Contents

Introduction

This chapter sets the stage for working with embedded systems, guiding users through the essential processes of module flashing, OS installation, network setup, and device programming. It introduces the scope of the guide and outlines the tools and knowledge required to get started.

⭐GUI first look ...

Who is this for?

This guide is tailored for embedded developers, system integrators, and technical learners who need to configure, program, and maintain hardware modules and microprocessor units. It's ideal for those working with both Linux and Windows based environments.

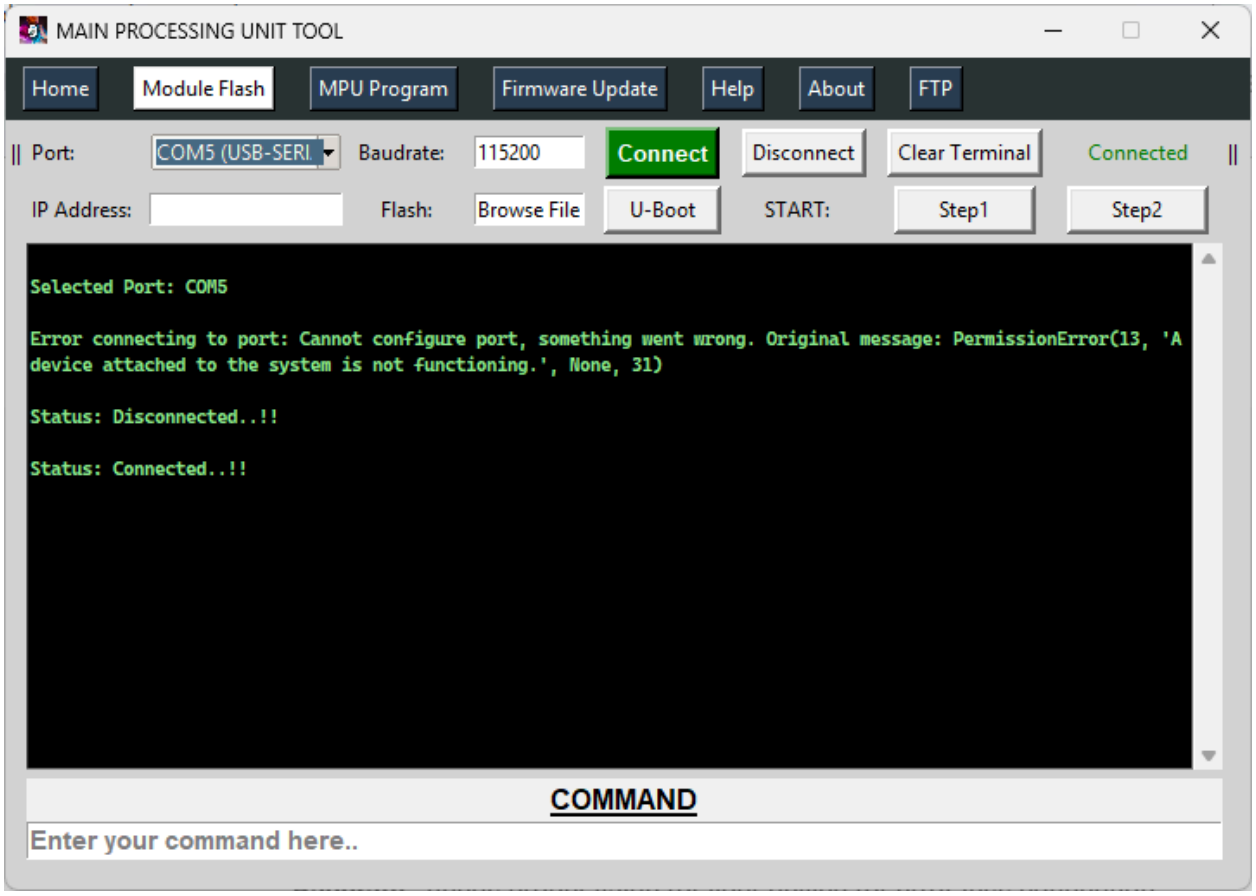
⚠️ **A user should have a basic knowledge of computers.**

Module Flash

Flashing a module is a foundational step in embedded system development. It involves writing firmware to a device to enable its core functionality and prepare it for further configuration and programming.

- Brief Intro
  Module flashing ensures that the hardware runs the correct software version. It's typically done via serial or USB interfaces using specialized tools that write firmware to the device's memory.

- GUI Overview
  The flashing GUI provides a user-friendly interface to select easy installer's recovery files, configure ports, and monitor progress.



Port - gives you access to select the correct serial port.
Baudrate - needs proper value for your device for error-less connection.
Connect - establish serial communication between your devices.
Disconnect - breaks the communication.
Clear Terminal - clears all the logs from the dedicated terminal.
Status - displays connectivity status, i.e., connected/disconnected
IP Address - field asks for the correct ip address that you want to set to your devices.
Flash - asks you to choose easy installer's files for module recovery.
U-Boot - sends your device in recovery mode.
START | Step1 - Hard reset or Factory reset.
START | Step2 - Network configurations and making your device ready-to-use.
COMMAND - sends shell command to your device for launching, allowing special permission to your application running or ready to run on your device.

These things simplify the process and reduce the chance of errors during flashing.

- Connection
  Establishing a stable connection between the host system and the module is crucial. This includes selecting the correct communication protocol (e.g., UART, USB), setting baud rates, and verifying device detection.

- Understanding Logs
  Logs displayed in the terminal are also generated during flashing and offer insights into the process, helping users identify issues, confirm successful operations, and troubleshoot errors effectively.

- What is U-Boot?
  U-Boot is a widely used bootloader in embedded systems. It initializes hardware, loads the operating system, and provides a command-line interface for debugging and configuration.
  However, U-Boot, in our case, is sending the target device into recovery mode i.e., u-boot mode to flash the main processing unit.

- Getting started with Module Flashing

To begin flashing, users must prepare the Toradex Apalis EasyInstallers recovery files, connect the device, and use the flashing tool to write the firmware. This section walks through each step with practical tips and precautions.
Hands-on training : First, choose the usb port, then easy installer's directory and then click on U-Boot to factory reset your device. Thereafter, it makes your device ready for OS installation.
Keynotes: While flashing your module, you may face otg driver issues. In such case, please download and install otg driver on your system and try this step again. However, the OTG driver here refers to none other than Google ADB driver.

- Journey of OS Installation
  Installing an OS involves partitioning storage, copying system files, and configuring boot parameters. This journey transforms a bare module into a functional computing device.
  In our case, we'll follow a different approach to install OS in my Apalis iMX6Q module.
  Hands-on training: First, choose serial port, baud rate, and connect your device. Next, enter IP Address to set a static ip address to your device and then proceed with Step1 to install Toradex BSP (embedded linux) to your device. Within a few seconds, it'll configure your device network and launch a user interface, listing your available software. If you can't see any software listed, then please use an offline-usb-drive which should contain Toradex BSP inside it. Once you get available software listed in the installation widget, choose it and then start the os installation.
  It will take approx 2 minutes to install. Once, the installation is complete, close the widget and power off of your device.
  Again power on your device and connect it. Once it gets connected, you'll get boot logs in the terminal.

- Network configurations
  Network setup includes assigning IP addresses, configuring gateways, and enabling services like SSH. Proper configuration ensures reliable communication and remote access.
  Hands-on training: Once your device is powered on, from the command line, set network configuration to your device. Just type and execute the below command to set ip address to your device —
  Option1: *ifconfig eth0 192.168.1.404 netmask 255.255.255.0*

Option2: *ip addr new 192.168.1.404/24 dev eth0*
Option3: *config <ethernet-cable-address> - - ipv4 manual 192.168.1.404 255.255.255.0 192.168.1.1*
Executing any one of the above commands, will set a static ip address to your device which is important for the further operations.

- Making your device ready
  Once flashed and configured, the device must be tested for functionality. This includes verifying hardware components, checking connectivity, and ensuring the OS boots correctly. Clicking on **Step2**, will do all things automatically.

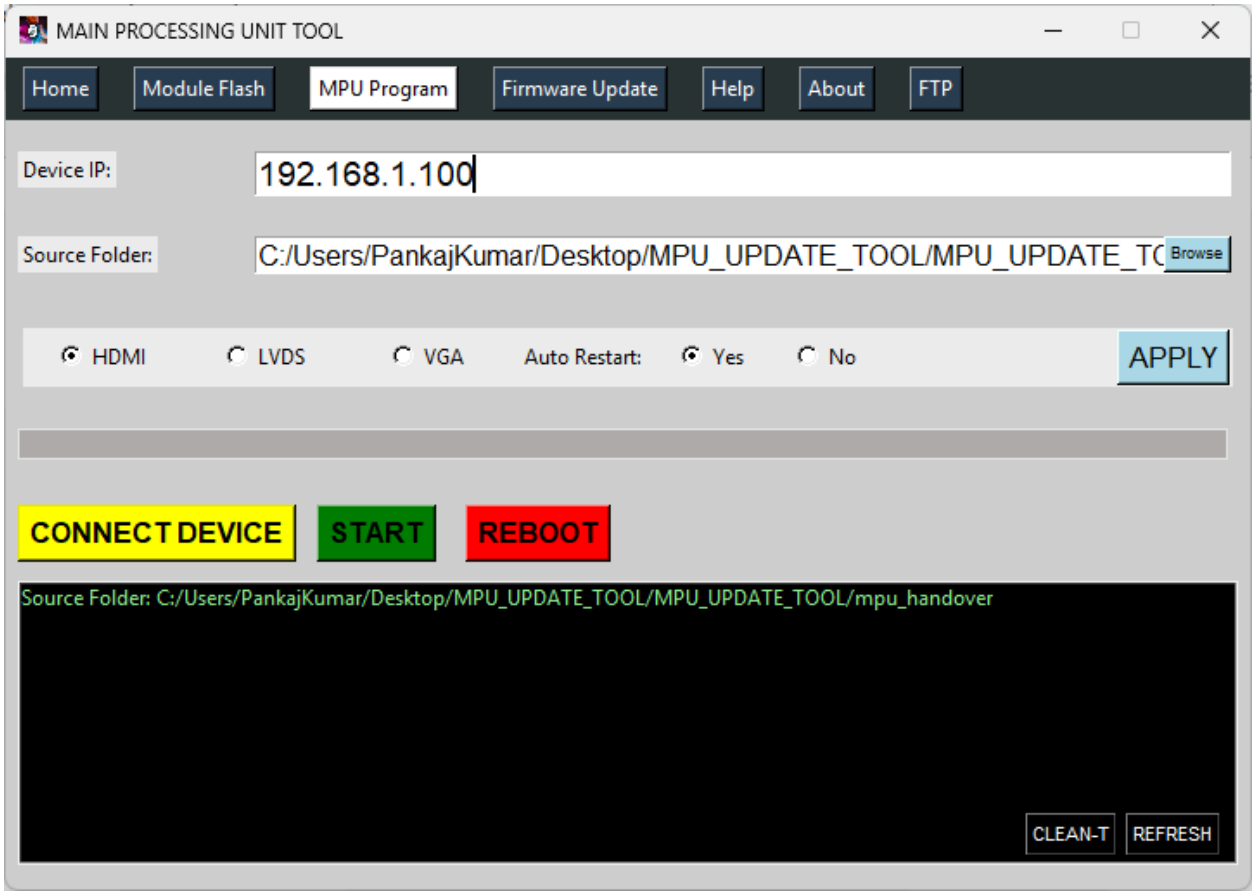
- Hands on practice with linux commands
  Linux command-line skills are essential for embedded development. Practicing commands like 'ls, cd, chmod', and 'dmesg' helps users manage files, monitor system status, and troubleshoot issues.

MPU Program

Programming the Microprocessor Unit (MPU) allows developers to control device behavior, interact with peripherals, and implement custom logic. This chapter covers the tools and techniques for MPU development.

- Brief Intro
  An MPU is a compact processor used in embedded systems. Programming it involves writing firmware that handles tasks like sensor input, data processing, and communication.

- GUI Overview | Programming
  The MPU programming GUI offers features like code editing, compiling, and uploading. It streamlines development and provides feedback during the programming process.



Device IP - requires the IP address of your device you want to program.
Source Folder - this is the main folder which contains all the necessary files, such as config, kernel, quectel, json, sounds, routes and many other libraries. Please make sure, you must have a source folder which should contain all the programming files.
Display | AutoRestart - This feature allows you to choose the desired display on which you want to stream, as well as autorestart selection can be done to automate the process.
Connect Device - This lets you check the device connectivity for further operations.
START - To proceed with this step, make sure everything is set to avoid programming errors. Clicking on START will start programming your device. It'll auto reboot your system/device multiple times to apply effect. Once programming is completed, you'll get a success message after a few minutes.
Programming may take a maximum of 3 to 5 minutes. So till then, you may have a cup of coffee.. 🍵😊

REBOOT - In case, you didn't select the autorestart option, you can manually restart your device to apply effect.

- How to Program?
  Programming involves writing code in languages like C or Rust, compiling it with a cross-compiler, and uploading it to the MPU. Debugging tools help refine and test the code.
  Here, we've simplified MPU programming to a one-click process. Just enter your device IP Address, Choose Source Folder, Select Display Type [HDMI, LVDS, VGA], and finally hit the Connect Device and Start button to automate the MPU Programming. Within a few minutes, it'll make your device ready. You can check the programming logs in the dedicated terminal. While programming, you'll be able to see the programming status in the progress bar.

- Keynotes
  Important considerations include understanding the MPU architecture, managing devices efficiently, handling interrupts, and ensuring real-time performance. The main key points are as following:
  → Check device connectivity before starting programming

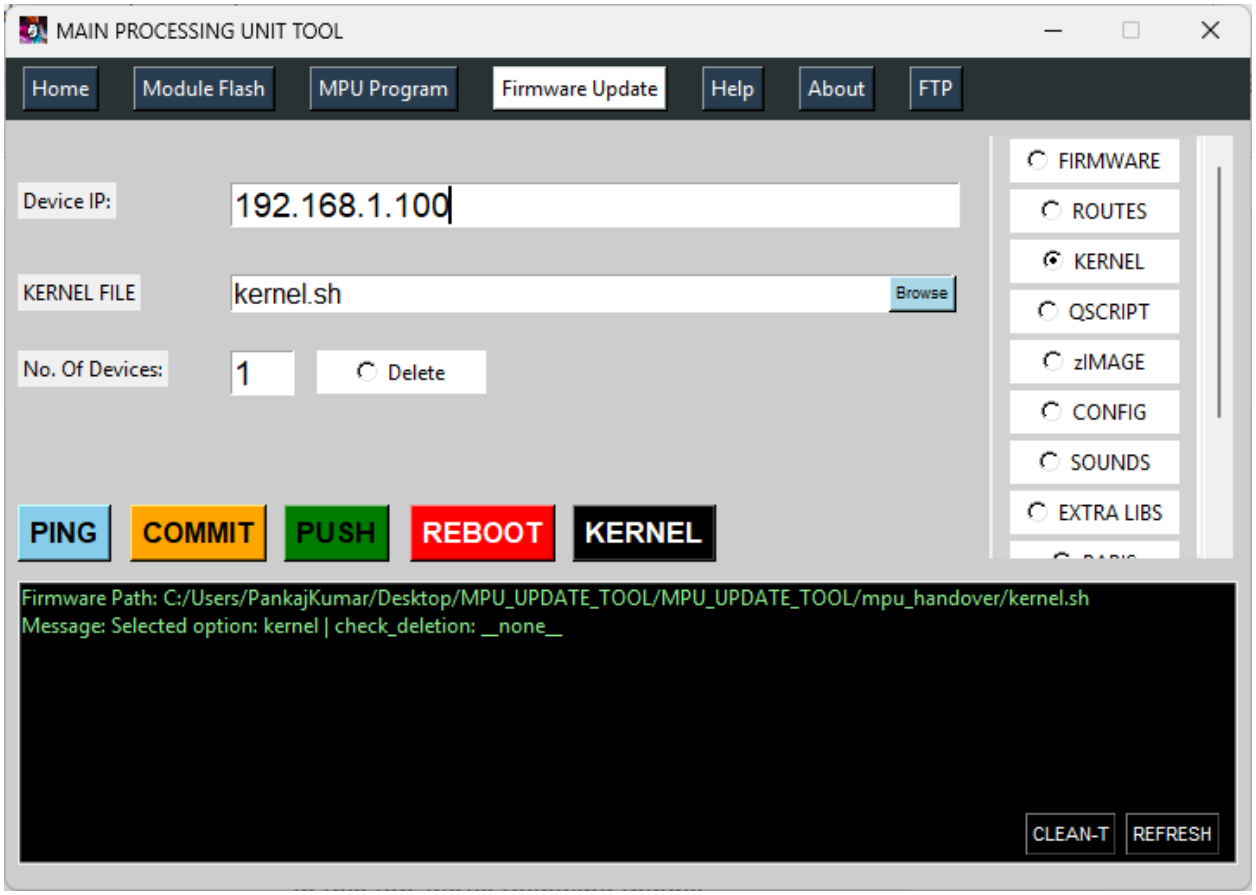
→ If facing issues in connectivity, please make sure you've configured your development machine's static ip address as per your device. Also you check the loose connections.
→ Choose Yes/No for Auto Restart for automation
→ Check the programming logs in the terminal and note down the errors if you get any. It'll help you to debug easily.
→ If everything remains fine, you'll have a successful message at the end of programming.

Firmware Update

Firmware updates are essential for maintaining device performance, security, and compatibility. This chapter explains how to safely update firmware and manage versions.

- Brief Intro
  Updating firmware involves replacing the existing software on a device with a newer version. It can fix bugs, add features, and improve stability.

- GUI Overview
  The firmware update GUI allows users to select update files, initiate the process, and monitor progress. It often includes rollback options and version checks.



In this tab, we've following things:
Device IP - It takes ip address of the target device
Firmware Path - This allows you to choose the firmware path. However, this field is interchanged based on the selected option from the right pane.
No. Of Devices - This is the quantity of your targeted devices. You can perform update operations to multiple devices.
Delete - If this radio button is checked, then it will do delete operation of the file/folders.
PING - This checks the device connectivity.
COMMIT - This makes everything ready to execute the final operations.
PUSH - It transfers the desired files/folders to the target device, gives execute permissions and updates your device as intended.
REBOOT - You can restart your device manually.
KERNEL - It executes the device kernel and refreshes everything to sync data and make it ready to use.
RightPanel - This panel has a list of options, such as Firmware, Routes, Kernel, QScript, zImage, Config, Sounds, etc. On selection of these things will update the related things on your device.
CLEAN-T/REFRESH - This clears all the logs and fields data entered.

- How to Use?
  To update your device (MPU), please follow the stepwise procedures —
  → First, enter your targeted device's ip address.
  → Then PING it, to check the connectivity. If it doesn't respond to a success message, please check the connection.
  → Next, from the right-pane, select the desired option that you want to update in MPU. For example, choose Firmware. However, by default Firmware remains selected.
  → Browse files, and visit your firmware location in your local system. Once found, select it. After selection, you'll be displayed the file path in the terminal.
  → Let the **No. Of Devices** be the same as it is (i.e., 1), and **Delete** unchecked.
  → Then, click COMMIT to get ready for the next operation.
  → Once you're sure that everything is set, then hit the PUSH button to start the update operation. You'll get the update message in the terminal. And after a few seconds or minutes, your device will get updated. Your device may restart automatically or you need to restart it to apply changes.

Hence, you can update your device.

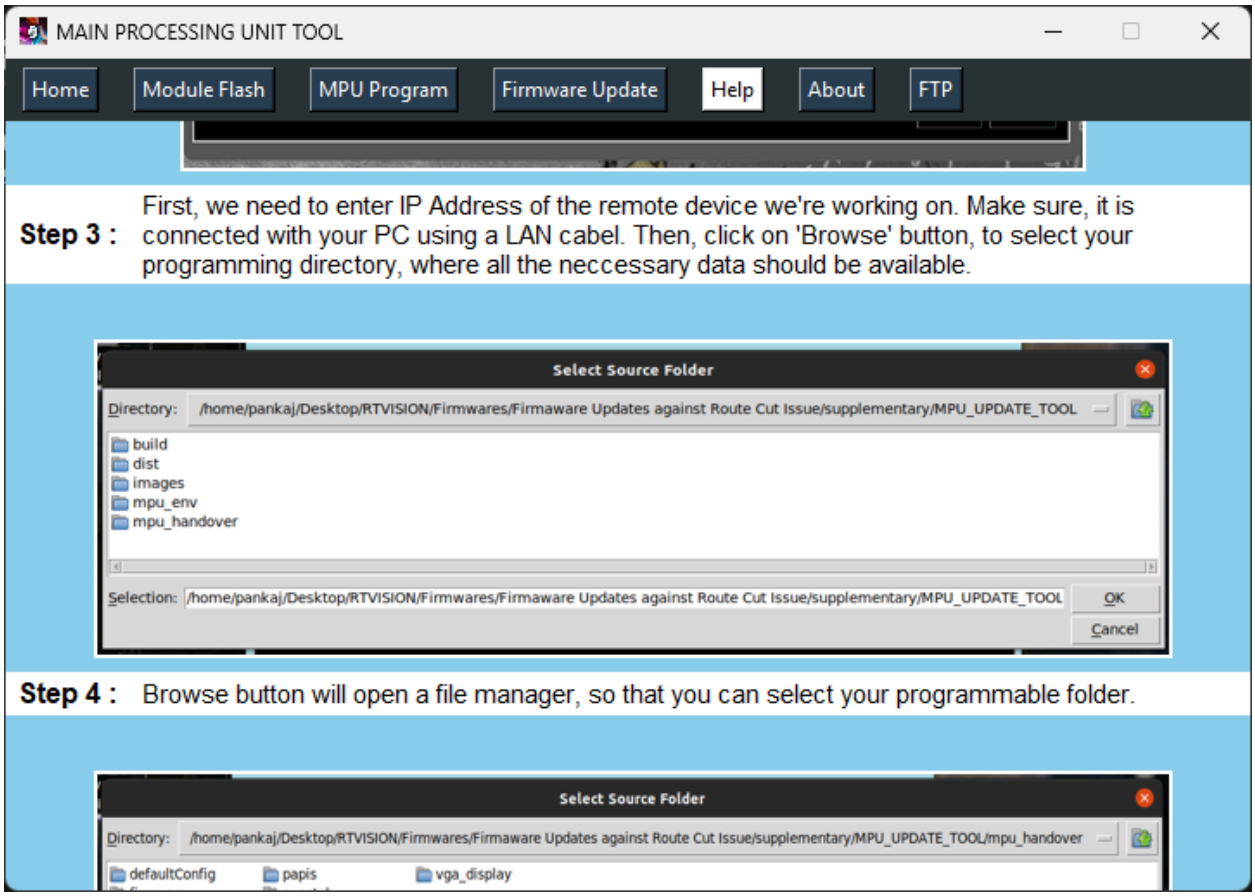
- Key features
  → You're required to use IP Address for data transfer between devices
  → You must have two different types of firmwares (in the case of debugging).
  → You must select the required options from the right pane before file/folder selection
  → You can increase the value of No. Of Devices if you're working with multiple devices.
  → Enable Delete operation only when you want to delete the related files/folders
  → Before PUSH, COMMIT operation should be executed first.
  → Both, REBOOT and KERNEL are two manual commands, and can be used as per your requirements.

- Terminal logs
  Terminal logs during updates provide real-time feedback, showing progress, errors, and system responses. They are vital for troubleshooting and validation.

Help

This chapter offers support resources, troubleshooting tips, and guidance for resolving common issues. It ensures users can find assistance when needed.

- User Guide
  The user guide includes detailed instructions, screenshots, and examples for each process. It helps users navigate the application confidently and efficiently.

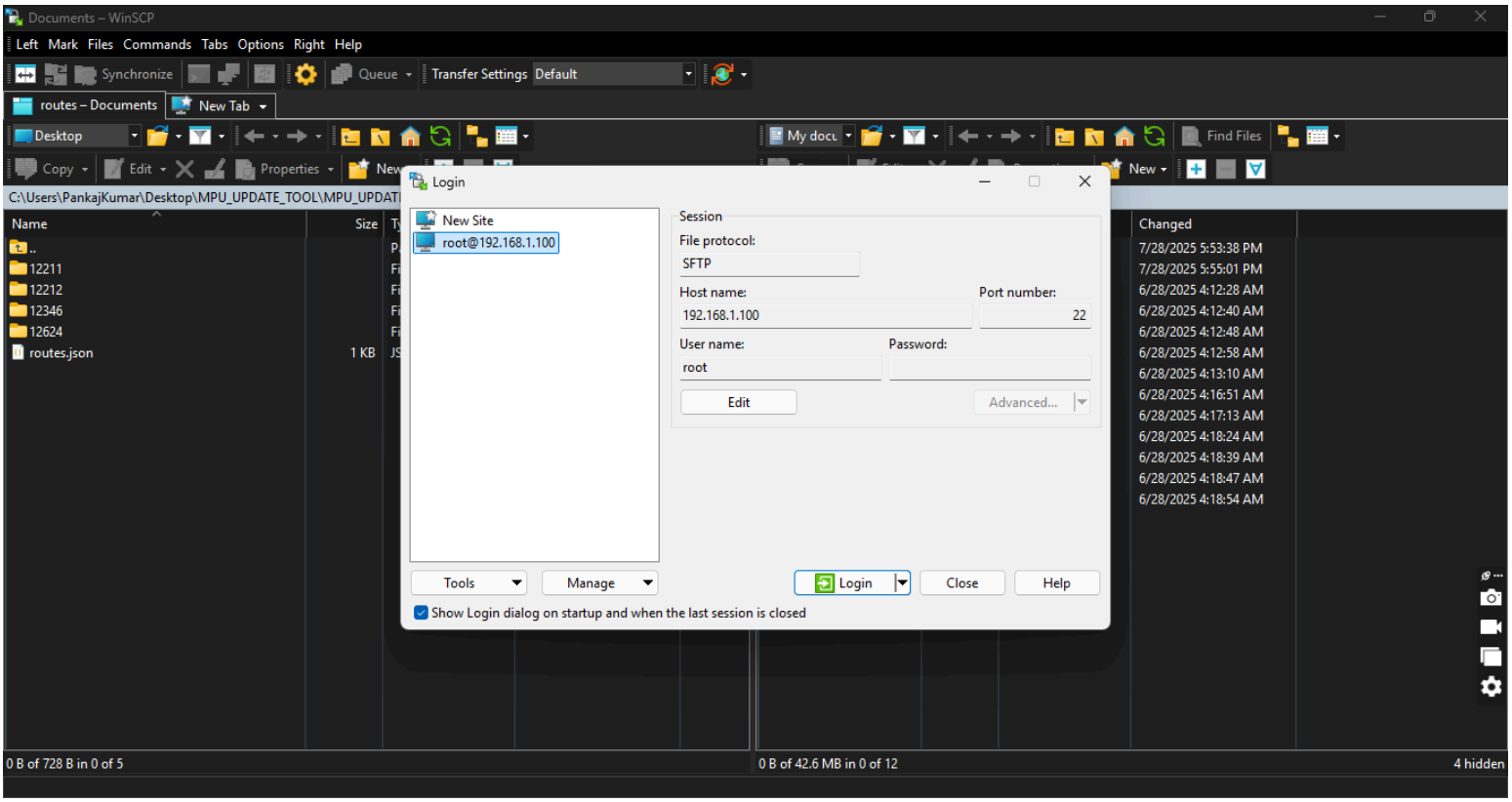


In this application's help tab, you'll find everything and will guide you how to use this application. You will get a copy of the same document.

FTP

FTP (File Transfer Protocol) is used to transfer files between devices over a network. This chapter explains how to use FTP for firmware uploads, log downloads, and remote file management.

- What is FTP?
  FTP is a standard protocol for exchanging files between a client and server. It supports authentication, directory navigation, and file operations over TCP/IP.

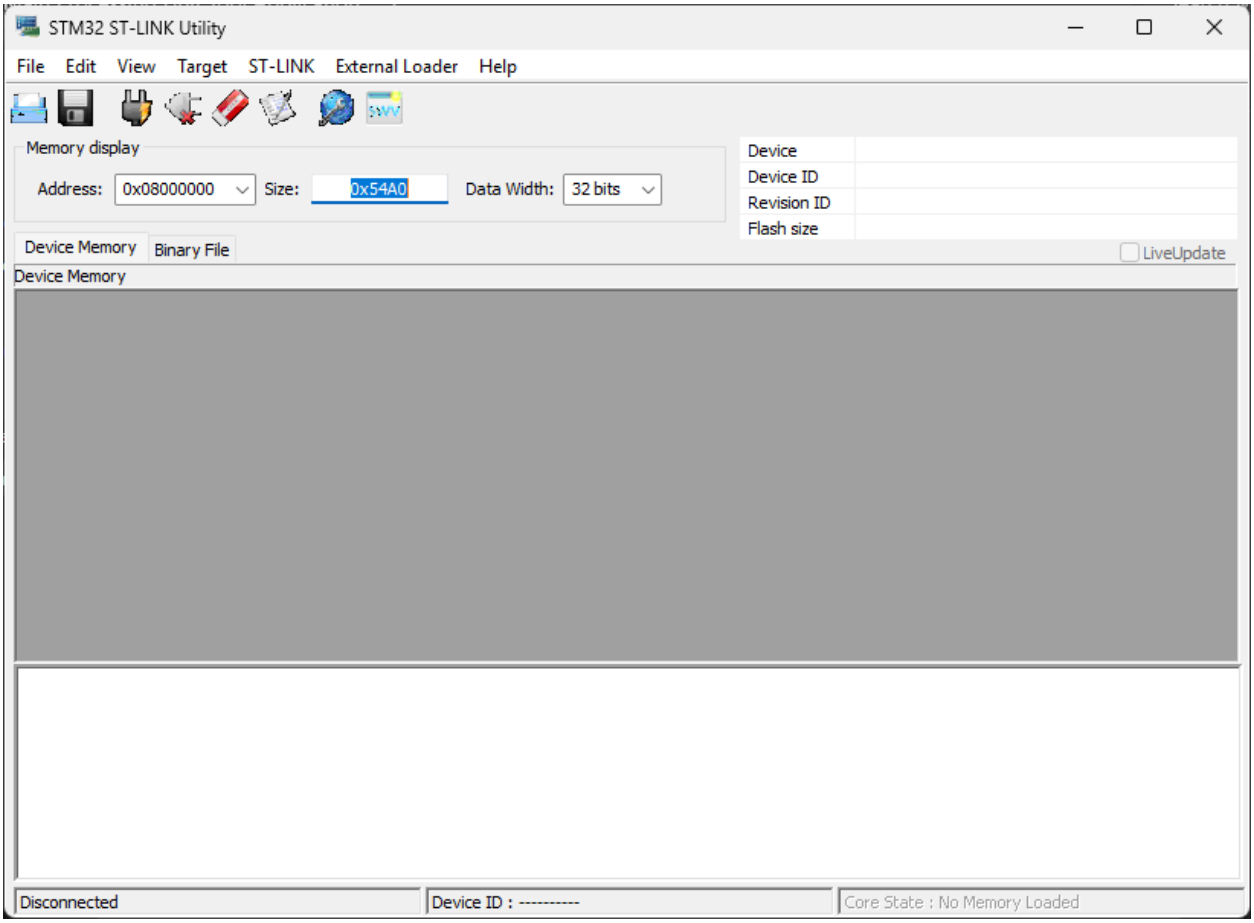


In our application, the FTP tab takes you to an external application named ‘WinSCP”, which is specially developed for data transmission over TCP/IP protocols. This application gives a cool interface for the both local system and remote device. You can just drag and drop from your local system to your remote device, and it will do the rest of your work.

STM32

STM32 is a family of microcontrollers developed by STMicroelectronics, based on the ARM Cortex-M processor architecture. These microcontrollers are widely used in embedded systems due to their high performance, low power consumption, and rich set of features. The STM32 family includes a variety of models that cater to different application needs, ranging from simple tasks to complex real-time applications.

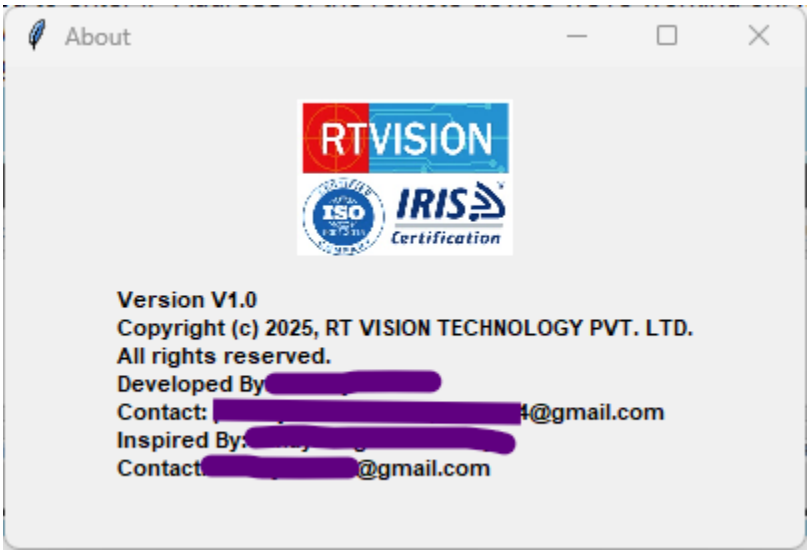
- This Tool is required to update the MPU Evaluation Board.



.

About

This guide was created to support embedded system developers in managing hardware modules and MPUs. It combines technical depth with practical examples for hands-on learning.



The about tab in this application gives you information about the app version, developers, application's owner, its copy rights and many things.

Summary

In summary, this guide covers module flashing, OS installation, network setup, MPU programming, and firmware updates. It equips users with the skills and tools needed for embedded development.

Glossary

The glossary provides quick access to topics and keywords covered in the guide. It helps users locate specific sections and navigate the content efficiently.

- Contents
- U-Boot
- Network configurations
- starting programming
- PUSH
- RightPanel
- FTP
- about
- basic knowledge
- EasyInstallers
- Connect Device
- COMMIT
- KERNEL
- User Guide
- WinSCP
- glossary