Eclipse/GCC/ST-LINK IDE
for
STMicroelectronics 32bit ARM
Teaser/Demo
Eclipse/GCC/ST-LINK/STM32Fx Teaser/Demos

• Teaser demos for experienced programmers
  • Eclipse/GCC/ST-LINK IDEs for STM32Fx
  • Model of Eclipse

• Demos showing some interesting TrueSTUDIO capabilities
  • Using STMicroelectronics STMCubeFn - libraries
  • Using STMicroelectronics STMCubeMX - plugin
  • Using Eclipse Target Management Terminal - plugin

• Demos will not show you how to
  • Install Atollic TrueSTUDIO or other Eclipse/GCC/ST-LINK systems
  • Install STMicroelectronics STMCubeFx - libraries
  • Install STMicroelectronics STMCubeMX – plugin
  • Install Eclipse Target Management Terminal – plugin

Like the text book says: “These exercises are beyond the scope of this book and therefore left as an exercise for the student.”
Eclipse/GCC/ST-LINK IDEs for STM32Fx

• Atollic TrueSTUDIO - http://timor.atollic.com/
• Roll Your Own
  • Setting up Eclipse and Code Sourcery lite for STM32 Discovery Development
  • Setting up a GCC/Eclipse toolchain for STM32Nucleo
• ***
Model of Eclipse

- **Workspace** – your project tub
  - **Purpose:** Eclipse root directory for Projects
  - TrueSTUDIO default: `C:\Users\<user_name>\Atollic\TrueSTUDIO\ARM_workspace_5.4`
  - Import - external projects (e.g. from STM32Cube_FW_FN_VX.Y.Z directory)
    - **Warning:** you’ll be modifying the original, so be sure to keep a copy of original
  - Export - to save projects

- **Perspectives** – DMS room (e.g. wood working, metal working, ...)
  - **Purpose:** Eclipse setup for specific development tasks
  - C/C++ (default)
  - Debug
  - STM32CubeMX
  - Git Repository

- **Views** – where you're looking (e.g. signal generator, oscilloscope, volt meter, ...)
  - **Purpose:** Eclipse panel/window for examining specific information
  - Project Explorer
  - Problems/Tasks/Console/Properties
  - Other
    - Terminal
    - ECalculator
Useful STMicroelectronics Documents

• STM32CubeMX
  • “STM32CubeMX for STM32 configuration and Initialization C code generation” User Manual – UM1718 – DM00104712

• STM32CubeF1
  • “STM32Cube firmware examples for STM32F1 Series” App Note - AN4724 - DM00209695
  • “Getting started with STM32CubeF1 firmware package for STM32F1 series” User Manual - UM1847 - DM00151047
  • “Description of STM32F1xx HAL drivers” User Manual - UM1850 - DM00154093

• STM32CubeF4
  • “STM32Cube firmware examples for STM32F4 Series” App Note AN4739 - DM00213525
  • “Getting started with STM32CubeF4 firmware package” User Manual - UM1730 - DM00107720
  • “Description of STM32F4xx HAL drivers” User Manual - UM1725 - DM00105879

• STM32 Nucleo Boards
  • “Getting started with STM32 Nucleo board software development tools” User Manual – UM1727 - DM00105928
True STUDIO/STM32Fx Teaser/Demo

- STMicroelectronics Software
  - STMCubeMX – standalone and eclipse plugin
  - STM32CubeFx – embedded software

- Atollic TrueSTUDIO IDE
  - Using STM32CubeF1/F4 embedded software
  - Using STMCubeMX plugin
  - Using Eclipse Target Management Terminal plugin

- Sample of STM32Fx Processors/Boards (STM32 MCU with Boot Loader, NVIC, DMA, and Bit Banding)
  - Small – 48pin: STM32F103C8T6 Minimum System Development Board (Aliexpress.com)
    - 72MHz, 64KB Flash, 20K RAM, 2 ADCs, 4 Timers, 2 I2Cs, 2 SPIs, 3 UARTs, FS USB
  - Medium – 64pin: Nucleo F411RE board
    - 100MHz, 512KB Flash, 128K RAM, 1 ADCs, 11 Timers, 3 I2Cs, 5 SPIs, 5 UARTs, FS USB, FPU, MPU
  - Large – 144pin: Nucleo 429ZI board
    - 180MHz, 2MB Flash, 256K RAM, 3 ADCs, 2 DACs, 17 Timers, 3 I2Cs, 6 SPIs, 8 UARTs, FS & HS UBSs, FPU, MPU

- Nucleo Demonstration Shield
  - Adafruit 1.8" Color TFT Shield w/microSD and Joystick
# STM32F103C8T6 Mini Dev Board/ST-Link

![STM32F103C8T6 Mini Dev Board/ST-Link](image)

<table>
<thead>
<tr>
<th>Pin</th>
<th>Nucleo CN4 SW Connector</th>
<th>Designation</th>
<th>Mini Dev Board</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>VDD_TARGET</td>
<td>VDD from application</td>
<td>3.3 V</td>
</tr>
<tr>
<td>2</td>
<td>SWCLK</td>
<td>SWD clock</td>
<td>DCLK</td>
</tr>
<tr>
<td>3</td>
<td>GND</td>
<td>Ground</td>
<td>GND</td>
</tr>
<tr>
<td>4</td>
<td>SWDIO</td>
<td>SWD data input/output</td>
<td>DIO</td>
</tr>
<tr>
<td>5</td>
<td>NRST</td>
<td>RESET of target MCU</td>
<td>Reset</td>
</tr>
<tr>
<td>6</td>
<td>SWO</td>
<td>Reserved</td>
<td>n/a</td>
</tr>
</tbody>
</table>

Note: A Nucleo board’s left JP1 pin can be used to supply 3.3V to a target MCU
STM32F103C8T6 Mini Dev Board Pinout
STM32F103C8T6 Mini Dev Board Schematic
STMicroelectronics Software
Download STCubeMX and STM32CubeFx Embedded Software


• Under “Get Software”
  • STMCubeMX (stand alone) – Part Number: STM32CubeMX > Download

• Under “Related Tools and Software”
  • STMCubeMX (eclipse plugin) – Part Number: STSW-STM32095 > Download
  • STM32CubeFn (embedded software) – Part Number: STMCubeFn > Download
Atollic TrueSTUDIO
Download/Install and Plugin Installation Instructions

• Download/Install Atollic TrueSTUDIO (need version: 5.4.1 or later):
  http://timor.atollic.com/truestudio/
  Select: “Download Now – No Registration Required”

• STM32 Cube MX – plugin installation instructions:
  http://timor.atollic.com/resources/application-notes/
  Select/Download “Integrate STM32CubeMx with Atollic TrueSTUDIO” App Note

• Eclipse Target Management Terminal – plugin download/install instructions:
STM32CubeFx Firmware Components

- **Board Demonstrations**
  - Evaluation boards
  - Discovery boards
  - Nucleo boards

- **Middleware level examples**
  - Networking
    - LwIP TCP/IP & Polar SSL
  - USB Host & Device
  - Graphics STemWin
  - File system FATFS
  - RTOS FreeRTOS

- **HAL level examples**
  - Hardware Abstraction Layer APIs
  - Boards Support Packages

- **Utilities**
- **CMSIS**

- **STM32 Series**
  - STM32F0
  - STM32F1
  - STM32F2
  - STM32F3
  - STM32F4
  - STM32F7
  - STM32L0
  - STM32L1
  - STM32L4
STM32Cube_FW_Fn_Vx.y.z Directory

- `_htmresc` – images
- **Documentation** - STM32CubeFnGettingStarted.pdf
- **Drivers**
  - BSP – Board Support Packages (e.g. Adafruit_Shield*, Components, STM32Fnxx_Nucleo)
  - CMSIS - Cortex™ Microcontroller Software Interface Standard
  - STM32Fnxx_HAL_Driver – Hardware Abstraction Layer (.h/.c libraries for MUC peripherals)
- **Middlewares**
  - ST – Graphics, Audio, USB Device/Host Libraries
  - ThirdParty – SD Fat Fs (file system), FreeRTOS, LibJPEG (jpeg), LwIP (TCP/IP), PolarSSL (security)
- **Projects**
  - STM32Fn-Discovery – Discovery board Applications, Demonstrations, Examples, Templates
  - ...
  - **STM32Fnxx-Nucleo – Nucleo board Applications, Demonstrations, Examples, Templates**
    - ...
    - STM32CubeProjectsList.html – Description of STM32Cube firmware
- **Utilities**
  - CPU – cpu utilities
  - Fonts – 8bit, 12bit, 16bit, 20bit, 24bit wide character pixel patterns
  - Log – LCD routines
  - Media – Audio, Pictures, Video material
  - PC_Software – SSL server, STM32Cube updater

*Nucleo board Demonstration applications use Adafruit’s 1.8” color tft shield w/microsd and joystick*
./STM32Fnxx-Nucleo Directory

- .metadata
- Applications
  - EEPROM > EEPROM_Emulation > vender specific project directories/files
- Demonstrations* > vender specific project directories/files
- Examples
  - GPIO > GPIO_IOToggle > vender specific project directories/files
  - HAL > HAL_TimeBase > vender specific project directories/files
  - PWR > PWR_CurrentConsumption > vender specific project directories/files
  - RCC > RCC_ClockConfig > vender specific project directories/files
  - RTC > RTC_Calendar > vender specific project directories/files
  - TIM > TIM_PWMInput > vender specific project directories/files
  - UART > UART_Printf > vender specific project directories/files
- Templates > vender specific project directories/files

*Nucleo board Demonstrations use Adafruit’s 1.8” color tft shield w/microsd and joystick
STM32Fnxx-Nucleo Vender Specific Project Files

• EWARM – environment for EWARM
• Inc – include/header (*.h) files for project
• MDK-ARM – environment for MDK-ARM
• Src – source (*.c) files for project
• SW4STM32 – environment for SW4STM32

➢ TrueSTUDIO – environment for TrueSTUDIO
  • STM32Fnxx-Nucleo
    • .settings (dir)
    • .cproject
    • .project
    • STM32Fnxxx_Nucleo.elf.launch
    • STM32Fnxxx_FLASH.ld
  • startup_stm32fnxx.s
Getting Started TrueSTUDIO/STM32F4 Examples
(using Nucleo-F411RE)

➢ Nucleo Blink LED – embedded computer equivalent to “Hello World”
   Check basic tool chain end-to-end test using Nucleo F411RE example (done)
➢ Nucleo UART printf() – useful for debugging embedded applications
   Check basic tool chain with serial output using Nucleo F411RE example (done)
➢ STM32CubeMX Blink LED – useful for developing new application
   Check enhanced tool chain end-to-end test using Nucleo F411RE board (done)
   • STM32CubeMX UART printf() – useful for debugging new application
     Check enhanced tool chain with serial output using Nucleo F411RE board (wip)
   • STM32CubeMX UART printf() and Decoder – useful for quadrature encoders
     Demonstrate using TIMer quadrature decoder using Nucleo F411RE board (tbd)
➢ Nucleo Demonstration Shield – example of STM32F411’s capabilities
   Using Nucleo F411RE with Adafruit 1.8" Color TFT Shield w/microSD and Joystick (binary)
Nucleo-F411RE Blink LED Example
(opening Nucleo example in TrueSTUDIO and compiling)

• Opening Nucleo Example in TrueSTUDIO
  • Open your STM32Cube_FW_F4_V1.11.0 directory > Projects >
    STM32F411RE-Nucleo > Examples > GPIO > GPIO IOToggle >
    TrueSTUDIO > STM32Fxx-Nucleo > .project
  • Double click .project to launch TrueSTUDIO’s Workspace Launcher and wait
  • Browse/Select desired workspace and click OK (e.g. default -
    C:\Users\<user_name>\Atollic\TrueSTUDIO\ARM_workspace_5.4)
  • Wait for Upgrade screen and select “Close Window”
  • Close “Information Center” window

• Compiling Example
  • In Project Explorer: select STM32F4xx-Nucleo
  • Select Rebuild Project
  • Select Problems View to check for compile errors (there shouldn’t be any)
  • Select Console View to check compile time, and code size/RAM usage
Nucleo-F411RE Blink LED Example - cont  
(downloading and debugging using STM32F411RE-Nucleo’s built-in ST-Link)

**Downloading/Debugging**

- In **Project Explorer**: select STM32F4xx-Nucleo
- Select **Debug** – this will switch to Debug Perspective
  
  May get error message “Launching STM32F103C8Tx_Board.elf has encountered problem”
  
  “Error in initializing ST-Link device.” “Reason: (-6) ST-Link firmware upgrade required.”...
  
  If so, click OK and for “ST-Link firmware upgrade required”, click Yes, and Yes again
- The Debugger will download the program and stop on line 75 at “HAL_Init();” breakpoint in “main.c”
- In “main.c” set a breakpoint on line 96 at “HAL_Delay(100);”
- Click Resume (F8), LD2 should light
- In **Breakpoint View**, uncheck “main.c[line 96]” breakpoint
  
  Click **Resume** (F8), LD2 should start blinking
- Click **Terminate** (Ctrl+F2) to stop and return to C/C++ Perspective

**Cleanup for next Example**

- In **Project Explorer**: select STM32F4xx-Nucleo
- Select **Delete** to delete this test case (don't select “Delete from file system)
Nucleo-F411RE UART printf() Example
(opening Nucleo example in TrueSTUDIO and change UART settings)

• Opening Nucleo Example in TrueSTUDIO
  • Open your STM32Cube_FW_F4_V1.11.0 directory > Projects >
    STM32F411RE-Nucleo > Examples > UART > UART_Printf >
    TrueSTUDIO > STM32Fxx-Nucleo > .project
  • Double click .project to launch TrueSTUDIO’s Workspace Launcher
  • If you’ve not closed TrueSTUDIO, this new STM32F4xx-Nucleo will appear

• Change UART settings
  • In Project Explorer: Select/Open STM32F4xx-Nucleo > Example > User > main.c
  • Line 98: change BaudRate to 115200
  • Line 99: change WordLength to UART_WORDLENGTH_9B
  • Line 101: change Parity to UART_PARITY_NONE
  • Update documentation (lines 91 to 94) to reflect these changes
Nucleo-F411RE UART printf() Example - cont

(compile and download using STM32F411RE-Nucleo’s built-in ST-Link)

• Compile Example
  • In Project Explorer: select STM32F4xx-Nucleo
  • Select Rebuild Project
  • Select Problems View to check for compile errors (there shouldn’t be any)
  • Select Console View to check compile time, and code size/RAM usage

• Download
  • In Project Explorer: select STM32F4xx-Nucleo
  • Select Debug – this will switch to Debug Perspective
    May get error message “Launching STM32F103C8Tx.Board.elf has encountered problem”
    “Error in initializing ST-Link device.” “Reason: (-6) ST-Link firmware upgrade required.”...
    If so, click OK and for “ST-Link firmware upgrade required”, click Yes, and Yes again
  • The Debugger will download the program and stop on line 76 at start of “int main()”
Nucleo-F411RE UART printf() Example - cont
(debugging using STM32F411RE-Nucleo’s built-in ST-Link)

• Debugging
  • Open Windows Device Manager
  • Open Ports (COM & LPT)
  • Select Properties under STMicroelectronics STLink Virtual COM Port (COMn)
  • Under Port Settings, select 115200 baud, 8 bits, Parity None, Flow control None
  • Open Show View: select Terminal > Terminal > OK
  • Open Terminal Settings: select Connection Type > Serial
  • Under Settings, select COMn, 115200 baud, 8 bits, Parity None, Flow control None
  • In the Debugger, select Resume (F8)
  • If you’re lucky, “UART Printf Example: ... “ will appear in Terminal View
  • If not lucky, terminate and restart debugger
    May also have to disconnect/reconnect terminal

• Cleanup for next Example
  • In Project Explorer: select STM32F4xx-Nucleo
  • Select Delete to delete this test case (don’t select “Delete from file system)
STM32CubeMX Nucleo-F411 Blink LED Example

(generate configuration and initialization code using STM32CubeMX)

• Using STM32CubeMX to generate configuration and initialization code
  • In TrueSTUDIO, **Open Perspective** > STM32CubeMX > OK
  • Select **New Project** > Board Selector > Nucleo64 > NUCLEO-F411RE > OK
  • Select **Project** > Settings > Project > Project Location (workspace), Project Name, Toolchain IDE > TrueSTUDIO, and **OK** (note Firmware Package Name and Version)
  
    May get error message “The Firmware Package (STM32Cube FW_F4 V1.11.0) or one of its dependencies required ...” “Do you want to download this now?” **Yes**
  
    Wait for it to download/unzip.
  
• Under **Pinout**, note Nucleo pinouts, including LD2 [Green Led] are already defined
• Under **Clock Configuration**, note Nucleo clock configuration is already defined
• Select **Project** > Generate Code and after generation **Open Project**
STM32CubeMX Nucleo-F411 Blink LED Example - cont
(add user Blink LED Example code)

- Open **C/C++ Perspective**
- In **Project Explorer** > STM32F411RE_Nucleo_Blink project
- View TMSTM32CubeMX generated definitions
  - Open **Includes** > <workspace>/<board name>/inc > mxconstants.h
    
    ```
    #define LD2_Pin GPIO_PIN_5
    #define LD2_GPIO_Port GPIOA
    ```
- View STM32Cube_FW_F4_V1.11.0 firmware definitions
  - Open **Includes** > <workspace>/<board name>/Drivers/STM32F4xx_HAL_Driver/inc > stm32f4xx_hal_gpio.h
    
    ```
    HAL_GPIO_TogglePin(GPIO_TypeDef*, uint16_t) : void
    ```
  - Open **Includes** > <workspace>/<board name>/Drivers/STM32F4xx_HAL_Driver/inc > stm32f4xx_hal.h
    
    ```
    HAL_Delay(volatile uint32_t Delay) : void
    ```
- Update “**main.c**” with blink LD2 code
  - Open Example > User > main.c – line 86 above “/* USER CODE END WHILE */”
  - Insert `HAL_GPIO_TogglePin(LD2_GPIO_Port, LD2_Pin);`
  - Insert `HAL_Delay(100);`
Compile Example

- In **Project Explorer:** select the STM32CubeMX Blink LED project
- Select **Rebuild Project**
- Select **Problems View** to check for compile errors (there shouldn’t be any)
- Select **Console View** to check compile time, and code size/RAM usage

Download/Debug

- In **Project Explorer:** select the STM32CubeMX Blink LED project
- Select **Debug** – this will switch to Debug Perspective
  
  May get error message “Launching STM32F103C8Tx_Board.elf has encountered problem”
  
  “Error in initializing ST-Link device.” “Reason: (-6) ST-Link firmware upgrade required.”...
  
  If so, click OK and for “ST-Link firmware upgrade required”, click Yes, and Yes again

- The Debugger will download the program and stop on line 76 at “MX_GPIO_Init();” breakpoint in “main.c”
- In “main.c” set a breakpoint on line 87 at “HAL_Delay(100);”
- Click Resume (F8), LD2 should light
- In **Breakpoint View**, uncheck “main.c[line 87]” breakpoint
- Click **Resume** (F8), LD2 should start blinking