임베디드 시스템 설계 및 실험 보고서

4 주차 실험_릴레이 모듈을 이용한 자동차 구동

분반 : 001 분반

교수님 : 정 상화 교수님

조교님 : 유 동화 조교님

실험일 : 2019-09-23

제출일 : 2019-09-30

00. 목차

7 조

장 수현

박 창조

임 다영

이 힘찬

01. 실험 목적

- 스캐터 파일의 이해 및 플래시 프로그래밍
- 릴레이 모듈의 이해 및 임베디드 펌웨어를 통한 동작

02. 실험 과제

2.1 주과제

릴레이 모듈을 이용한 자동차 구동

2.2 세부과제

- 개발 환경 구축
- DS-5 에서 프로젝트 생성 및 설정
- Datasheet 및 Reference Manual 을 참고하여 해당 레지스터 및 주소에 대한 설정 이해
- 스캐터 파일을 통해 플래시 메모리에 프로그램 다운로드
- 플래시 메모리에 올려진 프로그램 정상적인 동작 유무 확인
- 오실로스코프 사용 방법 학습

03. 실험 준비

3.1 실험 시 주의 사항

3.1.1 반드시 아래의 보드 연결 및 해제 순서를 지켜 실험을 진행해야 한다.

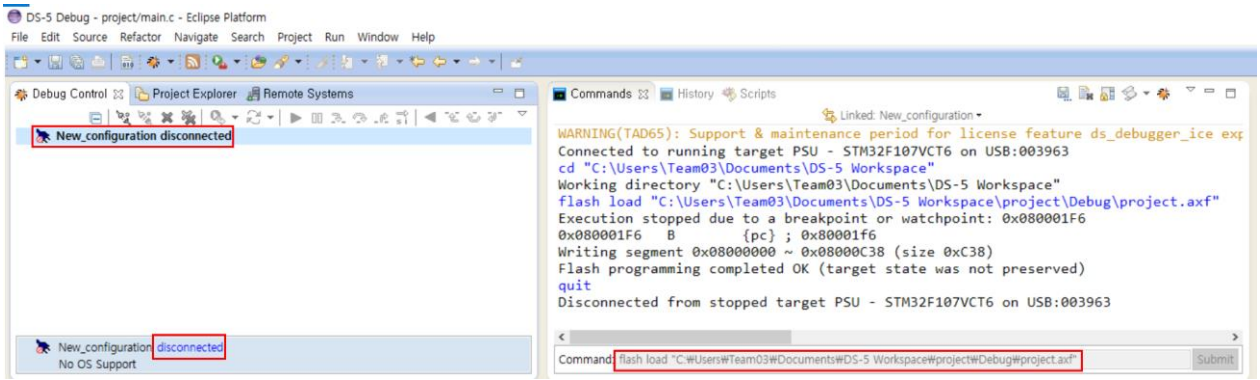
보드 연결 및 해제 순서

연결 과정	분리 과정
• 보드와 DSTREAM JTAG 연결 • 보드 전원선만 연결 (보드의 전원은 OFF 상태) • DSTREAM 전원 연결 및 ON • DSTREAM Status LED 점등 확인 • 보드 전원 ON • DSTREAM Target LED 점등 확인 • DS-5에서 'connect target'	• DS-5에서 'disconnect target' • 보드 전원 OFF • DSTREAM 전원 해제 및 OFF • 보드 전원선 분리 • DSTREAM과 보드 JTAG 분리

2

3.1.2 수정된 axf 파일을 보드에 다시 올리는 경우에는 반드시 "flashclear.axf"을 사용해서 보드를 리셋하고 난 뒤 올려야 한다. 그리고 파일을 올린 후에는 완전히 disconnected 상태로 만든 후 보드의 전원을 껐다가 다시 켜서 동작을 확인한다.

```
// flash load "C:\Users\Team03\Documents\DS-5 Workspace\project\Debug\flashclear\flashclear.axf"
```

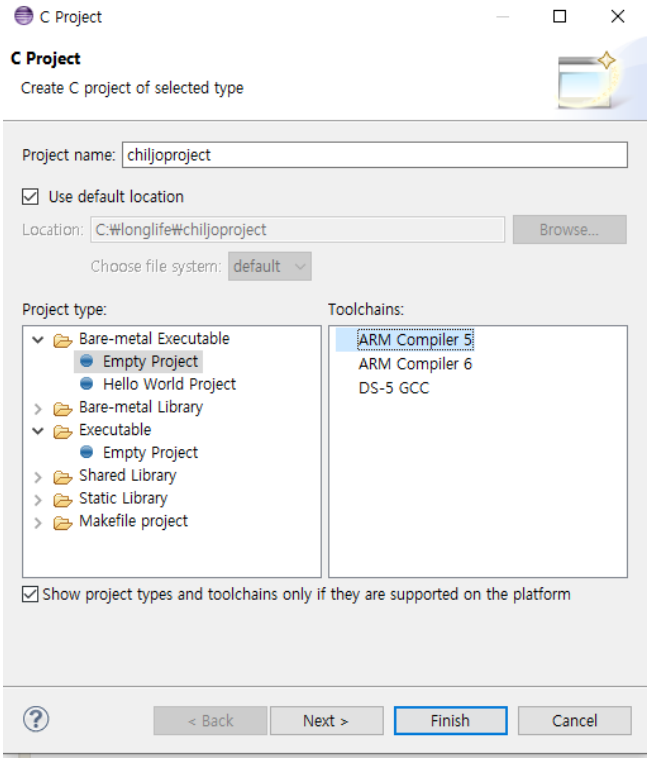


3.2 환경 설정

실험을 진행하기 전 아래의 프로젝트 생성 및 기본 환경 설정을 해주어야 한다.

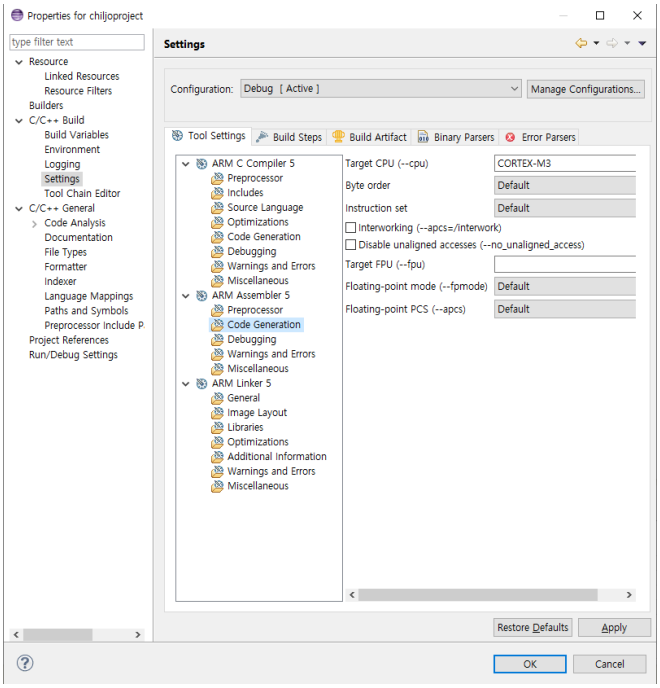
3.2.1 프로젝트 생성

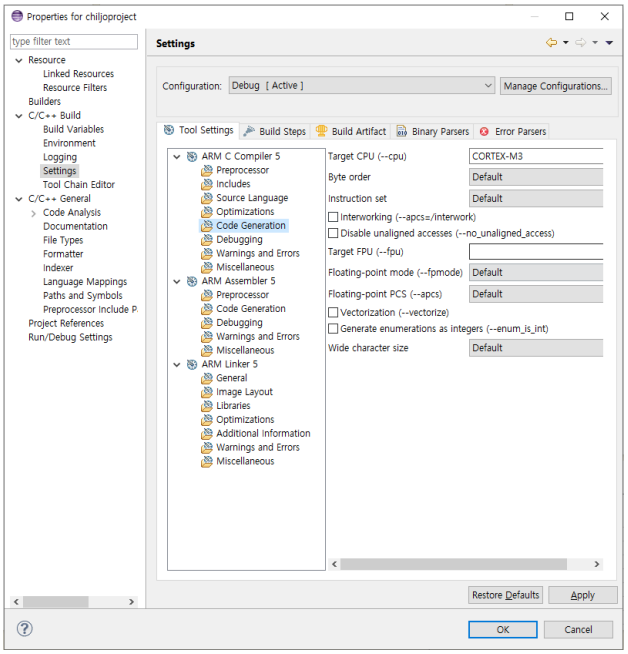
C 언어로 소스코드를 작성하기 때문에 C++프로젝트가 아닌 C 프로젝트를 생성해준다. Project type 은 Bare-metal Executable -〉 Empty Project 로, Toolchains 은 Arm Compiler 5 로 설정한다.

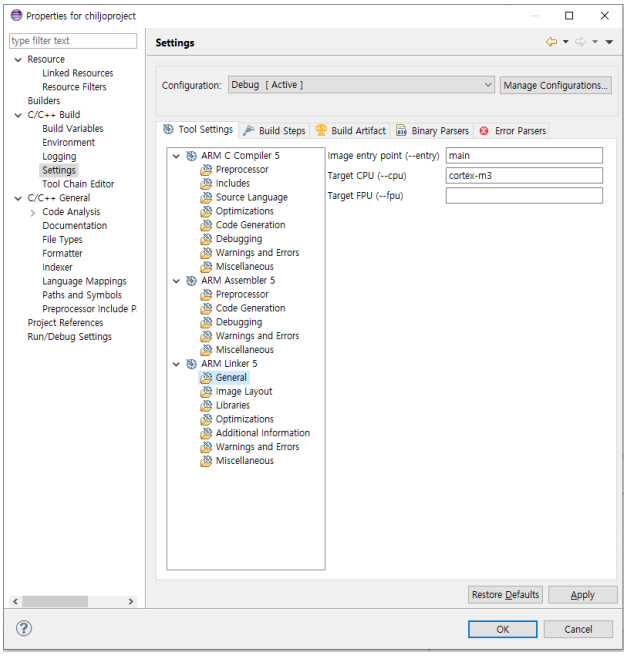


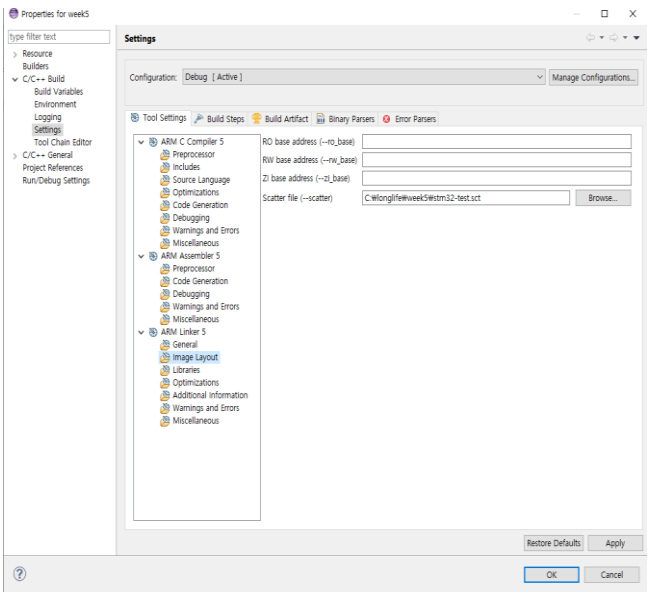
3.2.2 프로젝트 Properties-Settings

프로젝트 속성값을 세팅한다. 3 주차 실험에선 RO(Read Only) base address 와 RW(Read Write) base address 에 직접 값을 설정해주었는데 4 주차 실험에선 스캐터 파일을 이용한다.




Properties for chiljoproject
type filter text
Resource
Linked Resources
Resource Filters
Builders
C/C++ Build
Build Variables
Environment
Logging
Settings
Tool Chain Editor
C/C++ General
Code Analysis
Documentation
File Types
Formatter
Indexer
Language Mappings
Paths and Symbols
Preprocessor Include P.
Project References
Run/Debug Settings
Settings
Configuration: Debug [Active]
Manage Configurations...
Tool Settings
Build Steps
Build Artifact
Binary Parsers
Error Parsers
ARM C Compiler 5
Preprocessor
Includes
Source Language
Optimizations
Code Generation
Debugging
Warnings and Errors
Miscellaneous
ARM Assembler 5
Preprocessor
Code Generation
Debugging
Warnings and Errors
Miscellaneous
ARM Linker 5
General
Image Layout
Libraries
Optimizations
Additional Information
Warnings and Errors
Miscellaneous
Target CPU (--cpu)
CORTEX-M3
Byte order
Default
Instruction set
Default
Interworking (--apcs=/interwork)
Disable unaligned accesses (--no_unaligned_access)
Target FPU (--fpu)
Floating-point mode (--fpmode)
Default
Floating-point PCS (--apcs)
Default
Vectorization (--vectorize)
Generate enumerations as integers (--enum_is_int)
Wide character size
Default
Restore Defaults
Apply
OK
Cancel


Properties for chiljoproject
type filter text
Resource
Linked Resources
Resource Filters
Builders
C/C++ Build
Build Variables
Environment
Logging
Settings
Tool Chain Editor
C/C++ General
Code Analysis
Documentation
File Types
Formatter
Indexer
Language Mappings
Paths and Symbols
Preprocessor Include P.
Project References
Run/Debug Settings
Settings
Configuration: Debug [Active]
Manage Configurations...
Tool Settings
Build Steps
Build Artifact
Binary Parsers
Error Parsers
ARM C Compiler 5
Preprocessor
Includes
Source Language
Optimizations
Code Generation
Debugging
Warnings and Errors
Miscellaneous
ARM Assembler 5
Preprocessor
Code Generation
Debugging
Warnings and Errors
Miscellaneous
ARM Linker 5
General
Image Layout
Libraries
Optimizations
Additional Information
Warnings and Errors
Miscellaneous
Image entry point (--entry)
main
Target CPU (--cpu)
cortex-m3
Target FPU (--fpu)
Restore Defaults
Apply
OK
Cancel


Properties for week5
type filter text
Resource
Builders
C/C++ Build
Build Variables
Environment
Logging
Settings
Tool Chain Editor
C/C++ General
Project References
Run/Debug Settings
Settings
Configuration: Debug [Active]
Manage Configurations...
Tool Settings
Build Steps
Build Artifact
Binary Parsers
Error Parsers
ARM C Compiler 5
Preprocessor
Includes
Source Language
Optimizations
Code Generation
Debugging
Warnings and Errors
Miscellaneous
ARM Assembler 5
Preprocessor
Code Generation
Debugging
Warnings and Errors
Miscellaneous
ARM Linker 5
General
Image Layout
Libraries
Optimizations
Additional Information
Warnings and Errors
Miscellaneous
RO base address (--ro_base)
RW base address (--rw_base)
ZI base address (--zi_base)
Scatter file (--scatter)
C:\#longlife#week5\#stm32-test.sct
Browse...
Restore Defaults
Apply
OK
Cancel

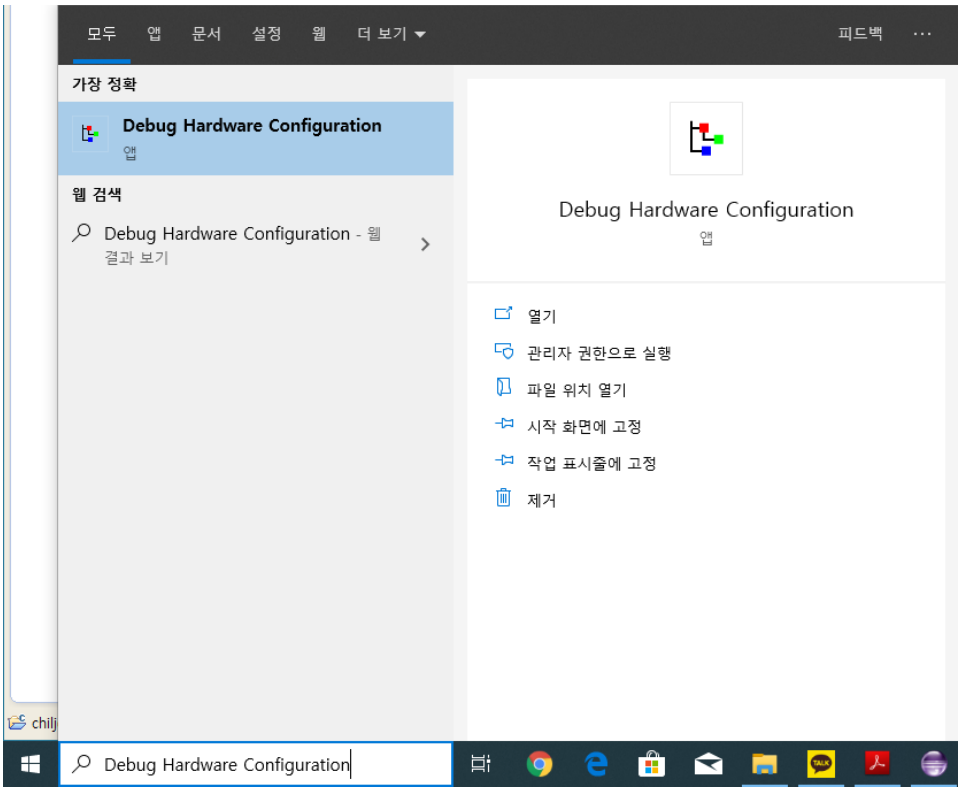
3.2.3 보드 연결

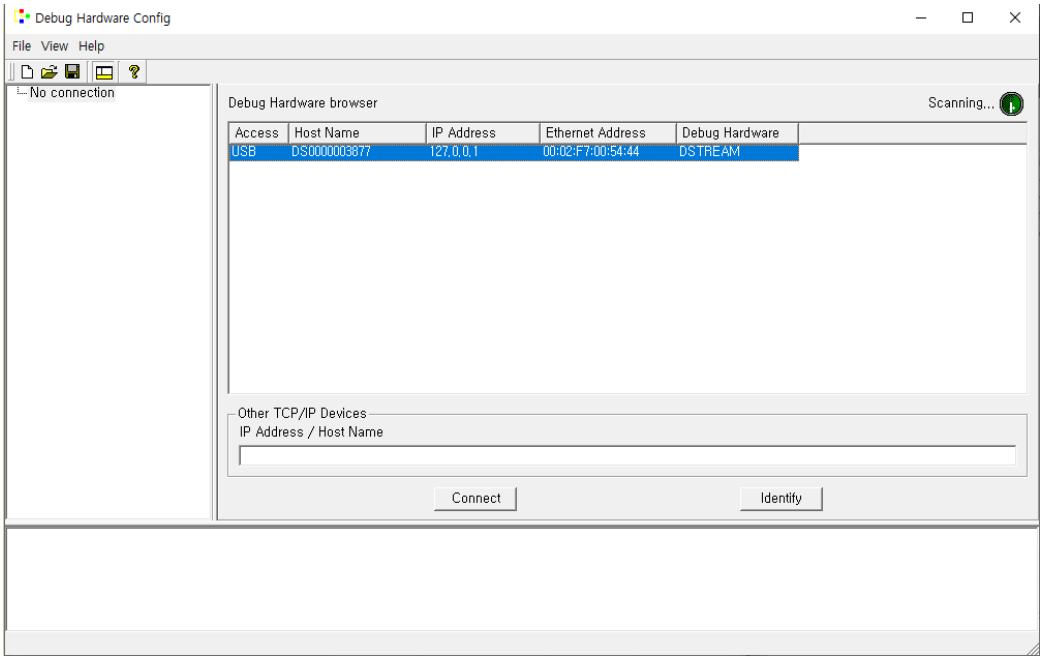
보드 연결 시에는 반드시 연결 순서를 지켜 연결하고, 맞는 전원선을 사용해야한다.

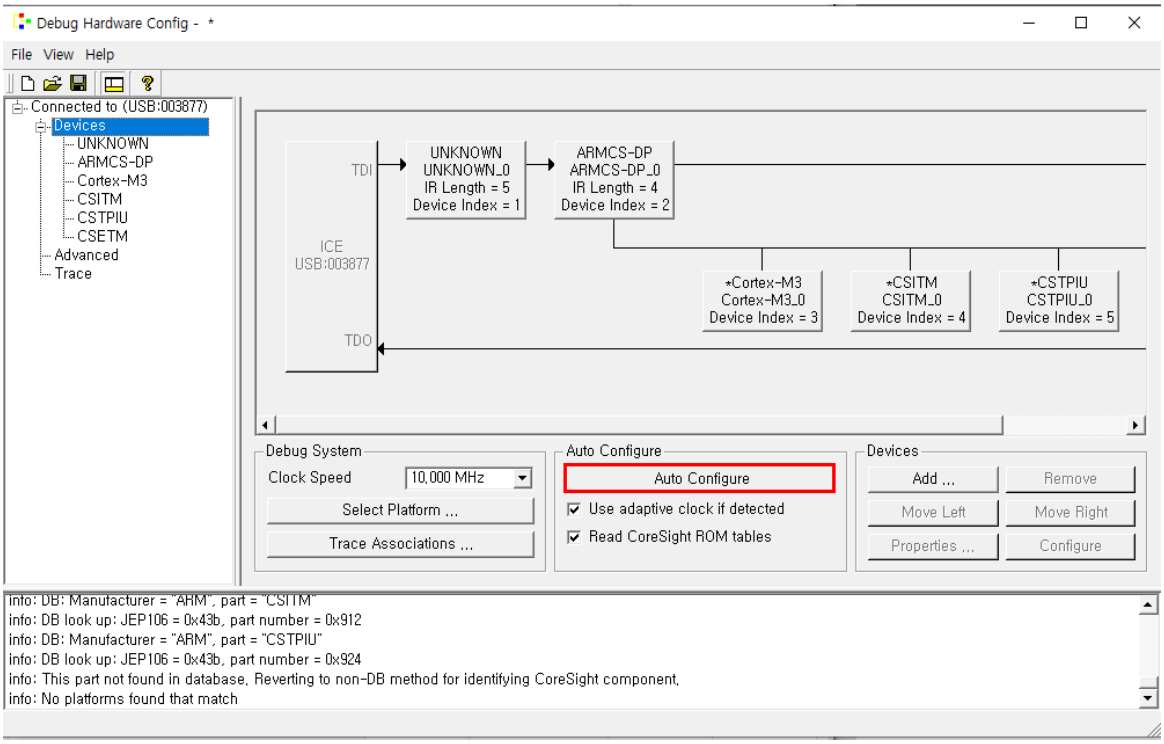
연결 순서 : 보드와 Dstream JTAG 연결 -〉 보드전원선 연결(보드 전원은 OFF) -〉 Dstream 전원 연결 및 ON -〉 Dstream Status LED 점등 확인 후 보드 전원 ON -〉 Dstream Target LED 점등 확인 후 DS-5 에서 'connect target'

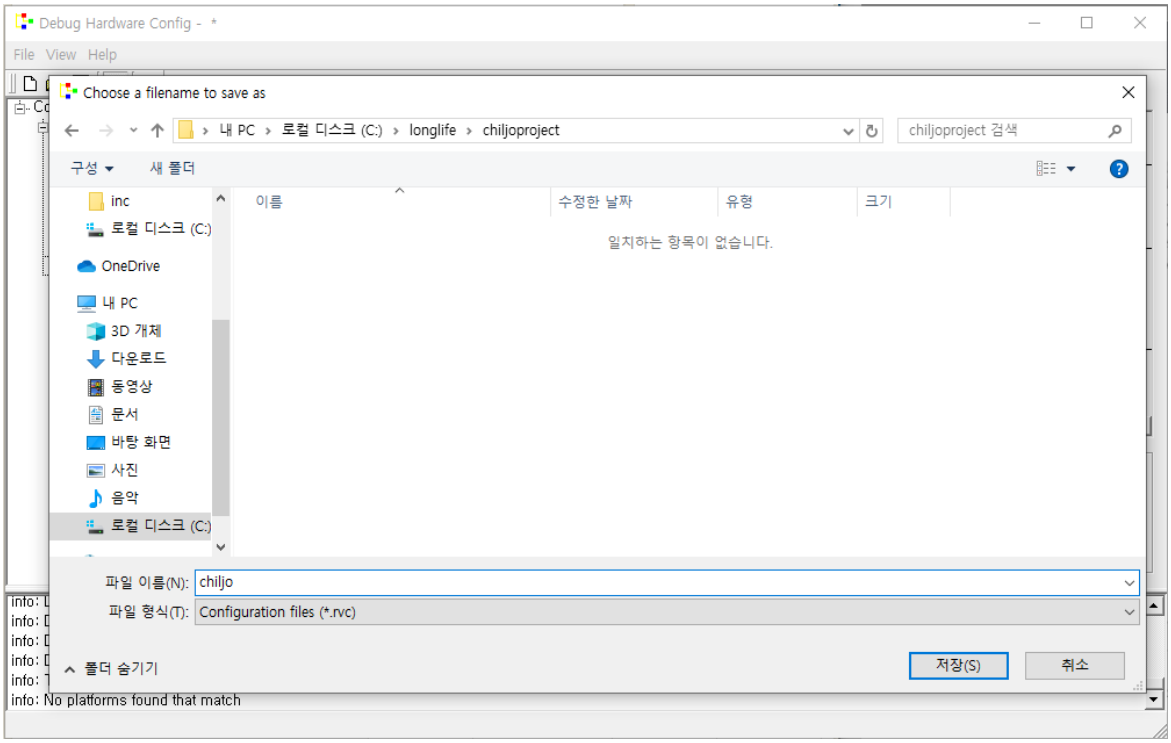
3.2.4 데이터 베이스 설정

Dstream 를 USB 로 컴퓨터에 연결하고 Debug Hardware config 에서 보드를 connect 한다. 보드 연결 후에는 Auto Configure 를 클릭하여 설정을 진행하고, rvc 파일로 저장한다. 이때 주의해야 할 점은 rvc 파일을 저장하는 파일의 경로에 한글 경로가 들어가면 안된다는 것이다.



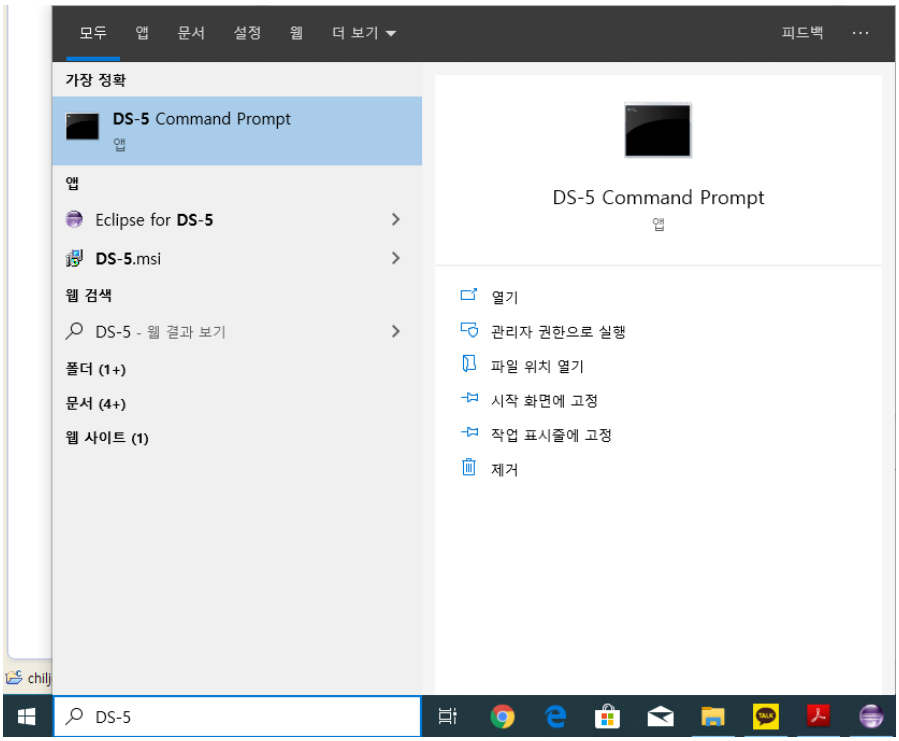



Debug Hardware Config - *
File View Help
Connected to (USB:003877)
Devices
UNKNOWN
ARMCS-DP
Cortex-M3
CSITM
CSTPIU
CSETM
Advanced
Trace
TDI
ICE
USB:003877
TDO
UNKNOWN
UNKNOWN_0
IR Length = 5
Device Index = 1
ARMCS-DP
ARMCS-DP_0
IR Length = 4
Device Index = 2
*Cortex-M3
Cortex-M3_0
Device Index = 3
*CSITM
CSITM_0
Device Index = 4
*CSTPIU
CSTPIU_0
Device Index = 5
Debug System
Clock Speed
10.000 MHz
Select Platform ...
Trace Associations ...
Auto Configure
Auto Configure
Use adaptive clock if detected
Read CoreSight ROM tables
Devices
Add ...
Remove
Move Left
Move Right
Properties ...
Configure
info: DB: Manufacturer = "ARM", part = "CSITM"
info: DB look up: JEP106 = 0x43b, part number = 0x912
info: DB: Manufacturer = "ARM", part = "CSTPIU"
info: DB look up: JEP106 = 0x43b, part number = 0x924
info: This part not found in database. Reverting to non-DB method for identifying CoreSight component.
info: No platforms found that match


Debug Hardware Config - *
File View Help
Choose a filename to save as
내 PC › 로컬 디스크 (C:) › longlife › chiljoproject
chiljoproject 검색
구성
새 폴더
inc
로컬 디스크 (C:)
OneDrive
내 PC
3D 개체
다운로드
동영상
문서
바탕 화면
사진
음악
로컬 디스크 (C:)
이름
수정한 날짜
유형
크기
일치하는 항목이 없습니다.
파일 이름(N): chiljo
파일 형식(T): Configuration files (*.rvc)
폴더 숨기기
저장(S)
취소
info: No platforms found that match

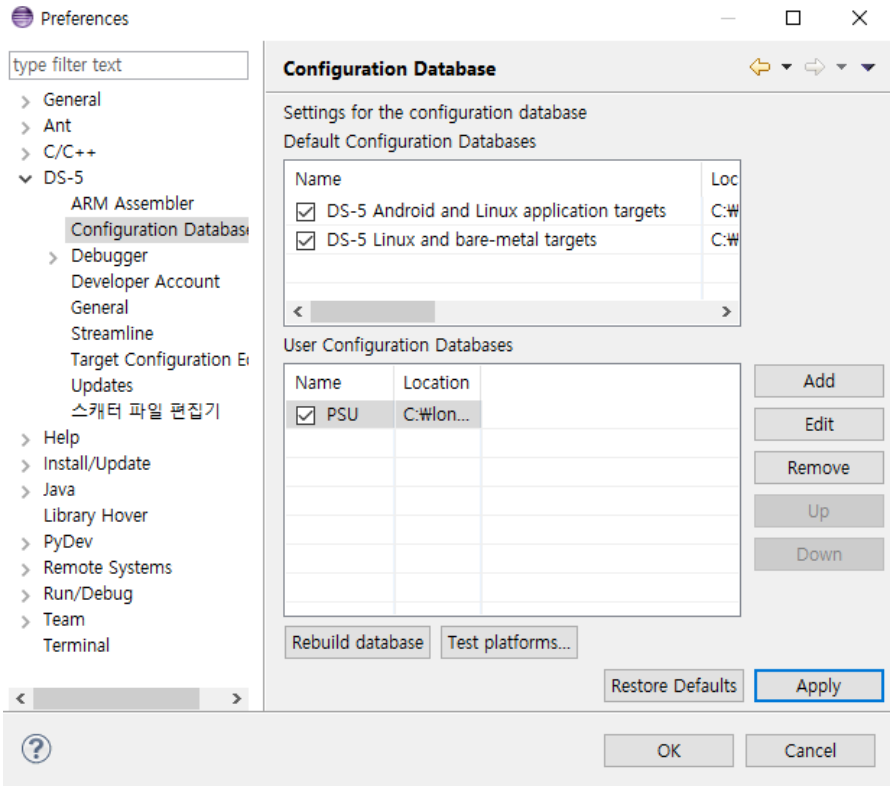
3.2.5 cdbimporter

DS-5 Command Prompt 에서 RVC 파일이 있는 폴더로 이동 후 아래와 같이 명령문을 작성한다.



3.2.6 데이터 베이스 등록

아래와 같이 데이터 베이스 등록을 진행한다.



3.2.7 라이브러리 추가

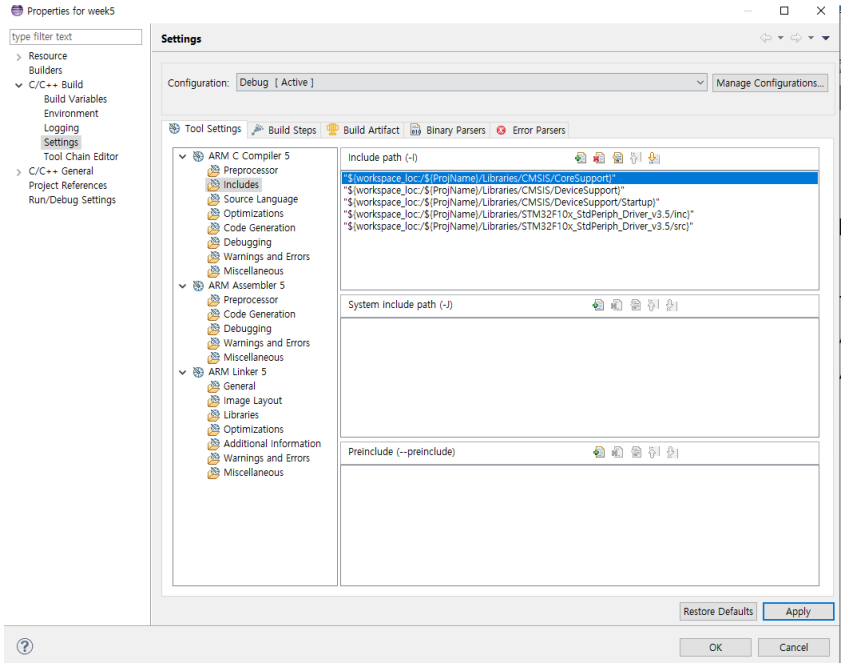
다음경로에 라이브러리를 추가해준다

CMSIS\CoreSupport

CMSIS\DeviceSupport

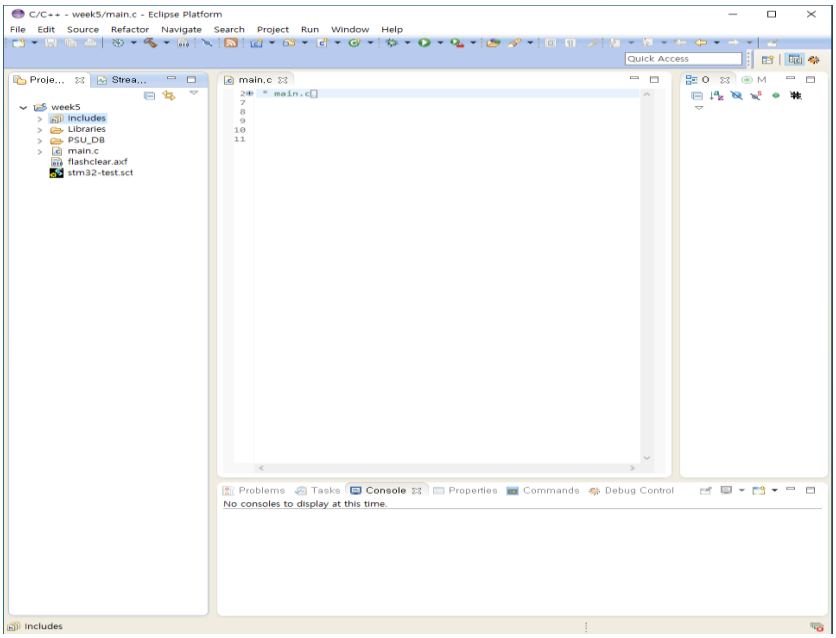
CMSIS\DeviceSupport\Startup STM32F10xStdPeriph_Driver_v3.5\inc

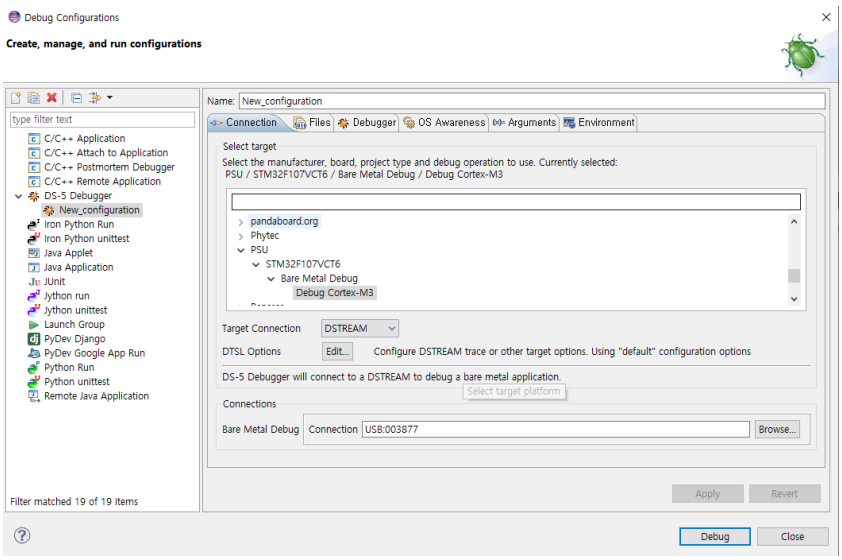
STM32F10xStdPeriph_Driver_v3.5\src

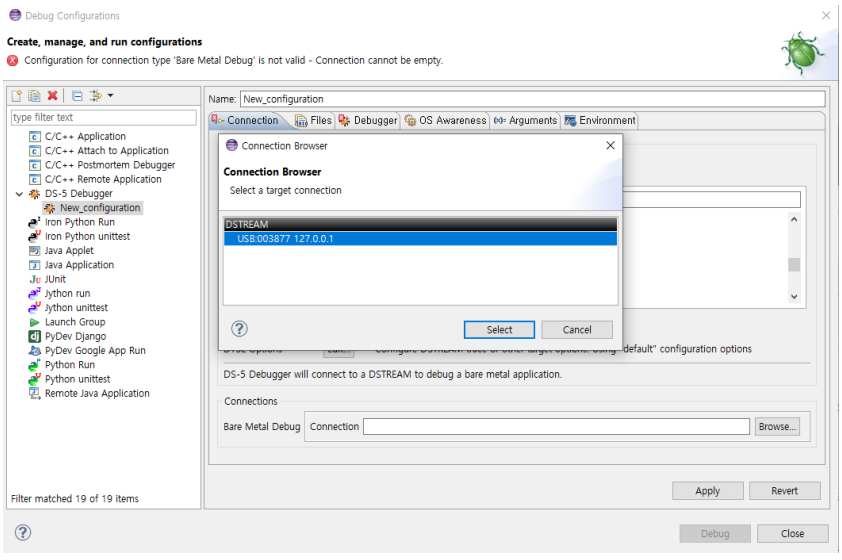


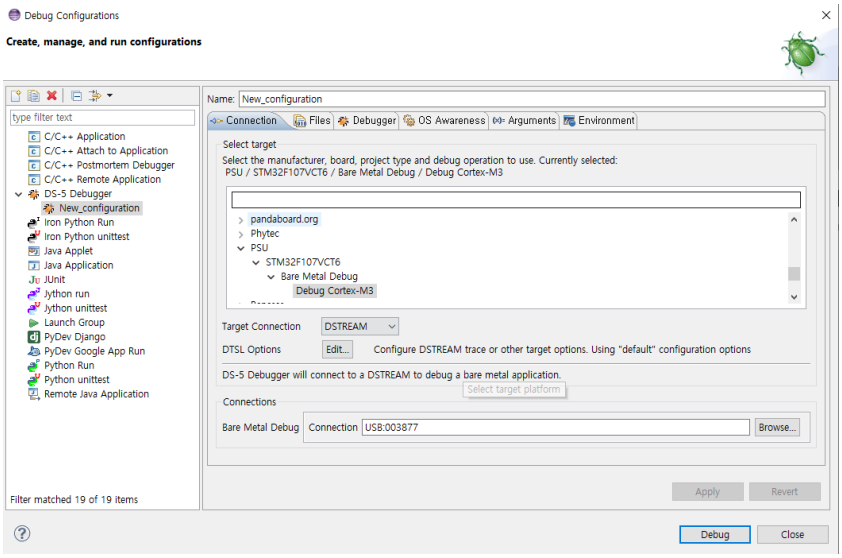
3.2.8 디버그 설정

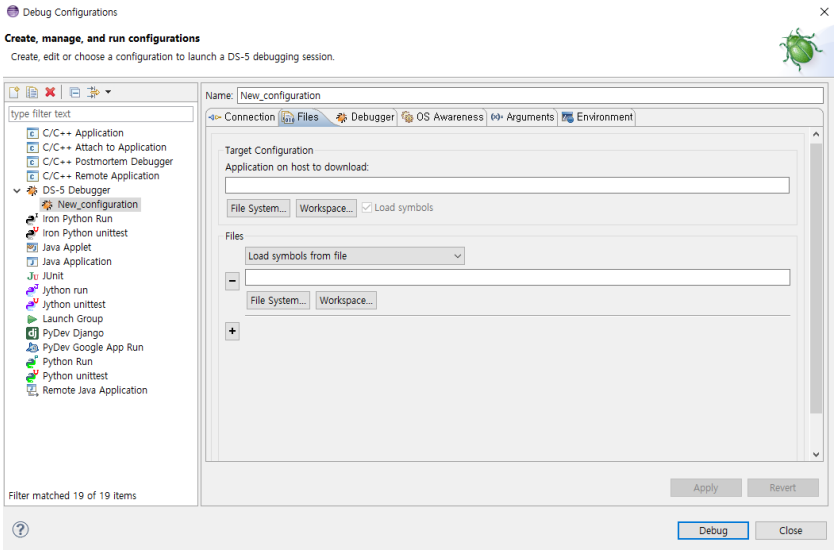
C 언어 소스파일을 만들어 빌드하고 디버그 설정을 한다. 3 주차와 달리 디버거 설정에서 axf file 을 추가하지 않는다. 디버거 “Run control”에서는 “Connect Only”선택한다.

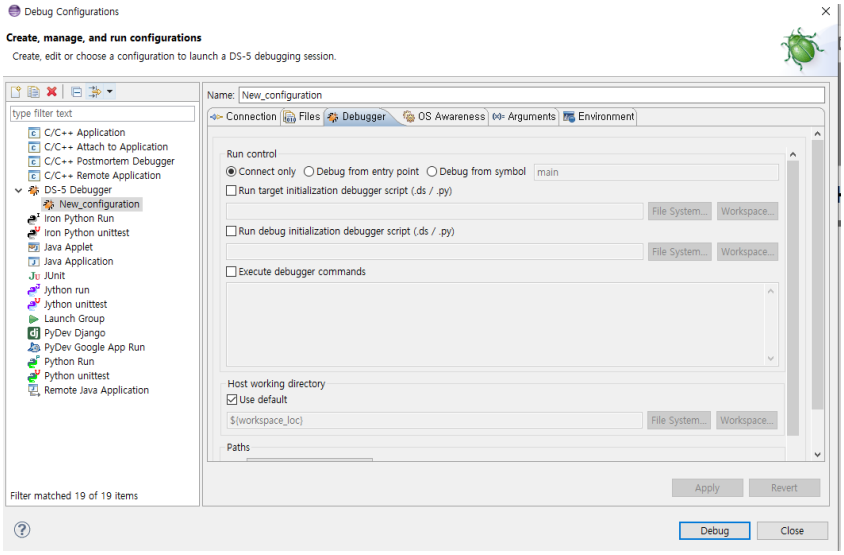







Debug Configurations
Create, manage, and run configurations
type filter text
C/C++ Application
C/C++ Attach to Application
C/C++ Postmortem Debugger
C/C++ Remote Application
DS-5 Debugger
New_configuration
Iron Python Run
Iron Python unittest
Java Applet
Java Application
JUnit
Jython run
Jython unittest
Launch Group
PyDev Django
PyDev Google App Run
Python Run
Python unittest
Remote Java Application
Filter matched 19 of 19 items
Name: New_configuration
Connection
Files
Debugger
OS Awareness
Arguments
Environment
Select target
Select the manufacturer, board, project type and debug operation to use. Currently selected:
PSU / STM32F107VCT6 / Bare Metal Debug / Debug Cortex-M3
pandaboard.org
Phytec
PSU
STM32F107VCT6
Bare Metal Debug
Debug Cortex-M3
Target Connection
DSTREAM
DTSL Options
Edit...
Configure DSTREAM trace or other target options. Using "default" configuration options
DS-5 Debugger will connect to a DSTREAM to debug a bare metal application.
Select target platform
Connections
Bare Metal Debug
Connection
USB:003877
Browse...
Apply
Revert
Debug
Close


Debug Configurations
Create, manage, and run configurations
Create, edit or choose a configuration to launch a DS-5 debugging session.
type filter text
C/C++ Application
C/C++ Attach to Application
C/C++ Postmortem Debugger
C/C++ Remote Application
DS-5 Debugger
New_configuration
Iron Python Run
Iron Python unittest
Java Applet
Java Application
JUnit
Jython run
Jython unittest
Launch Group
PyDev Django
PyDev Google App Run
Python Run
Python unittest
Remote Java Application
Filter matched 19 of 19 items
Name: New_configuration
Connection
Files
Debugger
OS Awareness
Arguments
Environment
Target Configuration
Application on host to download:
File System...
Workspace...
Load symbols
Files
Load symbols from file
File System...
Workspace...
Apply
Revert
Debug
Close

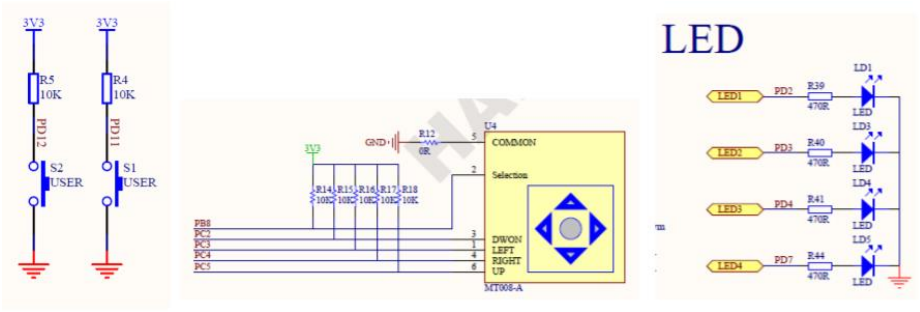

Debug Configurations
Create, manage, and run configurations
Create, edit or choose a configuration to launch a DS-5 debugging session.
type filter text
C/C++ Application
C/C++ Attach to Application
C/C++ Postmortem Debugger
C/C++ Remote Application
DS-5 Debugger
New_configuration
Iron Python Run
Iron Python unittest
Java Applet
Java Application
JUnit
Jython run
Jython unittest
Launch Group
PyDev Django
PyDev Google App Run
Python Run
Python unittest
Remote Java Application
Filter matched 19 of 19 items
Name: New_configuration
Connection
Files
Debugger
OS Awareness
Arguments
Environment
Run control
Connect only
Debug from entry point
Debug from symbol
main
Run target initialization debugger script (.ds / .py)
Run debug initialization debugger script (.ds / .py)
Execute debugger commands
Host working directory
Use default
${workspace_loc}
Paths
File System...
Workspace...
Apply
Revert
Debug
Close

04. 실험 및 과제 해결

4.1 GPIO(General purpose Input Output) 레지스터 클럭 인가

4.1.1 RCC(reset and clock control) 클럭 인가

RCC 로 사용하고자 하는 GPIO 에 클럭을 인가한다. 이 과제의 경우에는 입출력 디바이스로 User Button, 조이스틱, LED, 릴레이 모듈을 사용한다. User Button 에 연결된 레지스터는 D 포트의 11 번 레지스터, 조이스틱에 연결된 레지스터는 C 포트의 3~5 번 레지스터, LED 에 연결된 레지스터는 D 포트의 2, 7 번 레지스터, 릴레이 모듈을 사용하기 위한 레지스터는 C 포트의 8, 9 번 레지스터이다. 따라서 RCC 로 C, D 포트에 클럭을 인가해 주어야 한다. RCC 는 Datasheet 의 메모리 매핑 그림을 보면 RCC 의 시작 주소 값을 알 수 있고, 레퍼런스를 보고 RCC 의 OFFSET 값과 초기값, 연결되어 있는 포트 정보 등을 알 수 있다.



User Button : PD11

Up : PC5 **Down** : PC2, **Left** : PC3, **Right** : PC4

릴레이 모듈 사용을 위한 GPIO : PC8, PC9

LED

1 : PD2
2 : PD3
3 : PD4
4 : PD7

Figure 5. Memory map

AHB	Reserved	0x5000 0400 - 0x5FFF FFFF
	USB OTG FS	0x5000 0000 - 0x5003 FFFF
	Reserved	0x4003 0000 - 0x4FFF FFFF
	Ethernet	0x4002 8000 - 0x4002 9FFF
	Reserved	0x4002 3400 - 0x4002 7FFF
	CRC	0x4002 3000 - 0x4002 33FF
	Reserved	0x4002 2400 - 0x4002 2FFF
	Flash interface	0x4002 2000 - 0x4002 23FF
	Reserved	0x4002 1400 - 0x4002 1FFF
	RCC	0x4002 1000 - 0x4002 13FF
	Reserved	0x4002 0800 - 0x4002 0FFF
	DMA2	0x4002 0400 - 0x4002 07FF
	DMA1	0x4002 0000 - 0x4002 03FF
	Reserved	0x4001 3C00 - 0x4001 FFFF
	USART1	0x4001 3800 - 0x4001 3BFF

7.3.7 APB2 peripheral clock enable register (RCC_APB2ENR)

Address: 0x18

Reset value: 0x0000 0000

Access: word, half-word and byte access

No wait states, except if the access occurs while an access to a peripheral in the APB2 domain is on going. In this case, wait states are inserted until the access to APB2 peripheral is finished.

Note: When the peripheral clock is not active, the peripheral register values may not be readable by software and the returned value is always 0x0.

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved										TIM11 EN	TIM10 EN	TIM9 EN	Reserved		
										rw	rw	rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADC3 EN	USART 1EN	TIM8 EN	SPI1 EN	TIM1 EN	ADC2 EN	ADC1 EN	IOPG EN	IOPF EN	IOPE EN	IOPD EN	IOPC EN	IOPB EN	IOPA EN	Res.	AFIO EN
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw		rw

위의 정보들로 RCC 의 시작 주소 값 + OFFSET 을 하여 기준 주소 값을 알아낸 뒤에 C, D 포트에 각각 클럭 인가를 해준다는 0x30 이라는 값을 넣어주면 된다.

4.1.2 C 포트 클럭 인가

C 포트로 사용하고자 하는 레지스터에 클럭 인가를 해준다. C 포트는 조이스틱을 사용하기 위한 3~5 번 레지스터와 릴레이 모듈을 사용하기 위한 8, 9 번 레지스터를 사용한다. RCC 처럼 Datasheet 의 메모리 매핑 그림을 보면 C 포트의 시작 주소 값을 알 수 있고, 레퍼런스를 보고 각 포트의 GPIO 레지스터 OFFSET 값과 초기값, 연결되어 있는 레지스터 정보 등을 알 수 있다. 이때 조이스틱을 위한 C 포트는 3~5 번 레지스터를 사용하므로 GPIO_CRL(Configuration Register Low)를 봐야하고 릴레이 모듈을 위한 C 포트는 8,9 번 레지스터를 사용하므로 GPIO_CRH(Configuration Register High)를 봐야한다.

APB2	DMA1	0x4002 0000 - 0x4002 03FF
	Reserved	0x4001 3C00 - 0x4001 FFFF
	USART1	0x4001 3800 - 0x4001 3BFF
	Reserved	0x4001 3400 - 0x4001 37FF
	SPI1	0x4001 3000 - 0x4001 33FF
	TIM1	0x4001 2C00 - 0x4001 2FFF
	ADC2	0x4001 2800 - 0x4001 2BFF
	ADC1	0x4001 2400 - 0x4001 27FF
	Reserved	0x4001 1C00 - 0x4001 23FF
	Port E	0x4001 1800 - 0x4001 1BFF
	Port D	0x4001 1400 - 0x4001 17FF
	Port C	0x4001 1000 - 0x4001 13FF
	Port B	0x4001 0C00 - 0x4001 0FFF
	Port A	0x4001 0800 - 0x4001 0BFF
	EXTI	0x4001 0400 - 0x4001 07FF
	AFIO	0x4001 0000 - 0x4001 3FFF
	Reserved	0x4000 7800 - 0x4000 FFFF

9.2.1 Port configuration register low (GPIOx_CRL) (x=A..G)

Address offset: 0x00

Reset value: 0x4444 4444

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNF7[1:0]		MODE7[1:0]		CNF6[1:0]		MODE6[1:0]		CNF5[1:0]		MODE5[1:0]		CNF4[1:0]		MODE4[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNF3[1:0]		MODE3[1:0]		CNF2[1:0]		MODE2[1:0]		CNF1[1:0]		MODE1[1:0]		CNF0[1:0]		MODE0[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits 31:30, 27:26, 23:22, 19:18, 15:14, 11:10, 7:6, 3:2 **CNFy[1:0]:** Port x configuration bits (y= 0 .. 7)

These bits are written by software to configure the corresponding I/O port.

Refer to *Table 20: Port bit configuration table on page 161.*

In input mode (MODE[1:0]=00):

00: Analog mode

01: Floating input (reset state)

10: Input with pull-up / pull-down

11: Reserved

In output mode (MODE[1:0] > 00):

00: General purpose output push-pull

01: General purpose output Open-drain

10: Alternate function output Push-pull

11: Alternate function output Open-drain

Bits 29:28, 25:24, 21:20, 17:16, 13:12, 9:8, 5:4, 1:0 **MODEy[1:0]:** Port x mode bits (y= 0 .. 7)

These bits are written by software to configure the corresponding I/O port.

Refer to *Table 20: Port bit configuration table on page 161.*

00: Input mode (reset state)

01: Output mode, max speed 10 MHz.

10: Output mode, max speed 2 MHz.

11: Output mode, max speed 50 MHz.

9.2.2 Port configuration register high (GPIOx_CRH) (x=A..G)

Address offset: 0x04

Reset value: 0x4444 4444

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNF15[1:0]		MODE15[1:0]		CNF14[1:0]		MODE14[1:0]		CNF13[1:0]		MODE13[1:0]		CNF12[1:0]		MODE12[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNF11[1:0]		MODE11[1:0]		CNF10[1:0]		MODE10[1:0]		CNF9[1:0]		MODE9[1:0]		CNF8[1:0]		MODE8[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits 31:30, 27:26, 23:22, 19:18, 15:14, 11:10, 7:6, 3:2 **CNFy[1:0]:** Port x configuration bits (y= 8 .. 15)

These bits are written by software to configure the corresponding I/O port.

Refer to *Table 20: Port bit configuration table on page 161.*

In input mode (MODE[1:0]=00):

00: Analog mode

01: Floating input (reset state)

10: Input with pull-up / pull-down

11: Reserved

In output mode (MODE[1:0] > 00):

00: General purpose output push-pull

01: General purpose output Open-drain

10: Alternate function output Push-pull

11: Alternate function output Open-drain

Bits 29:28, 25:24, 21:20, 17:16, 13:12, 9:8, 5:4, 1:0 **MODEy[1:0]:** Port x mode bits (y= 8 .. 15)

These bits are written by software to configure the corresponding I/O port.

Refer to *Table 20: Port bit configuration table on page 161.*

00: Input mode (reset state)

01: Output mode, max speed 10 MHz.

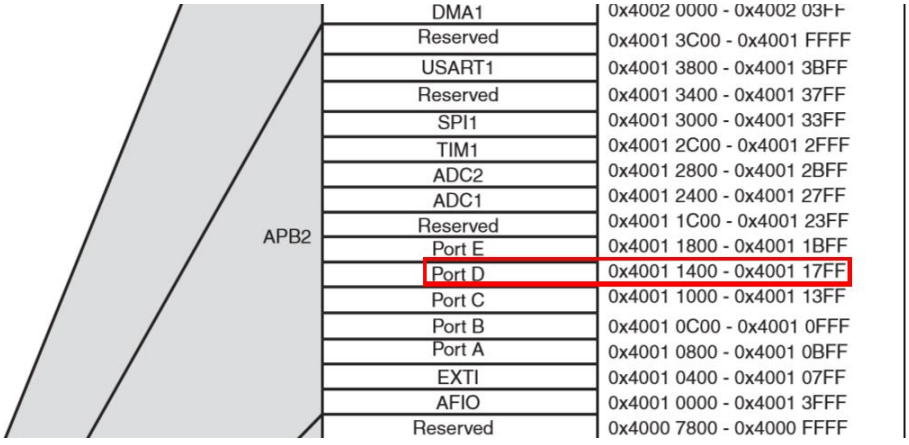
10: Output mode, max speed 2 MHz.

11: Output mode, max speed 50 MHz.

위의 정보들로 조이스틱은 C 포트의 시작 주소 값 + OFFSET 을 하여 기준 주소 값을 알아낸 뒤에 3~5, 8, 9 번 레지스터에 클럭 인가를 해준다. 이때 조이스틱은 보드기준으로 input 의 동작하고 릴레이 모듈은 output 의 동작한다. 따라서 조이스틱의 경우에는 mode 는 00 이고 CNF 값은 10 이 되므로 C 포트의 CRL 에 0x00888000 이라는 값으로, 릴레이 모듈의 경우에는 mode 는 11 이고 CNF 값은 00 이 되므로 C 포트의 CRH 에 0x00000033 이라는 값을 넣어주어서 클럭인가를 하면 된다.

4.1.4 D 포트 클럭 인가

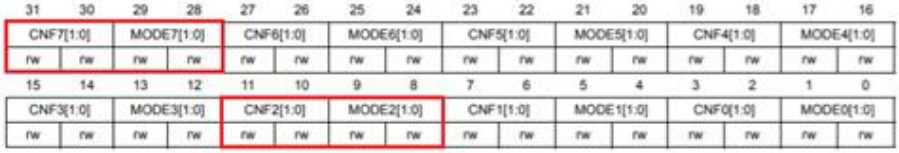
D 포트로 사용하고자 하는 레지스터에 클럭 인가를 해준다. 이번 실험에서는 LED 1, 4 번이 사용되므로 D 포트는 LED 에 연결된 2, 7 번 레지스터와 User Button 을 위한 11 번 레지스터를 사용한다. RCC 처럼 Datasheet 의 메모리 매핑 그림을 보면 D 포트의 시작 주소 값을 알 수 있고, 레퍼런스를 보고 각 포트의 GPIO 레지스터 OFFSET 값과 초기값, 연결되어 있는 레지스터 정보 등을 알 수 있다. 이때 LED 를 위한 D 포트는 2, 7 번 레지스터를 사용하므로 GPIO_CRL(Configuration Register Low)를 봐야하고 User Button 을 위한 D 포트는 11 번 레지스터를 사용하므로 GPIO_CRH(Configuration Register High)를 봐야한다.



9.2.1 Port configuration register low (GPIOx_CRL) (x=A..G)

Address offset: 0x00

Reset value: 0x4444 4444



31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNF7[1:0]		MODE7[1:0]		CNF6[1:0]		MODE6[1:0]		CNF5[1:0]		MODE5[1:0]		CNF4[1:0]		MODE4[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNF3[1:0]		MODE3[1:0]		CNF2[1:0]		MODE2[1:0]		CNF1[1:0]		MODE1[1:0]		CNF0[1:0]		MODE0[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

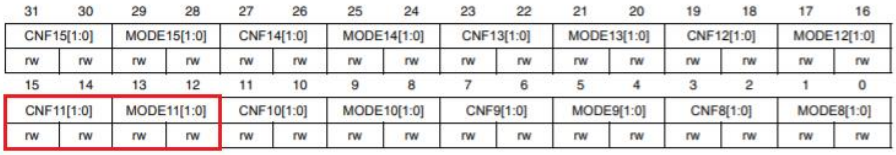
Bits 31:30, 27:26, 23:22, 19:18, 15:14, 11:10, 7:6, 3:2 **CNFy[1:0]:** Port x configuration bits (y= 0 .. 7)
These bits are written by software to configure the corresponding I/O port.
Refer to *Table 20: Port bit configuration table on page 161.*
In input mode (MODE[1:0]=00):
00: Analog mode
01: Floating input (reset state)
10: Input with pull-up / pull-down
11: Reserved
In output mode (MODE[1:0] > 00):
00: General purpose output push-pull
01: General purpose output Open-drain
10: Alternate function output Push-pull
11: Alternate function output Open-drain

Bits 29:28, 25:24, 21:20, 17:16, 13:12, 9:8, 5:4, 1:0 **MODEy[1:0]:** Port x mode bits (y= 0 .. 7)
These bits are written by software to configure the corresponding I/O port.
Refer to *Table 20: Port bit configuration table on page 161.*
00: Input mode (reset state)
01: Output mode, max speed 10 MHz.
10: Output mode, max speed 2 MHz.
11: Output mode, max speed 50 MHz.

9.2.2 Port configuration register high (GPIOx_CRH) (x=A..G)

Address offset: 0x04

Reset value: 0x4444 4444



31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNF15[1:0]		MODE15[1:0]		CNF14[1:0]		MODE14[1:0]		CNF13[1:0]		MODE13[1:0]		CNF12[1:0]		MODE12[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNF11[1:0]		MODE11[1:0]		CNF10[1:0]		MODE10[1:0]		CNF9[1:0]		MODE9[1:0]		CNF8[1:0]		MODE8[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits 31:30, 27:26, 23:22, 19:18, 15:14, 11:10, 7:6, 3:2 **CNFy[1:0]:** Port x configuration bits (y= 8 .. 15)
These bits are written by software to configure the corresponding I/O port.
Refer to *Table 20: Port bit configuration table on page 161.*
In input mode (MODE[1:0]=00):
00: Analog mode
01: Floating input (reset state)
10: Input with pull-up / pull-down
11: Reserved
In output mode (MODE[1:0] > 00):
00: General purpose output push-pull
01: General purpose output Open-drain
10: Alternate function output Push-pull
11: Alternate function output Open-drain

Bits 29:28, 25:24, 21:20, 17:16, 13:12, 9:8, 5:4, 1:0 **MODEy[1:0]:** Port x mode bits (y= 8 .. 15)
These bits are written by software to configure the corresponding I/O port.
Refer to *Table 20: Port bit configuration table on page 161.*
00: Input mode (reset state)
01: Output mode, max speed 10 MHz.
10: Output mode, max speed 2 MHz.
11: Output mode, max speed 50 MHz.

위의 정보들로 D 포트의 시작 주소 값 + OFFSET 을 하여 기준 주소 값을 알아낸 뒤에 2, 7, 11 번 레지스터에 클럭 인가를 해준다. 이때 LED 는 보드기준으로 output 의 동작을 하고 User Button 은 input 의 동작을 한다. 따라서 LED 를 위한 D 포트는 mode 를 00 이외의 값들 중에 11 로 하고, CNF 값은 00 이 되므로 D 포트의 CRL 에 0x30000300 이라는 값을 넣어주고 User Button 을 위한 D 포트는 mode 를 00 으로 하고, CNF 값은 10 이 되므로 D 포트의 CRH 에 0x00008000 이라는 값을 넣어주어서 클럭인가를 하면 된다.

4.2 GPIO(General purpose Input Output) 조작

4.2.1 IDR(Input Data Register) 조작

IDR 에는 조이스틱에 연결된 C 포트의 레지스터와 User Button 에 연결된 D 포트의 레지스터에 Input 값을 저장한다. IDR 에는 write 는 할 수 없고 read 만 할 수 있다. IDR 에 들어오는 값을 읽어온 뒤 차를 이용해서 조이스틱과 버튼의 조작을 인식하는 것에 사용할 수 있다. 그러나 이번 실험에서는 3 주차에서처럼 차를 이용하여 코딩을 했을 때 동작이 되지 않아서 IDR 로 값을 읽어와 비트를 &연산을 한 뒤 비교하여 동작을 인식하도록 하였다. IDR 의 주소 값은 위의 경우처럼 각 포트의 시작 주소 값에 OFFSET 를 더하는 것으로 구할 수 있다.

9.2.3 Port input data register (GPIOx_IDR) (x=A..G)

Address offset: 0x08h

Reset value: 0x0000 XXXX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IDR15	IDR14	IDR13	IDR12	IDR11	IDR10	IDR9	IDR8	IDR7	IDR6	IDR5	IDR4	IDR3	IDR2	IDR1	IDR0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bits 31:16 Reserved, must be kept at reset value.

Bits 15:0 **IDRy:** Port input data (y= 0 .. 15)
These bits are read only and can be accessed in Word mode only. They contain the input value of the corresponding I/O port.

4.2.2 BSRR(Bit Set/Reset Register) 조작

BSRR 로는 레지스터의 값을 세팅하거나 리셋 할 수 있다. 0~15 번 비트로는 셋을 16~31 번 비트로는 리셋을 하는데 같은 레지스터의 위치에 셋과 리셋에 값을 동시에 1 로 값을 넣어줄 경우 충돌하게 된다. 따라서 세팅을 할 때만 BSRR 을 이용하는 것이 좋고, 리셋을 할 때는 다음 항목에 나올 BRR 을 이용하는 것이 좋다. BSRR 로 C 포트의 릴레이 모듈과 연결된 8, 9 번 레지스터와 D 포트의 LED 와 연결된 2, 7 번 레지스터에 값을 넣음으로써 릴레이 모듈과 LED 점등 조작이 가능하다. 그리고 BSRR 의 주소 값은 위의 경우처럼 포트의 시작 주소 값에 OFFSET 을 더하는 것으로 구할 수 있다.

9.2.5 Port bit set/reset register (GPIOx_BSRR) (x=A..G)

Address offset: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BR15	BR14	BR13	BR12	BR11	BR10	BR9	BR8	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BS15	BS14	BS13	BS12	BS11	BS10	BS9	BS8	BS7	BS6	BS5	BS4	BS3	BS2	BS1	BS0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Bits 31:16 **BRy:** Port x Reset *bit y (y= 0 .. 15)*
These bits are write-only and can be accessed in Word mode only.
0: No action on the corresponding ODRx bit
1: Reset the corresponding ODRx bit
Note: If both BSx and BRx are set, BSx has priority.

Bits 15:0 **BSy:** Port x *Set bit y (y= 0 .. 15)*
These bits are write-only and can be accessed in Word mode only.
0: No action on the corresponding ODRx bit
1: Set the corresponding ODRx bit

↳ C 포트의 BSRR

9.2.5 Port bit set/reset register (GPIOx_BSRR) (x=A..G)

Address offset: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BR15	BR14	BR13	BR12	BR11	BR10	BR9	BR8	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BS15	BS14	BS13	BS12	BS11	BS10	BS9	BS8	BS7	BS6	BS5	BS4	BS3	BS2	BS1	BS0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Bits 31:16 **BRy:** Port x Reset *bit y (y= 0 .. 15)*
These bits are write-only and can be accessed in Word mode only.
0: No action on the corresponding ODRx bit
1: Reset the corresponding ODRx bit
Note: If both BSx and BRx are set, BSx has priority.

Bits 15:0 **BSy:** Port x *Set bit y (y= 0 .. 15)*
These bits are write-only and can be accessed in Word mode only.
0: No action on the corresponding ODRx bit
1: Set the corresponding ODRx bit

↳ D 포트의 BSRR

4.2.3 BRR(Bit Reset Register) 조작

BRR 로 레지스터의 값을 리셋 할 수 있다. BSRR 에서 같은 레지스터의 위치에 셋과 리셋에 값을 동시에 1 로 값을 넣어줄 경우 충돌하게 된다는 문제점 때문에 리셋을 할 때는 BRR 을 이용하는 것이 좋다. BRR 을 쓰게 될 경우 장점이 하나 더 있는데 BSRR 에 넣었던 값을 그대로 BRR 에 넣어주면 쉽게 리셋이 가능하다. BRR 로 C 포트의 릴레이 모듈과 연결된 8, 9 번 레지스터와 D 포트의 LED 와 연결된 2, 7 번 레지스터에 값을 넣음으로써 릴레이 모듈 멈춤과 LED 소등 조작이 가능하다. 그리고 BRR 의 주소 값은 위의 경우처럼 포트의 시작 주소 값에 OFFSET 을 더하는 것으로 구할 수 있다.

9.2.6 Port bit reset register (GPIOx_BRR) (x=A..G)

Address offset: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BR15	BR14	BR13	BR12	BR11	BR10	BR9	BR8	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Bits 31:16 Reserved

Bits 15:0 **BRy:** Port x Reset bit y (y= 0 .. 15)
These bits are write-only and can be accessed in Word mode only.
0: No action on the corresponding ODRx bit
1: Reset the corresponding ODRx bit

↳ C 포트의 BRR

9.2.6 Port bit reset register (GPIOx_BRR) (x=A..G)

Address offset: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BR15	BR14	BR13	BR12	BR11	BR10	BR9	BR8	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Bits 31:16 Reserved

Bits 15:0 **BRy:** Port x Reset bit y (y= 0 .. 15)

These bits are write-only and can be accessed in Word mode only.

0: No action on the corresponding ODRx bit

1: Reset the corresponding ODRx bit

↳ D 포트의 IDR

05. 실험 결과

5.1 소스코드

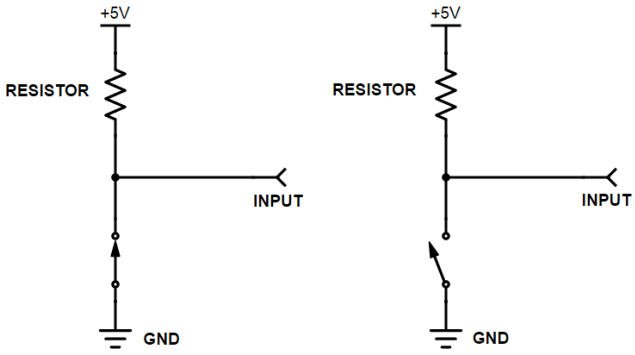
위의 GPIO 조작 내용을 바탕으로 작성한 소스코드는 아래와 같다.

```
#include "stm32f10x_gpio.h"
#include "stm32f10x.h"

#define RCC_B                   (*(volatile unsigned int*)0x40021018)

#define C_PORT_CRL        (*(volatile unsigned int*)0x40011000)
#define C_PORT_CRH              (*(volatile unsigned int*)0x40011004)
#define C_PORT_BRR              (*(volatile unsigned int*)0x40011014)
#define C_PORT_BSRR             (*(volatile unsigned int*)0x40011010)
#define C_PORT_IDR        (*(volatile unsigned int*)0x40011008)
#define D_PORT_CRL        (*(volatile unsigned int*)0x40011400)
#define D_PORT_CRH        (*(volatile unsigned int*)0x40011404)
#define D_PORT_IDR              (*(volatile unsigned int*)0x40011408)
#define D_PORT_BSRR       (*(volatile unsigned int*)0x40011410)
#define D_PORT_BRR        (*(volatile unsigned int*)0x40011414)

int main(void)
{
      int i;

      unsigned int before_C_IDR = C_PORT_IDR;
```

```
	/* 초기화 */
	RCC_B		=	0x00000000;
	C_PORT_CRL	=	0x44444444;
	D_PORT_CRL	=	0x44444444;

	/* 클럭 인가 */
	RCC_B		=	0x00000030;
	C_PORT_CRL	=	0x00888800;
	C_PORT_CRH	=0x00000033;
	D_PORT_CRL	=	0x30000300;
	D_PORT_CRH	=0x00008000;

	/* LED 불 다 꺼짐 */
	D_PORT_BRR	=	0x84; // 1000 0100

	/* 조이스틱 작동 */
	while(1){
		C_PORT_BSRR	=	0x300;
		while( !((C_PORT_IDR >> 5) &0x01) ) {// UP
			C_PORT_BRR = 0x300;
		}
		while( !((C_PORT_IDR >> 3) &0x01) ) { // LEFT
			C_PORT_BSRR = 0x100;
			C_PORT_BRR = 0x200;
		}
		while( !((C_PORT_IDR >> 4) &0x01) ) { // Right
			C_PORT_BSRR = 0x200;
			C_PORT_BRR = 0x100;
		}

		if( !((D_PORT_IDR >> 11) &0x01) ) {		// 버튼 누르면 안으로
			//버튼 때져 있으면 안으로, 버튼 누르면 밖으로 탈출
			for (i= 0;i<200000;i++);
			while( ((D_PORT_IDR >> 11) &0x01) ) {
				D_PORT_BSRR = 0x84;
				for (i= 0;i<2000000;i++);
				D_PORT_BRR	= 0x84;
				for (i= 0;i<2000000;i++);
			}
		}
	}
}
```

5.2 소스코드 해설

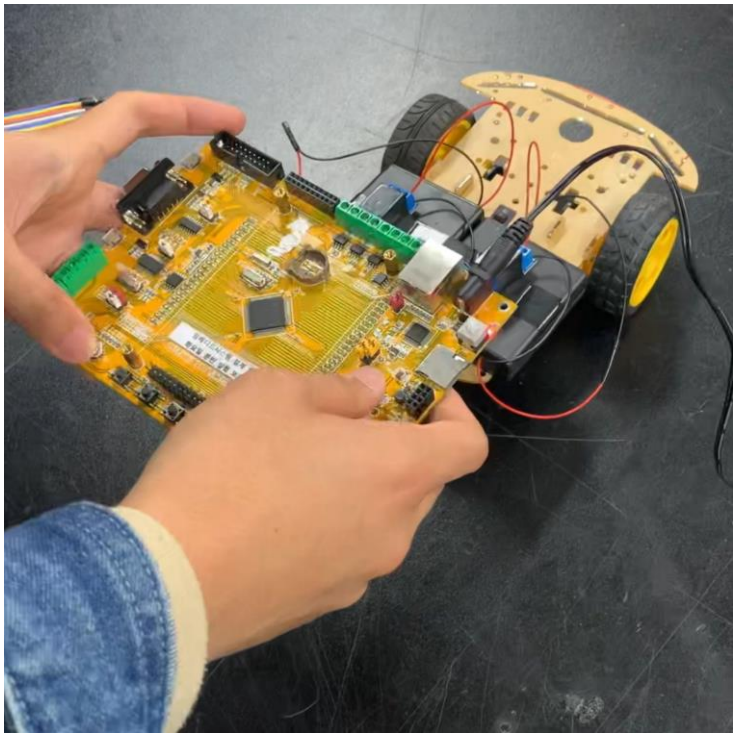
우선, 레지스터에 직접 접근해야 하므로 각 레지스터의 주소 값을 다 구해 #define 을 이용하여 변수처럼 사용한다. 그리고 초기화를 해준 뒤, 각 레지스터를 사용하겠다는 뜻을 밝히는 것으로 각 포트와 레지스터들에 클럭 인가를 해 주었다. 모든 릴레이 모듈과 LED 의 불을 소등 상태로 바꿔준 뒤, 조이스틱의 IDR 값과의 &연산에 따라 릴레이 모듈을 조작하고 User Button 의 IDR 값과의 &연산에 따라 LED 를 조작하는 코드를 작성하였다. 이때 신경 써야 할 점이 있다. 릴레이 모듈은 Pull Up 방식을 사용하여 0 일 때 스위치가 닫히고 1 일 때 스위치가 열린다. 따라서 릴레이 모듈은 BSRR 에 값을 넣어줬을 때 불이 꺼지고 BRR 에 값을 넣어줬을 때에서 불이 켜진다.



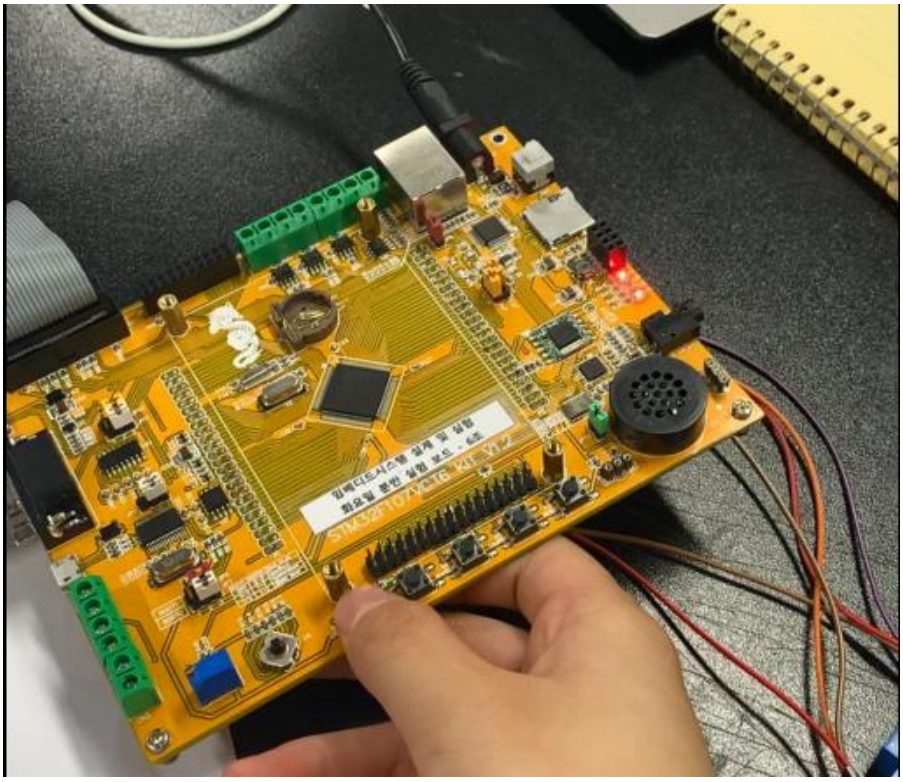
5.3 결과 확인 및 사진

해당 소스 코드대로 동작하는 프로그램은 다음과 같이 확인되었다.

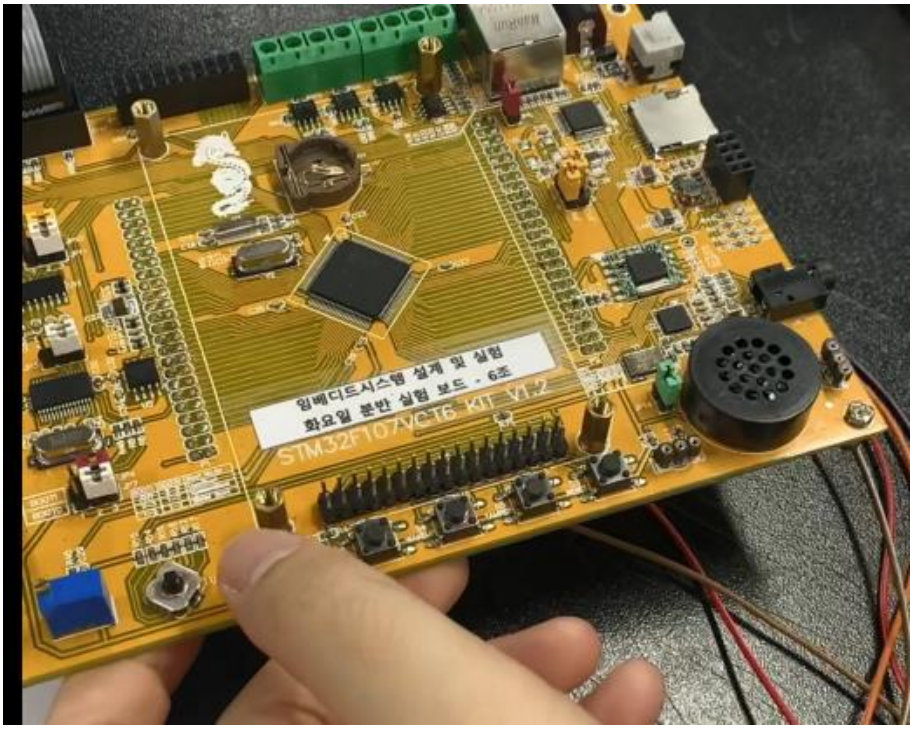
- 조이스틱 Up : Up 상태를 유지하는 동안 자동차 전진 (릴레이 모듈 8, 9 on)
- 조이스틱 Left : Left 상태를 유지하는 동안 자동차 좌회전 (릴레이 모듈 9 on)
- 조이스틱 Right : Right 상태를 유지하는 동안 자동차 우회전 (릴레이 모듈 8 on)
- 조이스틱 조작 하지 않을 때 : 자동차 정지 상태(릴레이 모듈 8, 9 off)
- User Button : User Button 눌렀을 때 LED 1, 4 번 on/off 반복(한 번 누르면 점멸 유지, 다시 누르면 off)



프로그램의 동작은 조교님께 확인 받았다.



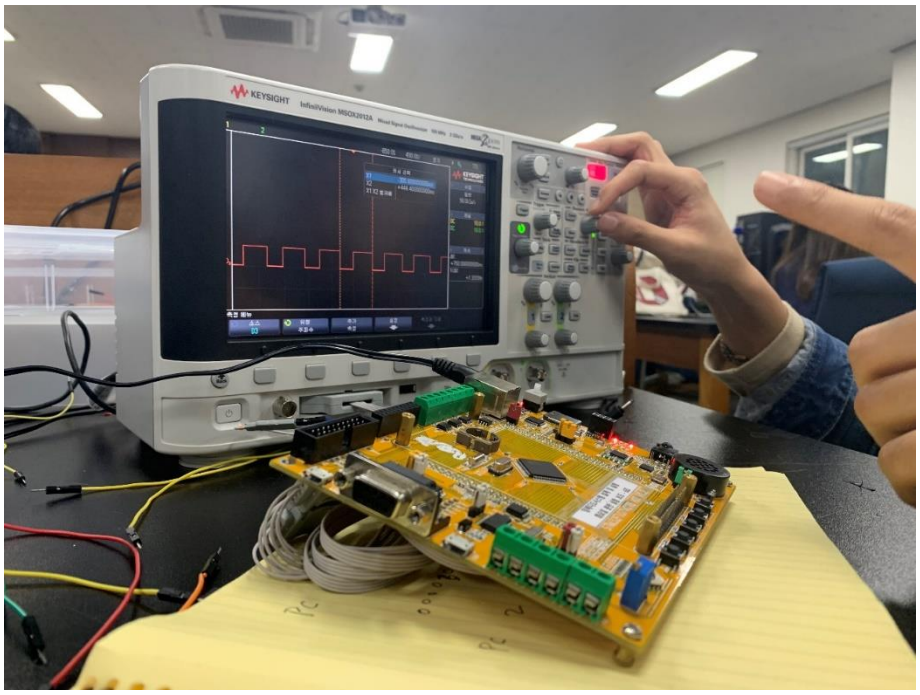
- LED 1, 4 on 사진

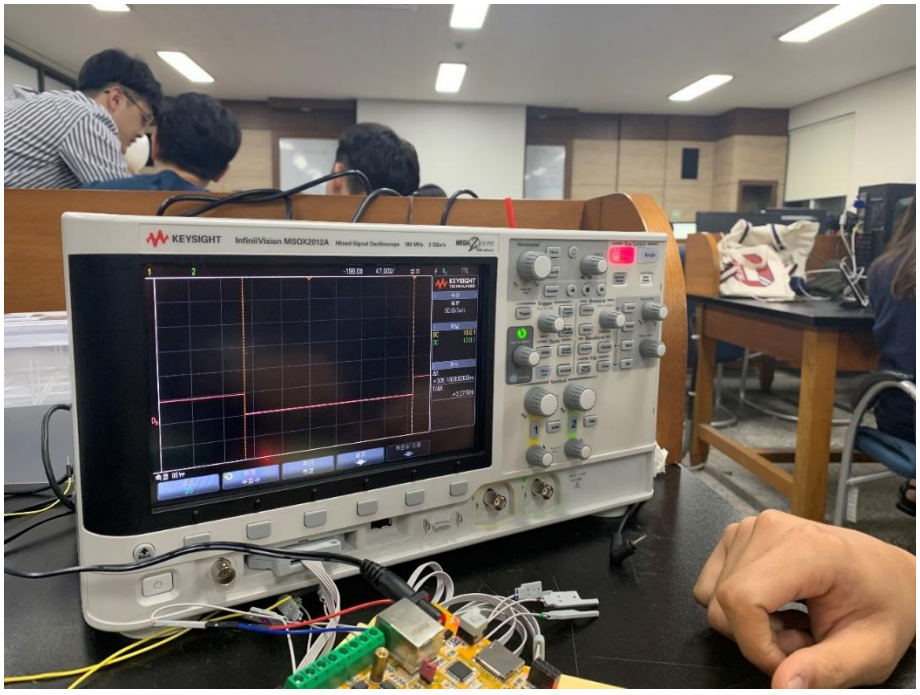


- LED 1, 4 off 사진

5.4 오실로스코프를 이용한 delay 측정

오실로스코프에서 트리거 모드: Single 을 주게 되면 트리거가 찍히게 된다. 그래프 사이의 초를 확인하려면 Measure 버튼을 눌러 x1 을 확인하고 커서를 맞춰준다. 그리고 x2 의 커서를 맞춰주게 되면 값이 대략 300ms 가 나온다. 따라서 delay 로 사용된 for (i= 0; i<2000000; i++);이 걸리는 시간은 대략 300ms 이다.





06. 결론

첫번째 실험과 예비 실험을 미리 하여 조금 더 수월하게 진행되었다. 릴레이 모듈을 사용해 보고 자동차에 연결하여 직접 움직이는 모습을 확인할 수 있었다. 그리고 첫번째 실험에서 하지 못했던 오실로스코프를 사용해 봄으로써 오실로스코프에 대해 배울 수 있었다.