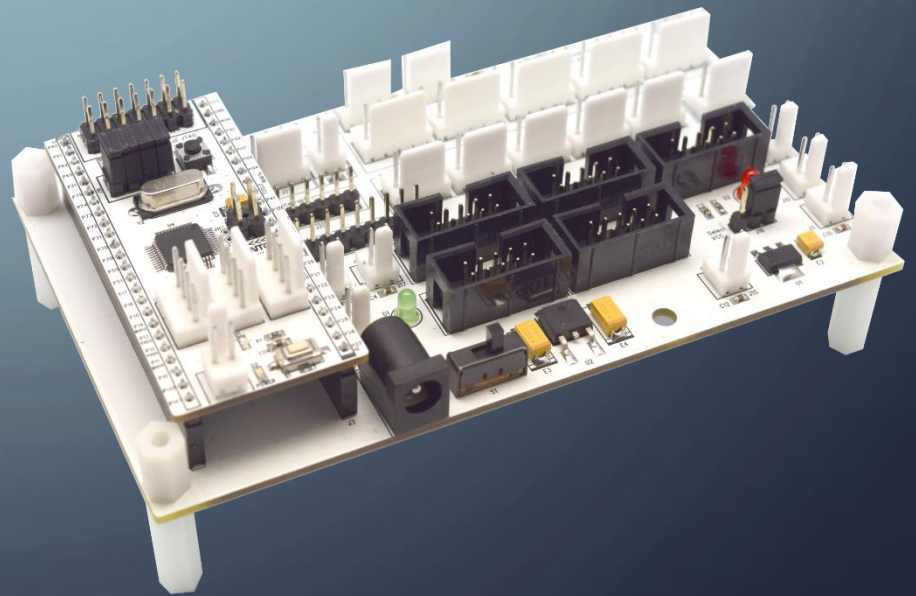


RL78/G13 새 프로젝트 생성하기

(주) 뉴티씨

대표이사 류대우





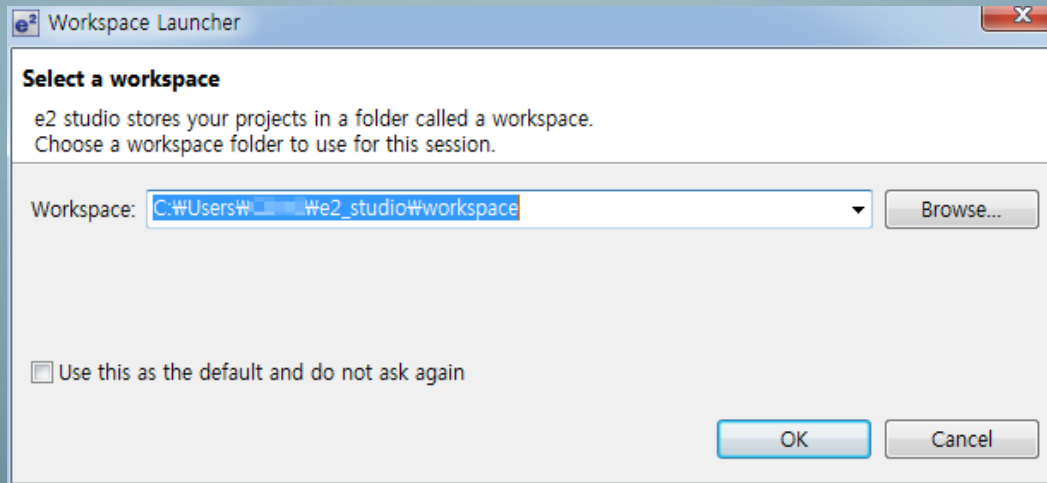
목차

1. IDE 처음 실행 설정
 2. 새 프로젝트 만들기
 3. 컴파일하여 바이너리 만들기
- 
- 

1. IDE 처음 실행 설정 개요

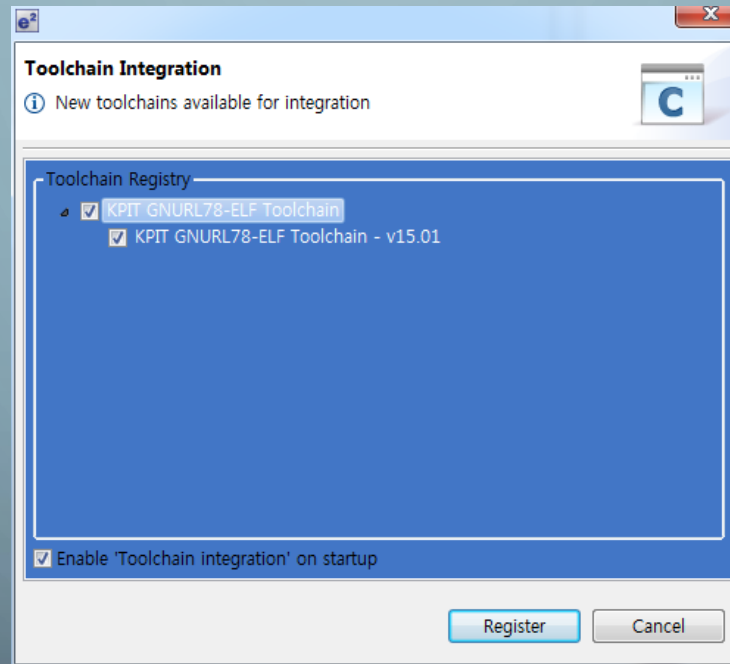
- 1) 툴체인(컴파일러 및 디버거 모음)을 e2 studio와 연동시킨다.
- 2) 코드 생성기를 컴퓨터에 등록시킨다.

1-(1)-A.작업 공간 설정



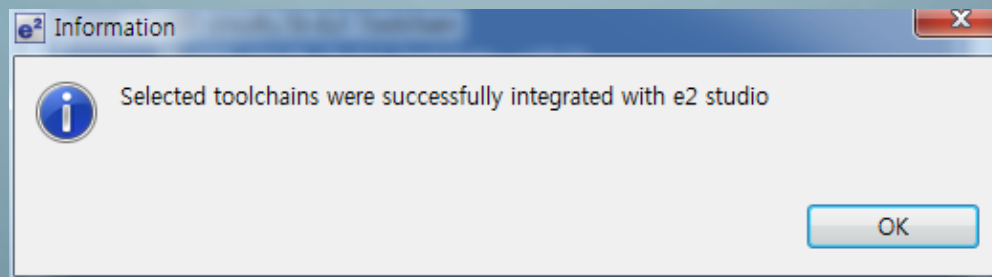
첫 작업 공간을 설정하고 OK 버튼 클릭

1-(1)-B.툴체인 등록



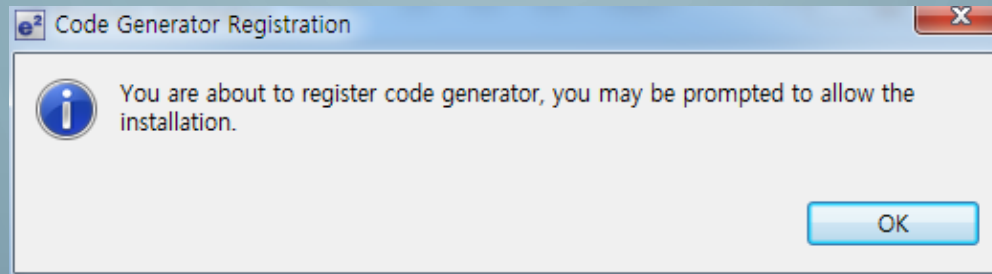
정보를 입력하고 Submit 버튼을 눌러 회원 가입

1-(1)-C.툴체인 등록



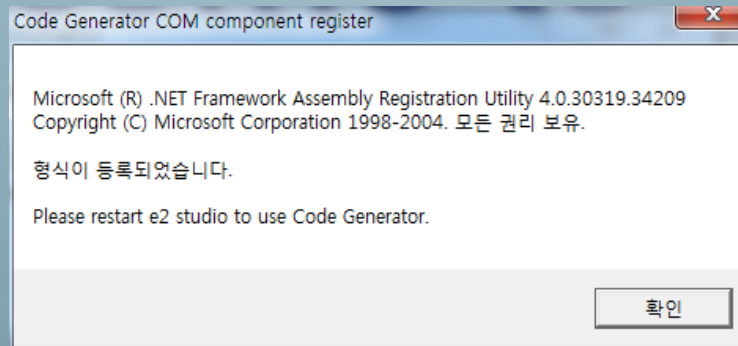
툴체인 등록 성공 메시지가 나타나면 OK 버튼 클릭

1-(2)-A. 코드 생성기 등록



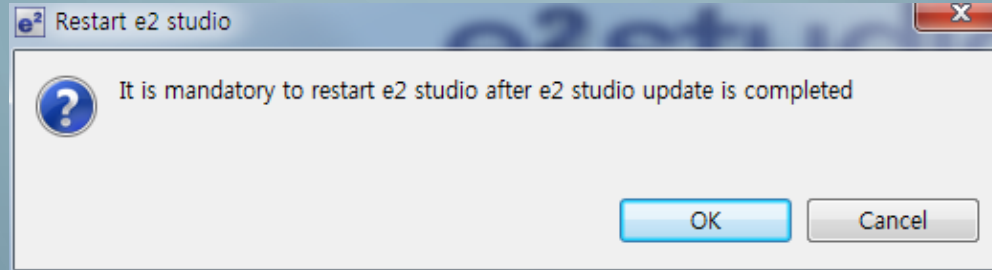
OK 버튼 클릭 후 관리자 권한
메시지가 나타나면 예(Y)를 눌러 동의

1-(2)-B.코드 생성기 등록



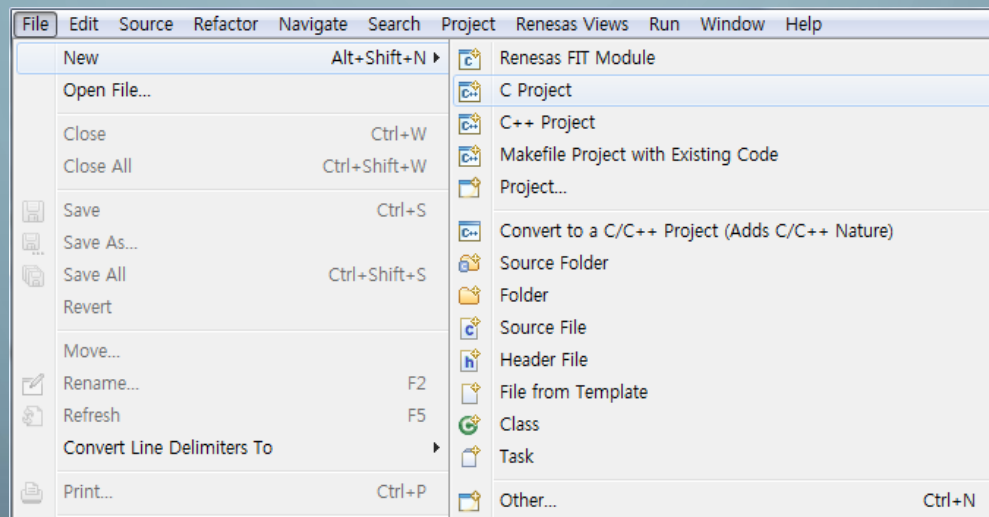
등록이 완료되면 확인 클릭

1-(2)-C.코드 생성기 등록



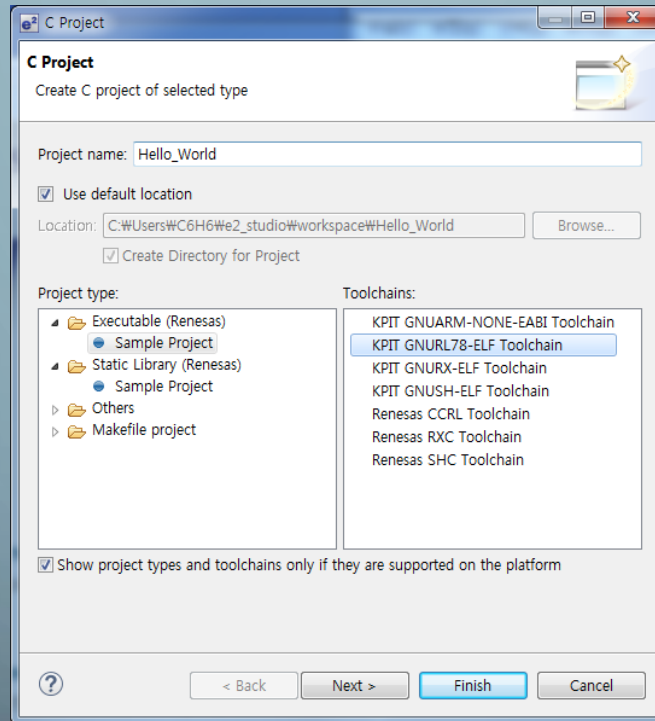
OK 버튼을 눌러 e2 studio 재시작

2-(1)-A. 새 프로젝트 생성



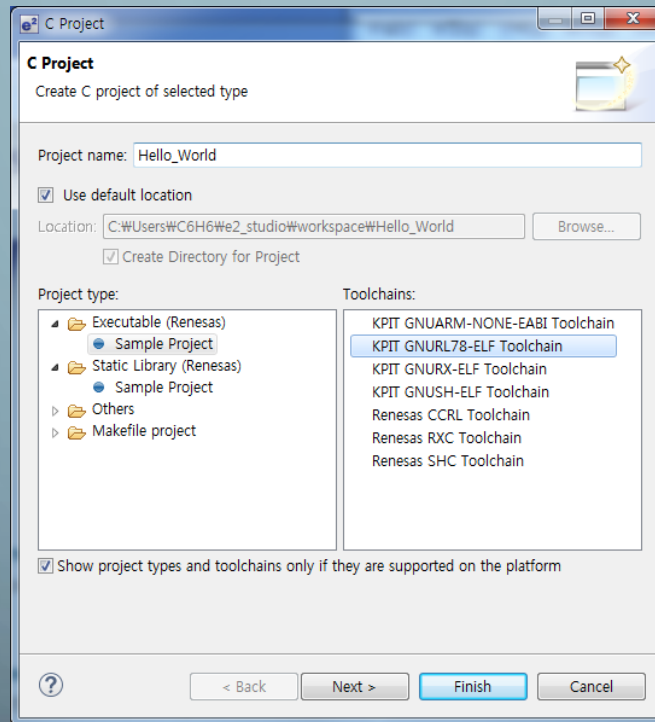
File → New → C Project 클릭

2-(1)-B. 새 프로젝트 생성



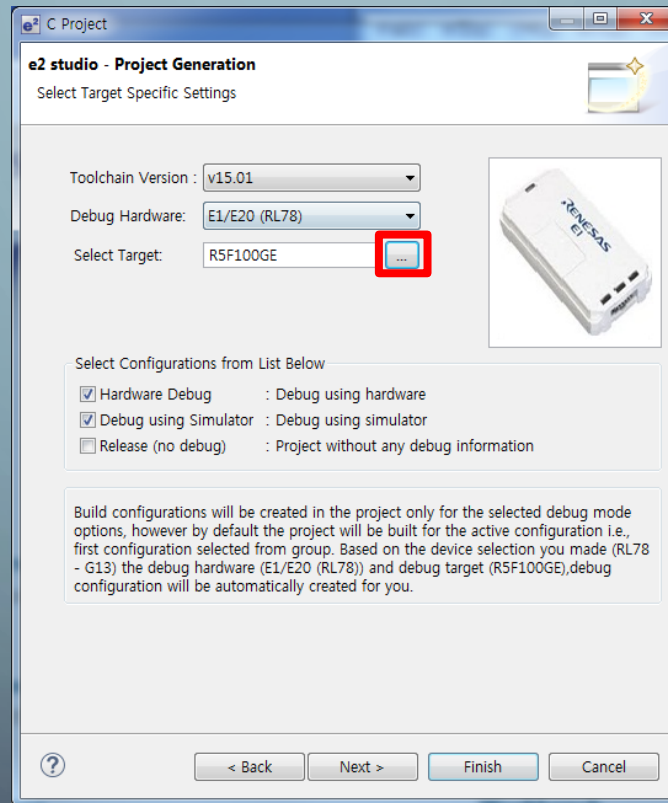
Project name → 프로젝트 명
Project type → Executable (Renesas)/Sample Project
Toolchains → KPIT GNURL78-ELF Toolchain

2-(1)-C. 새 프로젝트 생성



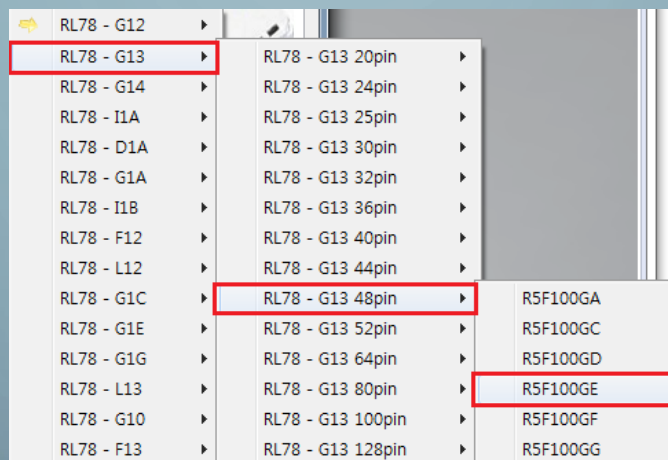
설정이 완료되면 Next 버튼 클릭

2-(1)-D. 새 프로젝트 생성



장치 지정을 위해 Select Target의 [...] 버튼 클릭

2-(1)-E. 새 프로젝트 생성



RL78 - G13 → RL78 - G13 48 pin → R5F100GE 클릭
장치가 설정되면 Next 버튼 클릭

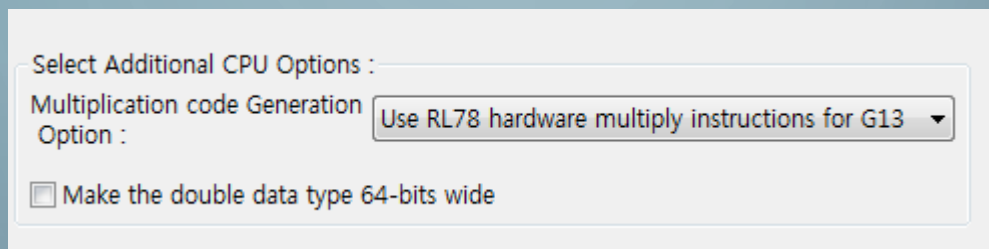
2-(1)-F. 새 프로젝트 생성

☐ Use Peripheral code Generator

The e2 studio peripheral code generator automatically generates programs (device drivers) for MCU peripheral functions (clocks, timers, serial interfaces, A/D converters, DMA controllers, etc.) based on settings entered via a graphical user interface (GUI). Functions are provided as application programming interfaces (APIs) and are not limited to initialization of peripheral functions.

코드 생성기 설정을 해제하고 Next 버튼 클릭

2-(1)-G. 새 프로젝트 생성



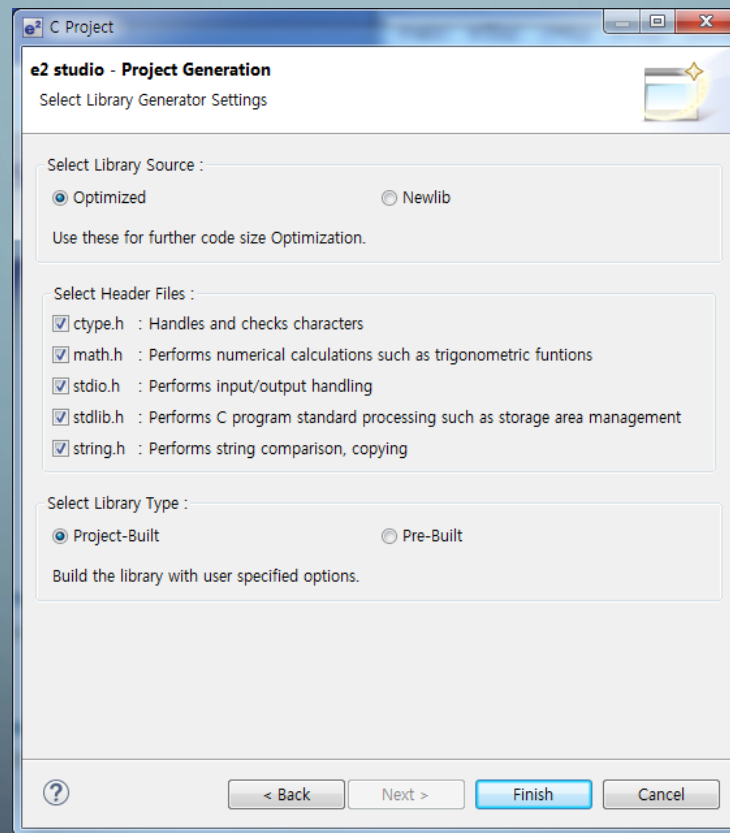
Select Additional CPU Options :

Multiplication code Generation Option : Use RL78 hardware multiply instructions for G13 ▼

☐ Make the double data type 64-bits wide

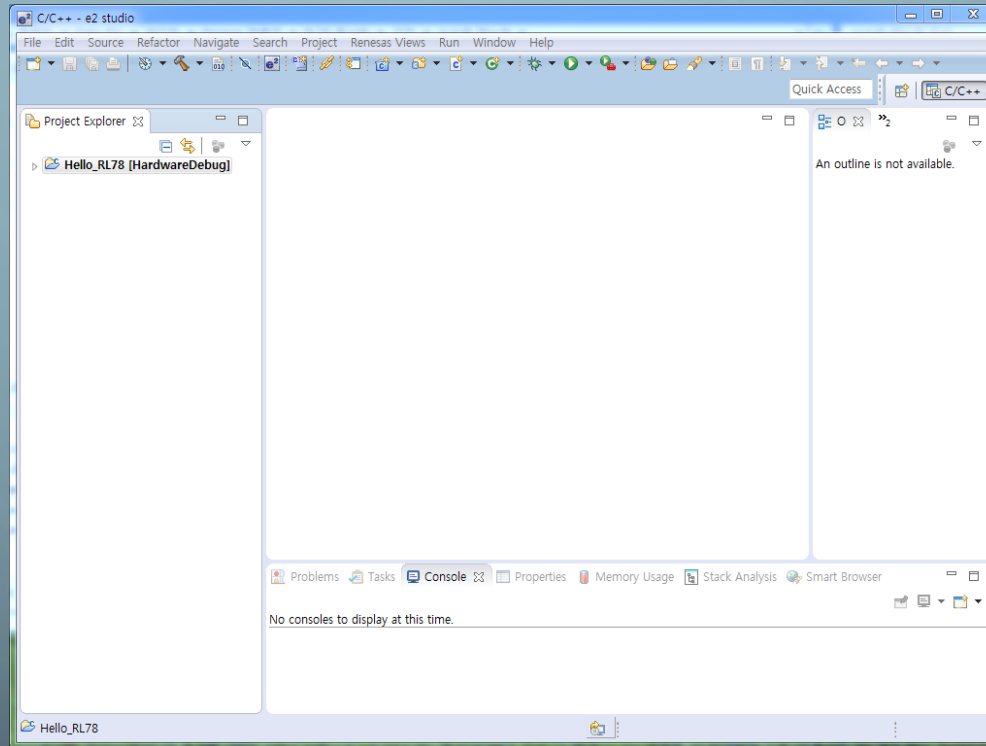
추가 CPU 옵션은 다음과 같이 설정하고 Next 버튼 클릭

2-(1)-H. 새 프로젝트 생성



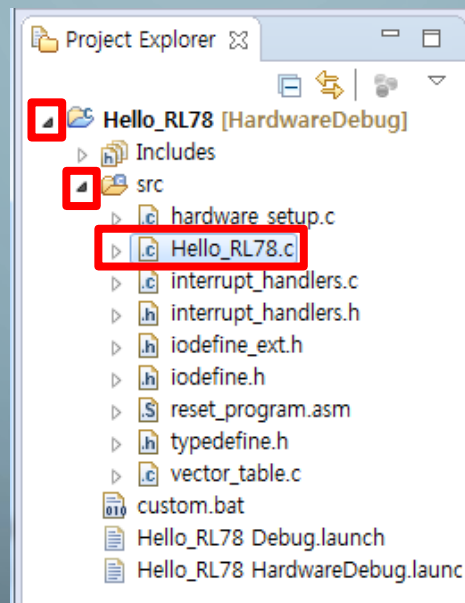
다음과 같이 설정하고 Finish 버튼을 클릭하여 프로젝트 생성

2-(1)-I. 새 프로젝트 생성



프로젝트 생성 완료

2-(2)-A. 코드 작성



프로젝트를 탐색하여 (프로젝트명).c 를 탐색

2-(2)-B. 코드 작성

```
/* PROJECT NAME : Hello_RL78 */

/* File Version: V1.00 */

#include "iodefine.h"

#ifdef CPPAPP
//Initialize global constructors
extern "C" void __main()
{
    static int initialized;
    if (! initialized)
    {
        typedef void (*pfunc) ();
        extern pfunc __ctors[];
        extern pfunc __ctors_end[];
        pfunc *p;

        initialized = 1;
        for (p = __ctors_end; p > __ctors; )
            (*--p) ();
    }
}
#endif

int main(void)
{
    // TODO: add application code here

    while (1) {
    }
    return 0;
}
```

코드 편집 창에 다음과 같은 코드가 있는지 확인

2-(2)-C. 코드 작성

```
#include "iodef.h"

void delay_ms(int ms)
{
    volatile int i,j;
    for(i=0;i<ms;i++)
    {
        for(j=0;j<2280;j++)
        {

        }
    }
}

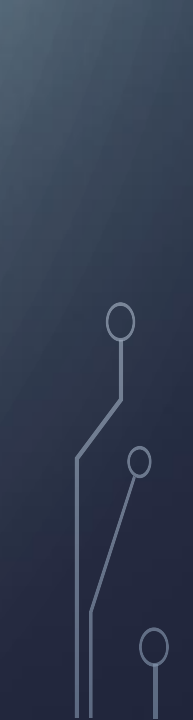
void delay_us(register int us)
{
    us<=1;
    us-=8;
    while(us)
    {
        us--;
    }
}
```

```
int main(void)
{
    // TODO: add application code here
    PM7=0x00;
    while (1)
    {
        P7=0x03;
        delay_ms(100);
        P7=~0x03;
        delay_ms(100);
    }
    return 0;
}
```

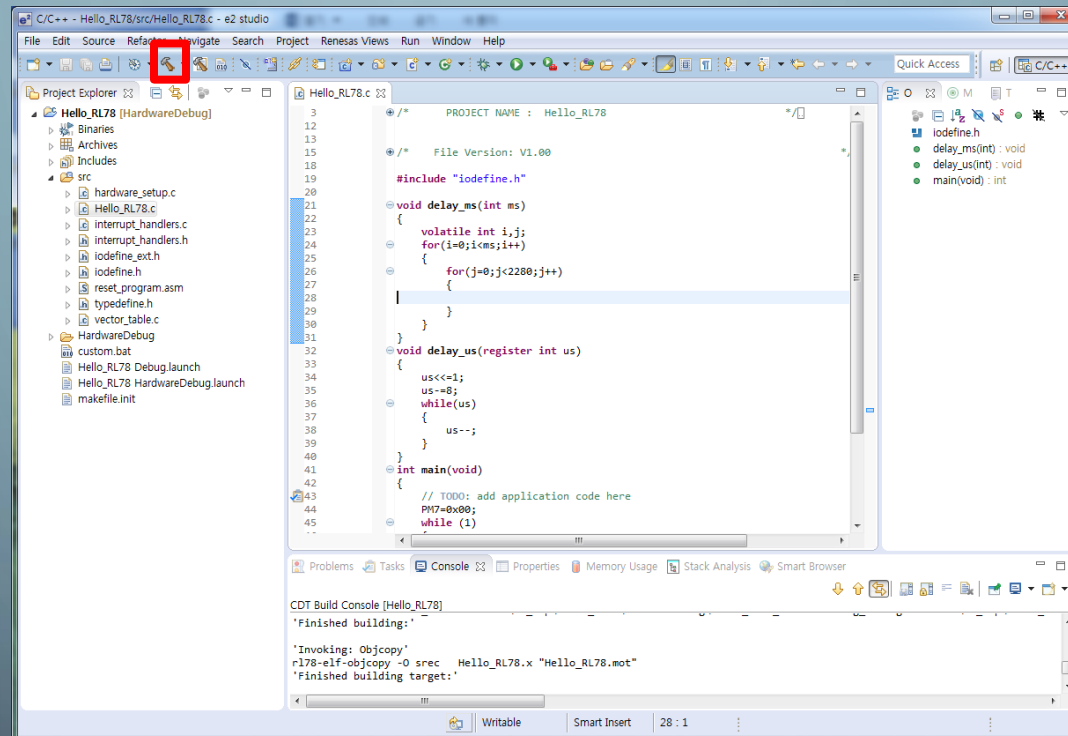
LED를 깜빡이기 위해 다음과 같은 코드를 작성



3. 컴파일 개요

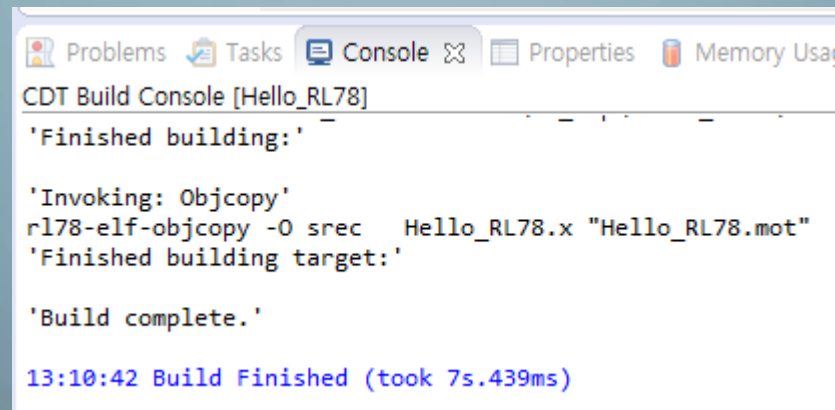
- 1) 컴파일 명령을 내려 컴파일을 수행한다.
- 
- 

3-A. 컴파일하기



망치 모양의 아이콘 () 을 클릭하거나 Ctrl + B 단축키 입력

3-B. 컴파일하기



```
Problems Tasks Console Properties Memory Usage
CDT Build Console [Hello_RL78]
'Finished building:'

'Invoking: Objcopy'
r178-elf-objcopy -O srec Hello_RL78.x "Hello_RL78.mot"
'Finished building target:'

'Build complete.'

13:10:42 Build Finished (took 7s.439ms)
```

하단 출력 창에 다음과 같이 나타나면 컴파일 성공