

Branch: master ▼

Find file

Copy path

MDS_DHKim_Docs / STM32_makefile.md

d-h-k mystg

0d855e4 32 seconds ago

1 contributor

Raw Blame History



115 lines (109 sloc) 6.68 KB

make로 STM32시리즈 펌웨어 개발하기

- 이 문서는 makefile 유틸리티로 STM32시리즈 펌웨어 개발하는 방법을 공유합니다
- 윈도우10 에서 리눅스와 거의 유사하게 Make기반 프로젝트를 생성해서 컴파일하는 튜토리얼 입니다.
- 만들어진 해당 프로젝트는 Windows, MAC, Linux(우분투)모든 OS에서 컴파일이 가능합니다.
- 영문버전 문서는 [여기로 English Version](#)
- 개발환경은 아래와 같습니다
 - OS : windows10 64bit
 - MCU : STM32L476VGT - Discovery board
 - 이외 사용 툴 : STM32CubeMX, Visual studio Code, (옵션 : Vim)

0. Overview

1. Makefile 설치
2. GCC ARM Cross compiler 설치
3. CubeMX에서 makefile 프로젝트 생성
4. 구동 및 테스트
5. 바이너리(펌웨어) 다운로드 및 구동

1. Makefile 설치

- CMake를 설치합니다 [여기링크](#)에서 CMake Windosw버전을 클릭하여 다운로드 합니다

The screenshot shows the CMake download page. It includes a table for source code downloads and a table for binary distributions. The source code table lists Unix/Linux Source and Windows Source with their respective tar.gz and zip files. The binary distributions table lists various operating systems and architectures with their corresponding installer or archive files. A download verification section at the bottom provides cryptographic hashes and PGP signatures for the downloads.

Platform	Files
Unix/Linux Source (has \n line feeds)	cmake-3.15.0-rc2.tar.gz cmake-3.15.0-rc2.tar.Z
Windows Source (has \r\n line feeds)	cmake-3.15.0-rc2.zip

Binary distributions:

Platform	Files
Windows win64-x64 Installer: Installer tool has changed. Uninstall CMake 3.4 or lower first!	cmake-3.15.0-rc2-win64-x64.msi
Windows win64-x64 ZIP	cmake-3.15.0-rc2-win64-x64.zip
Windows win32-x86 Installer: Installer tool has changed. Uninstall CMake 3.4 or lower first!	cmake-3.15.0-rc2-win32-x86.msi
Windows win32-x86 ZIP	cmake-3.15.0-rc2-win32-x86.zip
Mac OS X 10.7 or later	cmake-3.15.0-rc2-Darwin-x86_64.dmg cmake-3.15.0-rc2-Darwin-x86_64.tar.gz
Linux x86_64	cmake-3.15.0-rc2-Linux-x86_64.sh cmake-3.15.0-rc2-Linux-x86_64.tar.gz

Download verification:

Role	Files
Cryptographic Hashes	cmake-3.15.0-rc2-SHA-256.txt
PGP sig by EC8FEF3A7BFB4EDA	cmake-3.15.0-rc2-SHA-256.txt.asc

- 위 사진에서 windows OS를 위한 executable installer 를 다운받으시면 됩니다
- CMake 설치과정은 [여기링크](#)를 참고하시면 됩니다

2. GCC ARM Cross compiler 설치

- GCC ARM Cross compiler 설치를 위해 [여기링크](#)를 눌러 GCC ARM Cross compiler 를 다운받으면 됩니다. 글
- 작성시점(2019-06-21) 기준 아래 이름의 파일 입니다

gcc-arm-none-eabi-5_4-2016q3-20160926-win32.exe

Download project files

[How do I verify a download?](#)

1 → 10 of 18 releases First • Previous • **Next** • Last

5-2016-q3-update release from the 5.0 series released 2016-09-28

[Release information](#)

File	Description	Downloads
release.txt (md5)	Release notes	13,773 last downloaded today
gcc-arm-none-eabi-5_4-2016q3-20160926-win32.exe (md5)	Windows installer	571,628 last downloaded today
gcc-arm-none-eabi-5_4-2016q3-20160926-win32.zip (md5)	Windows zip package	251,889 last downloaded today
gcc-arm-none-eabi-5_4-2016q3-20160926-linux.tar.bz2 (md5)	Linux installation tarball	338,090 last downloaded today
gcc-arm-none-eabi-5_4-2016q3-20160926-mac.tar.bz2 (md5)	Mac installation tarball	105,727 last downloaded today
gcc-arm-none-eabi-5_4-2016q3-20160926-src.tar.bz2 (md5)	Source package	122,063 last downloaded today

- 컴파일러 설치과정은 [여기링크](#) 를 참고하시면 됩니다
- 컴파일러 설치가 완료되면 ARM Cross compiler 의 환경변수를 등록 해 줍니다

컴파일러의 환경변수 등록이 되지 않으면, 에러가 발생하므로 꼭 진행해야 합니다!! (많은 분들이 컴파일러의 환경변수 등록을 하지않는 에러를 겪으십니다..)

•

C:\Program Files (x86)\GNU Tools ARM Embedded\5.4 2016q3\bin

- [Windows키] + R을 눌러 실행창에서 "cmd"를 입력합니다

cmd

```
arm-none-eabi-gcc -v
```

```
C:\Users\dhkim>arm-none-eabi-gcc -v
Using built-in specs.
COLLECT_GCC=arm-none-eabi-gcc
COLLECT_LTO_WRAPPER=c:/program files\ (x86)\gnu\ tools\ arm\ embedded\ 5.4\ 2016q3\bin\../lib/gcc/arm-none-eabi/5.4.1/lto-wrapper.exe Target: arm-none-eabi

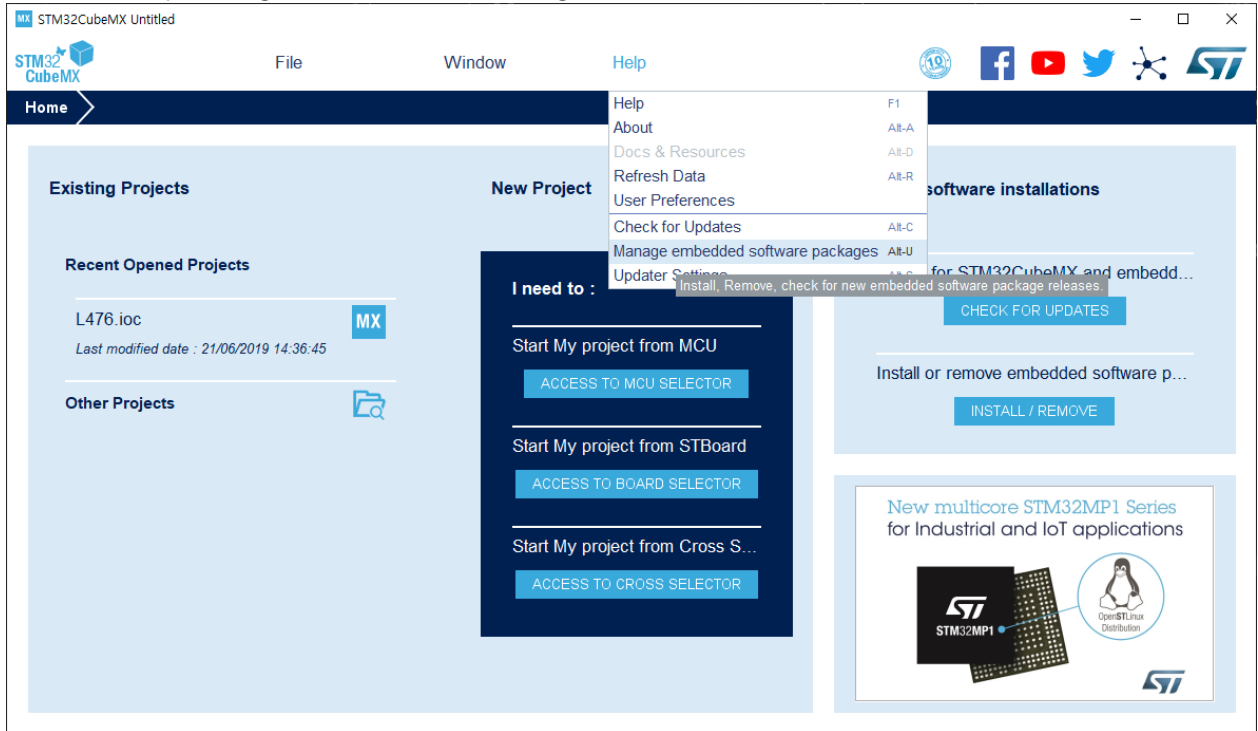
~ <<<<중략>>>> ~

usr --with-host-libstdcxx='-static-libgcc -Wl,-Bstatic,-lstdc++,-Bdynamic -lm' --with-pkgversion='GNU Tools for ARM Embedded Processors' -with-multilib-list=armv6-m,armv7-m,armv7e-m,armv7-r,armv8-m.base,armv8-m.main Thread model: single
gcc version 5.4.1 20160919 (release) [ARM/embedded-5-branch revision 240496] (GNU Tools for ARM Embedded Processors)
```

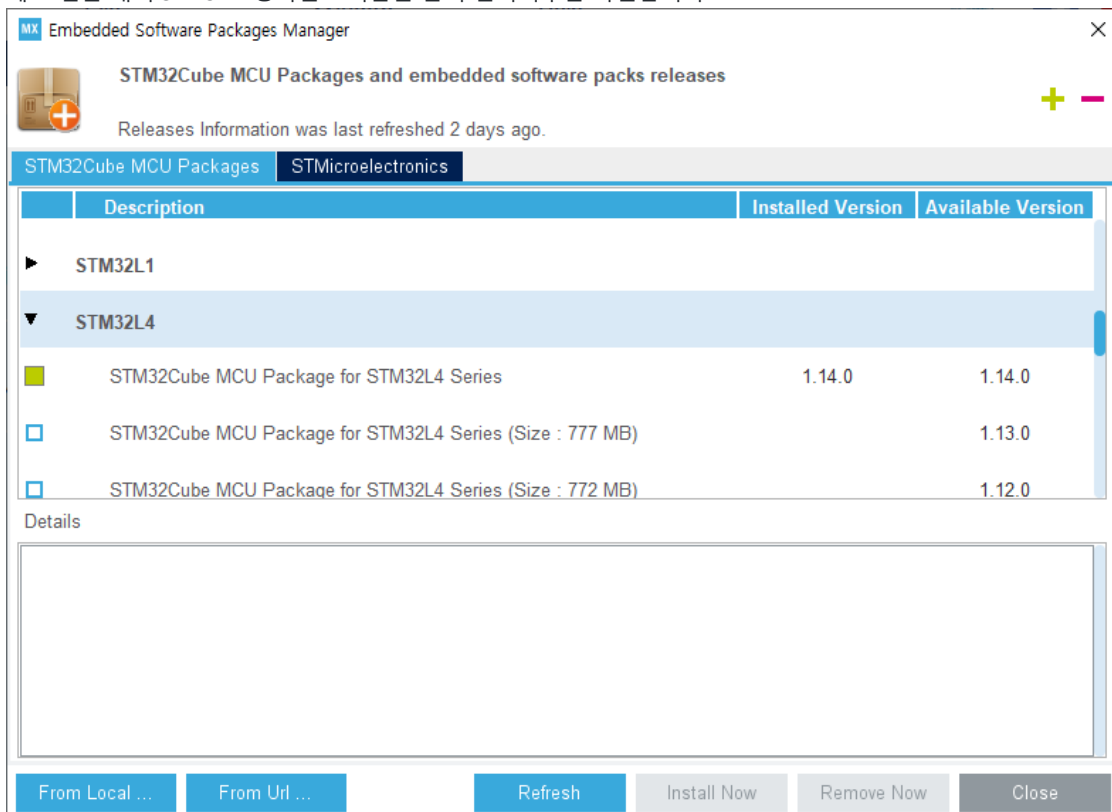
- 위와같은 메시지를 만나지 못한경우, 2번항목을 다시 반복하여 문제를 해결해야 합니다.

3. CubeMX에서 makefile 프로젝트 생성

- CubeMX 다운로드 및 설치를 진행합니다, 링크참조
- 이제 STM32CubeMX 에서 라이브러리 코드 및 BSP소스, Makefile을 생성해보겠습니다 (이런 툴을 제공해주는 ST는 정말 좋은 칩 벤더 입니다!)
- CubeMX버전은 5.2.1 입니다
- 홈 화면에서 Help->Manage embedded Software Packages 메뉴를 클릭합니다

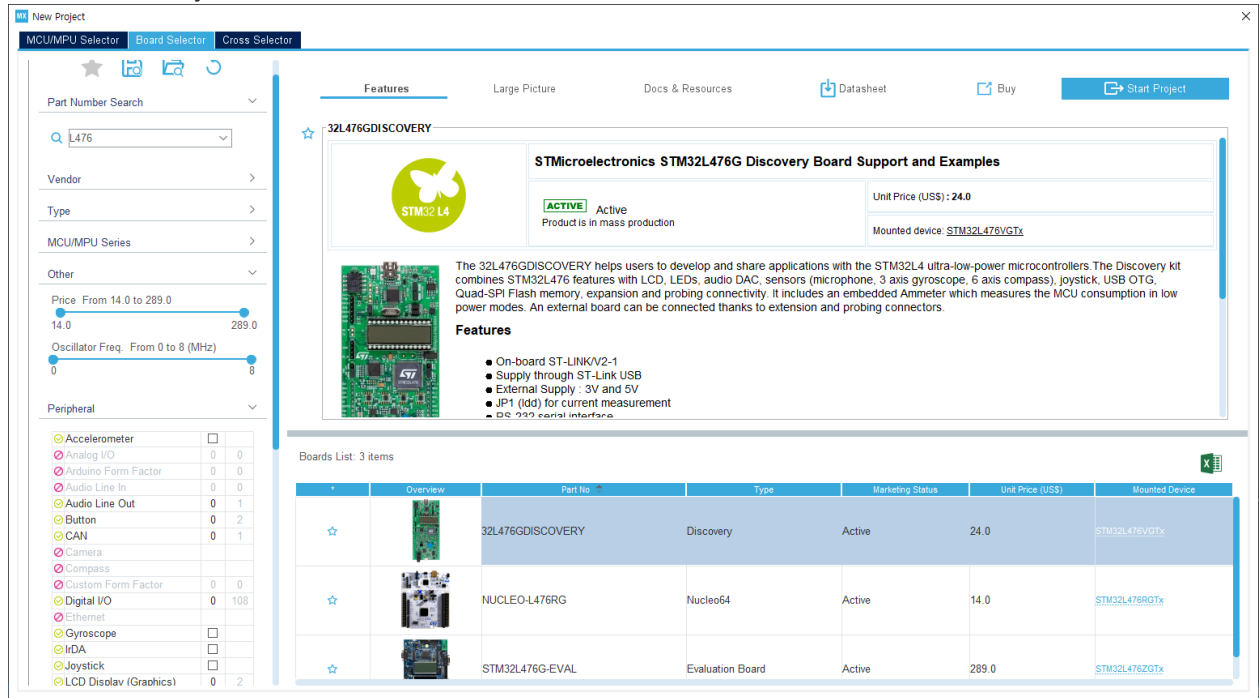


- 세로 탭을 내려 STM32L4항목을 ▼버튼을 눌러 설치여부를 확인합니다

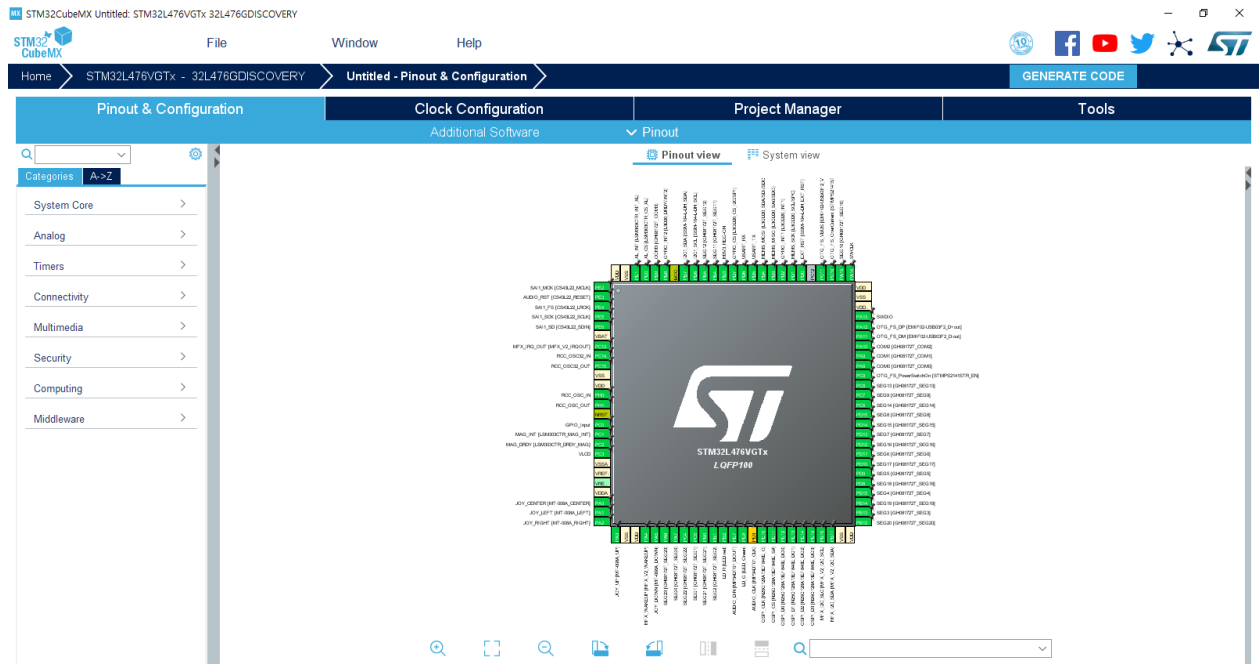


- File->New Project를 눌러 새로운 프로젝트를 시작합니다

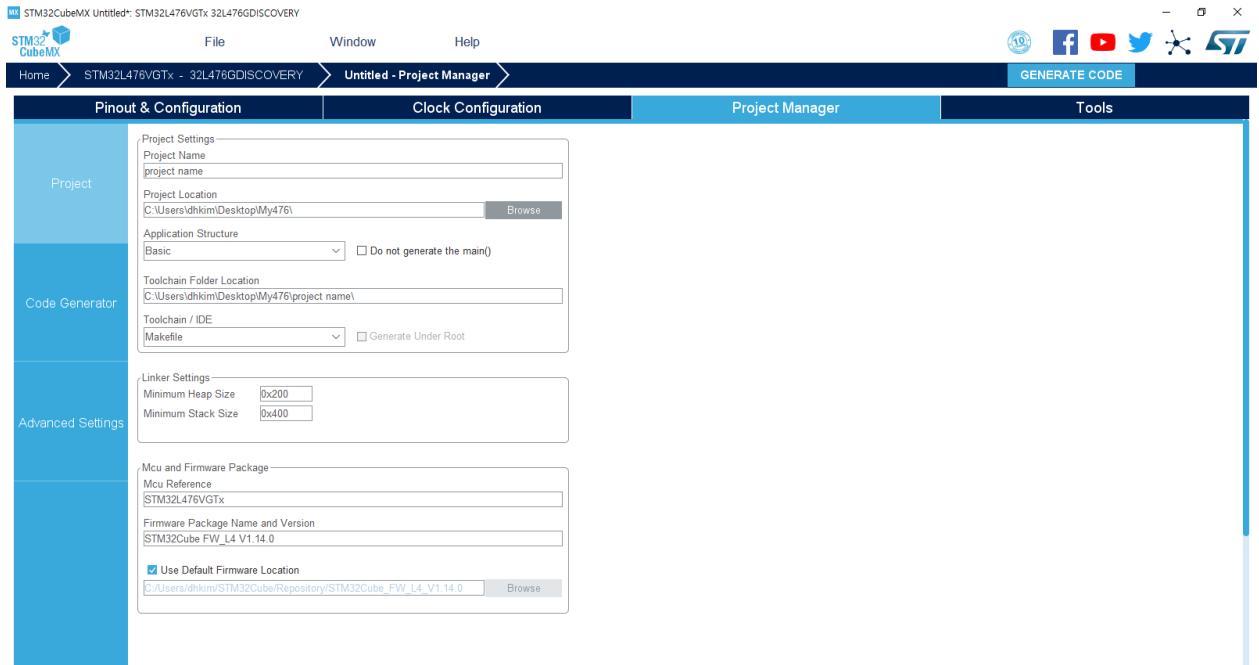
- 이때 상단바에서 Board Selector 탭으로 이동하여 "L746" 으로 검색하여 나오는 첫번째 보드를 선택합니다
- 우측 상단 Start Project 를 클릭합니다



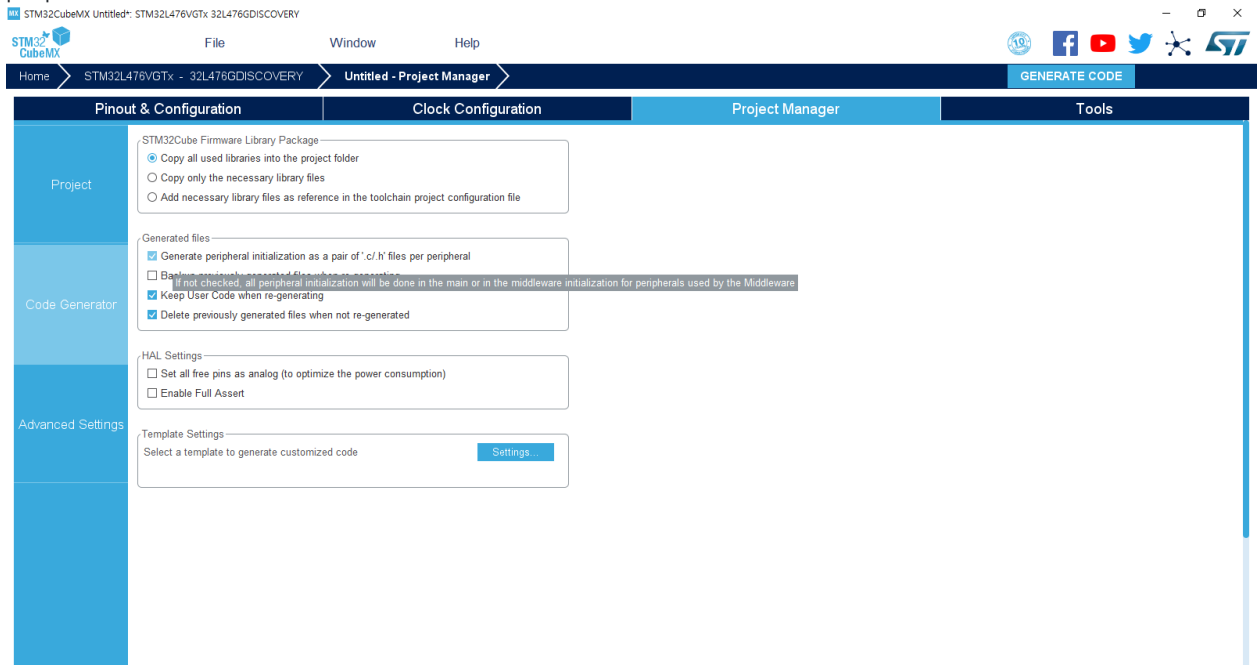
- 버튼을 눌러 프로젝트 생성 시 하단과 같은 기본값 생성에 YES를 클릭합니다
- 프로젝트가 생성되었고, Pin들이 Board에 맞게 셋팅된 모습을 확인할 수 있습니다



- 상단에 위치한 탭중 Project Manager 탭으로 이동합니다
- Project Setting 항목에서 Project Name과 Location을 자신이 원하는 위치에 지정해 줍니다(디렉토리 경로에 한글이 들어가 있으면 오동작 위험이 있습니다. 영문으로 하시기를 권해드립니다)



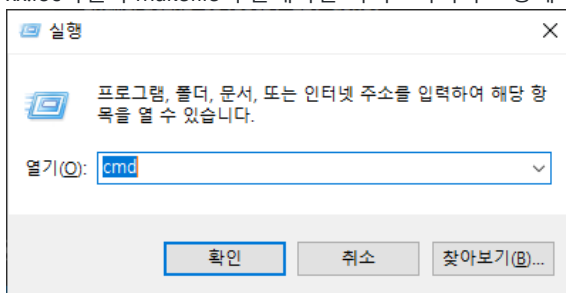
- Code Generator 항목에서 Generated files 항목에 첫번째요소 Generate peripheral initialization as a pair of '.c/.h' files per peripheral 요소를 체크합니다



-여기까지 완료후 Generation Code버튼을 눌러(우측 상단) 코드를 자동생성합니다

4. 구동 및 테스트

- xx.ioc파일과 makefile이 존재하는 디렉토리의 주소창에서 "cmd"를 입력하여 커맨드 창을 실행합니다



- 커맨드 창에서 make를 입력합니다

```
C:\Windows\System32\cmd.exe
C:\Users\dhkim\Desktop\My476\L476>make
```

make

- make (프로젝트 빌드)가 완료된 모습입니다

```
C:\Windows\System32\cmd.exe
arm-none-eabi-gcc -c -mcpu=cortex-m4 -mthumb -mfpv4-sp-d16 -mfloat-abi=hard -DUSE_HAL_DRIVER -DSTM32L476xx -DUSE_HAL_DRIVER -DSTM32L476xx -ICore/Inc -IUSB_HOST/App -IUSB_HOST/Target -IDrivers/STM32L4xx_HAL_Driver/Inc -IDrivers/STM32L4xx_HAL_Driver/Inc/Legacy -IMiddlewares/ST/STM32_USB_Host_Library/Core/Inc -IMiddlewares/ST/STM32_USB_Host_Library/Class/CDC/Inc -IDrivers/CMSIS/Device/ST/STM32L4xx/Include -IDrivers/CMSIS/Include -IDrivers/CMSIS/Include -Og -Wall -fdata-sections -ffunction-sections -g -gdwarf-2 -MMD -MP -MF"build/usbh_cdc.d" -Wa,-a,-ad,-alms=build/usbh_cdc.lst Middlewares/ST/STM32_USB_Host_Library/Class/CDC/Src/usbh_cdc.c -o build/usbh_cdc.o
arm-none-eabi-gcc -x assembler-with-cpp -c -mcpu=cortex-m4 -mthumb -mfpv4-sp-d16 -mfloat-abi=hard -DUSE_HAL_DRIVER -DSTM32L476xx -DUSE_HAL_DRIVER -DSTM32L476xx -ICore/Inc -IUSB_HOST/App -IUSB_HOST/Target -IDrivers/STM32L4xx_HAL_Driver/Inc -IDrivers/STM32L4xx_HAL_Driver/Inc/Legacy -IMiddlewares/ST/STM32_USB_Host_Library/Core/Inc -IMiddlewares/ST/STM32_USB_Host_Library/Class/CDC/Inc -IDrivers/CMSIS/Device/ST/STM32L4xx/Include -IDrivers/CMSIS/Include -IDrivers/CMSIS/Include -Og -Wall -fdata-sections -ffunction-sections -g -gdwarf-2 -MMD -MP -MF"build/startup_stm32l476xx.d" startup_stm32l476xx.s -o build/startup_stm32l476xx.o
arm-none-eabi-gcc build/main.o build/gpio.o build/i2c.o build/lcd.o build/quadspi.o build/sai.o build/spi.o build/usart.o build/stm32l4xx_it.o build/stm32l4xx_hal_msp.o build/usb_host.o build/usb_conf.o build/usbh_platform.o build/stm32l4xx_hal_hcd.o build/stm32l4xx_ll_usb.o build/stm32l4xx_hal_i2c.o build/stm32l4xx_hal_i2c_ex.o build/stm32l4xx_hal_lcd.o build/stm32l4xx_hal_qspi.o build/stm32l4xx_hal_sai.o build/stm32l4xx_hal_sai_ex.o build/stm32l4xx_hal_spi.o build/stm32l4xx_hal_spi_ex.o build/stm32l4xx_hal_tim.o build/stm32l4xx_hal_tim_ex.o build/stm32l4xx_hal_uart.o build/stm32l4xx_hal_uart_ex.o build/stm32l4xx_hal.o build/stm32l4xx_hal_rcc.o build/stm32l4xx_hal_rcc_ex.o build/stm32l4xx_hal_flash.o build/stm32l4xx_hal_flash_ex.o build/stm32l4xx_hal_flash_ramfunc.o build/stm32l4xx_hal_gpio.o build/stm32l4xx_hal_dma.o build/stm32l4xx_hal_dma_ex.o build/stm32l4xx_hal_pwr.o build/stm32l4xx_hal_pwr_ex.o build/stm32l4xx_hal_cortex.o build/stm32l4xx_hal_exti.o build/system_stm32l4xx.o build/usbh_core.o build/usbh_ctlreq.o build/usbh_ioreq.o build/usbh_pipes.o build/usbh_cdc.o build/startup_stm32l476xx.o -mcpu=cortex-m4 -mthumb -mfpv4-sp-d16 -mfloat-abi=hard -specs=nano.specs -TSTM32L476VGtx_FLASH.ld -lc -lm -lnosys -Wl,-Map=build/L476.map,--cref -Wl,--gc-sections -o build/L476.elf
arm-none-eabi-size build/L476.elf
text      data      bss      dec      hex filename
24320     160     4048     28528     6f70 build/L476.elf
arm-none-eabi-objcopy -O ihex build/L476.elf build/L476.hex
arm-none-eabi-objcopy -O binary -S build/L476.elf build/L476.bin
C:\Users\dhkim\Desktop\My476\L476>
```

- make를 한번 더 콘솔에 입력하면, 변경점 없음을 인식한 make 유틸리티가 컴파일을 다시 하지 않는것을 확인합니다.

```
C:\Windows\System32\cmd.exe
C:/Inc -IDrivers/CMSIS/Device/ST/STM32L4xx/Include -IDrivers/CMSIS/Include -IDrivers/CMSIS/Include -Og -Wall -fdata-sections -ffunction-sections -g -gdwarf-2 -MMD -MP -MF"build/usbh_cdc.d" -Wa,-a,-ad,-alms=build/usbh_cdc.lst -Ml build/usbh_cdc.o
arm-none-eabi-gcc -x assembler-with-cpp -c -mcpu=cortex-m4 -mthumb -mfpv4-sp-d16 -mfloat-abi=hard -DUSE_HAL_DRIVER -DSTM32L476xx -DUSE_HAL_DRIVER -DSTM32L476xx -ICore/Inc -IUSB_HOST/App -IUSB_HOST/Target -IDrivers/STM32L4xx_HAL_Driver/Inc -IDrivers/STM32L4xx_HAL_Driver/Inc/Legacy -IMiddlewares/ST/STM32_USB_Host_Library/Core/Inc -IMiddlewares/ST/STM32_USB_Host_Library/Class/CDC/Inc -IDrivers/CMSIS/Device/ST/STM32L4xx/Include -IDrivers/CMSIS/Include -IDrivers/CMSIS/Include -Og -Wall -fdata-sections -ffunction-sections -g -gdwarf-2 -MMD -MP -MF"build/startup_stm32l476xx.d" startup_stm32l476xx.s -o build/startup_stm32l476xx.o
arm-none-eabi-gcc build/main.o build/gpio.o build/i2c.o build/lcd.o build/quadspi.o build/sai.o build/spi.o build/usart.o build/stm32l4xx_it.o build/stm32l4xx_hal_msp.o build/usb_host.o build/usbh_conf.o build/usbh_platform.o build/stm32l4xx_hal_hcd.o build/stm32l4xx_ll_usb.o build/stm32l4xx_hal_i2c.o build/stm32l4xx_hal_i2c_ex.o build/stm32l4xx_hal_lcd.o build/stm32l4xx_hal_qspi.o build/stm32l4xx_hal_sai.o build/stm32l4xx_hal_sai_ex.o build/stm32l4xx_hal_spi.o build/stm32l4xx_hal_spi_ex.o build/stm32l4xx_hal_tim.o build/stm32l4xx_hal_tim_ex.o build/stm32l4xx_hal_uart.o build/stm32l4xx_hal_uart_ex.o build/stm32l4xx_hal.o build/stm32l4xx_hal_rcc.o build/stm32l4xx_hal_rcc_ex.o build/stm32l4xx_hal_flash.o build/stm32l4xx_hal_flash_ex.o build/stm32l4xx_hal_flash_ramfunc.o build/stm32l4xx_hal_gpio.o build/stm32l4xx_hal_dma.o build/stm32l4xx_hal_dma_ex.o build/stm32l4xx_hal_pwr.o build/stm32l4xx_hal_pwr_ex.o build/stm32l4xx_hal_cortex.o build/stm32l4xx_hal_exti.o build/system_stm32l4xx.o build/usbh_core.o build/usbh_ctlreq.o build/usbh_ioreq.o build/usbh_pipes.o build/usbh_cdc.o build/startup_stm32l476xx.o -mcpu=cortex-m4 -mthumb -mfpv4-sp-d16 -mfloat-abi=hard -specs=nano.specs -TSTM32L476VGTX_FLASH.ld -lc -lm -lnosys -Wl,-Map=build/L476.map,--cref -Wl,--gc-sections -o build/L476.elf
arm-none-eabi-size build/L476.elf
   text    data     bss     dec     hex filename
  24320    160    4048   28528    6f70 build/L476.elf
arm-none-eabi-objcopy -O ihex build/L476.elf build/L476.hex
arm-none-eabi-objcopy -O binary -S build/L476.elf build/L476.bin

C:\Users\dhkim\Desktop\My476\476>make
make: Nothing to be done for 'all'.

C:\Users\dhkim\Desktop\My476\476>
```

- 프로젝트를 재 컴파일 하기 위해서는 make clean 명령어를 입력합니다

```
C:\Windows\System32\cmd.exe
arm-none-eabi-gcc -x assembler-with-cpp -c -mcpu=cortex-m4 -mthumb -mfpv4-sp-d16 -mfloat-abi=hard -DUSE_HAL_DRIVER -DSTM32L476xx -DUSE_HAL_DRIVER -DSTM32L476xx -ICore/Inc -IUSB_HOST/App -IUSB_HOST/Target -IDrivers/STM32L4xx_HAL_Driver/Inc -IDrivers/STM32L4xx_HAL_Driver/Inc/Legacy -IMiddlewares/ST/STM32_USB_Host_Library/Core/Inc -IMiddlewares/ST/STM32_USB_Host_Library/Class/CDC/Inc -IDrivers/CMSIS/Device/ST/STM32L4xx/Include -IDrivers/CMSIS/Include -IDrivers/CMSIS/Include -Og -Wall -fdata-sections -ffunction-sections -g -gdwarf-2 -MMD -MP -MF"build/startup_stm32l476xx.d" startup_stm32l476xx.s -o build/startup_stm32l476xx.o
arm-none-eabi-gcc build/main.o build/gpio.o build/i2c.o build/lcd.o build/quadspi.o build/sai.o build/spi.o build/usart.o build/stm32l4xx_it.o build/stm32l4xx_hal_msp.o build/usb_host.o build/usbh_conf.o build/usbh_platform.o build/stm32l4xx_hal_hcd.o build/stm32l4xx_ll_usb.o build/stm32l4xx_hal_i2c.o build/stm32l4xx_hal_i2c_ex.o build/stm32l4xx_hal_lcd.o build/stm32l4xx_hal_qspi.o build/stm32l4xx_hal_sai.o build/stm32l4xx_hal_sai_ex.o build/stm32l4xx_hal_spi.o build/stm32l4xx_hal_spi_ex.o build/stm32l4xx_hal_tim.o build/stm32l4xx_hal_tim_ex.o build/stm32l4xx_hal_uart.o build/stm32l4xx_hal_uart_ex.o build/stm32l4xx_hal.o build/stm32l4xx_hal_rcc.o build/stm32l4xx_hal_rcc_ex.o build/stm32l4xx_hal_flash.o build/stm32l4xx_hal_flash_ex.o build/stm32l4xx_hal_flash_ramfunc.o build/stm32l4xx_hal_gpio.o build/stm32l4xx_hal_dma.o build/stm32l4xx_hal_dma_ex.o build/stm32l4xx_hal_pwr.o build/stm32l4xx_hal_pwr_ex.o build/stm32l4xx_hal_cortex.o build/stm32l4xx_hal_exti.o build/system_stm32l4xx.o build/usbh_core.o build/usbh_ctlreq.o build/usbh_ioreq.o build/usbh_pipes.o build/usbh_cdc.o build/startup_stm32l476xx.o -mcpu=cortex-m4 -mthumb -mfpv4-sp-d16 -mfloat-abi=hard -specs=nano.specs -TSTM32L476VGTX_FLASH.ld -lc -lm -lnosys -Wl,-Map=build/L476.map,--cref -Wl,--gc-sections -o build/L476.elf
arm-none-eabi-size build/L476.elf
   text    data     bss     dec     hex filename
  24320    160    4048   28528    6f70 build/L476.elf
arm-none-eabi-objcopy -O ihex build/L476.elf build/L476.hex
arm-none-eabi-objcopy -O binary -S build/L476.elf build/L476.bin

C:\Users\dhkim\Desktop\My476\476>make
make: Nothing to be done for 'all'.

C:\Users\dhkim\Desktop\My476\476>make clean
rm -fr build

C:\Users\dhkim\Desktop\My476\476>
```

make clean

- 이 문서에서 필요한 make관련 명령어는 두가지입니다

```
make clean
make
```

5. 바이너리(펌웨어) 다운로드 및 구동

- STM32CubeProg 프로그램을 다운받습니다 : [STM32CubeProg](#) 다운로드 링크
- 빌드파일이 생성된 디렉토리로 이동합니다
- ~My476프로젝트 폴더 경로\L476\build 위치에 존재합니다.
- 해당위치에 XX.bin, XX.elf, XX.hex 파일이 존재하며 xx.elf파일은 디버깅이 필요할때, ST-link에 연결하여 디버깅하고, .bin파일은 디버깅 없이 프로그램 실행시 사용합니다.

- 자신이 필요한 파일을 STM32CubeProg 프로그램으로 다운로드해 실행시킵니다.

