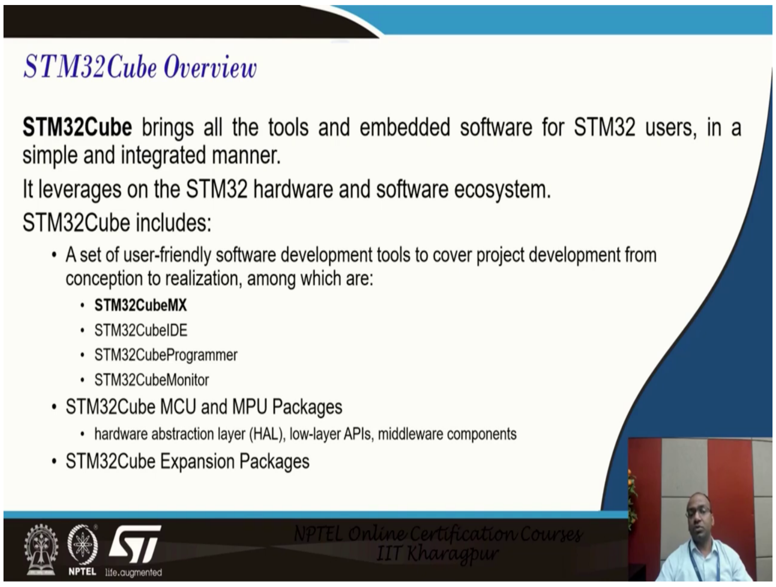
CONCEPTS COVERED

- STM32Cube and STM32CubeMX Overview
- STM32 Nucleo Development boards
- STM32CubeMX Installation
- STM32CubeMX Hands-on experience
- Lab-I – STM32CubeMX – GPIO Toggle

In this lecture, we will be starting with the overview of STM32Cube and STM32CubeMX where STM32Cube is the name of the complete environment and also an open source from STMicroelectronics; where STM32CubeMX is one of the tools it generates.

The C code we will be discussing in detail in this coming slide. Then we will be focusing our discussion on STM32-Nucleo boards. Which will be used for testing the code generated from STM32CubeMX. Then we will be moving step by step starting with the STM32Cube installation and we will be having hands-experience and various examples in detail where we will be doing the peripherals initializations, code generation, and programming the microcontroller on the Nucleo board.

(Refer Slide Time: 02:20)



STM32Cube Overview

STM32Cube brings all the tools and embedded software for STM32 users, in a simple and integrated manner.

It leverages on the STM32 hardware and software ecosystem.

STM32Cube includes:

- A set of user-friendly software development tools to cover project development from conception to realization, among which are:
  - **STM32CubeMX**
  - STM32CubeIDE
  - STM32CubeProgrammer
  - STM32CubeMonitor
- STM32Cube MCU and MPU Packages
  - hardware abstraction layer (HAL), low-layer APIs, middleware components
- STM32Cube Expansion Packages

Let us starts with the STM32Cube overview. STM32Cube brings all the tools and embedded software for STM32 users in a simple and integrated manner. STM32 open development environment or we can call it STM32ODE. It allows developers to verify their design assumptions and move quickly from ideas to proof of concept.

Moreover STM32ODE over a comprehensive entry point to choose the hardware, firmware, and software from the functions required by the applications. To complete the picture STMicroelectronics offers a business-friendly solution. Like there are no license terms and conditions developers can be benefited from free and open-source software.

Also, the embedded software and software tools from ST are available at no cost to developers and can be openly shared or provided they are using STM32 devices. On top of that quality remains the top priority. Hence the core components are tested by ST before their release and ST is committed to following good practices guidelines like MISRA, code checkers, validation reports etcetera. So, in a nutshell, STM32 is a comprehensive ties time-saving solution for STM32 users.

STM32 includes a set of user-friendly software tools to cover project development from conception to realization. These tools include the first one is the STM32CubeMX. It’s a configuration tool for any STM32 device. This easy-to-use graphical user interface where the user can initialize the peripherals and the tool generates the initialization c code. Particularly for this tool, we will be having a dedicated slide and we will be having several hands-on sessions with several examples.

The second one is the STM32CubeIDE. It’s an integrated development environment based on the open source solutions like an eclipse. This id includes compilation, reporting features, and advanced debugging features. It is similar to IAR, Keil, and other available ids.

The most important is it’s open source. The third one is the STM32CubeProgrammer. It’s a programming tool, which provides an easy-to-use and efficient environment for reading, writing, and verifying devices or external memories via a wide variety of available communication media.

It can be JTAG, SWD, UART, US, USB I2C, SPI etcetera. The fourth one is the STM32CubeMonitor. It includes powerful monitoring tools that help developers fine-tune the behavior and performance of their applications in real time. Apart from these software development tools STM32Cube also offers certain packages. The first one is the STM32Cube MCU and MPU Packages. As we all know MCU is the microcontroller and MPU stands for the microprocessor.

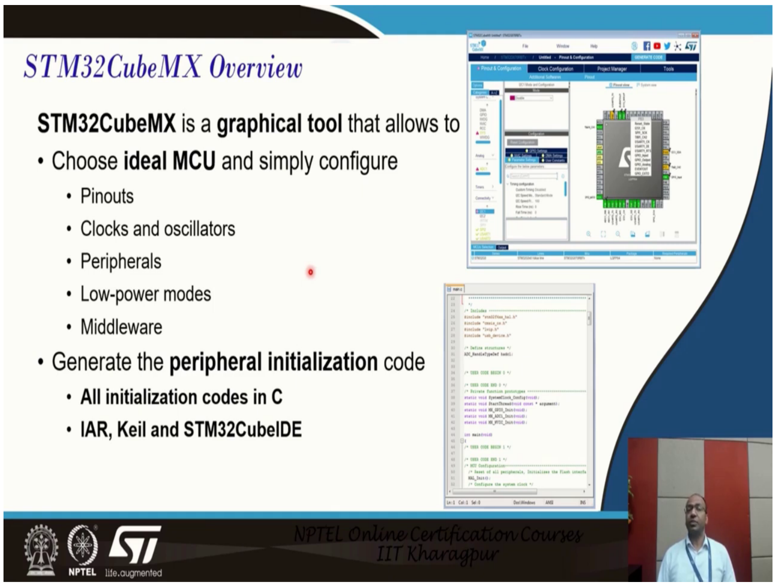
So, this package offers all the required embedded software bricks required to operate the available set of STM32Cube peripherals. They include drivers, middleware, and a lot of examples of code used in a wide variety of real-world use cases. The driver includes the hardware abstraction layer, lower-level APIs, and middleware components. The Hardware Abstraction Layer or we can call it a HAL layer enables the portability between different STM32 devices via standardized API calls.

While the lower level APIs it’s a lightweight optimized expert oriented set of APIs designed for both performance and runtime efficiency, I would suggest that for most of the applications HAL or we can say Hardware Abstraction Layer will fulfill our requirements, but for the digital power applications, we can start with the hardware abstraction layer. Eventually, we have to move towards the lower-level APIs, because, in digital power applications for all the protections and the control algorithm, we have to take the decision very fast.

And the middleware includes the RTOS, USB library, file system, and TCPIP Stack, using the library or graphic library depending on the STM32 series we are using. It also includes the RS-Stacks such as Bluetooth, OpenThread, ZigBee, LoRa, and Sigfox again depending on the specific STM32 wireless series we are using. On top of it, there is also a STM32Cube Expansion Package.

These packages are application-oriented. Mainly they are complementing and expanding the STM32Cube MCU Package with an additional embedded software brick. The STM32Cube Expansion Packages come either from ST or approved partners to create an extensive and scalable embedded software offer around the STM32.

(Refer Slide Time: 07:45)



STM32CubeMX Overview

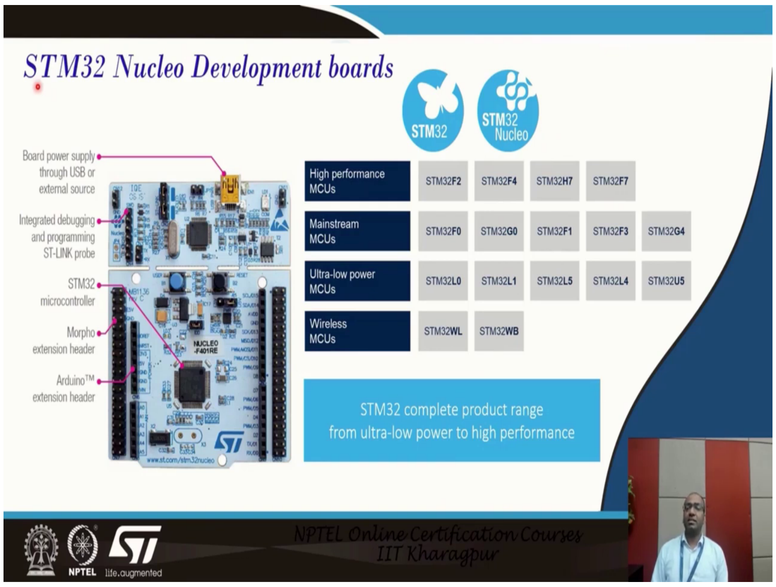
STM32CubeMX is a **graphical tool** that allows to

- Choose **ideal MCU** and simply configure
  - Pinouts
  - Clocks and oscillators
  - Peripherals
  - Low-power modes
  - Middleware
- Generate the **peripheral initialization** code
  - **All initialization codes in C**
  - **IAR, Keil and STM32CubeIDE**

STM32CubeMX as we have discussed is one of the software tools of STM, STM32Cube open development environment. and is of time important to us. It’s a graphical user interface tool. That guides the initial configuration of the firmware projects. Very soon we will be saying how exactly it helps in selecting the right microcontroller for our applications. How we can configure the pin-out configurations? The clock, peripherals, and different modes.

So, this is the screenshot of the pin configuration that is being done. Now, the most important thing the initialization code generates the C code. And it generates the project for different three different ids. The first one is the IAR, the second is the Keil and the third one is the STM32CubeIDE. STM32CubeIDE is an open source.

(Refer Slide Time: 08:36)



STM32 Nucleo Development boards

STM32
STM32 Nucleo
Board power supply through USB or external source
Integrated debugging and programming ST-LINK probe
STM32 microcontroller
Morpho extension header
Arduino™ extension header

High performance MCUs	STM32F2	STM32F4	STM32H7	STM32F7	
Mainstream MCUs	STM32F0	STM32G0	STM32F1	STM32F3	STM32G4
Ultra-low power MCUs	STM32L0	STM32L1	STM32L5	STM32L4	STM32U5
Wireless MCUs	STM32WL	STM32WB			

STM32 complete product range from ultra-low power to high performance

NPTEL Online Certification Courses
IIT Kharagpur

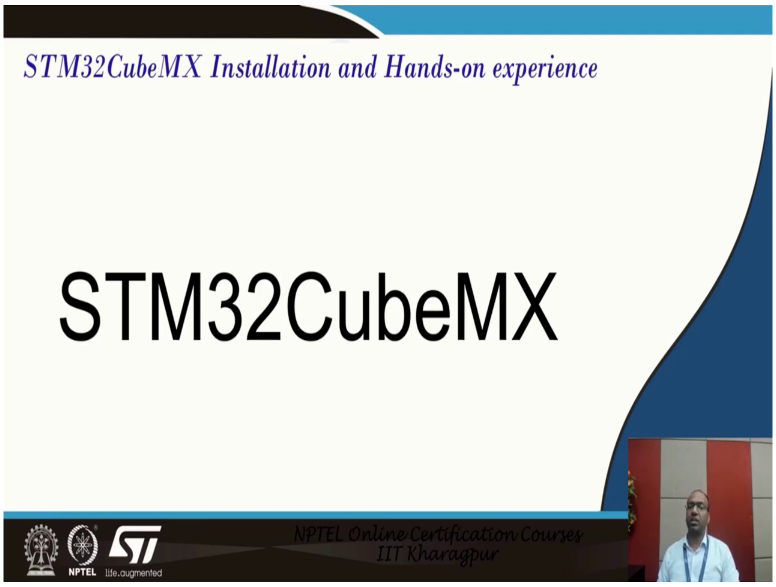
STM32 Nucleo Development boards. These Nucleo boards we have discussed in the last lecture, but it is worth mentioning here. As soon we will be starting with the actual working of the STM32CubeMX tool and we all know this tool helps in generating the C code and the entire firmware project.

So, this firmware needs to be run on some hardware to visualize the outcome of the firmware that has been written as right or not as per expectation. To demonstrate we will be using these Nucleo boards, though there is a large number of Nucleo boards, we are targeting digital power applications. So, we will be using an STM32G4 Microcontroller-based Nucleo board which you can see in my hand.

Just a quick glimpse about these Nucleo boards. These Nucleo boards allow developers to try out new ideas and quickly create prototypes. Thanks to various connectors present on these ports they can extend their capabilities using multiple application-related hardware add on's. When I say the hardware add on's it can be adding new shields and the X-Nucleo boards. There are three types of Nucleo boards.

Nucleo-64, Nucleo-144, Nucleo-32 boards. So, this number depends on the pin count of the microcontroller. This Nucleo board also integrates the ST-LINK debugger or the programmer. So, there is no dedicated hardware required to program the microcontroller.

(Refer Slide Time: 10:01)

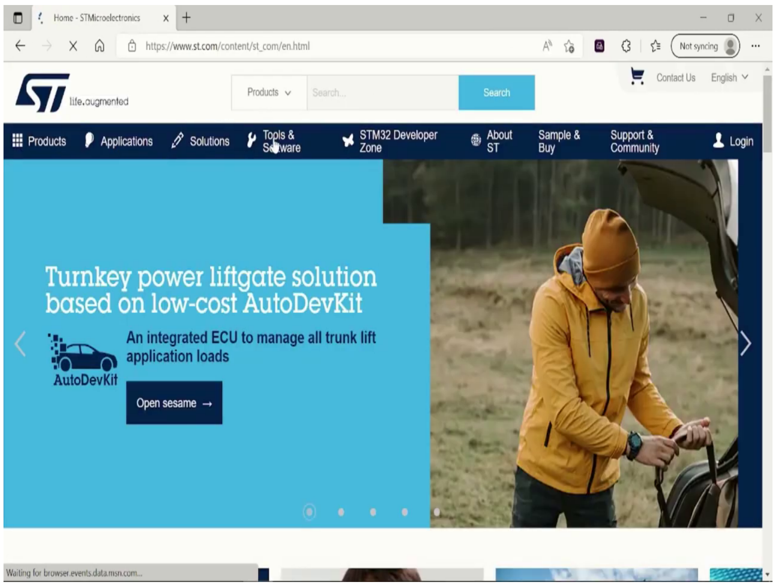

STM32CubeMX Installation and Hands-on experience

STM32CubeMX

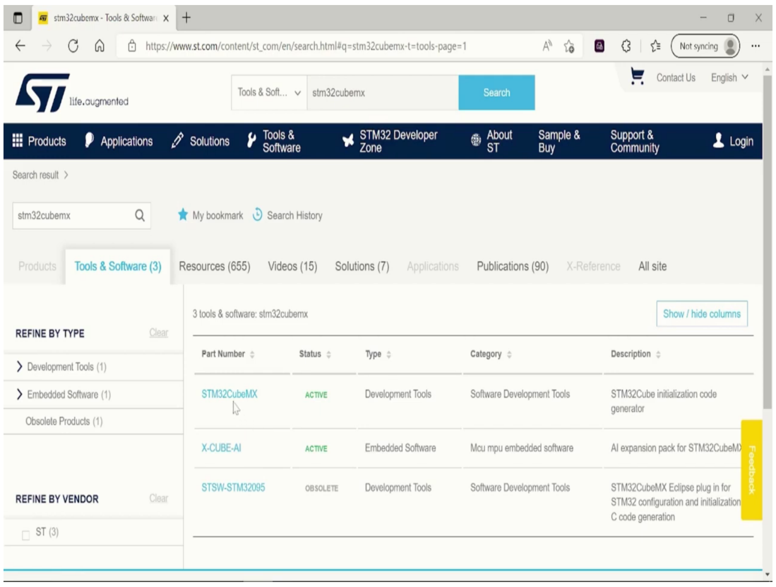
NPTEL
life.augmented
NPTEL Online Certification Courses
IIT Kharagpur

Let us start with the STM32CubeMX. Firstly, we will be starting with the downloading and installation process. Then we will be moving to the hand hands-on live experience.

(Refer Slide Time: 10:13)


Home - STMicroelectronics
https://www.st.com/content/st_com/en.html
life.augmented
Products
Search
Contact Us
English
Products
Applications
Solutions
Tools & Software
STM32 Developer Zone
About ST
Sample & Buy
Support & Community
Login
Turnkey power liftgate solution based on low-cost AutoDevKit
An integrated ECU to manage all trunk lift application loads
AutoDevKit
Open sesame →

(Refer Slide Time: 10:26)



Search result

stm32cubemx

My bookmark Search History

Products | Tools & Software (3) | Resources (655) | Videos (15) | Solutions (7) | Applications | Publications (90) | X-Reference | All site

REFINE BY TYPE
- Development Tools (1)
- Embedded Software (1)
- Obsolete Products (1)

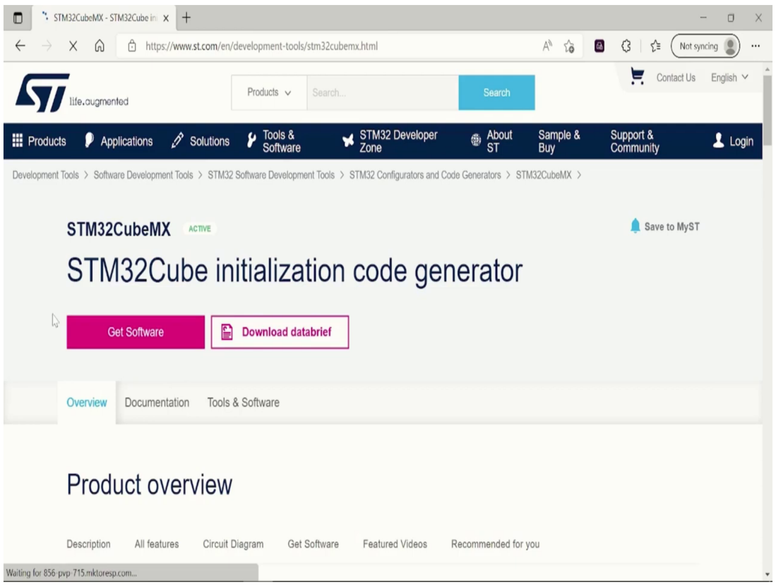
REFINE BY VENDOR
- ST (3)

3 tools & software: stm32cubemx

Part Number	Status	Type	Category	Description
STM32CubeMX	ACTIVE	Development Tools	Software Development Tools	STM32Cube initialization code generator
X-CUBE-AI	ACTIVE	Embedded Software	Mcu mpu embedded software	AI expansion pack for STM32CubeMX
STSW-STM32095	OBSOLETE	Development Tools	Software Development Tools	STM32CubeMX Eclipse plug in for STM32 configuration and initialization C code generation

To download the latest version of the STM32CubeMX tool. We can go to the st website. It is www dot st dot com. We can directly go to the tools and software sections or we can go to the search bar menu options and search for the stm32Cubemx tool. It will be showing a couple of options. Here we can select STM32CubeMX. Initialization code generator.

(Refer Slide Time: 10:45)



Development Tools > Software Development Tools > STM32 Software Development Tools > STM32 Configurators and Code Generators > STM32CubeMX >

STM32CubeMX ACTIVE

STM32Cube initialization code generator

Get Software | Download databrief

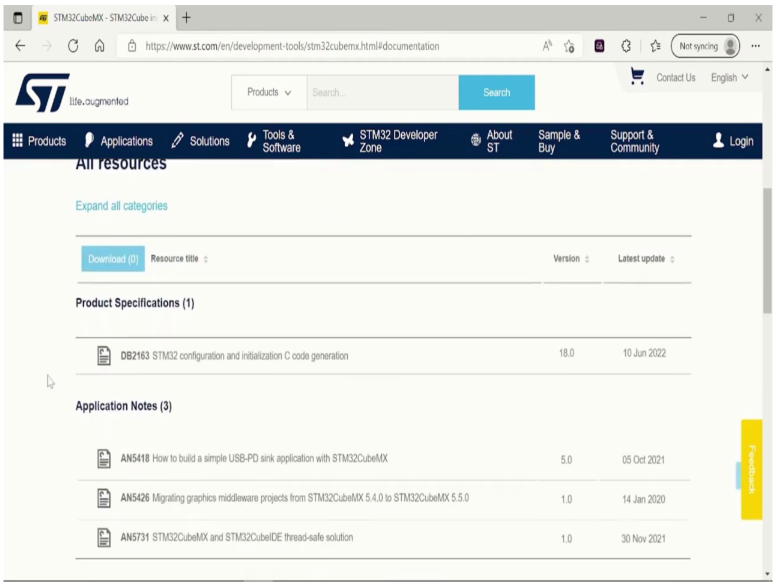
Overview | Documentation | Tools & Software

Product overview

Description | All features | Circuit Diagram | Get Software | Featured Videos | Recommended for you

Now, it will take us to the main page of the tool. Here, we see an option of getting the software and downloading the data brief.

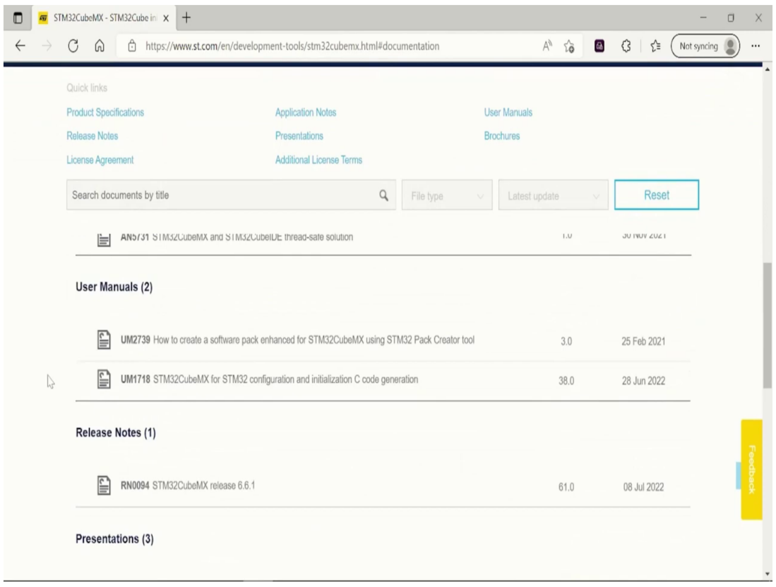
(Refer Slide Time: 10:59)


All resources

Expand all categories

Resource title	Version	Latest update
Product Specifications (1)		
DB2163 STM32 configuration and initialization C code generation	18.0	10 Jun 2022
Application Notes (3)		
AN5418 How to build a simple USB-PD sink application with STM32CubeMX	5.0	05 Oct 2021
AN5426 Migrating graphics middleware projects from STM32CubeMX 5.4.0 to STM32CubeMX 5.5.0	1.0	14 Jan 2020
AN5731 STM32CubeMX and STM32CubeIDE thread-safe solution	1.0	30 Nov 2021

(Refer Slide Time: 11:00)

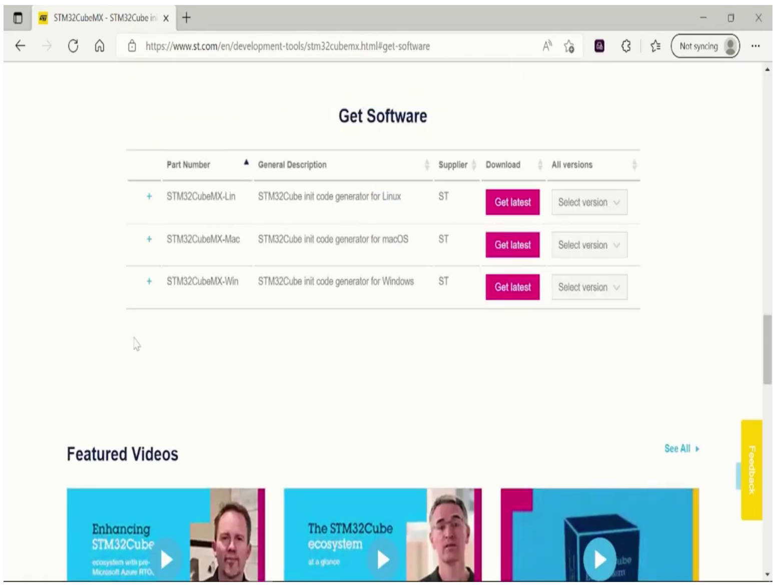

Quick links

Product Specifications	Application Notes	User Manuals
Release Notes	Presentations	Brochures
License Agreement	Additional License Terms	

Resource title	Version	Latest update
AN5731 STM32CubeMX and STM32CubeIDE thread-safe solution	1.0	30 Nov 2021
User Manuals (2)		
UM2739 How to create a software pack enhanced for STM32CubeMX using STM32 Pack Creator tool	3.0	25 Feb 2021
UM1718 STM32CubeMX for STM32 configuration and initialization C code generation	38.0	28 Jun 2022
Release Notes (1)		
RN0094 STM32CubeMX release 6.6.1	61.0	08 Jul 2022
Presentations (3)		

Below there is also an option of fore documentation, where we can see various application nodes, user manuals, release nodes, and the presentation.

(Refer Slide Time: 11:08)


Get Software

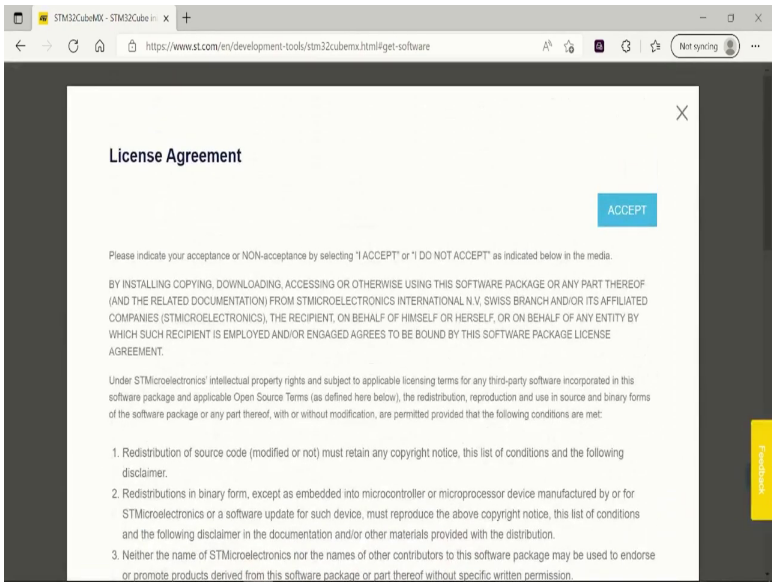
	Part Number	General Description	Supplier	Download	All versions
+	STM32CubeMX-Lin	STM32Cube init code generator for Linux	ST	Get latest	Select version
+	STM32CubeMX-Mac	STM32Cube init code generator for macOS	ST	Get latest	Select version
+	STM32CubeMX-Win	STM32Cube init code generator for Windows	ST	Get latest	Select version

Featured Videos

https://www.st.com/en/development-tools/stm32cubemx.html#get-software
See All
Feedback
Enhancing STM32Cube
The STM32Cube ecosystem

Now, I click on get software. It will be showing me three options thnLinuxAC or the Windows machine.

(Refer Slide Time: 11:20)


License Agreement

https://www.st.com/en/development-tools/stm32cubemx.html#get-software
ACCEPT
Feedback

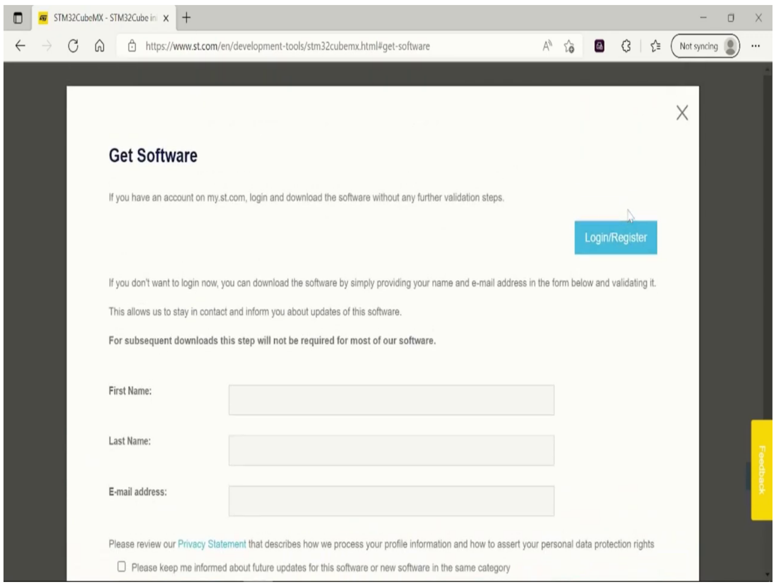
Please indicate your acceptance or NON-acceptance by selecting "I ACCEPT" or "I DO NOT ACCEPT" as indicated below in the media.

BY INSTALLING COPYING, DOWNLOADING, ACCESSING OR OTHERWISE USING THIS SOFTWARE PACKAGE OR ANY PART THEREOF (AND THE RELATED DOCUMENTATION) FROM STMICROELECTRONICS INTERNATIONAL N.V, SWISS BRANCH AND/OR ITS AFFILIATED COMPANIES (STMICROELECTRONICS), THE RECIPIENT, ON BEHALF OF HIMSELF OR HERSELF, OR ON BEHALF OF ANY ENTITY BY WHICH SUCH RECIPIENT IS EMPLOYED AND/OR ENGAGED AGREES TO BE BOUND BY THIS SOFTWARE PACKAGE LICENSE AGREEMENT.

Under STMicroelectronics' intellectual property rights and subject to applicable licensing terms for any third-party software incorporated in this software package and applicable Open Source Terms (as defined here below), the redistribution, reproduction and use in source and binary forms of the software package or any part thereof, with or without modification, are permitted provided that the following conditions are met:

1. Redistribution of source code (modified or not) must retain any copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form, except as embedded into microcontroller or microprocessor device manufactured by or for STMicroelectronics or a software update for such device, must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of STMicroelectronics nor the names of other contributors to this software package may be used to endorse or promote products derived from this software package or part thereof without specific written permission.

(Refer Slide Time: 11:23)


Get Software

If you have an account on my.st.com, login and download the software without any further validation steps.

If you don't want to login now, you can download the software by simply providing your name and e-mail address in the form below and validating it.

This allows us to stay in contact and inform you about updates of this software.

For subsequent downloads this step will not be required for most of our software.

First Name:

Last Name:

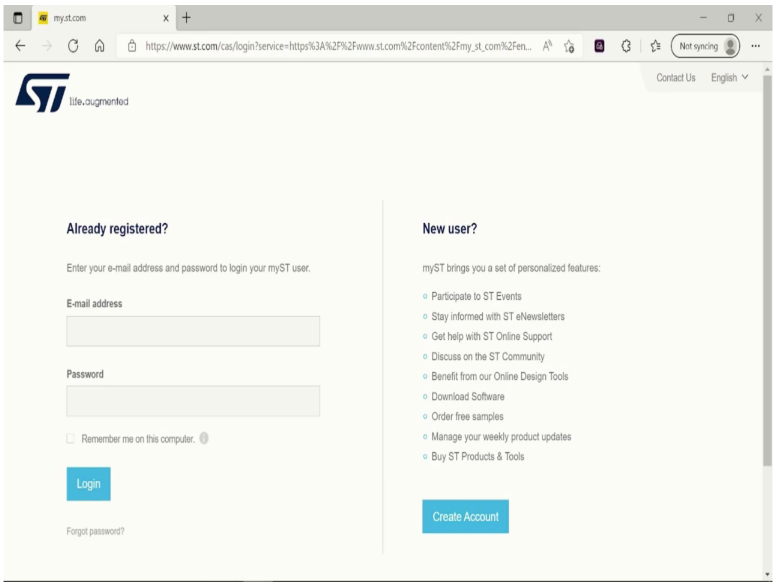
E-mail address:

Please review our Privacy Statement that describes how we process your profile information and how to assert your personal data protection rights

☐ Please keep me informed about future updates for this software or new software in the same category

So, as having the Window machines. So, I am clicking on the get the latest version for Windows. License agreement we can accept. Now, it will be taking a to a login or the register page as I am already having a login. So, I am clicking on the login, otherwise, we can fill in the details and register with a suitable email id.

(Refer Slide Time: 11:42)


Already registered?

Enter your e-mail address and password to login your myST user.

E-mail address

Password

☐ Remember me on this computer.

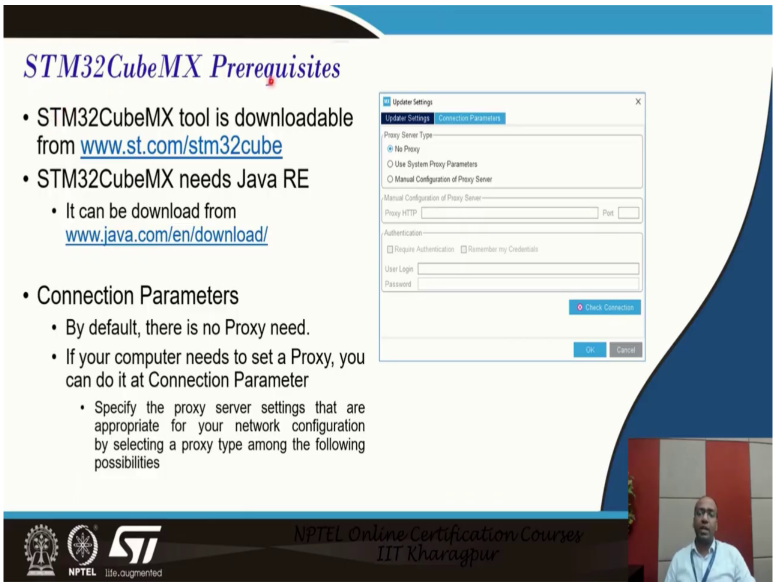
New user?

myST brings you a set of personalized features:

- Participate to ST Events
- Stay informed with ST eNewsletters
- Get help with ST Online Support
- Discuss on the ST Community
- Benefit from our Online Design Tools
- Download Software
- Order free samples
- Manage your weekly product updates
- Buy ST Products & Tools

Hereafter do the login or once you have registered. We can get the software download and installed on our PC.

(Refer Slide Time: 11:52)



STM32CubeMX Prerequisites

- STM32CubeMX tool is downloadable from www.st.com/stm32cube
- STM32CubeMX needs Java RE
  - It can be download from www.java.com/en/download/
- Connection Parameters
  - By default, there is no Proxy need.
  - If your computer needs to set a Proxy, you can do it at Connection Parameter
    - Specify the proxy server settings that are appropriate for your network configuration by selecting a proxy type among the following possibilities

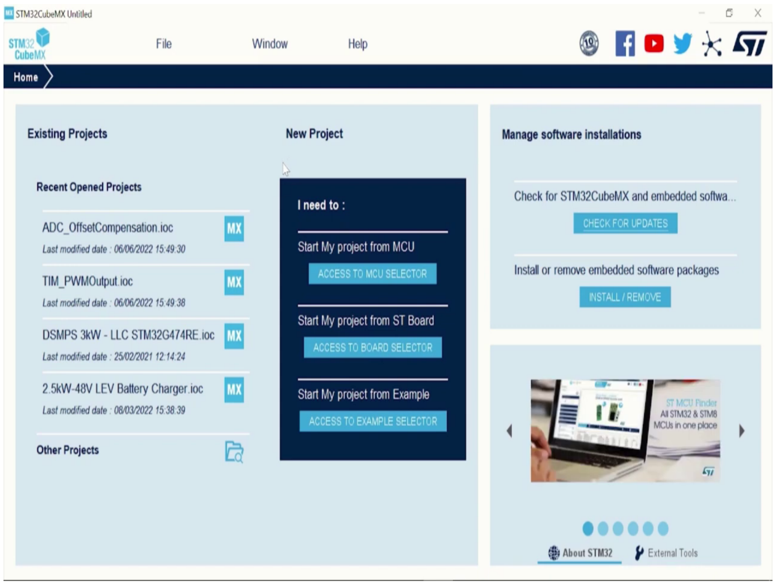
Updater Settings

Proxy Server Type: [✓ No Proxy] / Use System Proxy Parameters / Manual Configuration of Proxy Server

Some important points while installing the STM32CubeMX tool are that it requires Java RE to be installed on our machine.

Just in case you do not have it on your pc. You can download it from the given link and get it installed and in the updated settings there are two options. Update settings and the connection parameters by default there is nothing required to be changed here. Just in case you are using a workplace pc or work machine and you require to set some Proxy you can do it appropriately over here.

(Refer Slide Time: 12:24)



STM32CubeMX Untitled

Existing Projects

Recent Opened Projects

ADC_OffsetCompensation.ioc
Last modified date : 06/06/2022 15:49:30

TIM_PWMOutput.ioc
Last modified date : 06/06/2022 15:49:38

DSMPS 3kW - LLC STM32G474RE.ioc
Last modified date : 25/02/2021 12:14:24

2.5kW-48V LEV Battery Charger.ioc
Last modified date : 08/03/2022 15:38:39

Other Projects

New Project

I need to :

Start My project from MCU
ACCESS TO MCU SELECTOR

Start My project from ST Board
ACCESS TO BOARD SELECTOR

Start My project from Example
ACCESS TO EXAMPLE SELECTOR

Manage software installations

Check for STM32CubeMX and embedded softwa...
CHECK FOR UPDATES

Install or remove embedded software packages
INSTALL / REMOVE

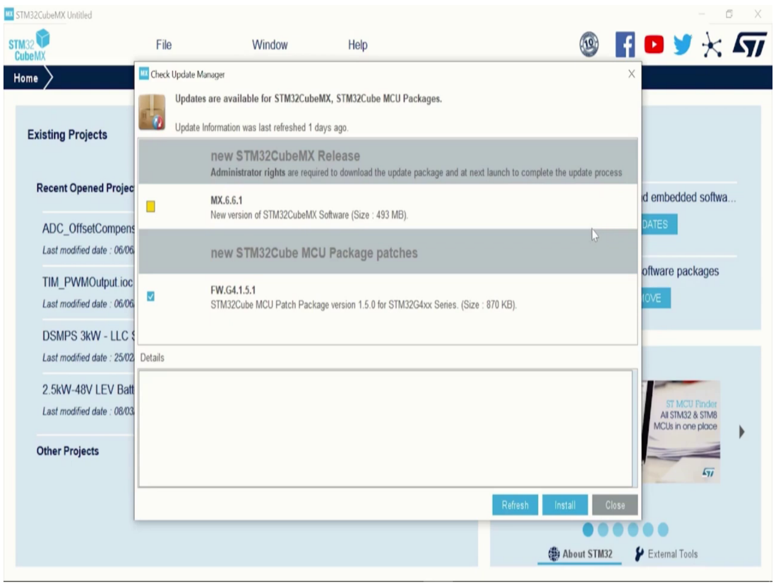
On successful installation of the STM32CubeMX tool, we can launch the tool and this is the home screen of the tool. In which we can see various options. On the left side, we can see the various recent projects we have been working on it is showing the list.

The other option is the new project, if you want to start the new project in that also there are three options if we have any preselected MCU in our mind or if we want to select the MCU for our application, we can use this. And depending on the peripherals required we can select the best MCU in terms of the cost or power or size or package whatever our requirement.

And accordingly, we can initialize the peripherals, and accordingly, we can generate the C code. We will be working on this section a bit later. The other option is the start my project from the ST board. When we say the ST board it can be the Nucleo boards or the discovery boards. When we use this option. So, the file that will be working on all the peripherals will be initialized as per the hardware. If you want to modify that also that option is available.

So, the third option is to start my project with examples. As I have been telling you that several examples have been predefined for several Nucleo boards and several discovery boards. We can use those particular examples and evaluate them.

(Refer Slide Time: 13:55)



Updates are available for STM32CubeMX, STM32Cube MCU Packages.

Update Information was last refreshed 1 days ago.

new STM32CubeMX Release

Administrator rights are required to download the update package and at next launch to complete the update process

MX.6.6.1
New version of STM32CubeMX Software (Size : 493 MB).

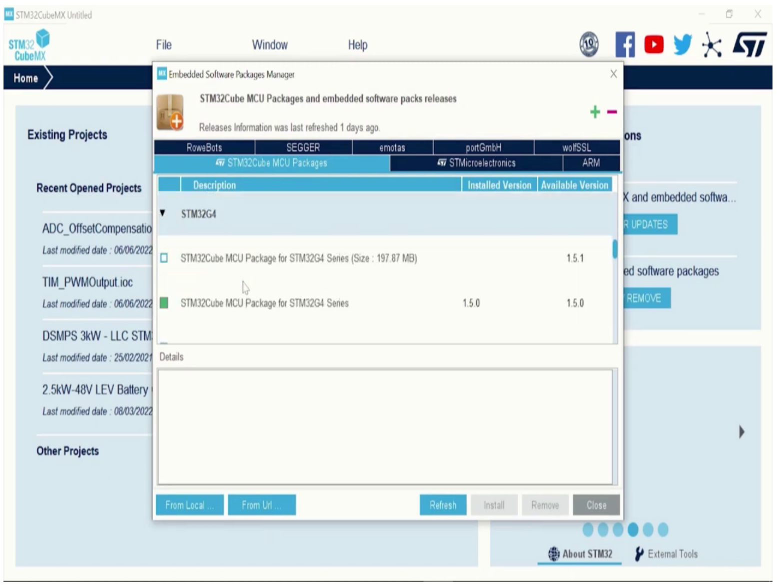
new STM32Cube MCU Package patches

✓ FW.G4.1.5.1
STM32Cube MCU Patch Package version 1.5.0 for STM32G4xx Series. (Size : 870 KB).

On the right hand on the right-hand side, we can see another option to check for updates. If we click on check for updates, it will be telling us what all software or packages we have installed and if the latest version or if there is another version that we need to update.

It will be telling that. So, the CubeMX is the latest version that has been installed, but the MCU Package is. So, for the G4 Package, it is telling me that there is another updated version that is available and I have the version installed its 1.50, while the available version is 1.5.1. So, if you want to install we can just click on install and it will be correspondingly installed on your pc.

(Refer Slide Time: 14:43)



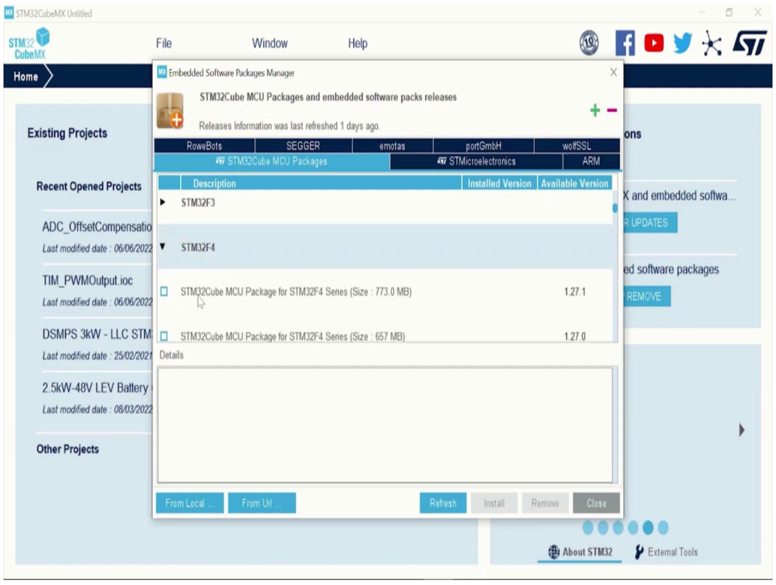
STM32Cube MCU Packages and embedded software packs releases

Releases Information was last refreshed 1 days ago.

	Description	Installed Version	Available Version
▼	STM32G4		
☐	STM32Cube MCU Package for STM32G4 Series (Size : 197.87 MB)		1.5.1
🟩	STM32Cube MCU Package for STM32G4 Series	1.5.0	1.5.0

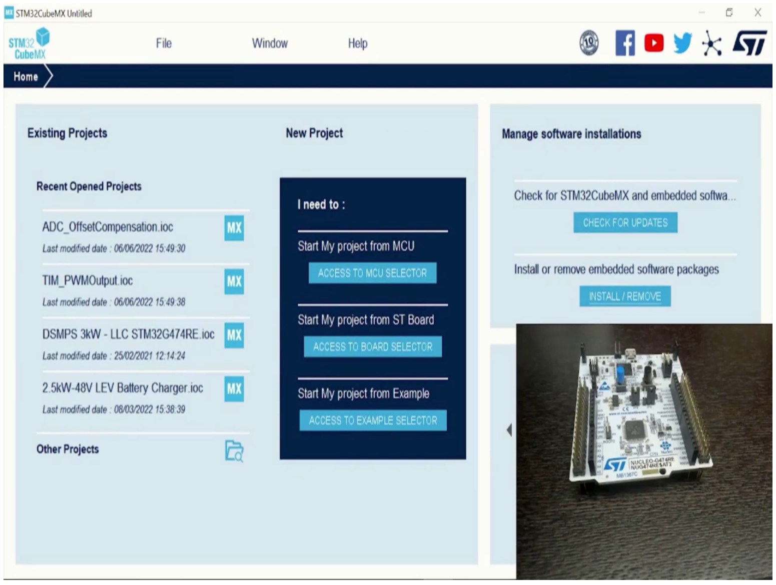
The other option is to install or remove embedded software packages. Depending on the microcontroller that we are working on. We can accordingly select the package select say we are we want to work on the STM32G4. So, as just mentioned that 1.50 is already installed, but if you want to install the latest. We can just right-click here and press install it will be correspondingly installed on your PC. Just remember the path where exactly all these files are getting stored. One thing important to mention here like in our case we are using STM32G4 Nucleo board.

(Refer Slide Time: 15:18)



So, I have installed the G4 Package here. As you can see here I have installed this, but in case you do not have the STM32G4 Nucleo board. If you have another other board. Let us say F4. So, you can correspondingly install the latest version STM32Cube Package.

(Refer Slide Time: 15:42)


STM32CubeMX Untitled
File
Window
Help
Home
Existing Projects
Recent Opened Projects
ADC_OffsetCompensation.ioc
Last modified date : 06/06/2022 15:49:30
TIM_PWMOutput.ioc
Last modified date : 06/06/2022 15:49:38
DSMPS 3kW - LLC STM32G474RE.ioc
Last modified date : 25/02/2021 12:14:24
2.5kW-48V LEV Battery Charger.ioc
Last modified date : 08/03/2022 15:38:39
Other Projects
New Project
I need to :
Start My project from MCU
ACCESS TO MCU SELECTOR
Start My project from ST Board
ACCESS TO BOARD SELECTOR
Start My project from Example
ACCESS TO EXAMPLE SELECTOR
Manage software installations
Check for STM32CubeMX and embedded softwa...
CHECK FOR UPDATES
Install or remove embedded software packages
INSTALL / REMOVE

Now, we will be started working on this Nucleo board based on STM32G4Microcontroller. This is the 64-pin microcontroller. These are the various connectors on the top side.

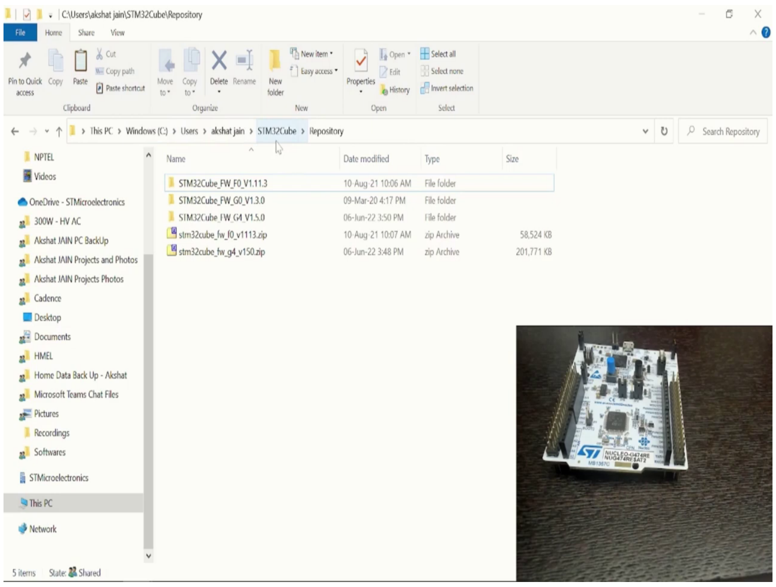
This is the ST-Link, basically the debugger. Also there are 2 push buttons one is for the reset, and the power reset, and the other is the user push button we can use it for various purposes. Also, there are few LEDs fa or indication.

For more details on the Nucleo board, we can always go to the ST website which is www dot st dot com and check for the relevant schematics and other documentation. Where we can find all the jumpers, all the signals, and the connector details. Now, we will be moving on to an example. Here, we have selected a very basic example of a GPIO toggle. The LED is connected to one of the GPIOs and we will be talking about it.

This example we will be covering in three ways with each way we are moving ahead. We will be getting deeper inside STM32CubeMX and we will be getting more aware of it. First, the first way we will be directly using the example provided by the STM32Cube Package. In a second way, we will be selecting this Nucleo board. Correspondingly all the peripherals that are being done in the hardware will be initialized.

And we will be only working on the application layer. While thirdly we will be selecting the MCU. We will be initializing the peripheral server cells and correspondingly we will be working on the application layer and make our example up and running. We will be using IAR as IDE integrated development environment. Let us start with the very first way.

(Refer Slide Time: 17:35)

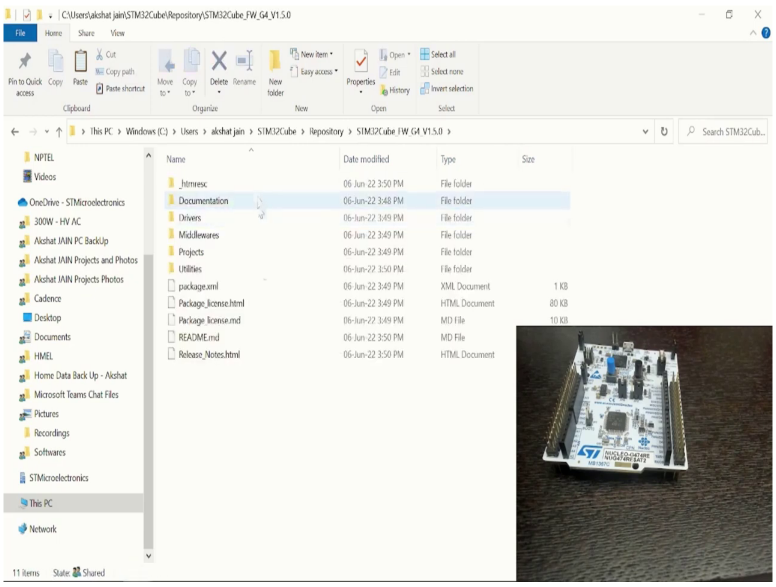

C:\Users\akshat jain\STM32Cube\Repository

This PC › Windows (C:) › Users › akshat jain › STM32Cube › Repository

Name	Date modified	Type	Size
STM32Cube_FW_F0_V1.11.3	10-Aug-21 10:06 AM	File folder	
STM32Cube_FW_G0_V1.3.0	09-Mar-20 4:17 PM	File folder	
STM32Cube_FW_G4_V1.5.0	06-Jun-22 3:50 PM	File folder	
stm32cube_fw_f0_v1113.zip	10-Aug-21 10:07 AM	zip Archive	58,524 KB
stm32cube_fw_g4_v150.zip	06-Jun-22 3:48 PM	zip Archive	201,771 KB

Let us move to the repository where exactly the package has been installed. By default, it will be installed in your C drive. Where there is a user you can find a folder STM32Cube and a corresponding repository. You can find all the packages that you have installed.

(Refer Slide Time: 17:54)



C:\Users\akshat jain\STM32Cube\Repository\STM32Cube_FW_G4_V1.5.0

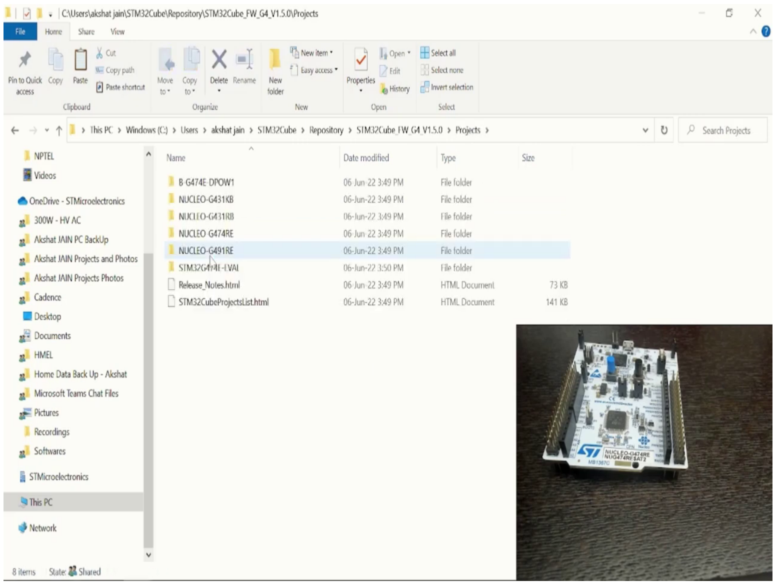
This PC › Windows (C:) › Users › akshat jain › STM32Cube › Repository › STM32Cube_FW_G4_V1.5.0 ›

Name	Date modified	Type	Size
_htmresc	06-Jun-22 3:50 PM	File folder	
Documentation	06-Jun-22 3:48 PM	File folder	
Drivers	06-Jun-22 3:49 PM	File folder	
Middlewares	06-Jun-22 3:49 PM	File folder	
Projects	06-Jun-22 3:49 PM	File folder	
Utilities	06-Jun-22 3:50 PM	File folder	
package.xml	06-Jun-22 3:49 PM	XML Document	1 KB
Package_license.html	06-Jun-22 3:49 PM	HTML Document	80 KB
Package_license.md	06-Jun-22 3:49 PM	MD File	10 KB
README.md	06-Jun-22 3:50 PM	MD File	
Release_Notes.html	06-Jun-22 3:50 PM	HTML Document	

11 items State: Shared

So, as we are using G4, 1.5 version. We can go inside and in that, we can see various folders of the middleware driver documentation. Now, we have to move to the projects.

(Refer Slide Time: 18:03)



C:\Users\akshat jain\STM32Cube\Repository\STM32Cube_FW_G4_V1.5.0\Projects

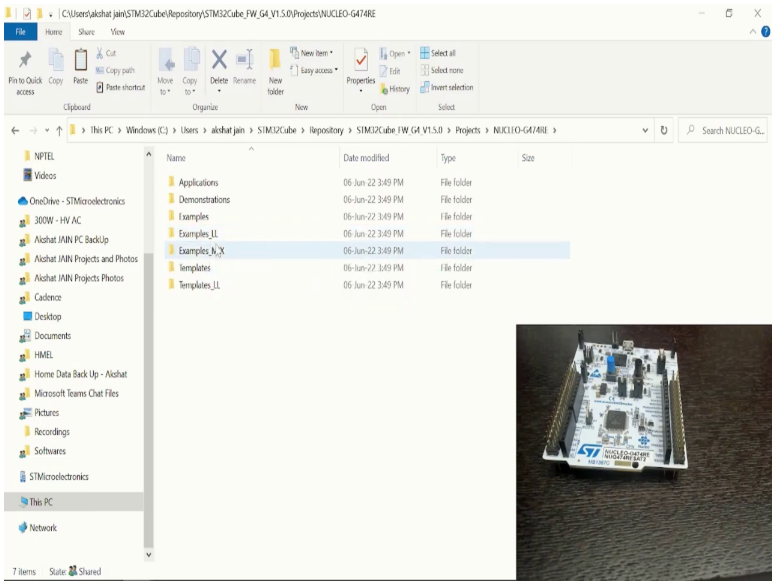
This PC › Windows (C:) › Users › akshat jain › STM32Cube › Repository › STM32Cube_FW_G4_V1.5.0 › Projects ›

Name	Date modified	Type	Size
B-G474E-DPOW1	06-Jun-22 3:49 PM	File folder	
NUCLEO-G431KB	06-Jun-22 3:49 PM	File folder	
NUCLEO-G431RB	06-Jun-22 3:49 PM	File folder	
NUCLEO-G474RE	06-Jun-22 3:49 PM	File folder	
NUCLEO-G491RE	06-Jun-22 3:49 PM	File folder	
STM32G474E-EVAL	06-Jun-22 3:50 PM	File folder	
Release_Notes.html	06-Jun-22 3:49 PM	HTML Document	73 KB
STM32CubeProjectsList.html	06-Jun-22 3:49 PM	HTML Document	141 KB

8 items State: Shared

In the projects also there are several Nucleo boards. In our case, we are using the Nucleo-G474RE board.

(Refer Slide Time: 18:12)



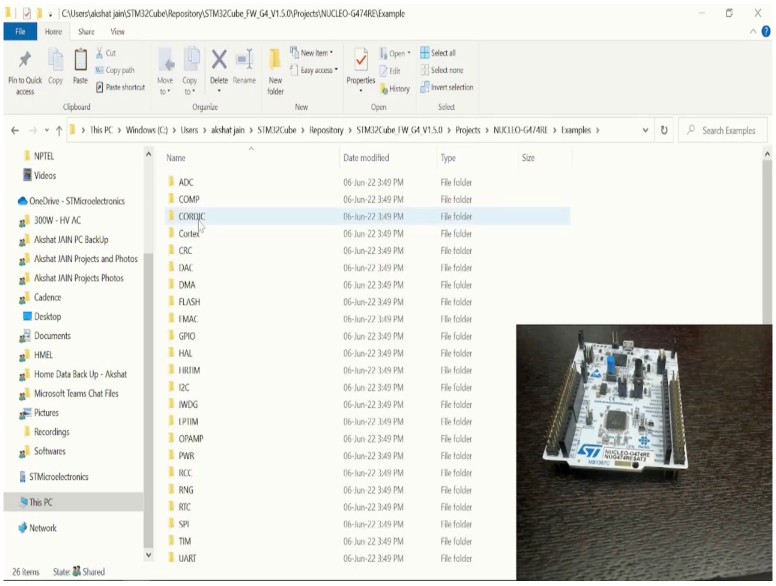
C:\Users\akshat jain\STM32Cube\Repository\STM32Cube_FW_G4_V1.5.0\Projects\NUCLEO-G474RE

Name	Date modified	Type	Size
Applications	06-Jun-22 3:49 PM	File folder	
Demonstrations	06-Jun-22 3:49 PM	File folder	
Examples	06-Jun-22 3:49 PM	File folder	
Examples_LL	06-Jun-22 3:49 PM	File folder	
Examples_MIX	06-Jun-22 3:49 PM	File folder	
Templates	06-Jun-22 3:49 PM	File folder	
Templates_LL	06-Jun-22 3:49 PM	File folder	

7 items State: Shared

In this board, there are several examples of the lower layer, mix, and templates. These templates are basically for if we are making some new project on our own. So, we can use these templates.

(Refer Slide Time: 18:31)

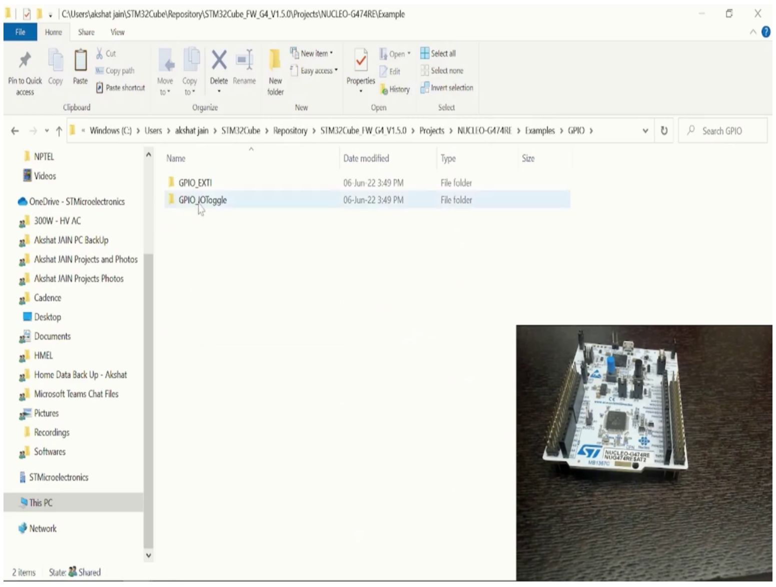


C:\Users\akshat jain\STM32Cube\Repository\STM32Cube_FW_G4_V1.5.0\Projects\NUCLEO-G474RE\Example

Name	Date modified	Type	Size
ADC	06-Jun-22 3:49 PM	File folder	
COMP	06-Jun-22 3:49 PM	File folder	
CORDIC	06-Jun-22 3:49 PM	File folder	
Cortex	06-Jun-22 3:49 PM	File folder	
CRC	06-Jun-22 3:49 PM	File folder	
DAC	06-Jun-22 3:49 PM	File folder	
DMA	06-Jun-22 3:49 PM	File folder	
FLASH	06-Jun-22 3:49 PM	File folder	
FMAC	06-Jun-22 3:49 PM	File folder	
GPIO	06-Jun-22 3:49 PM	File folder	
HAL	06-Jun-22 3:49 PM	File folder	
HRTIM	06-Jun-22 3:49 PM	File folder	
I2C	06-Jun-22 3:49 PM	File folder	
IWDG	06-Jun-22 3:49 PM	File folder	
LPTIM	06-Jun-22 3:49 PM	File folder	
OPAMP	06-Jun-22 3:49 PM	File folder	
PWR	06-Jun-22 3:49 PM	File folder	
RCC	06-Jun-22 3:49 PM	File folder	
RNG	06-Jun-22 3:49 PM	File folder	
RTC	06-Jun-22 3:49 PM	File folder	
SPI	06-Jun-22 3:49 PM	File folder	
TIM	06-Jun-22 3:49 PM	File folder	
UART	06-Jun-22 3:49 PM	File folder	

26 items State: Shared

(Refer Slide Time: 18:35)

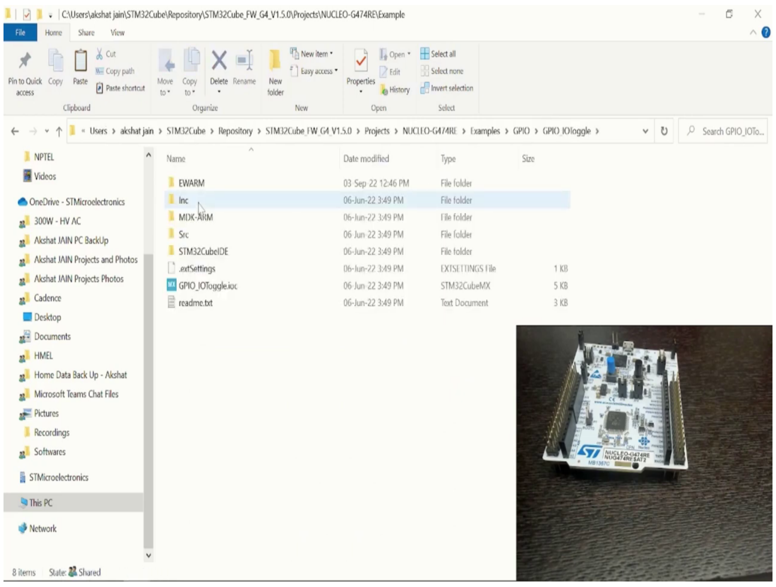

C:\Users\akshat jain\STM32Cube\Repository\STM32Cube_FW_G4_V1.5.0\Projects\NUCLEO-G474RE\Example

Windows (C:) › Users › akshat jain › STM32Cube › Repository › STM32Cube_FW_G4_V1.5.0 › Projects › NUCLEO-G474RE › Examples › GPIO ›

Name	Date modified	Type	Size
GPIO_EXTI	06-Jun-22 3:49 PM	File folder	
GPIO_IOToggle	06-Jun-22 3:49 PM	File folder	

2 items State: Shared

(Refer Slide Time: 18:37)


C:\Users\akshat jain\STM32Cube\Repository\STM32Cube_FW_G4_V1.5.0\Projects\NUCLEO-G474RE\Example

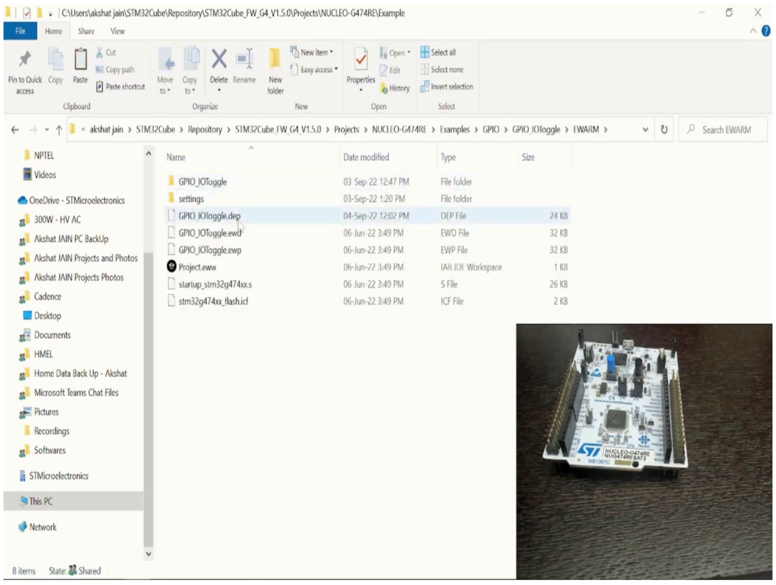
Users › akshat jain › STM32Cube › Repository › STM32Cube_FW_G4_V1.5.0 › Projects › NUCLEO-G474RE › Examples › GPIO › GPIO_IOToggle ›

Name	Date modified	Type	Size
EWARM	03-Sep-22 12:46 PM	File folder	
Inc	06-Jun-22 3:49 PM	File folder	
MDK-ARM	06-Jun-22 3:49 PM	File folder	
Src	06-Jun-22 3:49 PM	File folder	
STM32CubeIDE	06-Jun-22 3:49 PM	File folder	
.extSettings	06-Jun-22 3:49 PM	EXTSETTINGS File	1 KB
GPIO_IOToggle.ioc	06-Jun-22 3:49 PM	STM32CubeMX	5 KB
readme.txt	06-Jun-22 3:49 PM	Text Document	3 KB

8 items State: Shared

So, right now we can go to an example and we can go to an example of GPIO and GPIO-IO toggle. Now, as I just mentioned that we will be using then IAR. So, we can go to EWARM. Otherwise correspondingly we can go for the STM32CubeIDE or Keil IDE.

(Refer Slide Time: 18:51)

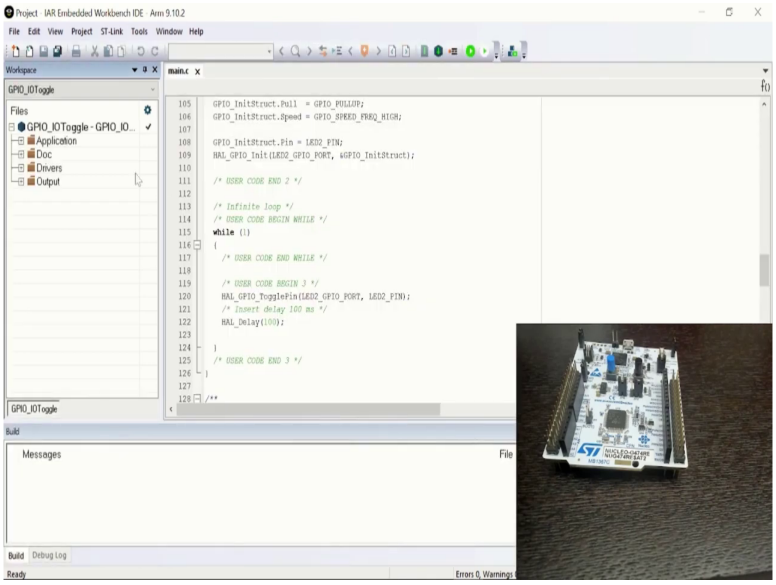

C:\Users\akshat jain\STM32Cube\Repository\STM32Cube_FW_G4_V1.5.0\Projects\NUCLEO-G474RE\Example

akshat jain › STM32Cube › Repository › STM32Cube_FW_G4_V1.5.0 › Projects › NUCLEO-G474RE › Examples › GPIO › GPIO_IOToggle › EWARM

Name	Date modified	Type	Size
GPIO_IOToggle	03-Sep-22 12:47 PM	File folder	
settings	03-Sep-22 1:20 PM	File folder	
GPIO_IOToggle.dep	04-Sep-22 12:02 PM	DEP File	24 KB
GPIO_IOToggle.ewd	06-Jun-22 3:49 PM	EWD File	32 KB
GPIO_IOToggle.ewp	06-Jun-22 3:49 PM	EWP File	32 KB
Project.eww	06-Jun-22 3:49 PM	IAR IDE Workspace	1 KB
startup_stm32g474xx.s	06-Jun-22 3:49 PM	S File	26 KB
stm32g474xx_flash.icf	06-Jun-22 3:49 PM	ICF File	2 KB

8 items State: Shared

(Refer Slide Time: 18:59)


Project - IAR Embedded Workbench IDE - Arm 9.10.2

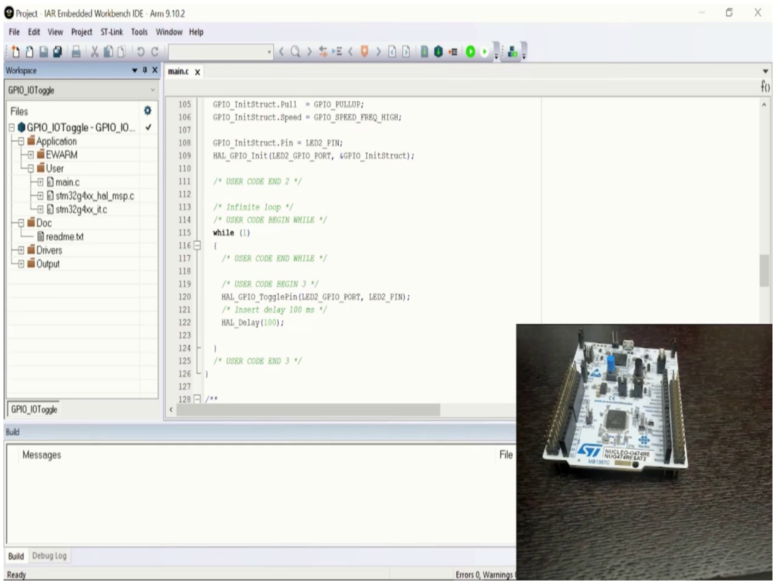
```c
  GPIO_InitStruct.Pull  = GPIO_PULLUP;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_HIGH;

  GPIO_InitStruct.Pin = LED2_PIN;
  HAL_GPIO_Init(LED2_GPIO_PORT, &GPIO_InitStruct);

  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */
    HAL_GPIO_TogglePin(LED2_GPIO_PORT, LED2_PIN);
    /* Insert delay 100 ms */
    HAL_Delay(100);

  }
  /* USER CODE END 3 */
}

/**
```

Messages

Ready   Errors 0, Warnings

So, for the IAR, we need to go to EWARM. So, let me copy this path and open this in an IAR workbench. So, this project has been opened here.

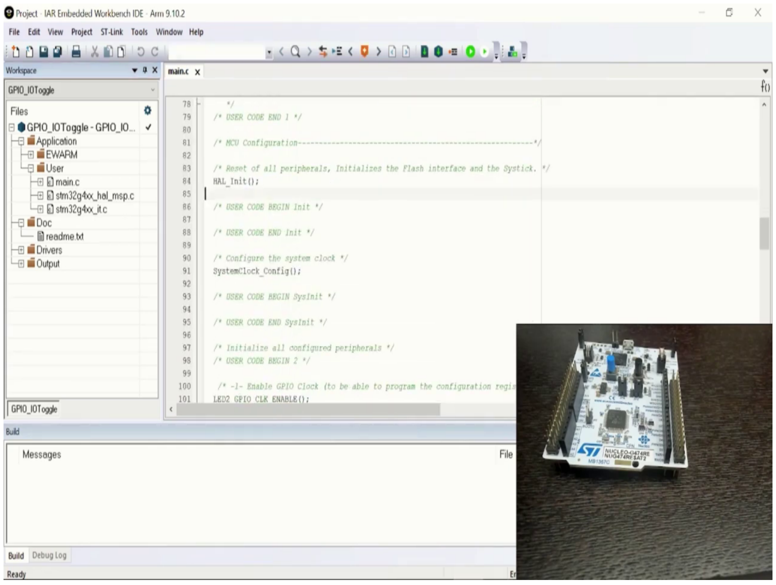
(Refer Slide Time: 19:19)



```
  GPIO_InitStruct.Pull  = GPIO_PULLUP;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_HIGH;

  GPIO_InitStruct.Pin = LED2_PIN;
  HAL_GPIO_Init(LED2_GPIO_PORT, &GPIO_InitStruct);

  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */
    HAL_GPIO_TogglePin(LED2_GPIO_PORT, LED2_PIN);
    /* Insert delay 100 ms */
    HAL_Delay(100);

  }
  /* USER CODE END 3 */
}

/**
```

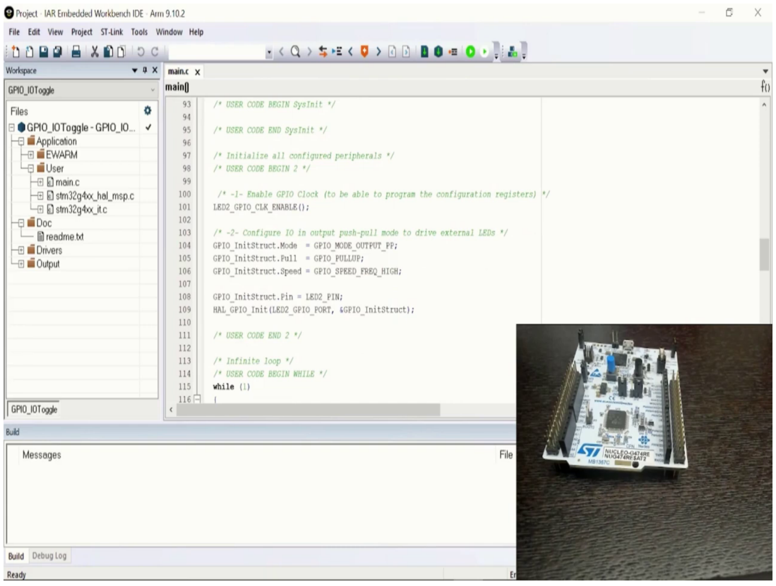
So, we can see this in the documentation. There is a readme file also where we can find all the details in the application and the user, there is a main dot c file, an interrupt dot c file, and the hall MSP file.

(Refer Slide Time: 19:42)



```
    */
  /* USER CODE END 1 */

  /* MCU Configuration--------------------------------------------------------*/

  /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
  HAL_Init();

  /* USER CODE BEGIN Init */

  /* USER CODE END Init */

  /* Configure the system clock */
  SystemClock_Config();

  /* USER CODE BEGIN SysInit */

  /* USER CODE END SysInit */

  /* Initialize all configured peripherals */
  /* USER CODE BEGIN 2 */

   /* -1- Enable GPIO Clock (to be able to program the configuration regis
  LED2_GPIO_CLK_ENABLE();
```

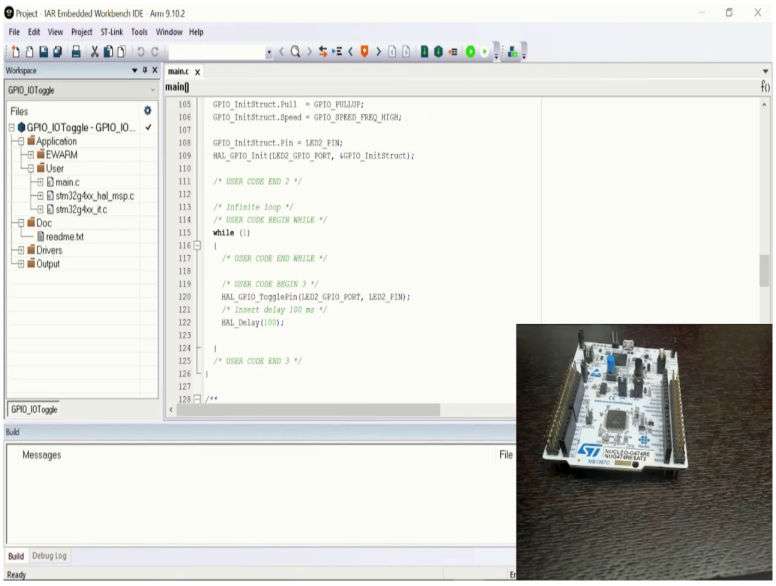
(Refer Slide Time: 19:47)



```
  /* USER CODE BEGIN SysInit */

  /* USER CODE END SysInit */

  /* Initialize all configured peripherals */
  /* USER CODE BEGIN 2 */

   /* -1- Enable GPIO Clock (to be able to program the configuration registers) */
  LED2_GPIO_CLK_ENABLE();

  /* -2- Configure IO in output push-pull mode to drive external LEDs */
  GPIO_InitStruct.Mode  = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull  = GPIO_PULLUP;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_HIGH;

  GPIO_InitStruct.Pin = LED2_PIN;
  HAL_GPIO_Init(LED2_GPIO_PORT, &GPIO_InitStruct);

  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
```

If we do so, the mean dot c file is already open. If we go from the top the main function has started the HAL initialization layer.

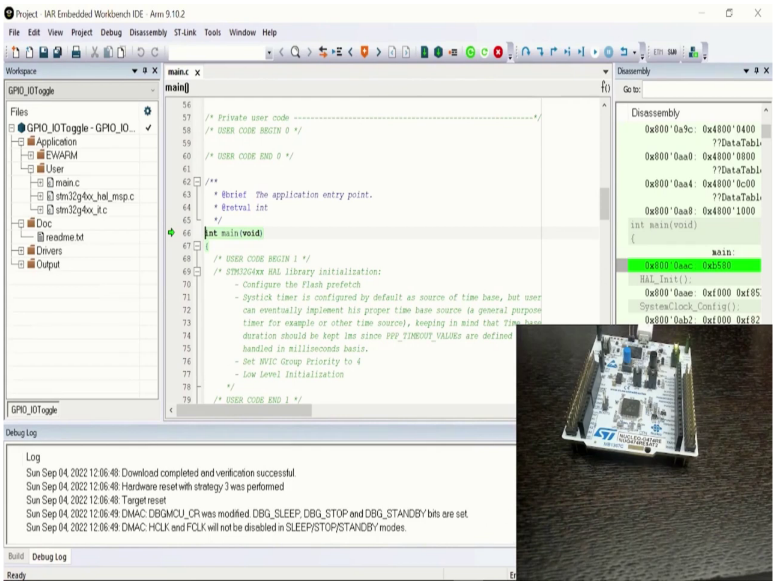
(Refer Slide Time: 19:59)



```
  GPIO_InitStruct.Pull  = GPIO_PULLUP;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_HIGH;

  GPIO_InitStruct.Pin = LED2_PIN;
  HAL_GPIO_Init(LED2_GPIO_PORT, &GPIO_InitStruct);

  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */
    HAL_GPIO_TogglePin(LED2_GPIO_PORT, LED2_PIN);
    /* Insert delay 100 ms */
    HAL_Delay(100);

  }
  /* USER CODE END 3 */
}

/**
```

Then the system clock as we have just mentioned is an example of GPIO togging where one led will be toggling. So, this one GPIO port is configured and correspondingly there will be a delay of 100 millisecond in the while one loop that we are working on. So, let me connect this Nucleo board. So, if you see it has been connected and now I will be programming this firmware onto this Nucleo board. You can see this LED is turn screens means the communication is happening.

(Refer Slide Time: 20:37)



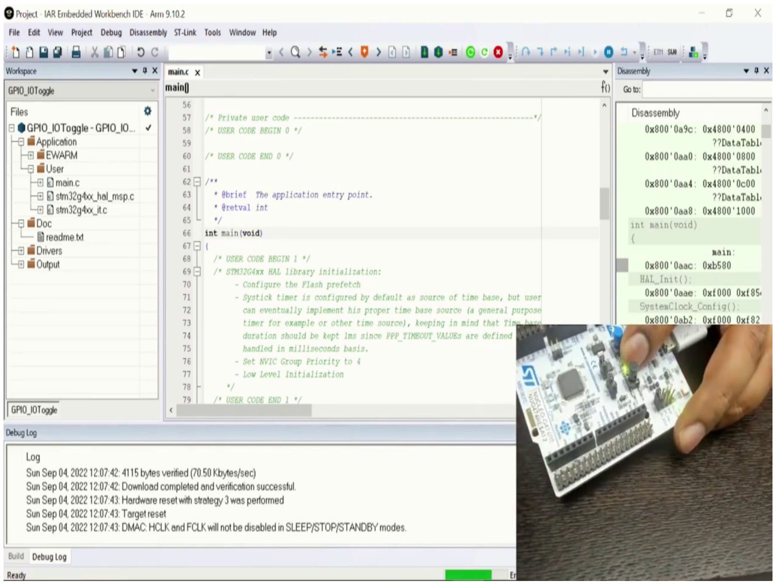
```
/* Private user code ---------------------------------------------------------*/
/* USER CODE BEGIN 0 */

/* USER CODE END 0 */

/**
  * @brief  The application entry point.
  * @retval int
  */
int main(void)
{
  /* USER CODE BEGIN 1 */
  /* STM32G4xx HAL library initialization:
       - Configure the Flash prefetch
       - Systick timer is configured by default as source of time base, but user
         can eventually implement his proper time base source (a general purpose
         timer for example or other time source), keeping in mind that Time base
         duration should be kept 1ms since PPP_TIMEOUT_VALUEs are defined
         handled in milliseconds basis.
       - Set NVIC Group Priority to 4
       - Low Level Initialization
     */
  /* USER CODE END 1 */
```

```
Disassembly
    0x800'0a9c:  0x4800'0400
                    ??DataTabl
    0x800'0aa0:  0x4800'0800
                    ??DataTabl
    0x800'0aa4:  0x4800'0c00
                    ??DataTabl
    0x800'0aa8:  0x4800'1000
int main(void)
{
                    main:
    0x800'0aac   0xb580
  HAL_Init();
    0x800'0aae:  0xf000 0xf85
  SystemClock_Config();
    0x800'0ab2:  0xf000 0xf82
```

```
Log
Sun Sep 04, 2022 12:06:48: Download completed and verification successful.
Sun Sep 04, 2022 12:06:48: Hardware reset with strategy 3 was performed
Sun Sep 04, 2022 12:06:48: Target reset
Sun Sep 04, 2022 12:06:49: DMAC: DBGMCU_CR was modified. DBG_SLEEP, DBG_STOP and DBG_STANDBY bits are set.
Sun Sep 04, 2022 12:06:49: DMAC: HCLK and FCLK will not be disabled in SLEEP/STOP/STANDBY modes.
```

Just a couple of seconds more. Now, the firmware has been loaded and now I make it run. Now, as you could see this LED starts blinking with a delay of 100 milliseconds. So, this is the direct example that we have used.

(Refer Slide Time: 20:54)



```
/* Private user code ---------------------------------------------------------*/
/* USER CODE BEGIN 0 */

/* USER CODE END 0 */

/**
  * @brief  The application entry point.
  * @retval int
  */
int main(void)
{
  /* USER CODE BEGIN 1 */
  /* STM32G4xx HAL library initialization:
       - Configure the Flash prefetch
       - Systick timer is configured by default as source of time base, but user
         can eventually implement his proper time base source (a general purpose
         timer for example or other time source), keeping in mind that Time base
         duration should be kept 1ms since PPP_TIMEOUT_VALUEs are defined
         handled in milliseconds basis.
       - Set NVIC Group Priority to 4
       - Low Level Initialization
     */
  /* USER CODE END 1 */
```

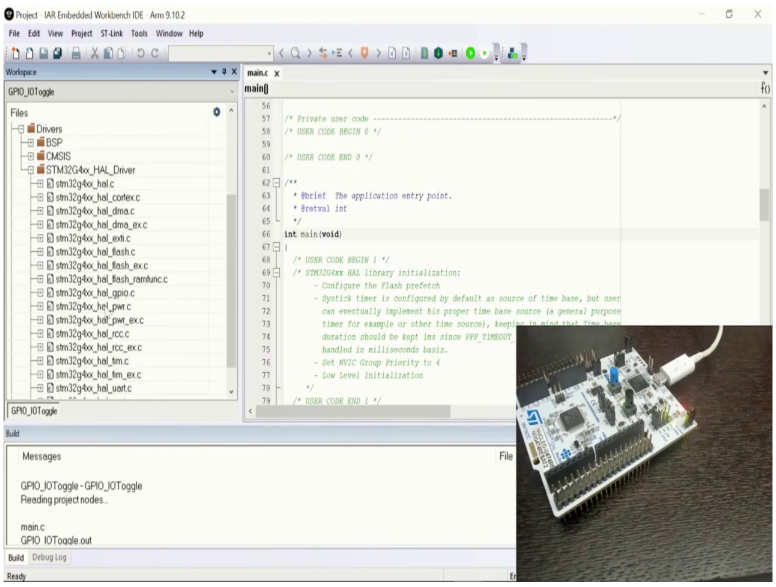
```
Disassembly
    0x800'0a9c:  0x4800'0400
                    ??DataTabl
    0x800'0aa0:  0x4800'0800
                    ??DataTabl
    0x800'0aa4:  0x4800'0c00
                    ??DataTabl
    0x800'0aa8:  0x4800'1000
int main(void)
{
                    main:
    0x800'0aac:  0xb580
  HAL_Init();
    0x800'0aae:  0xf000 0xf85
  SystemClock_Config();
    0x800'0ab2:  0xf000 0xf82
```

```
Log
Sun Sep 04, 2022 12:07:42: 4115 bytes verified (70.50 Kbytes/sec)
Sun Sep 04, 2022 12:07:42: Download completed and verification successful.
Sun Sep 04, 2022 12:07:43: Hardware reset with strategy 3 was performed
Sun Sep 04, 2022 12:07:43: Target reset
Sun Sep 04, 2022 12:07:43: DMAC: HCLK and FCLK will not be disabled in SLEEP/STOP/STANDBY modes.
```

Let me just change the delay to 1 second. So, this is the 1000 milliseconds corresponding to 1 second. I just reload the program again.

It takes a couple of seconds to reprogram. Now, if I run the program, now if you could see that the LED is blinking with a delay of 1 second. Now I just stop debugging. So, the main idea is that it generates not only the C code but the complete project. We can use directly the project as per our application requirements.

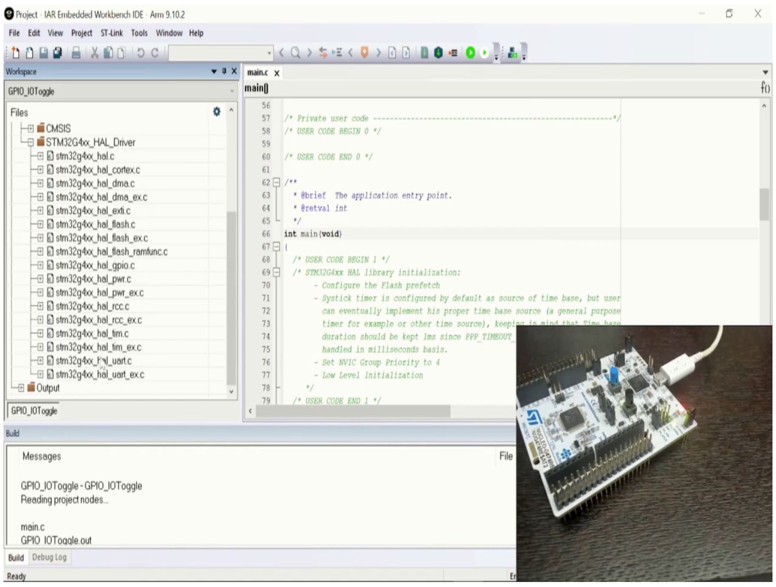
(Refer Slide Time: 21:48)



```c

/* Private user code ---------------------------------------------------------*/
/* USER CODE BEGIN 0 */

/* USER CODE END 0 */

/**
  * @brief  The application entry point.
  * @retval int
  */
int main(void)
{
  /* USER CODE BEGIN 1 */
  /* STM32G4xx HAL library initialization:
       - Configure the Flash prefetch
       - Systick timer is configured by default as source of time base, but user
         can eventually implement his proper time base source (a general purpose
         timer for example or other time source), keeping in mind that Time base
         duration should be kept 1ms since PPP_TIMEOUT_
         handled in milliseconds basis.
       - Set NVIC Group Priority to 4
       - Low Level Initialization
     */
  /* USER CODE END 1 */
```

```
Messages                                                                File

GPIO_IOToggle - GPIO_IOToggle
Reading project nodes...

main.c
GPIO_IOToggle.out
```

(Refer Slide Time: 21:59)

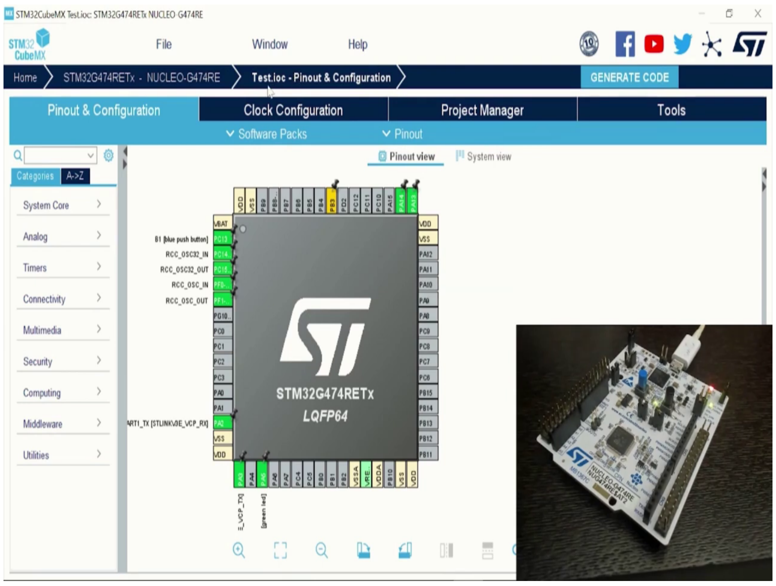


```c

/* Private user code ---------------------------------------------------------*/
/* USER CODE BEGIN 0 */

/* USER CODE END 0 */

/**
  * @brief  The application entry point.
  * @retval int
  */
int main(void)
{
  /* USER CODE BEGIN 1 */
  /* STM32G4xx HAL library initialization:
       - Configure the Flash prefetch
       - Systick timer is configured by default as source of time base, but user
         can eventually implement his proper time base source (a general purpose
         timer for example or other time source), keeping in mind that Time base
         duration should be kept 1ms since PPP_TIMEOUT_
         handled in milliseconds basis.
       - Set NVIC Group Priority to 4
       - Low Level Initialization
     */
  /* USER CODE END 1 */
```

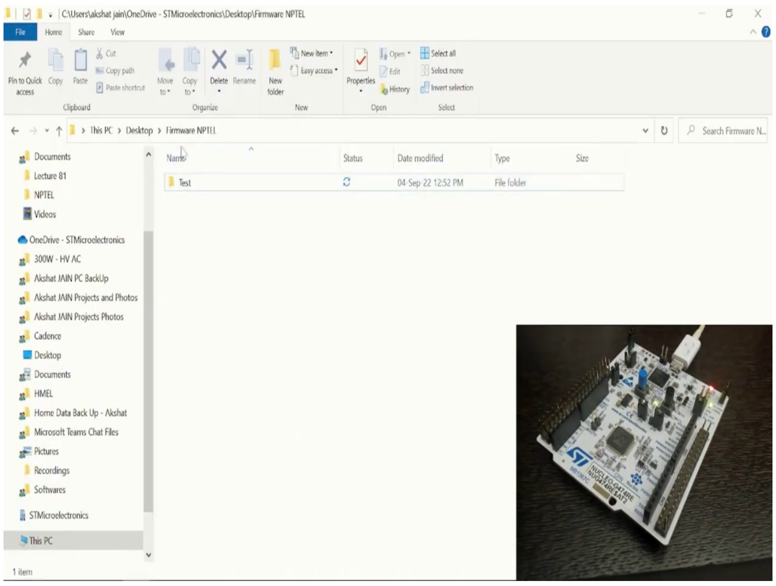
```
Messages                                                                File

GPIO_IOToggle - GPIO_IOToggle
Reading project nodes...

main.c
GPIO_IOToggle.out
```

There are several drivers. HAL drivers as we have mentioned in the very beginning regarding the HAL driver. Whatever various communication that we want to use either UART, timer, or whatever peripherals we are initializing. It will be correspondingly reflected here. Let us say if you are using I2C or SPI correspondingly driver files will be flashed here. And we can use direct function calls.

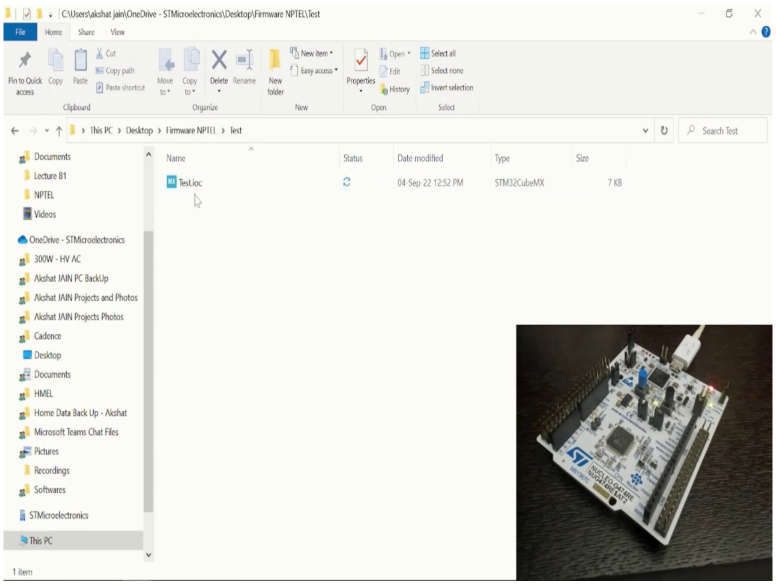
(Refer Slide Time: 22:17)



(Refer Slide Time: 22:24)


C:\Users\akshat jain\OneDrive - STMicroelectronics\Desktop\Firmware NPTEL
File
Home
Share
View
This PC > Desktop > Firmware NPTEL
Name
Status
Date modified
Type
Size
Test
04-Sep-22 12:52 PM
File folder
1 item

(Refer Slide Time: 22:29)


C:\Users\akshat jain\OneDrive - STMicroelectronics\Desktop\Firmware NPTEL\Test

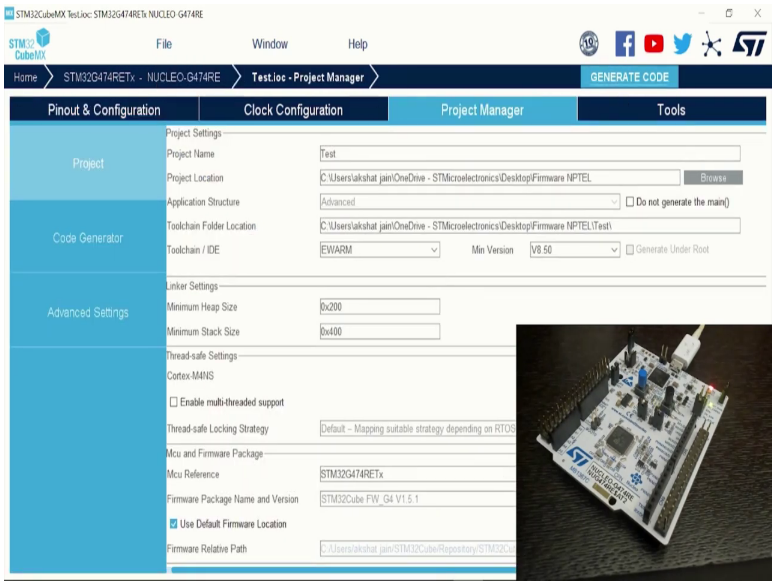
This PC › Desktop › Firmware NPTEL › Test

Name	Status	Date modified	Type	Size
Test.ioc		04-Sep-22 12:52 PM	STM32CubeMX	7 KB

1 item

Test. So, the file is saved as Test dot ioc. If you want to see I can see that in the Firmware NPTEL. There is a folder called Test and in that the file name the Test dot IOC has been saved now if I click on generate the code. So, it is asking if the latest Firmware Package is not being used. So, if you want to download it right now or not. So, I select no.

(Refer Slide Time: 23:03).


STM32CubeMX Test.ioc: STM32G474RETx NUCLEO-G474RE

Home › STM32G474RETx - NUCLEO-G474RE › Test.ioc - Project Manager

Project Settings
Project Name: Test
Project Location: C:\Users\akshat jain\OneDrive - STMicroelectronics\Desktop\Firmware NPTEL
Application Structure: Advanced ☐ Do not generate the main()
Toolchain Folder Location: C:\Users\akshat jain\OneDrive - STMicroelectronics\Desktop\Firmware NPTEL\Test\
Toolchain / IDE: EWARM Min Version: V8.50 ☐ Generate Under Root

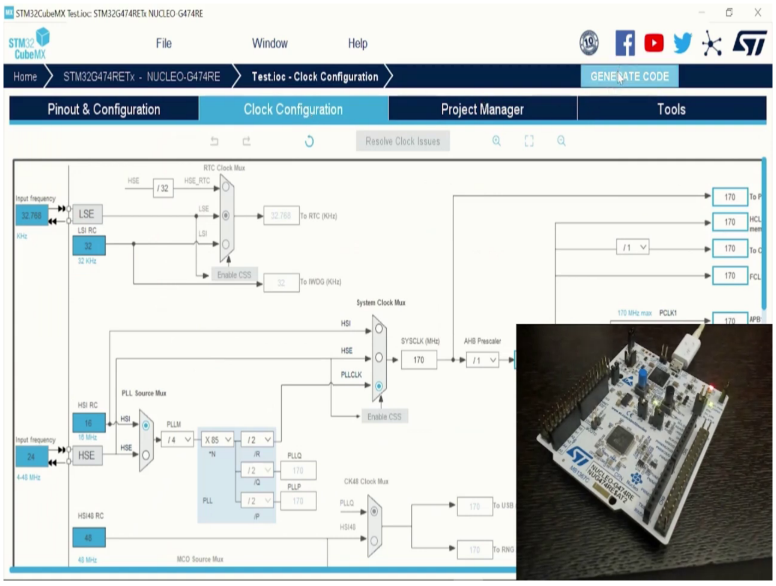
Linker Settings
Minimum Heap Size: 0x200
Minimum Stack Size: 0x400

Thread-safe Settings
Cortex-M4NS
☐ Enable multi-threaded support
Thread-safe Locking Strategy: Default – Mapping suitable strategy depending on RTOS

Mcu and Firmware Package
Mcu Reference: STM32G474RETx
Firmware Package Name and Version: STM32Cube FW_G4 V1.5.1
☑ Use Default Firmware Location
Firmware Relative Path: C:/Users/akshat jain/STM32Cube/Repository/STM32Cu

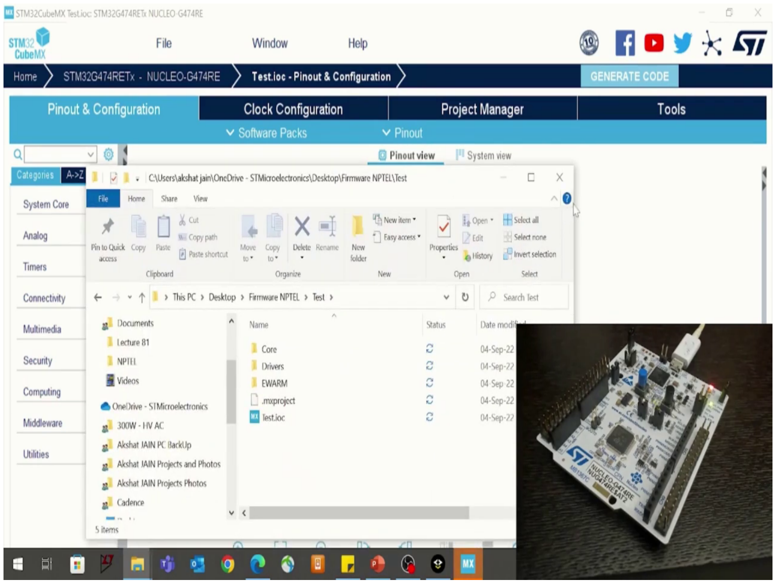
So, here we can choose if you want to use the latest version of the earlier version that we have. So, in the project manager, we can do all these settings.

(Refer Slide Time: 23:18)



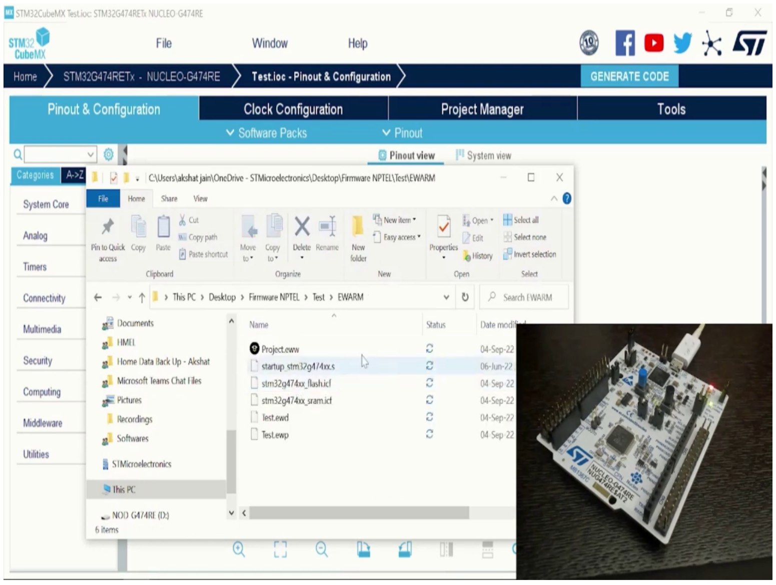
So, I will be telling you maybe in the next lecture on this project manager file. So, now if I click on the generated code. Again it will be taking a couple of seconds and generating the Firmware project of 3 different ids IAR, Keil, and STM32CubeIDE. So, in the meantime.

(Refer Slide Time: 23:56)

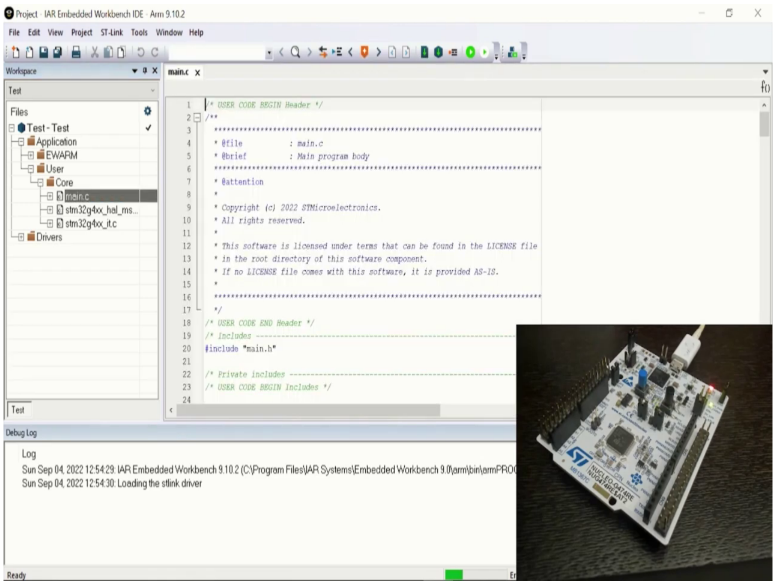


If I also show you the folder if you can see that several Test folders are being created here ok. So, it has been done. So, close it. So, it says if you want to open the project and open the folder. So, I do not want to open the project right now. I just want to open the folder ok.

(Refer Slide Time: 24:29)


STM32CubeMX Test.ioc: STM32G474RETx NUCLEO-G474RE
File
Window
Help
Home
STM32G474RETx - NUCLEO-G474RE
Test.ioc - Pinout & Configuration
GENERATE CODE
Pinout & Configuration
Clock Configuration
Project Manager
Tools
Software Packs
Pinout
Pinout view
System view
C:\Users\akshat jain\OneDrive - STMicroelectronics\Desktop\Firmware NPTEL\Test\EWARM
This PC › Desktop › Firmware NPTEL › Test › EWARM
Project.eww
startup_stm32g474xx.s
stm32g474xx_flash.icf
stm32g474xx_sram.icf
Test.ewd
Test.ewp

(Refer Slide Time: 24:35)

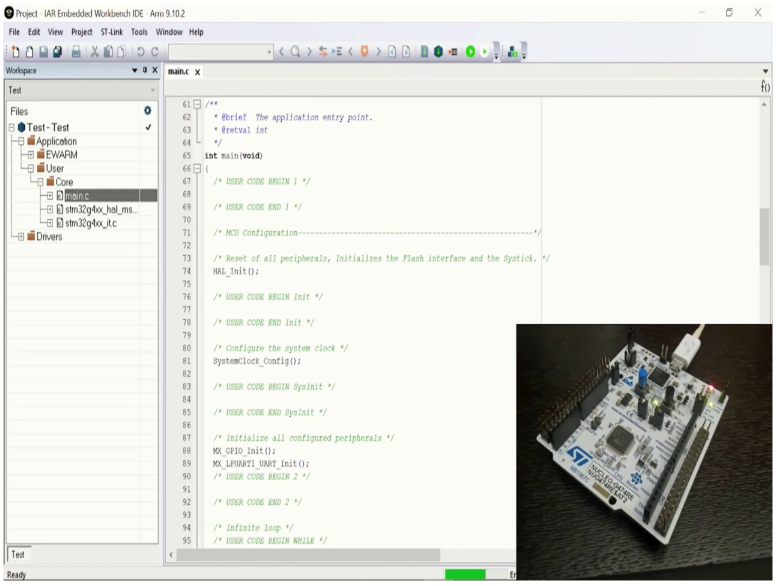


```
/* USER CODE BEGIN Header */
/**
  ******************************************************************************
  * @file           : main.c
  * @brief          : Main program body
  ******************************************************************************
  * @attention
  *
  * Copyright (c) 2022 STMicroelectronics.
  * All rights reserved.
  *
  * This software is licensed under terms that can be found in the LICENSE file
  * in the root directory of this software component.
  * If no LICENSE file comes with this software, it is provided AS-IS.
  *
  ******************************************************************************
  */
/* USER CODE END Header */
/* Includes ------------------------------------------------------------------
#include "main.h"

/* Private includes ----------------------------------------------------------
/* USER CODE BEGIN Includes */
```

```
Log
Sun Sep 04, 2022 12:54:29: IAR Embedded Workbench 9.10.2 (C:\Program Files\IAR Systems\Embedded Workbench 9.0\arm\bin\armPRO
Sun Sep 04, 2022 12:54:30: Loading the stlink driver
```

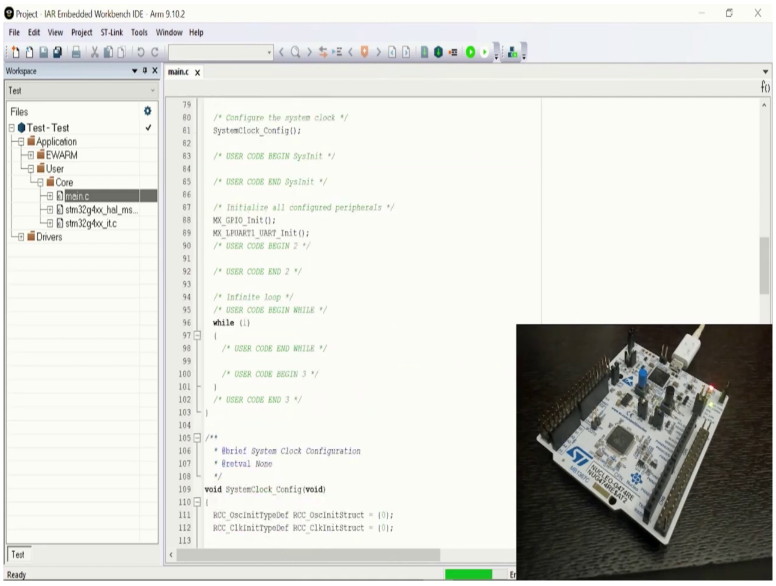
So, this is the same folder again I want to go to EWARM and I copy the path and open it in the IAR, one thing that we need to keep in mind is that right now all the peripherals that are preconfigured for this Nucleo board have been done here. So, there is no test example. So, in the application layer, we only have to write by ourselves to make the same example up and running.

(Refer Slide Time: 25:15)



```
/**
  * @brief  The application entry point.
  * @retval int
  */
int main(void)
{
  /* USER CODE BEGIN 1 */

  /* USER CODE END 1 */

  /* MCU Configuration--------------------------------------------------------*/

  /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
  HAL_Init();

  /* USER CODE BEGIN Init */

  /* USER CODE END Init */

  /* Configure the system clock */
  SystemClock_Config();

  /* USER CODE BEGIN SysInit */

  /* USER CODE END SysInit */

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  MX_LPUART1_UART_Init();
  /* USER CODE BEGIN 2 */

  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
```

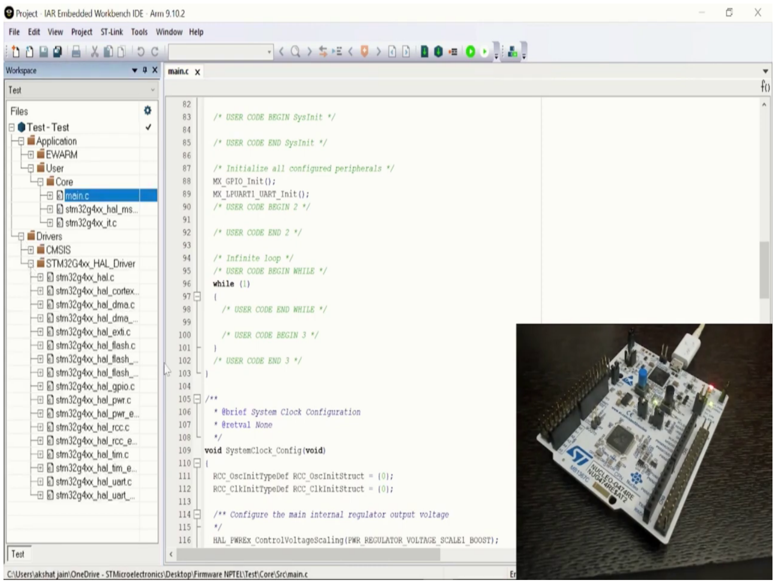
(Refer Slide Time: 25:18)



```

  /* Configure the system clock */
  SystemClock_Config();

  /* USER CODE BEGIN SysInit */

  /* USER CODE END SysInit */

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  MX_LPUART1_UART_Init();
  /* USER CODE BEGIN 2 */

  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */
  }
  /* USER CODE END 3 */
}

/**
  * @brief System Clock Configuration
  * @retval None
  */
void SystemClock_Config(void)
{
  RCC_OscInitTypeDef RCC_OscInitStruct = {0};
  RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};

```

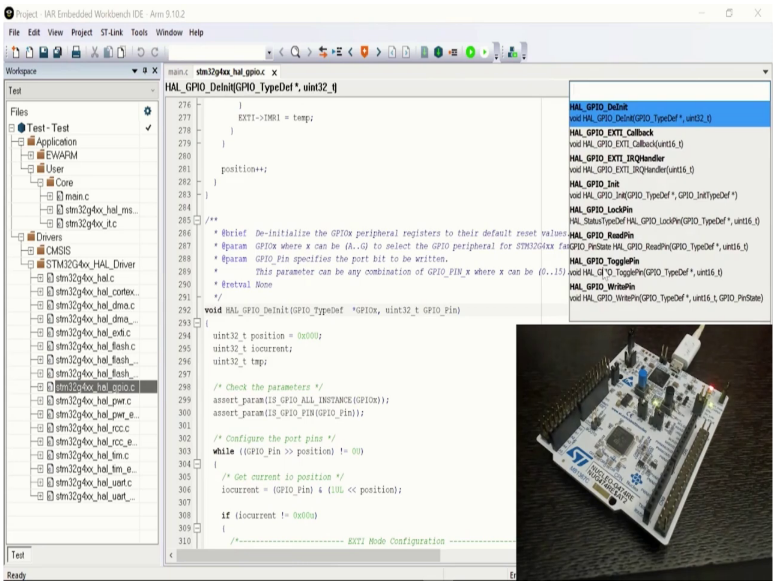
Again if you can see if you remember the initial code of the first way there also have the HAL in it, system clock config, GPIO config, and the UART, but if you remember there were a few lines written in the wild one with the HAL delay and the GPIO toggle which is not the case here. So, we have to write these lines right now.

(Refer Slide Time: 25:37)



```
  /* USER CODE BEGIN SysInit */

  /* USER CODE END SysInit */

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  MX_LPUART1_UART_Init();
  /* USER CODE BEGIN 2 */

  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */
  }
  /* USER CODE END 3 */
}

/**
  * @brief System Clock Configuration
  * @retval None
  */
void SystemClock_Config(void)
{
  RCC_OscInitTypeDef RCC_OscInitStruct = {0};
  RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};

  /** Configure the main internal regulator output voltage
  */
  HAL_PWREx_ControlVoltageScaling(PWR_REGULATOR_VOLTAGE_SCALE1_BOOST);
```

(Refer Slide Time: 25:44)



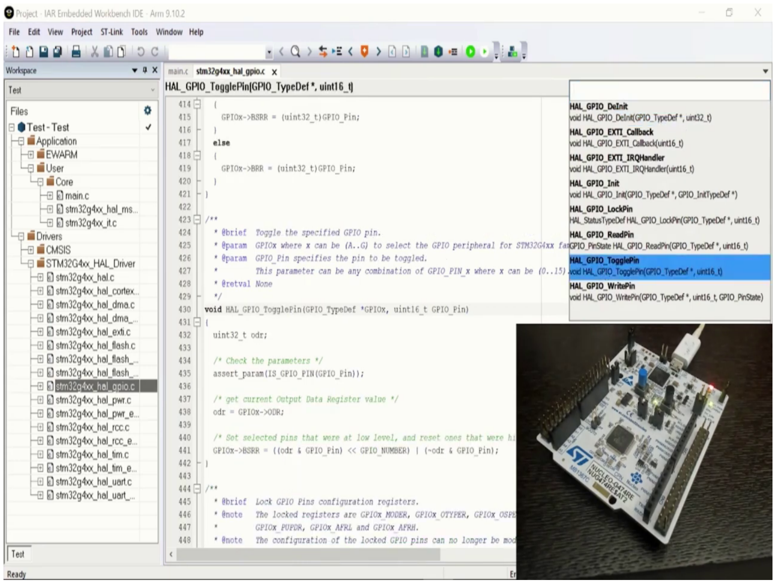
HAL_GPIO_DeInit(GPIO_TypeDef *, uint32_t)

```
        }
        EXTI->IMR1 = temp;
      }
    }

    position++;
  }
}

/**
  * @brief  De-initialize the GPIOx peripheral registers to their default reset values.
  * @param  GPIOx where x can be (A..G) to select the GPIO peripheral for STM32G4xx fa
  * @param  GPIO_Pin specifies the port bit to be written.
  *         This parameter can be any combination of GPIO_PIN_x where x can be (0..15).
  * @retval None
  */
void HAL_GPIO_DeInit(GPIO_TypeDef  *GPIOx, uint32_t GPIO_Pin)
{
  uint32_t position = 0x00U;
  uint32_t iocurrent;
  uint32_t tmp;

  /* Check the parameters */
  assert_param(IS_GPIO_ALL_INSTANCE(GPIOx));
  assert_param(IS_GPIO_PIN(GPIO_Pin));

  /* Configure the port pins */
  while ((GPIO_Pin >> position) != 0U)
  {
    /* Get current io position */
    iocurrent = (GPIO_Pin) & (1UL << position);

    if (iocurrent != 0x00u)
    {
      /*------------------------ EXTI Mode Configuration ---------------
```

```
HAL_GPIO_DeInit
void HAL_GPIO_DeInit(GPIO_TypeDef *, uint32_t)
HAL_GPIO_EXTI_Callback
void HAL_GPIO_EXTI_Callback(uint16_t)
HAL_GPIO_EXTI_IRQHandler
void HAL_GPIO_EXTI_IRQHandler(uint16_t)
HAL_GPIO_Init
void HAL_GPIO_Init(GPIO_TypeDef *, GPIO_InitTypeDef *)
HAL_GPIO_LockPin
HAL_StatusTypeDef HAL_GPIO_LockPin(GPIO_TypeDef *, uint16_t)
HAL_GPIO_ReadPin
GPIO_PinState HAL_GPIO_ReadPin(GPIO_TypeDef *, uint16_t)
HAL_GPIO_TogglePin
void HAL_GPIO_TogglePin(GPIO_TypeDef *, uint16_t)
HAL_GPIO_WritePin
void HAL_GPIO_WritePin(GPIO_TypeDef *, uint16_t, GPIO_PinState)
```

So, right now if I go to the driver and go to the HAL driver and go to the GPIO dot c file. In the GPIO dot c file on the right-hand side if you see that I could find all the functions that have been written here.

(Refer Slide Time: 25:55)



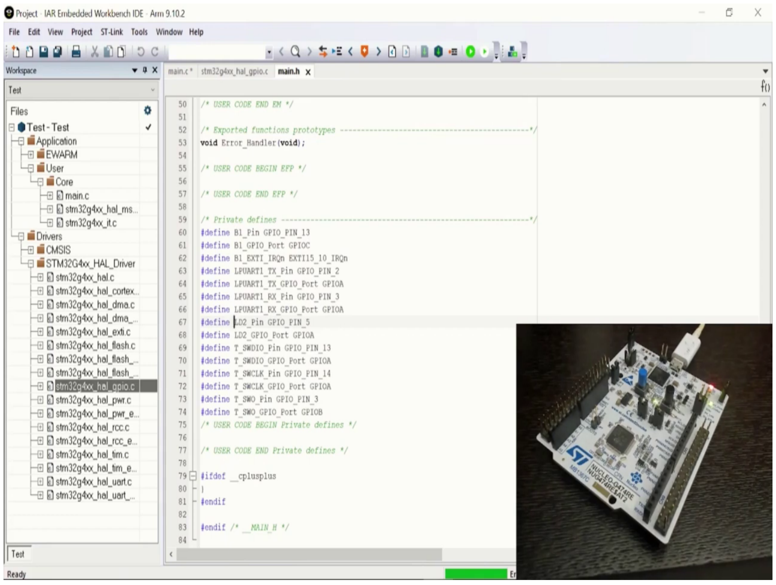
HAL_GPIO_TogglePin(GPIO_TypeDef *, uint16_t)

```
  {
    GPIOx->BSRR = (uint32_t)GPIO_Pin;
  }
  else
  {
    GPIOx->BRR = (uint32_t)GPIO_Pin;
  }
}

/**
  * @brief  Toggle the specified GPIO pin.
  * @param  GPIOx where x can be (A..G) to select the GPIO peripheral for STM32G4xx fa
  * @param  GPIO_Pin specifies the pin to be toggled.
  *         This parameter can be any combination of GPIO_PIN_x where x can be (0..15)
  * @retval None
  */
void HAL_GPIO_TogglePin(GPIO_TypeDef *GPIOx, uint16_t GPIO_Pin)
{
  uint32_t odr;

  /* Check the parameters */
  assert_param(IS_GPIO_PIN(GPIO_Pin));

  /* get current Output Data Register value */
  odr = GPIOx->ODR;

  /* Set selected pins that were at low level, and reset ones that were hi
  GPIOx->BSRR = ((odr & GPIO_Pin) << GPIO_NUMBER) | (~odr & GPIO_Pin);
}

/**
  * @brief  Lock GPIO Pins configuration registers.
  * @note   The locked registers are GPIOx_MODER, GPIOx_OTYPER, GPIOx_OSPE
  *         GPIOx_PUPDR, GPIOx_AFRL and GPIOx_AFRH.
  * @note   The configuration of the locked GPIO pins can no longer be mod
```

```
HAL_GPIO_DeInit
void HAL_GPIO_DeInit(GPIO_TypeDef *, uint32_t)
HAL_GPIO_EXTI_Callback
void HAL_GPIO_EXTI_Callback(uint16_t)
HAL_GPIO_EXTI_IRQHandler
void HAL_GPIO_EXTI_IRQHandler(uint16_t)
HAL_GPIO_Init
void HAL_GPIO_Init(GPIO_TypeDef *, GPIO_InitTypeDef *)
HAL_GPIO_LockPin
HAL_StatusTypeDef HAL_GPIO_LockPin(GPIO_TypeDef *, uint16_t)
HAL_GPIO_ReadPin
GPIO_PinState HAL_GPIO_ReadPin(GPIO_TypeDef *, uint16_t)
HAL_GPIO_TogglePin
void HAL_GPIO_TogglePin(GPIO_TypeDef *, uint16_t)
HAL_GPIO_WritePin
void HAL_GPIO_WritePin(GPIO_TypeDef *, uint16_t, GPIO_PinState)
```

So, I am selecting HAL GPIO toggle function here. So, I am copying this function here. And pasting it here now I want to go to which GPIO unit. Though I know it is PF5. So, I just want to go to LED2-Pin, I just want to configure ok.

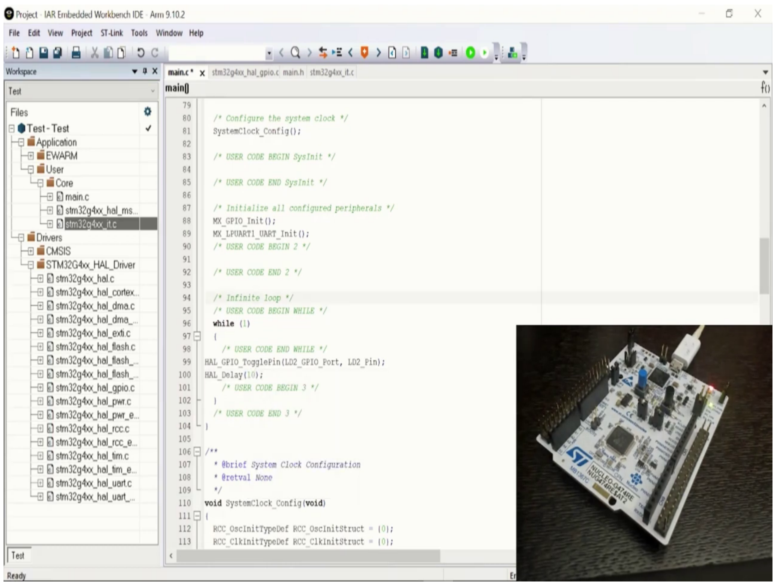
(Refer Slide Time: 26:25)



```
/* USER CODE END EM */

/* Exported functions prototypes ---------------------------------------------*/
void Error_Handler(void);

/* USER CODE BEGIN EFP */

/* USER CODE END EFP */

/* Private defines -----------------------------------------------------------*/
#define B1_Pin GPIO_PIN_13
#define B1_GPIO_Port GPIOC
#define B1_EXTI_IRQn EXTI15_10_IRQn
#define LPUART1_TX_Pin GPIO_PIN_2
#define LPUART1_TX_GPIO_Port GPIOA
#define LPUART1_RX_Pin GPIO_PIN_3
#define LPUART1_RX_GPIO_Port GPIOA
#define LD2_Pin GPIO_PIN_5
#define LD2_GPIO_Port GPIOA
#define T_SWDIO_Pin GPIO_PIN_13
#define T_SWDIO_GPIO_Port GPIOA
#define T_SWCLK_Pin GPIO_PIN_14
#define T_SWCLK_GPIO_Port GPIOA
#define T_SWO_Pin GPIO_PIN_3
#define T_SWO_GPIO_Port GPIOB
/* USER CODE BEGIN Private defines */

/* USER CODE END Private defines */

#ifdef __cplusplus
}
#endif

#endif /* __MAIN_H */
```

So, this is the LED2-Pin and this is the LED2- GPIO port. So, I copy the port first and paste it here while I the pin, I paste it here ok.

(Refer Slide Time: 26:41)



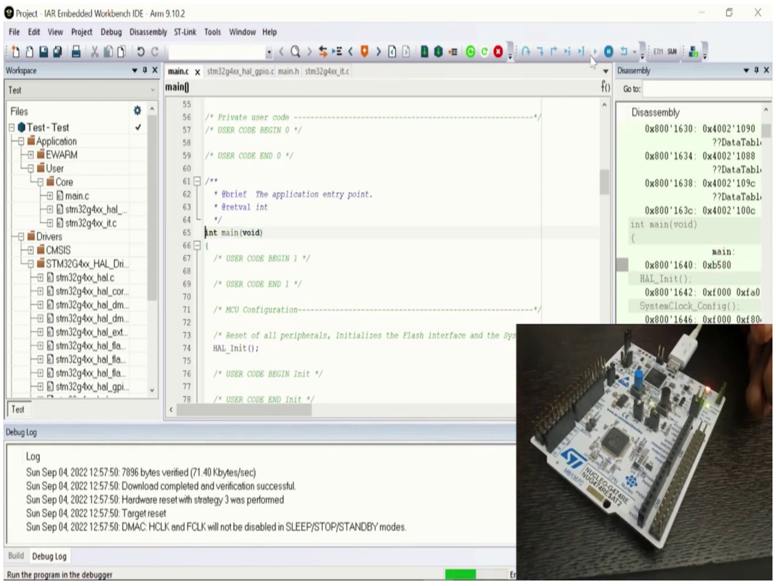
```
  /* Configure the system clock */
  SystemClock_Config();

  /* USER CODE BEGIN SysInit */

  /* USER CODE END SysInit */

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  MX_LPUART1_UART_Init();
  /* USER CODE BEGIN 2 */

  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
    /* USER CODE END WHILE */
HAL_GPIO_TogglePin(LD2_GPIO_Port, LD2_Pin);
HAL_Delay(10);
    /* USER CODE BEGIN 3 */
  }
  /* USER CODE END 3 */
}

/**
  * @brief System Clock Configuration
  * @retval None
  */
void SystemClock_Config(void)
{
  RCC_OscInitTypeDef RCC_OscInitStruct = {0};
  RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};
```

So, this has been done. Now, another function that I want to have is the HAL Delay. Just to update you by default. If you go until the interrupt dot c file. The stick handler this intake is of 1 millisecond.

So, whatever the value I pass here it will be correspondingly multiplied by 1 millisecond. So, if I just again pass maybe 10 milliseconds. So, the LED2 should blink very fast. Firstly, I need to basically rebuild my project and basically, it does the complete cleaning and you build the complete project. I request maybe if we can focus on this LED2-Pin which is blinking with a 1-second delay.

So, now my project has been built, and now if I download this firmware into my Nucleo board the Firmware has been downloaded.

(Refer Slide Time: 28:14)

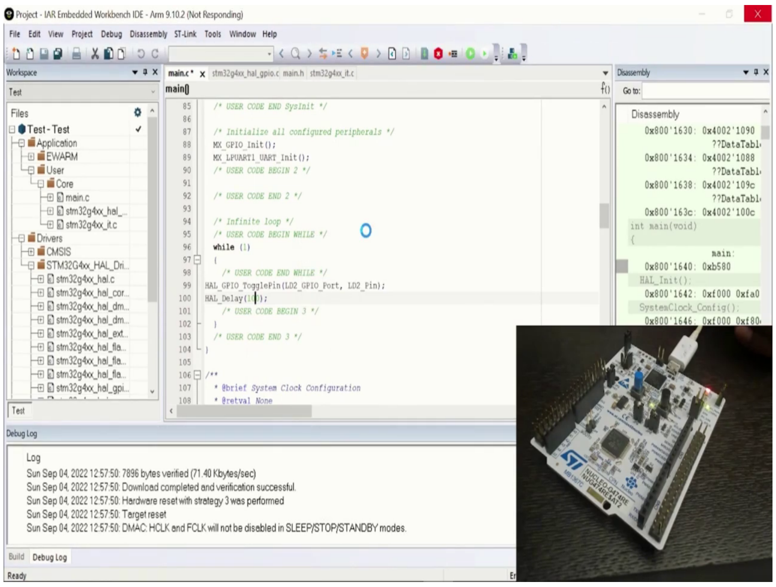


```
/* Private user code ---------------------------------------------------------*/
/* USER CODE BEGIN 0 */

/* USER CODE END 0 */

/**
  * @brief  The application entry point.
  * @retval int
  */
int main(void)
{
  /* USER CODE BEGIN 1 */

  /* USER CODE END 1 */

  /* MCU Configuration--------------------------------------------------------*/

  /* Reset of all peripherals, Initializes the Flash interface and the Sys
  HAL_Init();

  /* USER CODE BEGIN Init */

  /* USER CODE END Init */
```

```
Log
Sun Sep 04, 2022 12:57:50: 7896 bytes verified (71.40 Kbytes/sec)
Sun Sep 04, 2022 12:57:50: Download completed and verification successful.
Sun Sep 04, 2022 12:57:50: Hardware reset with strategy 3 was performed
Sun Sep 04, 2022 12:57:50: Target reset
Sun Sep 04, 2022 12:57:50: DMAC: HCLK and FCLK will not be disabled in SLEEP/STOP/STANDBY modes.
```

And if I make it run. As you could see the LED is almost just like it's completely glowing because the delay is very less.

(Refer Slide Time: 28:30)

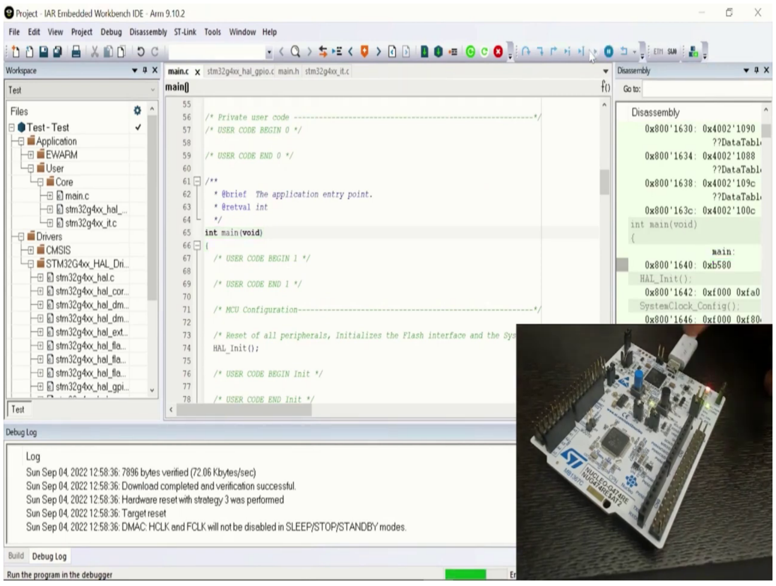


```
  /* USER CODE END SysInit */

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  MX_LPUART1_UART_Init();
  /* USER CODE BEGIN 2 */

  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
    /* USER CODE END WHILE */
HAL_GPIO_TogglePin(LD2_GPIO_Port, LD2_Pin);
HAL_Delay(100);
    /* USER CODE BEGIN 3 */
  }
  /* USER CODE END 3 */
}

/**
  * @brief System Clock Configuration
  * @retval None
```

```
Log
Sun Sep 04, 2022 12:57:50: 7896 bytes verified (71.40 Kbytes/sec)
Sun Sep 04, 2022 12:57:50: Download completed and verification successful.
Sun Sep 04, 2022 12:57:50: Hardware reset with strategy 3 was performed
Sun Sep 04, 2022 12:57:50: Target reset
Sun Sep 04, 2022 12:57:50: DMAC: HCLK and FCLK will not be disabled in SLEEP/STOP/STANDBY modes.
```

So, maybe I just increase the delay to 100 milliseconds and now I reprogram the microcontroller and see what exactly is the case. The Firmware is already downloaded and I click on run.

(Refer Slide Time: 29:15)



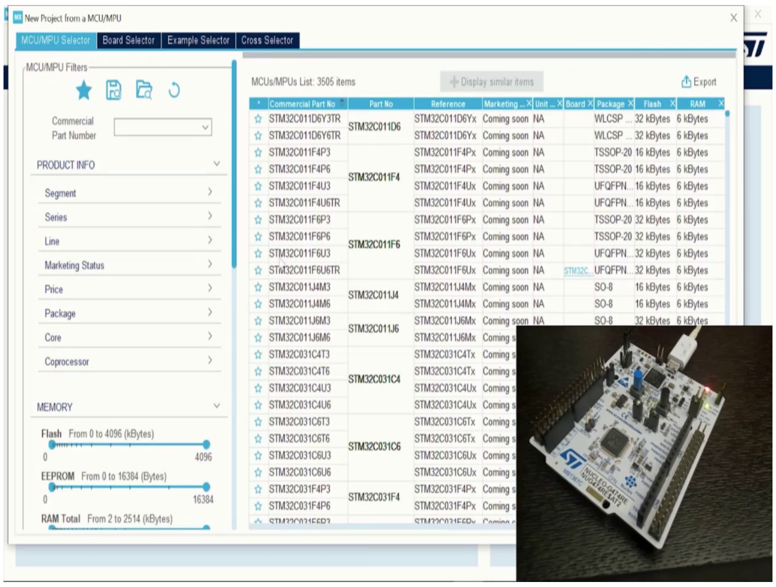
```
/* Private user code ---------------------------------------------------------*/
/* USER CODE BEGIN 0 */

/* USER CODE END 0 */

/**
  * @brief  The application entry point.
  * @retval int
  */
int main(void)
{
  /* USER CODE BEGIN 1 */

  /* USER CODE END 1 */

  /* MCU Configuration--------------------------------------------------------*/

  /* Reset of all peripherals, Initializes the Flash interface and the Sys
  HAL_Init();

  /* USER CODE BEGIN Init */

  /* USER CODE END Init */
```

```
Disassembly
    0x800'1630: 0x4002'1090
                     ??DataTabl
    0x800'1634: 0x4002'1088
                     ??DataTabl
    0x800'1638: 0x4002'109c
                     ??DataTabl
    0x800'163c: 0x4002'100c
int main(void)
{
                     main:
    0x800'1640: 0xb580
  HAL_Init();
    0x800'1642: 0xf000 0xfa0
  SystemClock_Config();
    0x800'1646: 0xf000 0xf80
```

```
Log
Sun Sep 04, 2022 12:58:36: 7896 bytes verified (72.06 Kbytes/sec)
Sun Sep 04, 2022 12:58:36: Download completed and verification successful.
Sun Sep 04, 2022 12:58:36: Hardware reset with strategy 3 was performed
Sun Sep 04, 2022 12:58:36: Target reset
Sun Sep 04, 2022 12:58:36: DMAC: HCLK and FCLK will not be disabled in SLEEP/STOP/STANDBY modes.
```

So, now if you could see that we can easily see the 100-millisecond delay. So, this is the second way of generating the code. Now, I stop the debugging and we move on to the third part for the last way we need to click on access to MCU selector. It will be displaying the complete list of the microcontroller. 32-bit microcontrollers. Though ST is having 8-bit microcontrollers also.

(Refer Slide Time: 29:58)

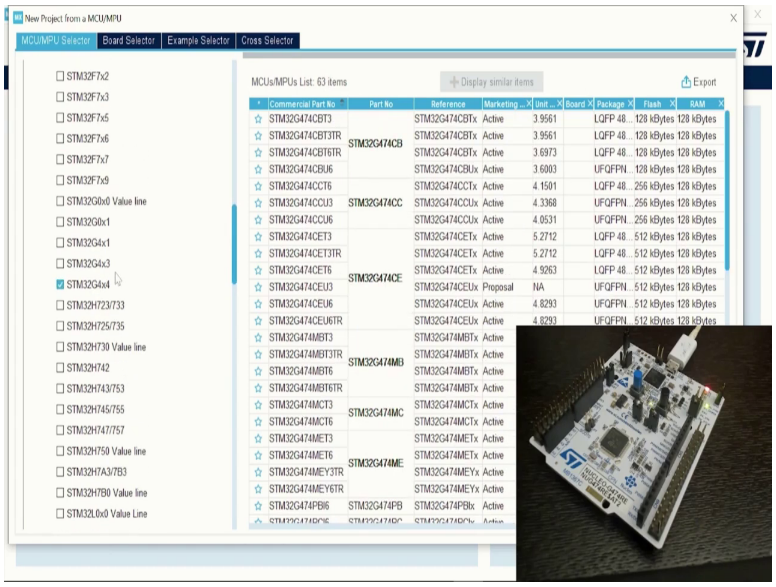


MCUs/MPUs List: 3505 items

*	Commercial Part No	Part No	Reference	Marketing ...	Unit ...	Board	Package	Flash	RAM
☆	STM32C011D6Y3TR	STM32C011D6	STM32C011D6Yx	Coming soon	NA		WLCSP ...	32 kBytes	6 kBytes
☆	STM32C011D6Y6TR		STM32C011D6Yx	Coming soon	NA		WLCSP ...	32 kBytes	6 kBytes
☆	STM32C011F4P3	STM32C011F4	STM32C011F4Px	Coming soon	NA		TSSOP-20	16 kBytes	6 kBytes
☆	STM32C011F4P6		STM32C011F4Px	Coming soon	NA		TSSOP-20	16 kBytes	6 kBytes
☆	STM32C011F4U3		STM32C011F4Ux	Coming soon	NA		UFQFPN...	16 kBytes	6 kBytes
☆	STM32C011F4U6TR		STM32C011F4Ux	Coming soon	NA		UFQFPN...	16 kBytes	6 kBytes
☆	STM32C011F6P3	STM32C011F6	STM32C011F6Px	Coming soon	NA		TSSOP-20	32 kBytes	6 kBytes
☆	STM32C011F6P6		STM32C011F6Px	Coming soon	NA		TSSOP-20	32 kBytes	6 kBytes
☆	STM32C011F6U3		STM32C011F6Ux	Coming soon	NA		UFQFPN...	32 kBytes	6 kBytes
☆	STM32C011F6U6TR		STM32C011F6Ux	Coming soon	NA	STM32C...	UFQFPN...	32 kBytes	6 kBytes
☆	STM32C011J4M3	STM32C011J4	STM32C011J4Mx	Coming soon	NA		SO-8	16 kBytes	6 kBytes
☆	STM32C011J4M6		STM32C011J4Mx	Coming soon	NA		SO-8	16 kBytes	6 kBytes
☆	STM32C011J6M3	STM32C011J6	STM32C011J6Mx	Coming soon	NA		SO-8	32 kBytes	6 kBytes
☆	STM32C011J6M6		STM32C011J6Mx	Coming s					
☆	STM32C031C4T3	STM32C031C4	STM32C031C4Tx	Coming s					
☆	STM32C031C4T6		STM32C031C4Tx	Coming s					
☆	STM32C031C4U3		STM32C031C4Ux	Coming s					
☆	STM32C031C4U6		STM32C031C4Ux	Coming s					
☆	STM32C031C6T3	STM32C031C6	STM32C031C6Tx	Coming s					
☆	STM32C031C6T6		STM32C031C6Tx	Coming s					
☆	STM32C031C6U3		STM32C031C6Ux	Coming s					
☆	STM32C031C6U6		STM32C031C6Ux	Coming s					
☆	STM32C031F4P3	STM32C031F4	STM32C031F4Px	Coming s					
☆	STM32C031F4P6		STM32C031F4Px	Coming s					

So, this is the complete list we can select the dedicated MCU that is mounted onto the Nucleo board, but we can squeeze down the list by using the several options available here.

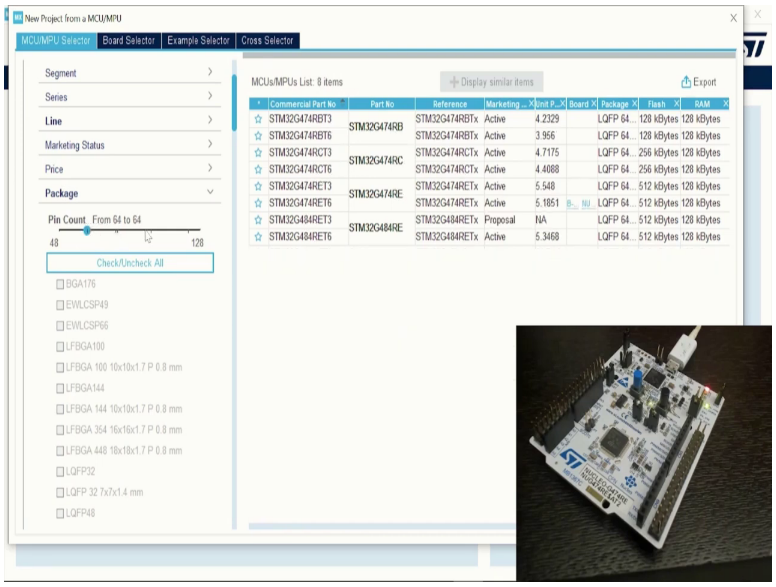
(Refer Slide Time: 30:17)



MCUs/MPUs List: 63 items

*	Commercial Part No	Part No	Reference	Marketing ...	Unit ...	Board	Package	Flash	RAM
☆	STM32G474CBT3	STM32G474CB	STM32G474CBTx	Active	3.9561		LQFP 48...	128 kBytes	128 kBytes
☆	STM32G474CBT3TR		STM32G474CBTx	Active	3.9561		LQFP 48...	128 kBytes	128 kBytes
☆	STM32G474CBT6TR		STM32G474CBTx	Active	3.6973		LQFP 48...	128 kBytes	128 kBytes
☆	STM32G474CBU6		STM32G474CBUx	Active	3.6003		UFQFPN...	128 kBytes	128 kBytes
☆	STM32G474CCT6	STM32G474CC	STM32G474CCTx	Active	4.1501		LQFP 48...	256 kBytes	128 kBytes
☆	STM32G474CCU3		STM32G474CCUx	Active	4.3368		UFQFPN...	256 kBytes	128 kBytes
☆	STM32G474CCU6		STM32G474CCUx	Active	4.0531		UFQFPN...	256 kBytes	128 kBytes
☆	STM32G474CET3	STM32G474CE	STM32G474CETx	Active	5.2712		LQFP 48...	512 kBytes	128 kBytes
☆	STM32G474CET3TR		STM32G474CETx	Active	5.2712		LQFP 48...	512 kBytes	128 kBytes
☆	STM32G474CET6		STM32G474CETx	Active	4.9263		LQFP 48...	512 kBytes	128 kBytes
☆	STM32G474CEU3		STM32G474CEUx	Proposal	NA		UFQFPN...	512 kBytes	128 kBytes
☆	STM32G474CEU6		STM32G474CEUx	Active	4.8293		UFQFPN...	512 kBytes	128 kBytes
☆	STM32G474CEU6TR		STM32G474CEUx	Active	4.8293		UFQFPN...	512 kBytes	128 kBytes
☆	STM32G474MBT3	STM32G474MB	STM32G474MBTx	Active					
☆	STM32G474MBT3TR		STM32G474MBTx	Active					
☆	STM32G474MBT6		STM32G474MBTx	Active					
☆	STM32G474MBT6TR		STM32G474MBTx	Active					
☆	STM32G474MCT3	STM32G474MC	STM32G474MCTx	Active					
☆	STM32G474MCT6		STM32G474MCTx	Active					
☆	STM32G474MET3	STM32G474ME	STM32G474METx	Active					
☆	STM32G474MET6		STM32G474METx	Active					
☆	STM32G474MEY3TR		STM32G474MEYx	Active					
☆	STM32G474MEY6TR		STM32G474MEYx	Active					
☆	STM32G474PBI6	STM32G474PB	STM32G474PBIx	Active					

(Refer Slide Time: 30:22)



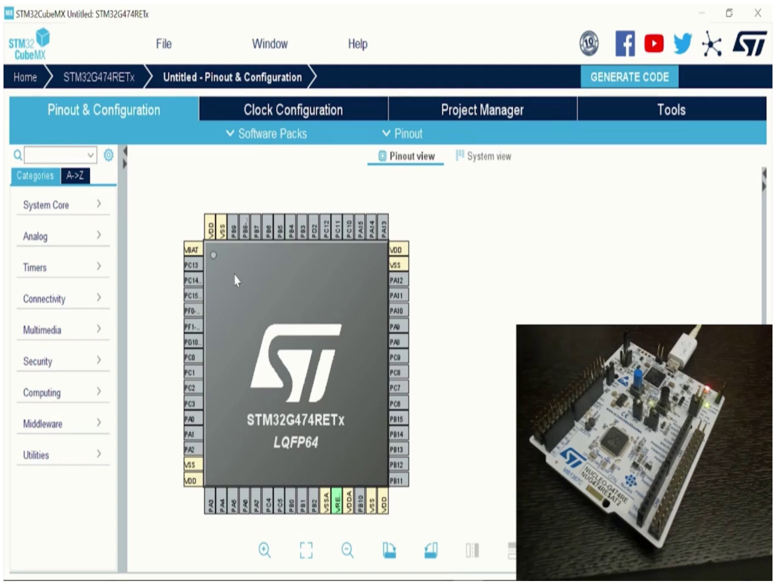
MCUs/MPUs List: 8 items

*	Commercial Part No	Part No	Reference	Marketing ...	Unit P...	Board	Package	Flash	RAM
☆	STM32G474RBT3	STM32G474RB	STM32G474RBTx	Active	4.2329		LQFP 64...	128 kBytes	128 kBytes
☆	STM32G474RBT6		STM32G474RBTx	Active	3.956		LQFP 64...	128 kBytes	128 kBytes
☆	STM32G474RCT3	STM32G474RC	STM32G474RCTx	Active	4.7175		LQFP 64...	256 kBytes	128 kBytes
☆	STM32G474RCT6		STM32G474RCTx	Active	4.4088		LQFP 64...	256 kBytes	128 kBytes
☆	STM32G474RET3	STM32G474RE	STM32G474RETx	Active	5.548		LQFP 64...	512 kBytes	128 kBytes
☆	STM32G474RET6		STM32G474RETx	Active	5.1851	B-... NU...	LQFP 64...	512 kBytes	128 kBytes
☆	STM32G484RET3	STM32G484RE	STM32G484RETx	Proposal	NA		LQFP 64...	512 kBytes	128 kBytes
☆	STM32G484RET6		STM32G484RETx	Active	5.3468		LQFP 64...	512 kBytes	128 kBytes

So, I can choose first. The line I can choose is STM32G474 and to squeeze down further. Let me choose the package. It’s a 64-pin package. So, we can use various options to squeeze down the list.

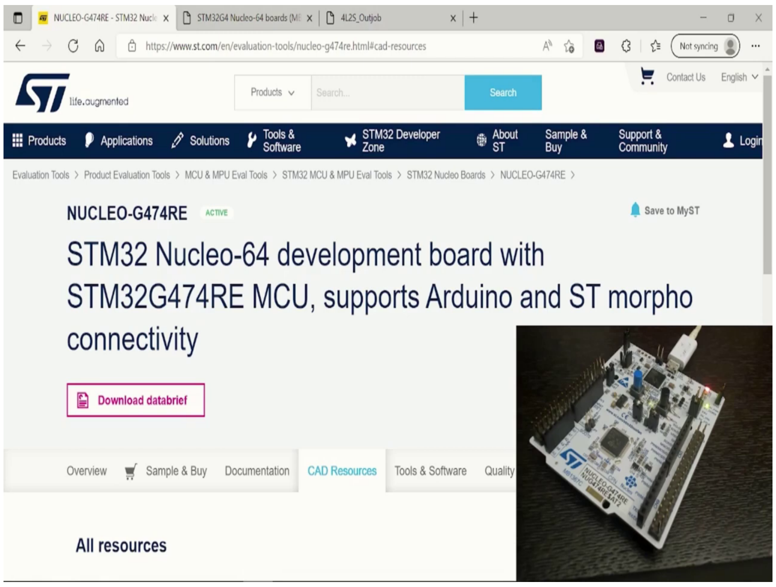
So, I am directly selecting the microcontroller which is mounted onto my Nucleo board. That is a STM32G474RET6. So, now, in the meantime, the file is generated. The file generated will be now completely blank with no peripherals initialized. So, we have to customize or do the configuration as per our application requirements. So, we wait for a couple of seconds till the blank file of this microcontroller generates. We are almost there and the file is here.

(Refer Slide Time: 31:29)

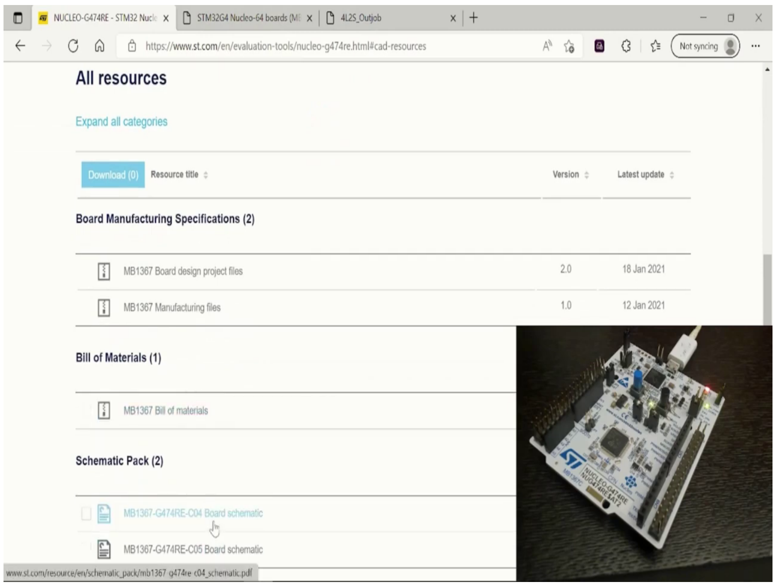


So, now you can see that in the microcontroller all pins are blank. Only the VCC and the ground pins have highlighted the Yellow. And we have to configure every pin as per our requirements. So, now, we are targeting an example of the GPIO or LED toggle.

(Refer Slide Time: 31:57)


https://www.st.com/en/evaluation-tools/nucleo-g474re.html#cad-resources
life.augmented
Products
Search
Contact Us
English
Products
Applications
Solutions
Tools & Software
STM32 Developer Zone
About ST
Sample & Buy
Support & Community
Evaluation Tools > Product Evaluation Tools > MCU & MPU Eval Tools > STM32 MCU & MPU Eval Tools > STM32 Nucleo Boards > NUCLEO-G474RE
NUCLEO-G474RE
ACTIVE
Save to MyST
STM32 Nucleo-64 development board with STM32G474RE MCU, supports Arduino and ST morpho connectivity
Download databrief
Overview
Sample & Buy
Documentation
CAD Resources
Tools & Software
Quality
All resources

(Refer Slide Time: 32:07)

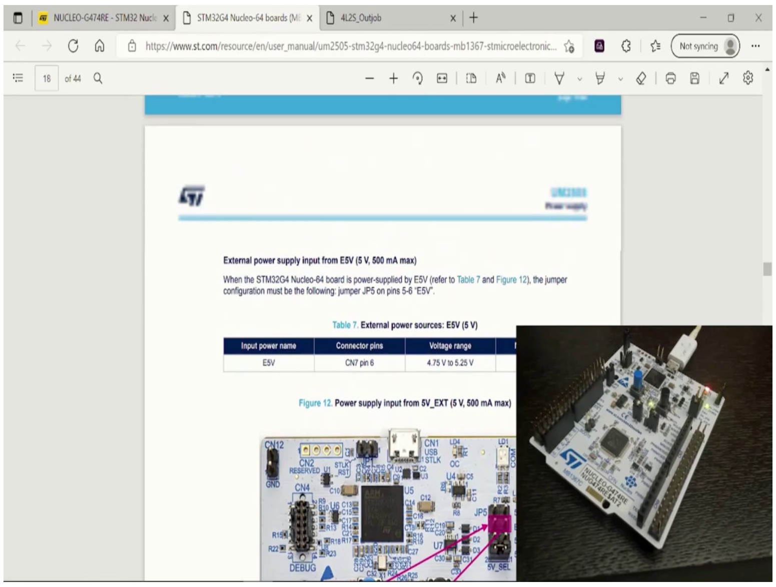


All resources

Expand all categories

Download (0) Resource title	Version	Latest update
Board Manufacturing Specifications (2)		
MB1367 Board design project files	2.0	18 Jan 2021
MB1367 Manufacturing files	1.0	12 Jan 2021
Bill of Materials (1)		
MB1367 Bill of materials		
Schematic Pack (2)		
MB1367-G474RE-C04 Board schematic		
MB1367-G474RE-C05 Board schematic		

(Refer Slide Time: 32:09)



External power supply input from E5V (5 V, 500 mA max)

When the STM32G4 Nucleo-64 board is power-supplied by E5V (refer to Table 7 and Figure 12), the jumper configuration must be the following: jumper JP5 on pins 5-6 "E5V".

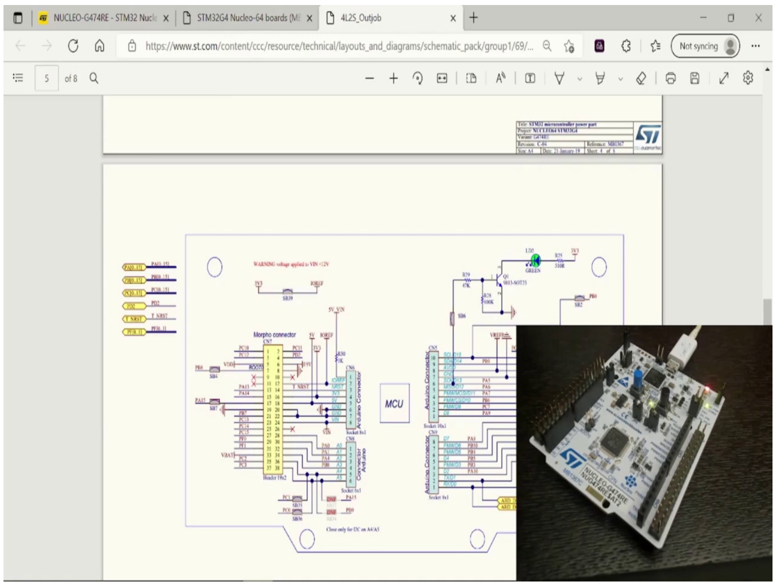
Table 7. External power sources: E5V (5 V)

Input power name	Connector pins	Voltage range
E5V	CN7 pin 6	4.75 V to 5.25 V

Figure 12. Power supply input from 5V_EXT (5 V, 500 mA max)

So, now we have to switch to the schematics. So, we can go on to the NucleoG47RET6 page, and there we can have the documentation, the cad resources, and the schematics in terms of PDF. In the documentation, we can have several jumpers and all those things in tails.

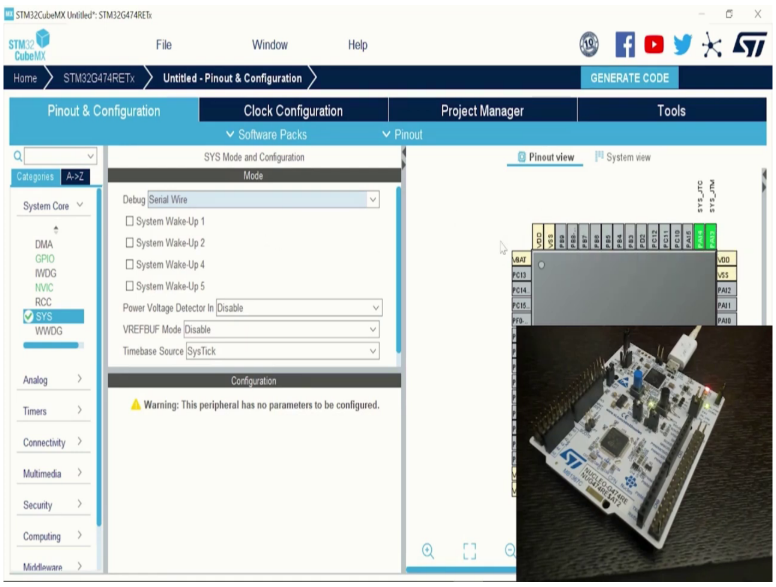
(Refer Slide Time: 32:15)



While in the schematic if you can see this LED2 which is now blinking with a 100-millisecond delay is connected to a PF5-Pin of the microcontroller.

So, we need to configure the PF5-Pin of the microcontroller. So, we need to configure it as GPIO output. So, we need to make it high and low to make leads on and off.

(Refer Slide Time: 32:43)



SYS Mode and Configuration

Mode

Debug Serial Wire

☐ System Wake-Up 1

☐ System Wake-Up 2

☐ System Wake-Up 4

☐ System Wake-Up 5

Power Voltage Detector In Disable

VREFBUF Mode Disable

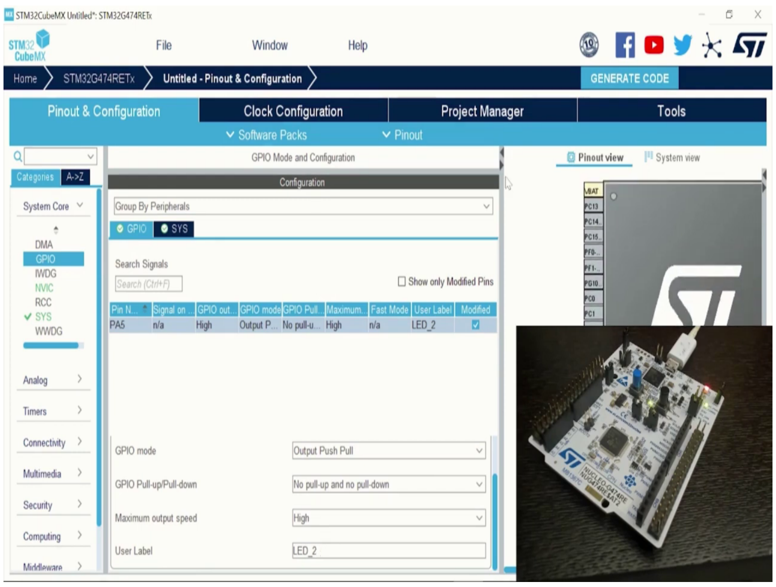
Timebase Source SysTick

Configuration

Warning: This peripheral has no parameters to be configured.

Also on the left side, there are several parameters. Firstly, I want to configure my debugging interface. The serial wire interface. So, you could see that as I have selected these 2 pins are already configured.

(Refer Slide Time: 33:00)



GPIO Mode and Configuration

Configuration

Group By Peripherals

GPIO | SYS

Search Signals

Search (Ctrl+F)

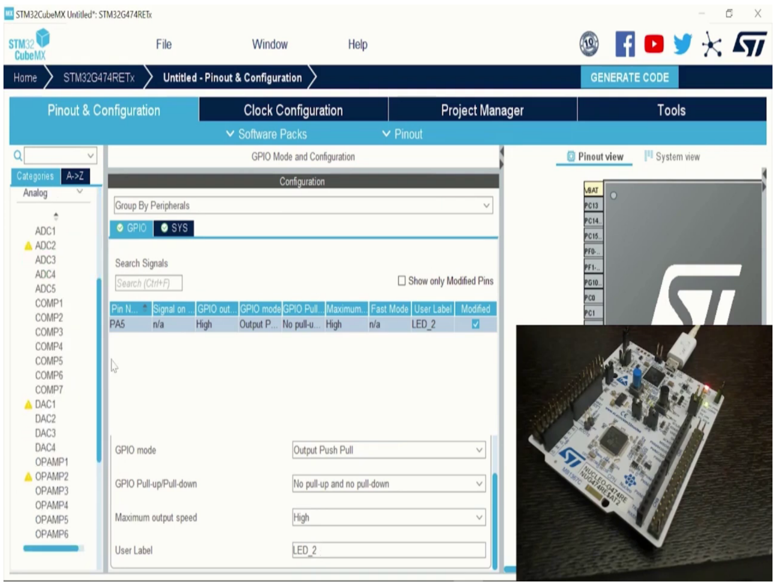
Show only Modified Pins

Pin N...	Signal on ...	GPIO out...	GPIO mode	GPIO Pull...	Maximum...	Fast Mode	User Label	Modified
PA5	n/a	High	Output P...	No pull-u...	High	n/a	LED_2	✓

GPIO mode	Output Push Pull
GPIO Pull-up/Pull-down	No pull-up and no pull-down
Maximum output speed	High
User Label	LED_2

And now for the GPIO, this PF5-Pin so, the output level firstly, let me keep it high output push-pull, yes I want to push-pull. So, speed. So, I can select the high speed. So, as per our requirement, we can select it in the user label. So, let me put LED2 underscore 2. So, just for our or basically as per the user requirement we can put the label here, and correspondingly this label will be updated here.

(Refer Slide Time: 33:43)



Group By Peripherals

GPIO | SYS

Search Signals

Search (Ctrl+F)

☐ Show only Modified Pins

Pin N...	Signal on ...	GPIO out...	GPIO mode	GPIO Pull...	Maximum...	Fast Mode	User Label	Modified
PA5	n/a	High	Output P...	No pull-u...	High	n/a	LED_2	✓

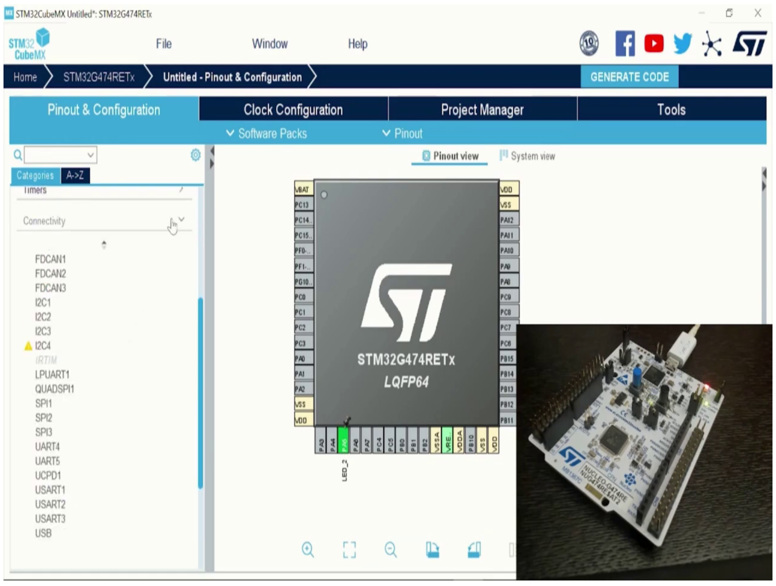
GPIO mode: Output Push Pull

GPIO Pull-up/Pull-down: No pull-up and no pull-down

Maximum output speed: High

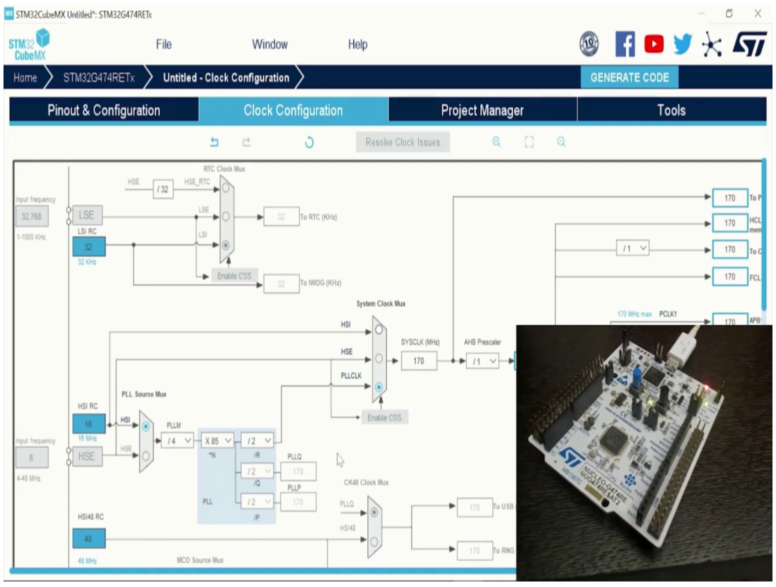
User Label: LED_2

(Refer Slide Time: 33:54)



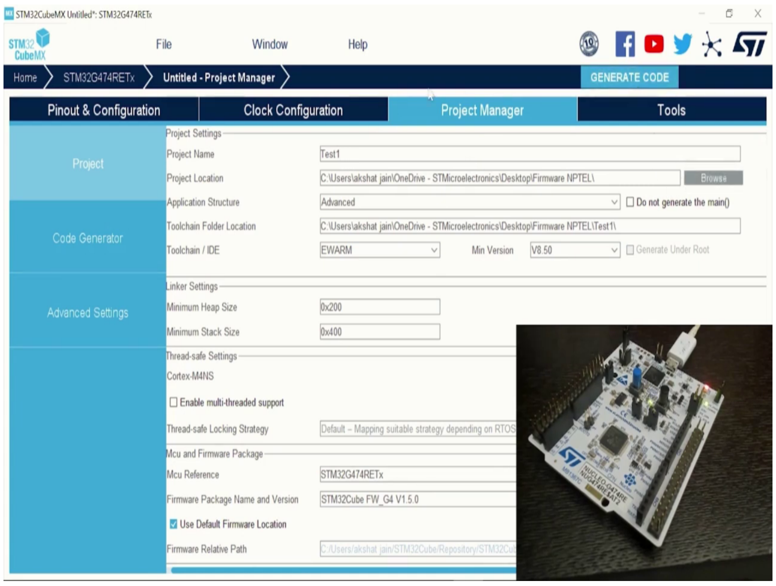
Apart from that there are ADCs, Tags, and Opamps comparator that is present in the analog section. Also, the timers will have several timers in the connectivity we have SPI, I2C, UART, and USB. While there are other options middleware and others that maybe it will be useful for another project, as of now we need to focus on the GPIO output. So, I will be covering a few of these topics in the next lecture. So, now, we go on to the Clock Configuration.

(Refer Slide Time: 34:16)



So, as I mentioned just in the second way that the Nucleo board is having external oscillators, and external crystals, but as of now I would not use the high-speed internal and the maximum speed that the controller can work is 170 megahertz. So, we can use the PPLL loop and make it faster. So, I already know the values, but we can play with it to have whatever frequency we want to make the microcontroller work. We can select that.

(Refer Slide Time: 34:56)

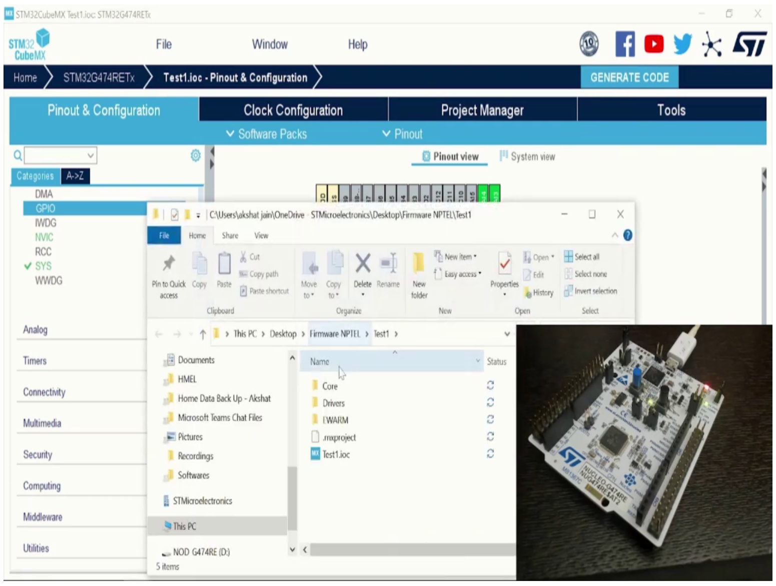


Project Settings
Project Name: Test1
Project Location: C:\Users\akshat jain\OneDrive - STMicroelectronics\Desktop\Firmware NPTEL\
Application Structure: Advanced ☐ Do not generate the main()
Toolchain Folder Location: C:\Users\akshat jain\OneDrive - STMicroelectronics\Desktop\Firmware NPTEL\Test1\
Toolchain / IDE: EWARM Min Version: V8.50 ☐ Generate Under Root
Linker Settings
Minimum Heap Size: 0x200
Minimum Stack Size: 0x400
Thread-safe Settings
Cortex-M4NS
☐ Enable multi-threaded support
Thread-safe Locking Strategy: Default – Mapping suitable strategy depending on RTOS
Mcu and Firmware Package
Mcu Reference: STM32G474RETx
Firmware Package Name and Version: STM32Cube FW_G4 V1.5.0
☑ Use Default Firmware Location
Firmware Relative Path: C:/Users/akshat jain/STM32Cube/Repository/STM32Cu

In the project manager for sure we can set the names you can set the location the main thing that we have already seen that I want to highlight here also.

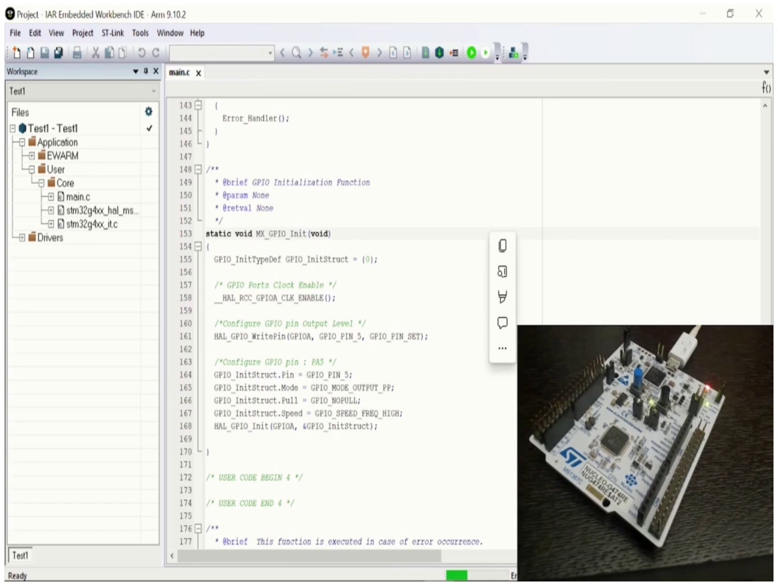
So, we can use the latest version if this option is stuck it is bound to give an error that the latest library is not downloaded, but if you uncheck this and we can select the earlier version that is installed on our PC we can select that. So, right now we can give the name Test1 for this and again I go to the pin configuration here and save the project. So, now, if you could see sorry in the Firmware NPTEL this Test1 folder is created.

(Refer Slide Time: 36:00)



And now if I generate the code. So, correspondingly you will be seeing that various files are being generated in the Test1 folder, just bear with a couple of seconds till ok. So, it’s done. So, I open the folder instead of directly opening the project. We can open the project, but I would go to the open folder to show you the various files. We can go to the EWARM again copy the path and we can open it in the IAR.

(Refer Slide Time: 36:39)

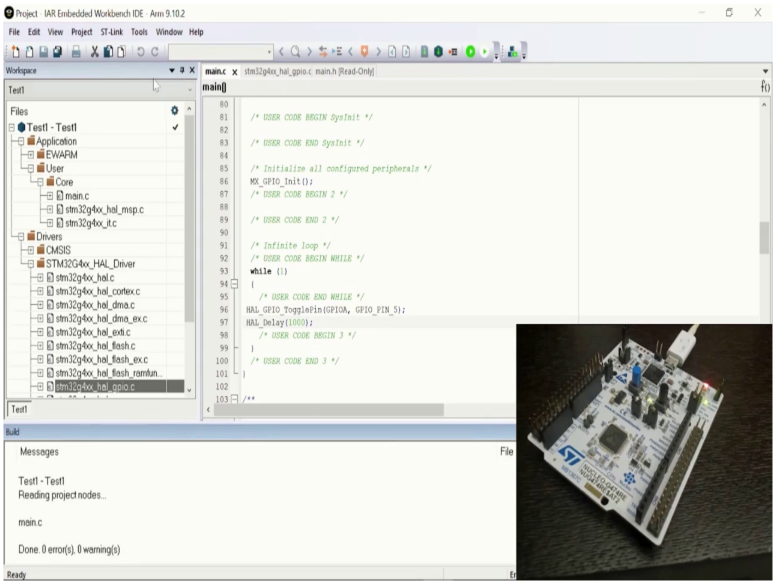


```c
  {
    Error_Handler();
  }
}

/**
  * @brief GPIO Initialization Function
  * @param None
  * @retval None
  */
static void MX_GPIO_Init(void)
{
  GPIO_InitTypeDef GPIO_InitStruct = {0};

  /* GPIO Ports Clock Enable */
  __HAL_RCC_GPIOA_CLK_ENABLE();

  /*Configure GPIO pin Output Level */
  HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, GPIO_PIN_SET);

  /*Configure GPIO pin : PA5 */
  GPIO_InitStruct.Pin = GPIO_PIN_5;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_HIGH;
  HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);

}

/* USER CODE BEGIN 4 */

/* USER CODE END 4 */

/**
  * @brief  This function is executed in case of error occurrence.
```

So, this is the project that we have created in which we have not only used the pre-initialized peripherals we have configured that by themselves. And right now we have to do the application layer also as we have done in a second way. So, again I go to the main dot c file. Where I could see that the HALL, reinitialize, system clock, and GPIO when it is there while again in the while 1. I have to write my application here

So, again I go to the drivers HAL drivers, where I go on to the GPIO dot c and the functions here I want to use the toggle pin example. So, I copied and paste it here. Again I can go to the header dot c file maybe not here. So, I go to the GPIO in order of C. So, it’s GPIO-Pin5 and GPA. So, right now there are no hash definitions I am searching for the main dot H5 and it is GPIO a function that we want to use is HAL.

(Refer Slide Time: 38:57)

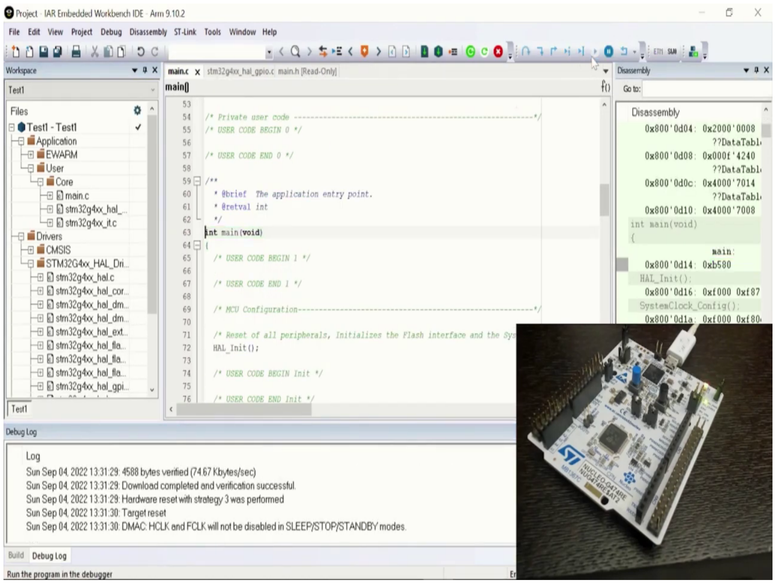


```
  /* USER CODE BEGIN SysInit */

  /* USER CODE END SysInit */

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  /* USER CODE BEGIN 2 */

  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
    /* USER CODE END WHILE */
 HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_5);
 HAL_Delay(1000);
    /* USER CODE BEGIN 3 */
  }
  /* USER CODE END 3 */
}

/**
```

```
Messages
Test1 - Test1
Reading project nodes...
main.c
Done. 0 error(s), 0 warning(s)
```

Let me just compile it ok. So, there are no errors. I rebuild it all again. So, right now if you see at the Nucleo board. The LED is flashing with the 100 millisecond delay and right now I have given a 1-second delay.

(Refer Slide Time: 39:24)



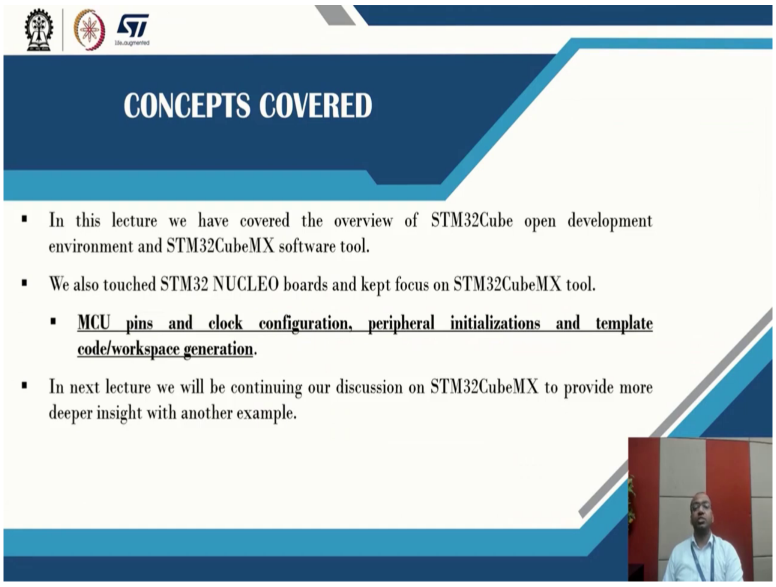
```
/* Private user code ---------------------------------------------------------*/
/* USER CODE BEGIN 0 */

/* USER CODE END 0 */

/**
  * @brief  The application entry point.
  * @retval int
  */
int main(void)
{
  /* USER CODE BEGIN 1 */

  /* USER CODE END 1 */

  /* MCU Configuration--------------------------------------------------------*/

  /* Reset of all peripherals, Initializes the Flash interface and the Sys
  HAL_Init();

  /* USER CODE BEGIN Init */

  /* USER CODE END Init */
```

```
Disassembly
    0x800'0d04: 0x2000'0008
                 ??DataTabl
    0x800'0d08: 0x000f'4240
                 ??DataTabl
    0x800'0d0c: 0x4000'7014
                 ??DataTabl
    0x800'0d10: 0x4000'7008
int main(void)
{
                  main:
    0x800'0d14: 0xb580
  HAL_Init();
    0x800'0d16: 0xf000 0xf87
  SystemClock_Config();
    0x800'0d1a: 0xf000 0xf80
```

```
Log
Sun Sep 04, 2022 13:31:29: 4588 bytes verified (74.67 Kbytes/sec)
Sun Sep 04, 2022 13:31:29: Download completed and verification successful.
Sun Sep 04, 2022 13:31:29: Hardware reset with strategy 3 was performed
Sun Sep 04, 2022 13:31:30: Target reset
Sun Sep 04, 2022 13:31:30: DMAC: HCLK and FCLK will not be disabled in SLEEP/STOP/STANDBY modes.
```

So, let me program the Nucleo board with this. I will make the Firmware run. So, now, again if you can see the LED the point at which my finger is pointing is blinking with a 1-second delay. So, this is the third way of generating the code.

(Refer Slide Time: 39:44)



CONCEPTS COVERED

- In this lecture we have covered the overview of STM32Cube open development environment and STM32CubeMX software tool.
- We also touched STM32 NUCLEO boards and kept focus on STM32CubeMX tool.
  - **MCU pins and clock configuration, peripheral initializations and template code/workspace generation**.
- In next lecture we will be continuing our discussion on STM32CubeMX to provide more deeper insight with another example.

In this lecture, we have covered the overview of STM32Cube which is an open development environment in which we have highlighted that it contains several software tools and several software packages. Then we focused our discussion on STM32CubeMX where we highlighted the Nucleo boards which we have used in demonstrating and we used GPIO toggle examples through which we have done MCU pin configuration, clock configuration, and generated the C code.

And in the next lecture also we will be continuing our discussion on STM32CubeMX to provide more deeper insight. See you in the next lecture.