

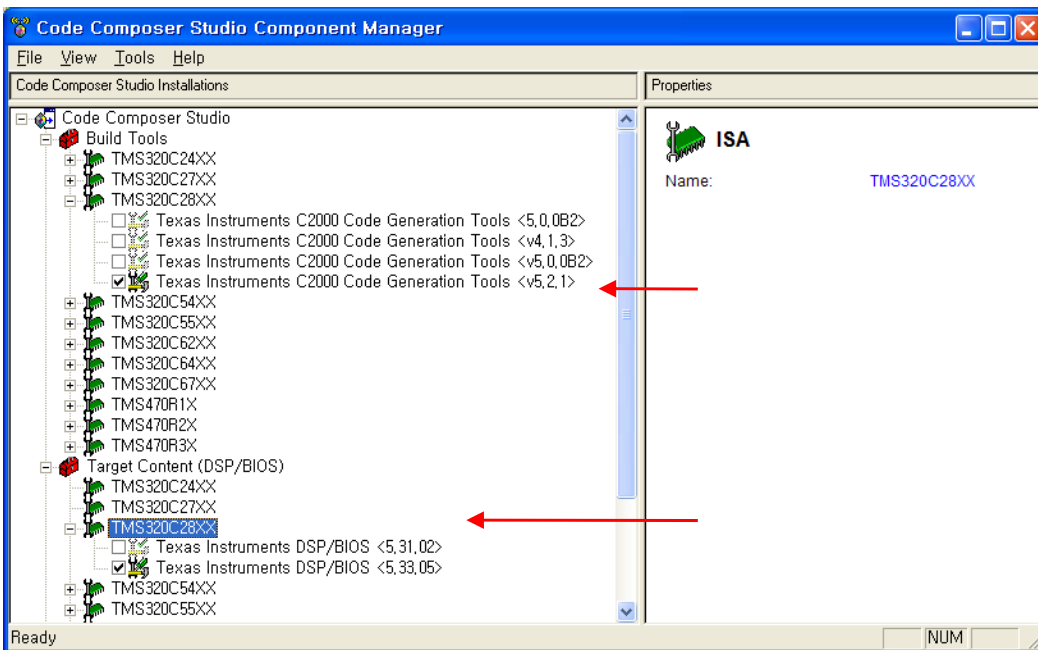
1. CCS3.3 DSP/BIOS TASK 생성

1. 디렉토리 구성

- ..Wcmd : Linker 커맨드 파일
- ..DSP2833x_headers : Chip관련 헤더 파일 및 헤더용 Linker 커맨드 파일
- ..Winclude : 사용자 인클루드 파일
- ..Wtestprj_1 : 사용자 프로젝트 파일 및 실행 파일(.HEX)
- ..Wtestsrc_1 : 사용자 소스 파일

2. CCS3.3 폴더에 설치된 DSP BIOS버전을 확인후 최신버전(5.33.xx 이상)으로 업그레이드 한다.
CCS3.3 C2000 Code Generation Tools 도 최신 버전으로 업그레이드 한다.

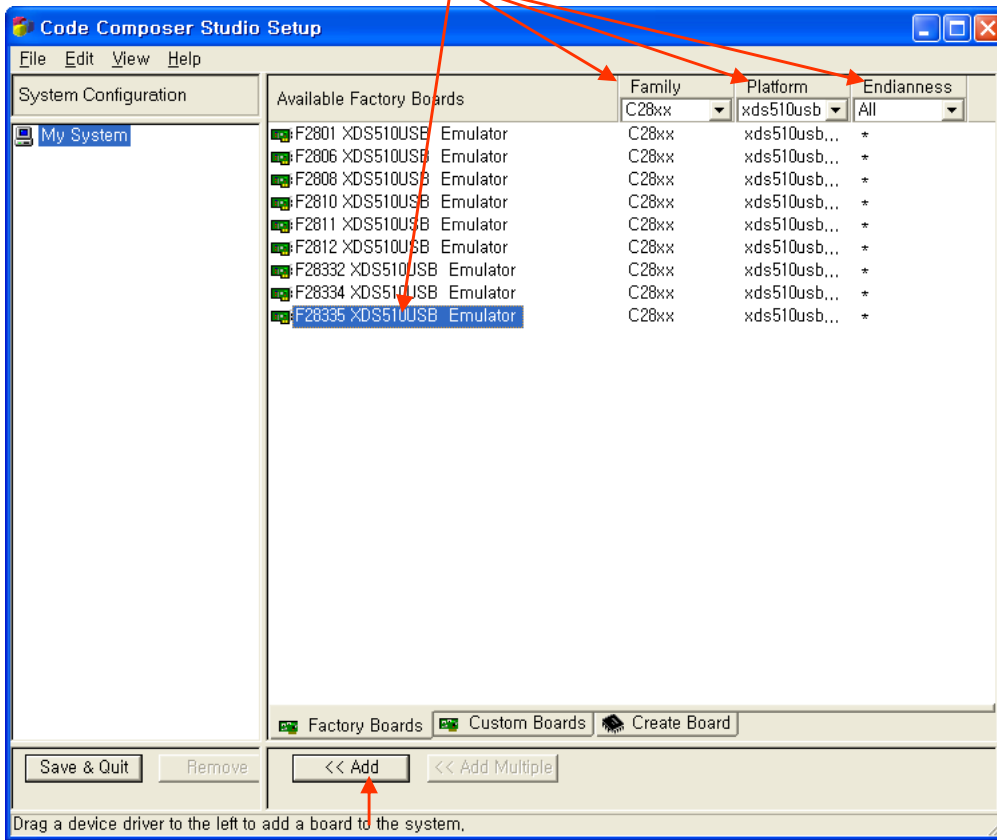
업그레이드된 BIOS버전을 Code Composer Manager에 등록 한다.

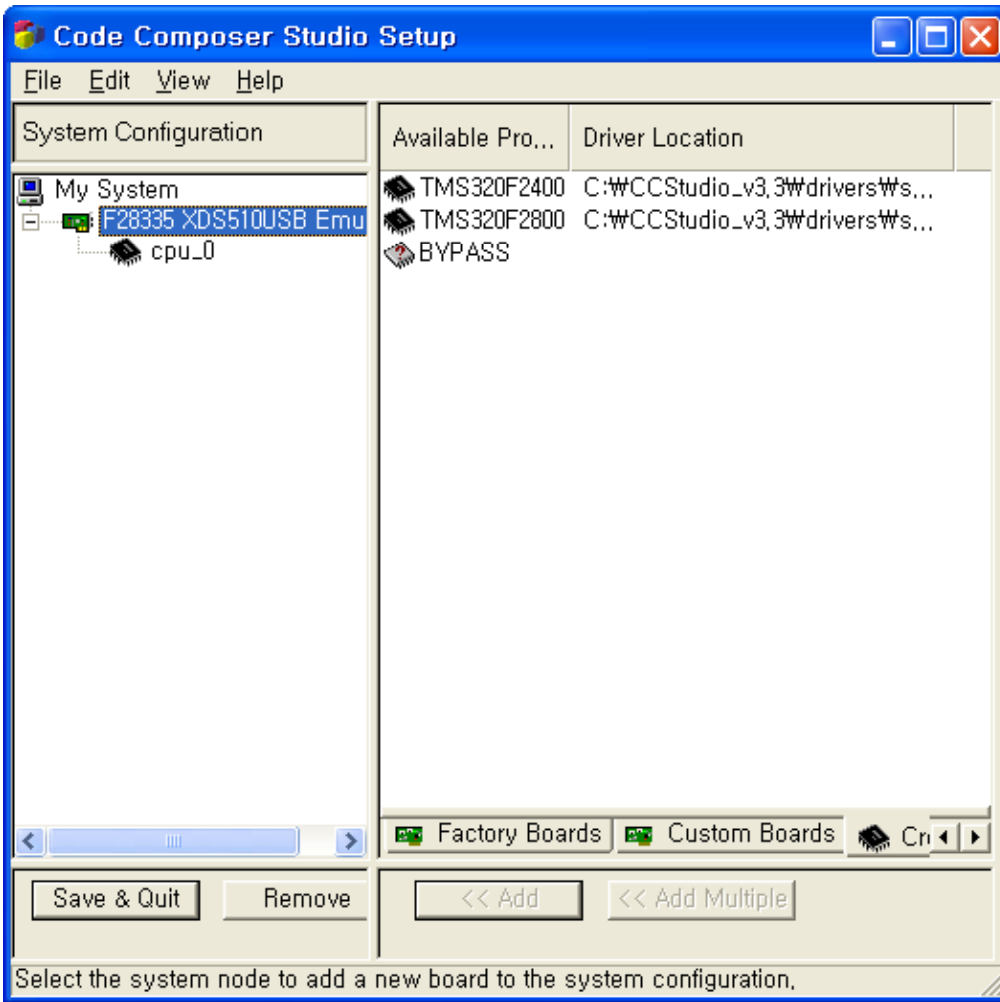


3. Setup CCStudio v3.3을 실행 합니다.

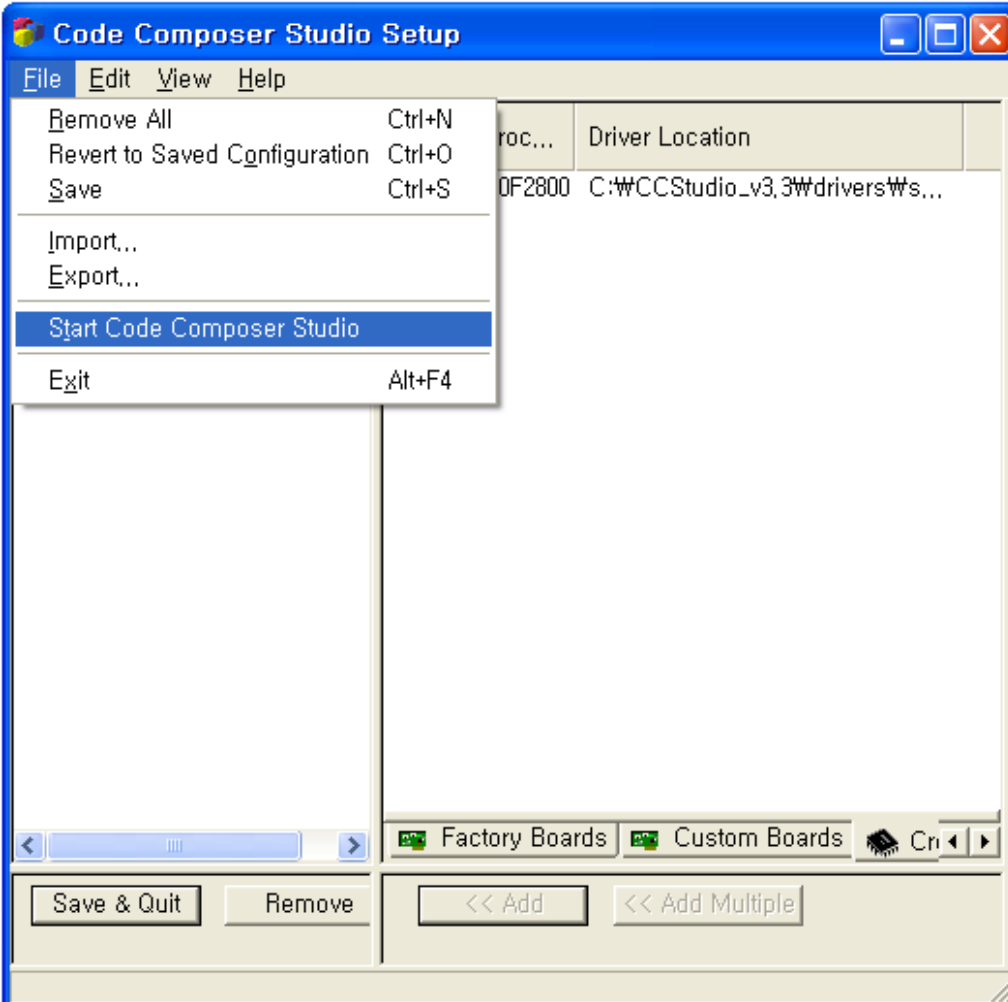


4. 프로그램 및 디버거에 사용할 장치를 선택 한후 Add버튼을 클릭 합니다.





5. My System에서 F283335 XDS510USB Emu를 선택후 Start Code Composer Studio를 실행 합니다.

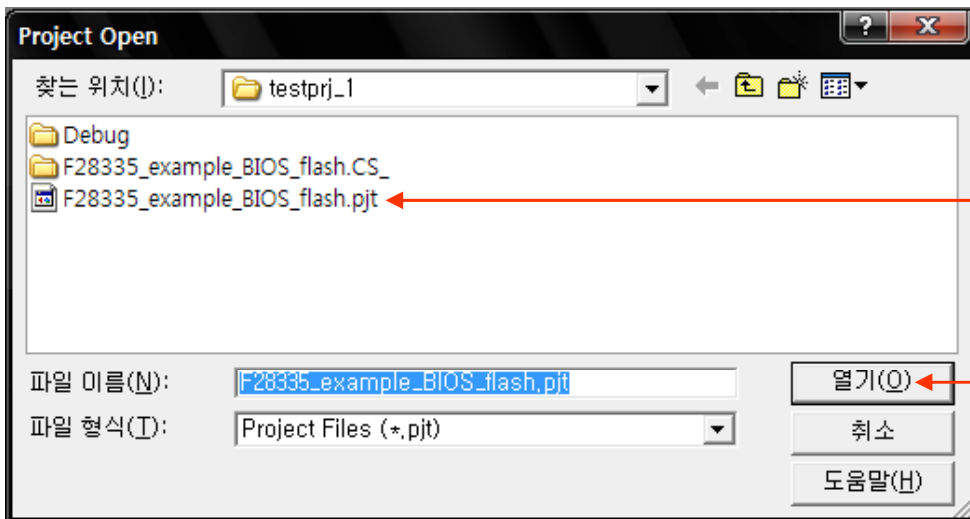
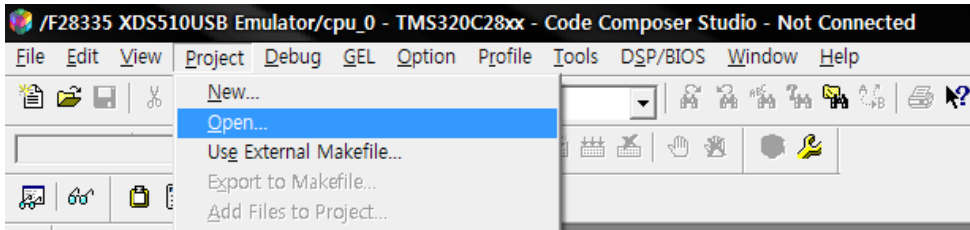


* CCS3.3 DSP/BIOS 구성 및 코드 설명

1. Setup CCStudio v3.3 이나 CCSstudio3.3을 실행 합니다.



2. 아래와 같이 Project를 오픈 합니다.(Project->Open)



3. Projects Source 파일 구성

The image shows a file explorer view of a project named 'F28335_example_BIOS_flash.pjt (Debug)'. The 'Source' folder is expanded, and several files are listed with red arrows pointing to their descriptions on the right. The descriptions explain the purpose of each file, such as 'CodeStartBranch.asm' for code entry point registration and 'Watchdog.c' for Watch-Dog related initialization.

File Name	Description
F28335_example_BIOS_flash.tcf	DSP/BIOS Config 파일
CodeStartBranch.asm	Watch-dog disable 후 C를 실행 할 때 사용, 필요시 project->build option에서 code entry point에서 등록하여 사용
DefaultIsr_BIOS.c	인터럽트 서비스 루틴
DelayUs.asm	DelayUs() 함수 지원
DSP2833x_GlobalVariableDefs.c	전역데이터 및 데이터 섹션 정의
Flash.c	Flash Memory 관련 지원 및 초기화
Gpio.c	CPU I/O핀 초기화
Main_BIOS.c	Main() 프로그램
Passwords.asm	Flash Passwords 관련
PieCtrl_BIOS.c	CPU 인터럽트 초기화
SetDBGIER.asm	BIOS 인터럽트 지원
SysCtrl.c	CPU 클럭 설정
Watchdog.c	Watch-Dog 관련 초기화
Xintf.c	외부 버스 초기화(wait 설정)
DSP2833x_Headers_BIOS.cmd	DSP/BIOS Config에서 컴파일시 생성
F28335_BIOS_flash.cmd	기본 CMD 파일 정의
F28335_example_BIOS_flashcfg.cmd	DSP/BIOS Config에서 컴파일시 생성

4. 소스코드 설명(Main_Bios.c)

```
#include "DSP2833x_Device.h" <- DSP 초기화 및 설정 관련
#include "F28335_example.h" <- 사용자 외부 함수, 변수, 정의 관리

void main(void)
{
    InitSysCtrl();           <- CPU 클럭 설정((30*10) / 2 = 150M)
    InitPieCtrl();          <- 인터럽트 관련 초기화
    InitWatchdog();         <- watch-dog 설정 및 초기화
    InitGpio();             <- CPU I/O 설정(IN,OUT,기본기능..) _EX_BUS_ON정의에 따라 외부 버스 ON
    InitXintf();           <- 내부 주변 디바이스 클럭 설정 및 외부 버스 타이밍 설정

    ** DSP/BIOS 관련 설정 **
    #ifdef EXAMPLE_FLASH
        memcpy(&secureRamFuncs_runstart,
               &secureRamFuncs_loadstart,&secureRamFuncs_loadend - &secureRamFuncs_loadstart);
        InitFlash();
    #endif

    ** DSP/BIOS에서 TINT2,DLOGINT를 사용 하므로 BIOS사용 인터럽트 허가 **
    SetDBGIER(IER | 0x6000); <- Enable everything in IER, plus TINT2 and DLOGINT
    *(volatile unsigned int *)0x00000C14 |= 0x0C00;<- Set TIMER2 FREE=SOFT=1

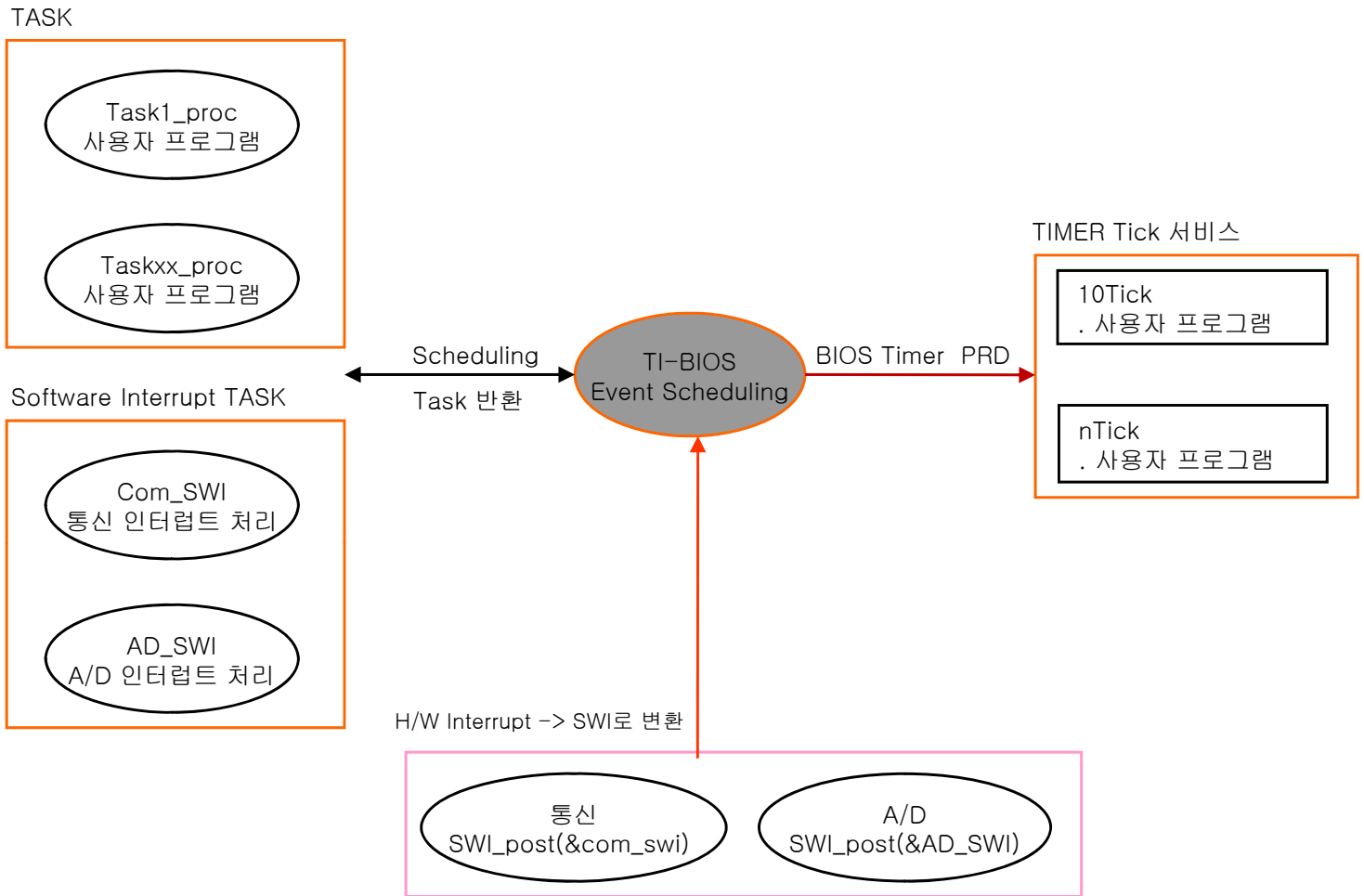
    ** 아래 main()를 종료 하면 DSP/BIOS가 동작.. **
}

void UserInit(void){
    <- 이 함수는 리셋시 DSP/BIOS 초기화 부분에서 한번 수행 후
    DSP/BIOS관련 및 사용자 초기화 함수 추가
}

}
```

- CCS3.3 DSP/BIOS TASK 생성

* TASK란 스케줄링 되는 최소 단위로서 DSP/BIOS에 등록된 우선순위에 따라 관리 된다.



1. Main_Bios.c 를 open후 아래 소스코드를 입력 후 저장한다.

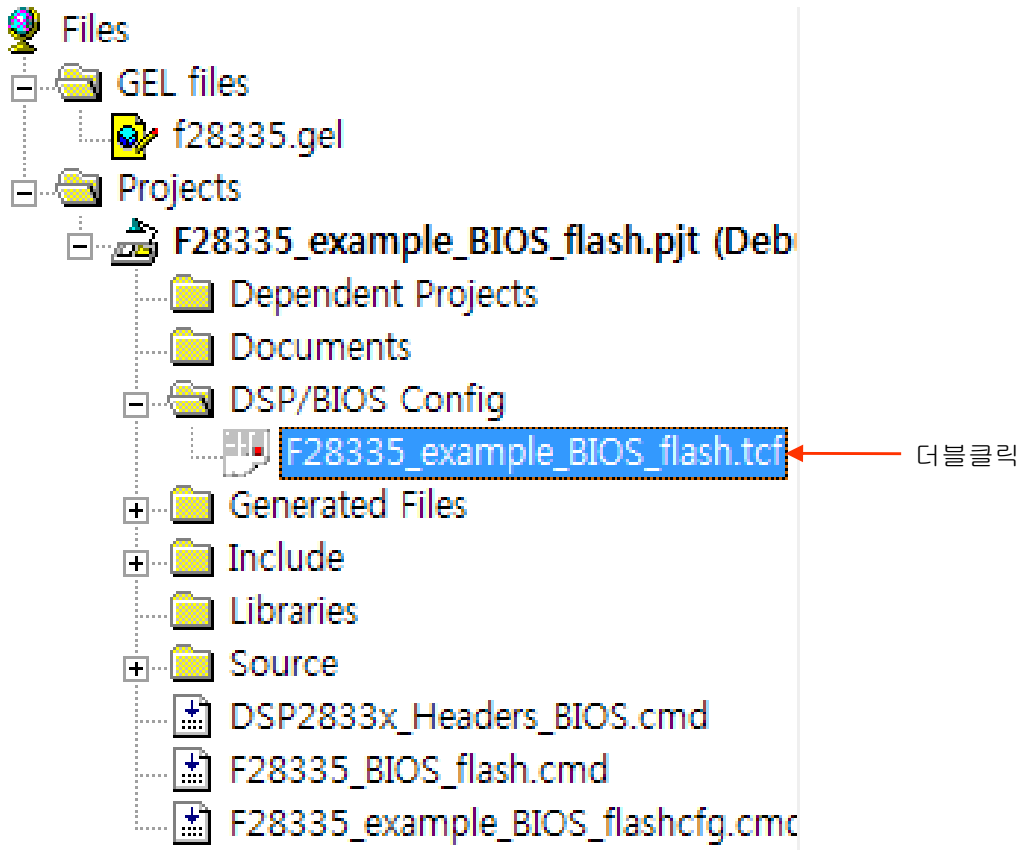
```
void task1_proc(void)
{
    while(1){
        TSK_sleep(1);
    }
}
```

<- TSK 함수 명

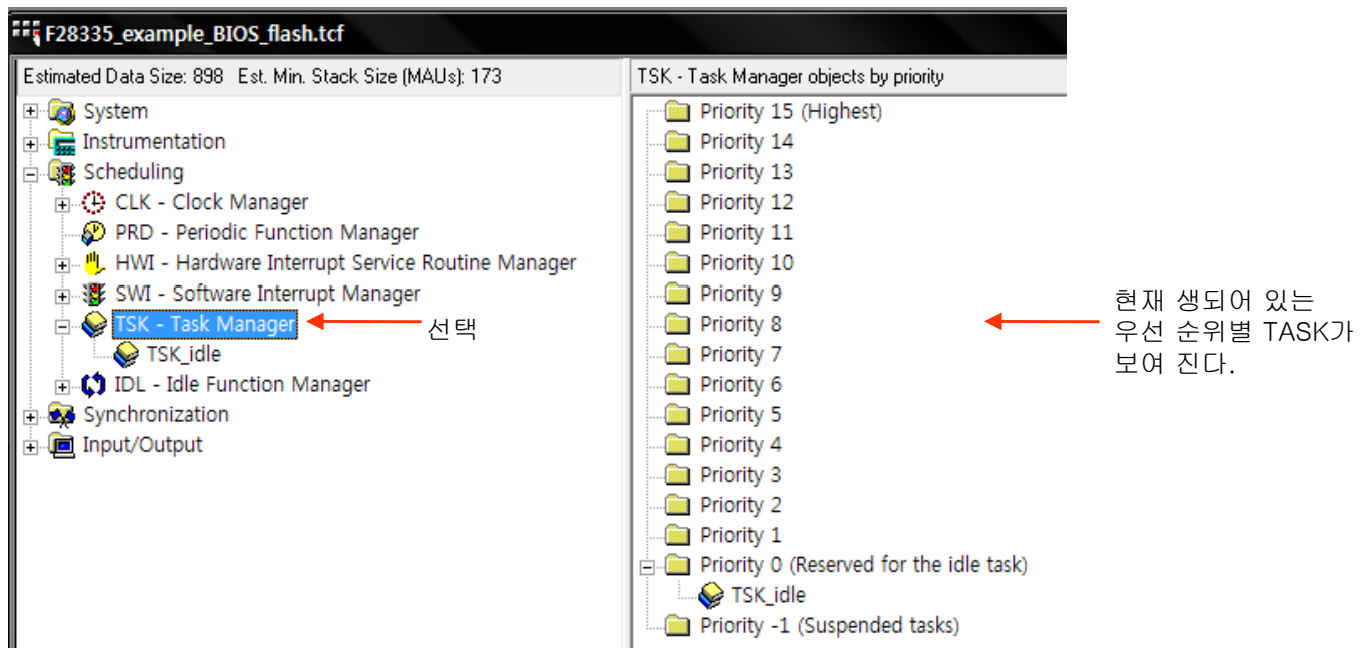
<- TASK 실행시 최초 실행 영역
변수 선언 및 초기화 작업을 할수 있다.

<- 사용자가 필요한 코드 추가 할수 있음.
<- 1TICK(1ms) 동안 Sleep모드로 전환
다른 TASK가 실행되도록 위함.

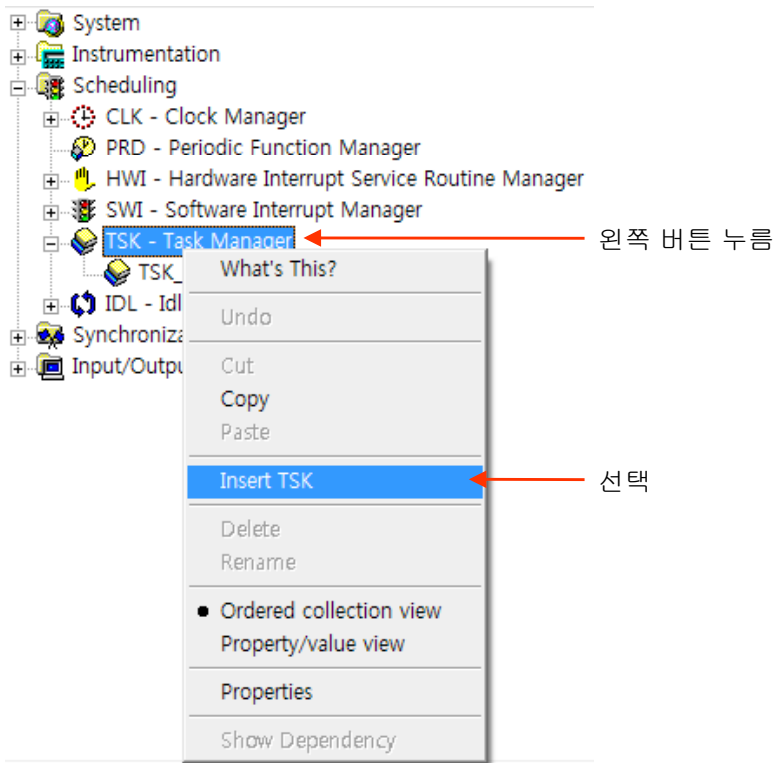
2. DSP/BIOS Config->*.tcf 를 실행 한다.



3. 스케줄링 Task Manager에서 아래와 같이 기존 TASK를 확인후 생성 한다.



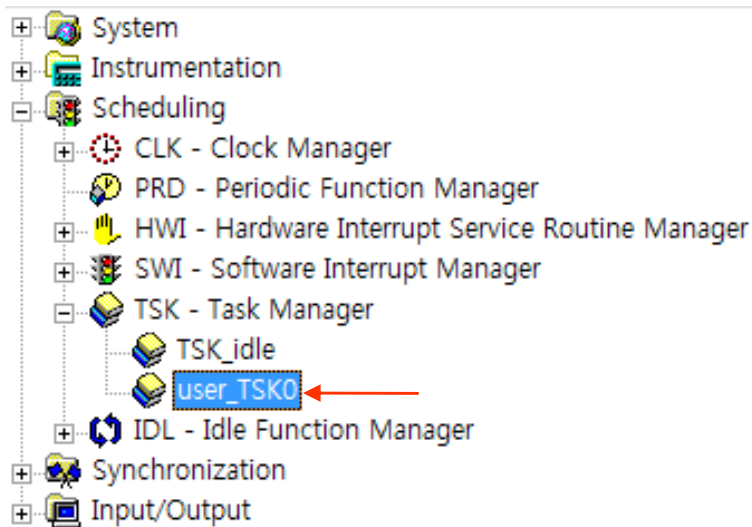
* TASK를 생성 한다.(TSK Task Manager-> Inser TSK)



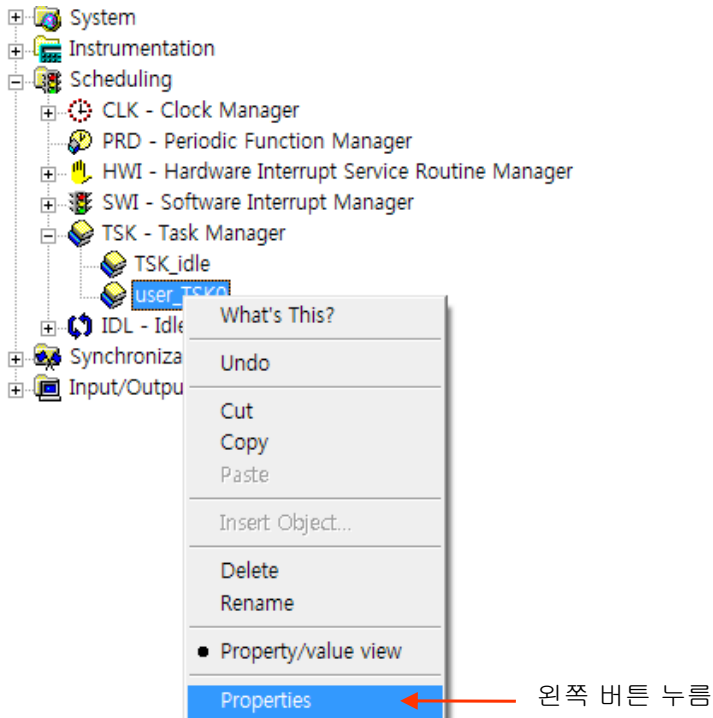
* Task 관리 명을 입력 한다



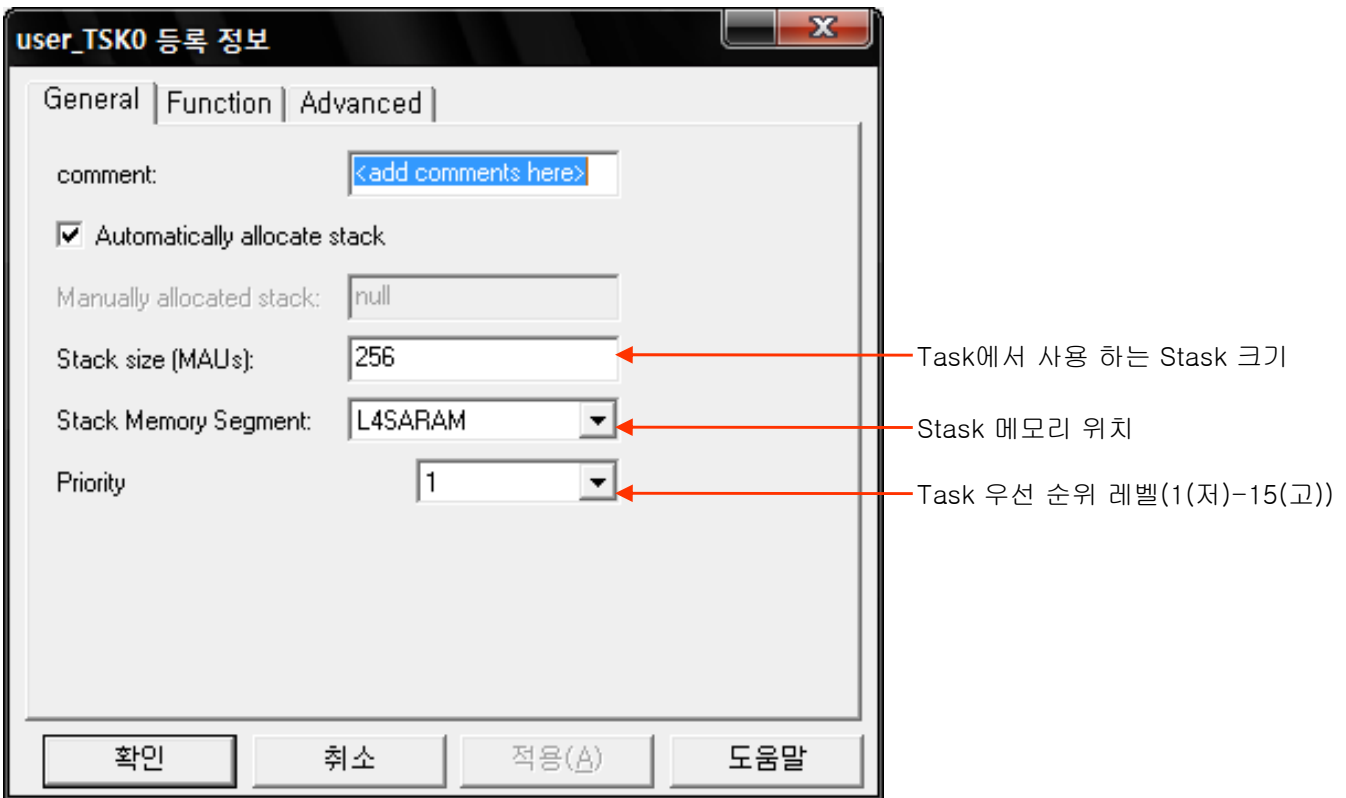
* Task 관리자 생성 확인



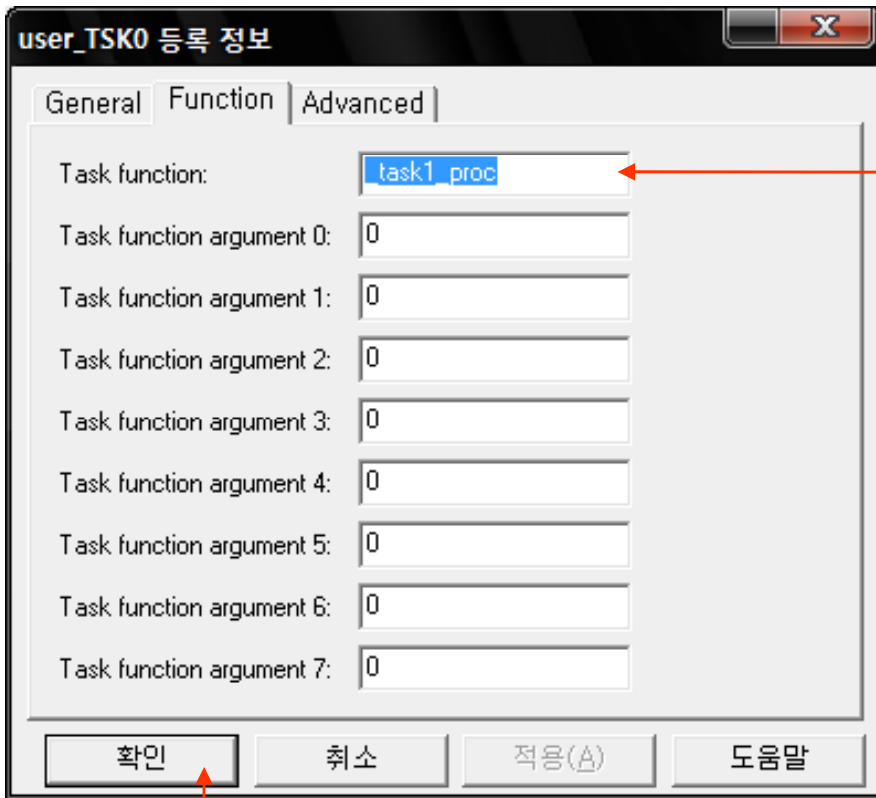
* 생성한 Task에 사용자 환경을 설정 한다.(user_TSK0 선택후 오른쪽 버튼)



* Genrnal 에서 기본 정보를 설정 한다.



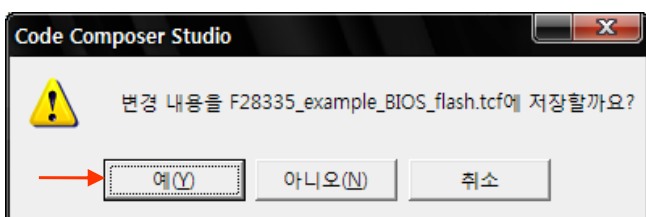
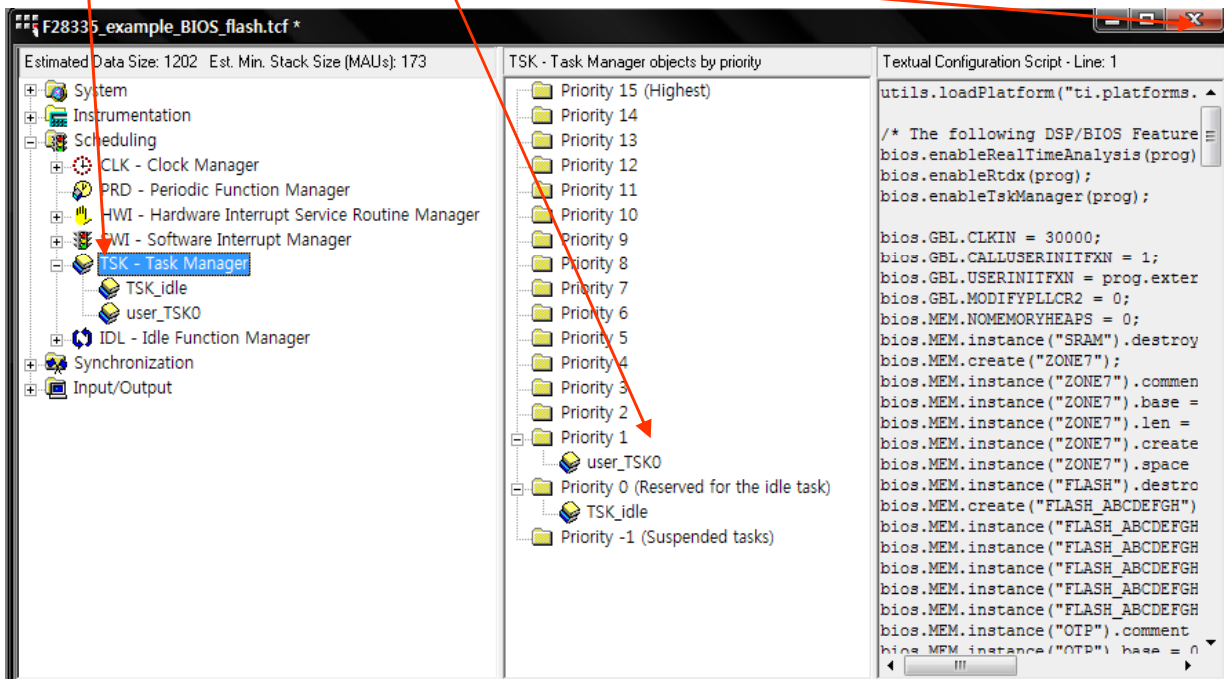
* Function 에서 실제 Task가 실행될 함수를 연결 합니다.



Main_Bios.c에서 작성된 Task함수를 등록 합니다. 함수명 앞에 _를 붙여야 함.

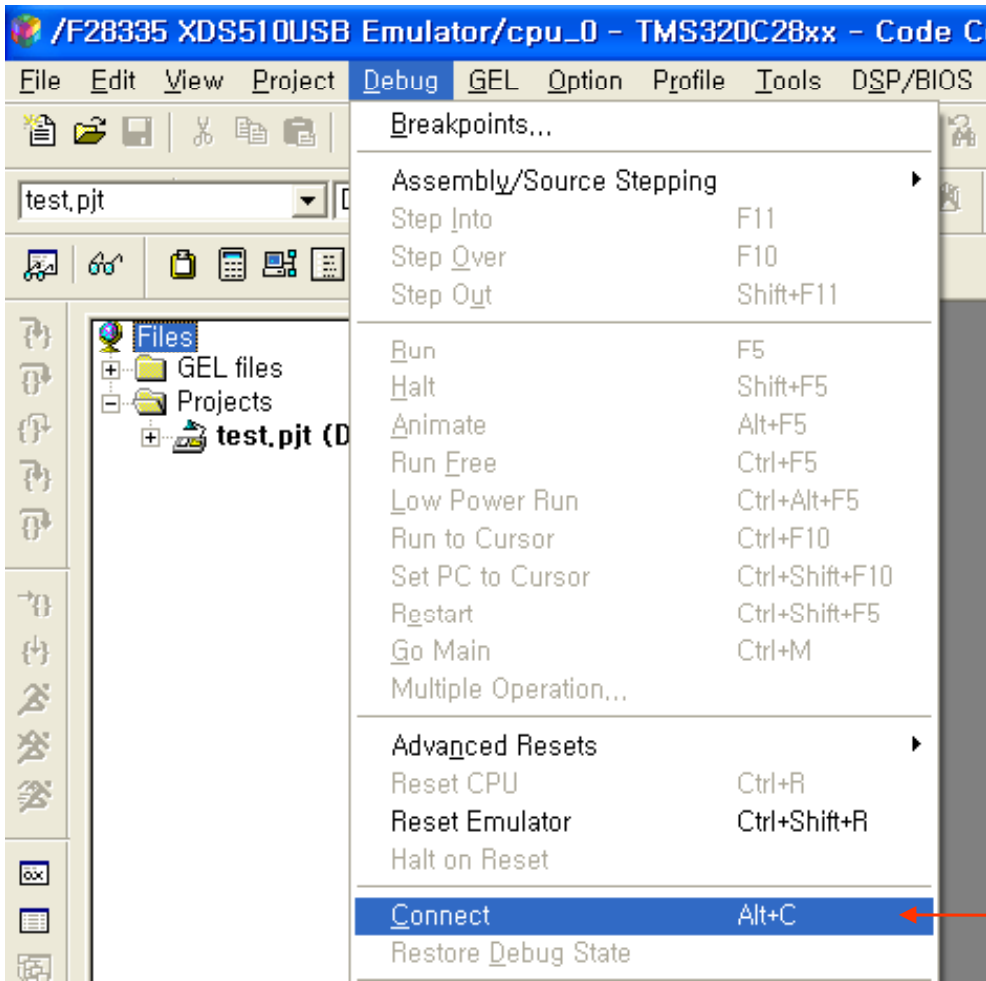
확인 버튼 클릭

* Task Manager에서 생성 된 Task를 확인후 *.tcf 파일을 종료 합니다.



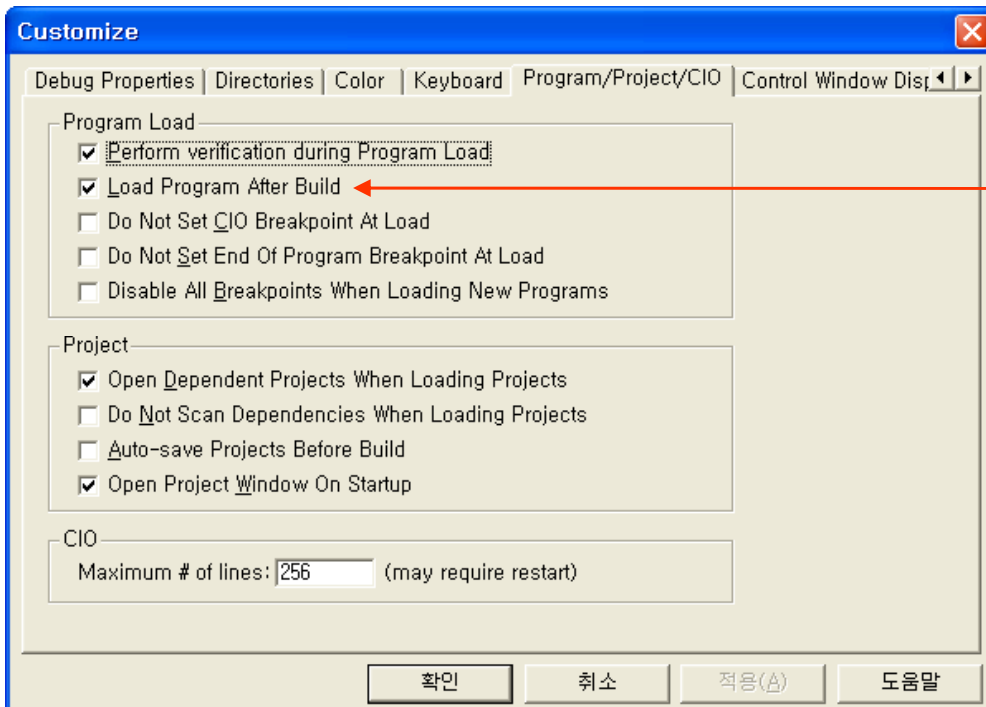
- CCS3.3 DSP/BIOS TASK 실행

1. JTAG 및 에뮬레이터를 연결 합니다.



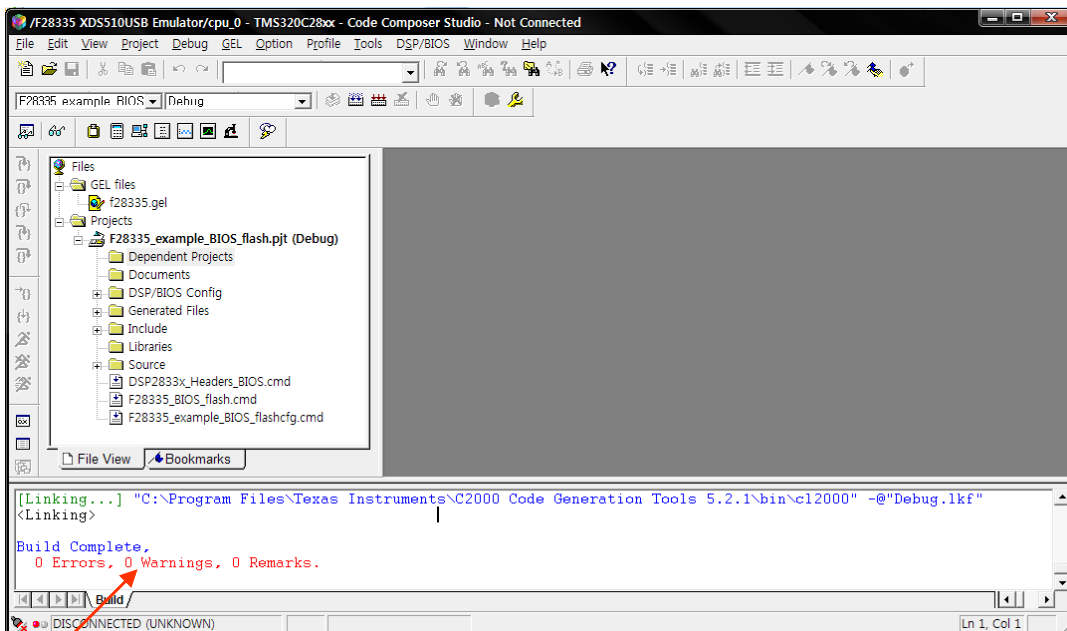
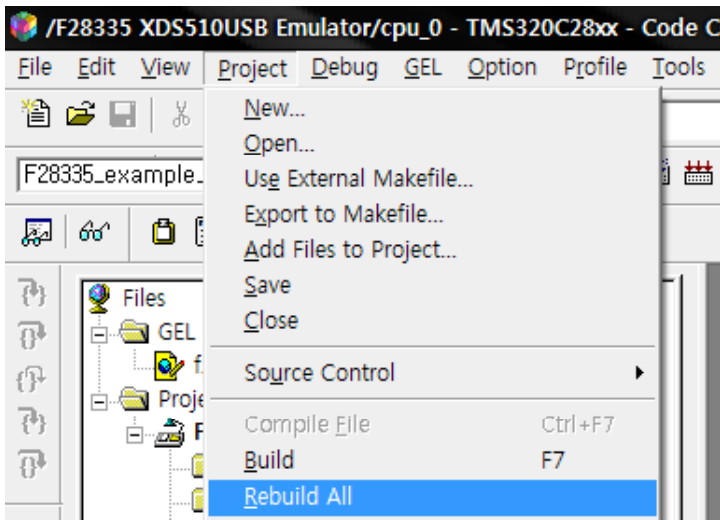
선택후
마우스 왼쪽 버튼 클릭

2. 내부럼 으로 프로그램을 실행할 경우 아래와 같이 설정 합니다.(Option->Customize)



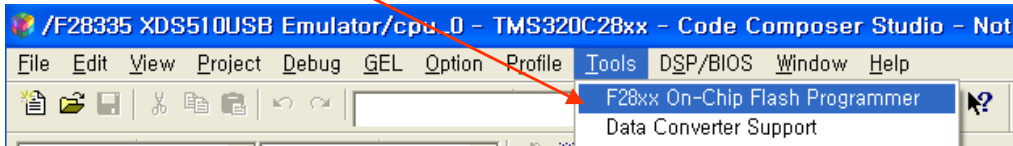
체크

3. 컴파일 하기(Project->Rebuild All)

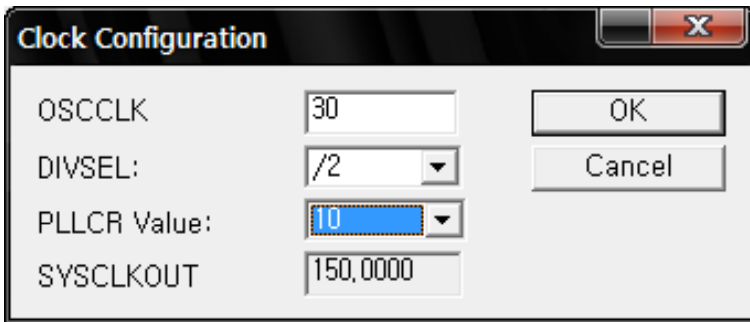


에러 확인

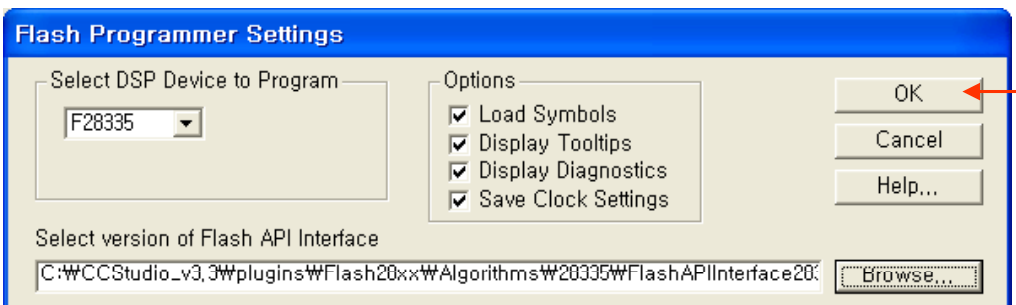
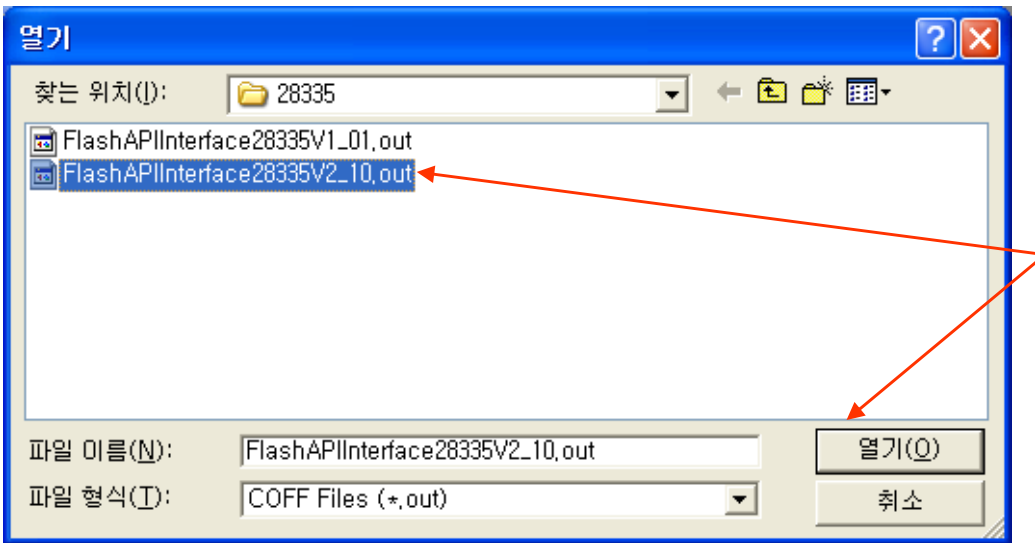
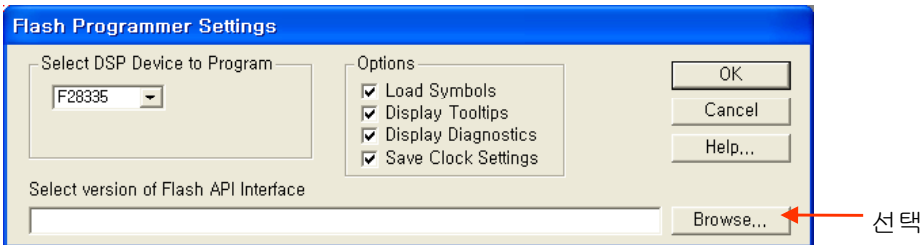
4. FLASH에 프로그램 하기

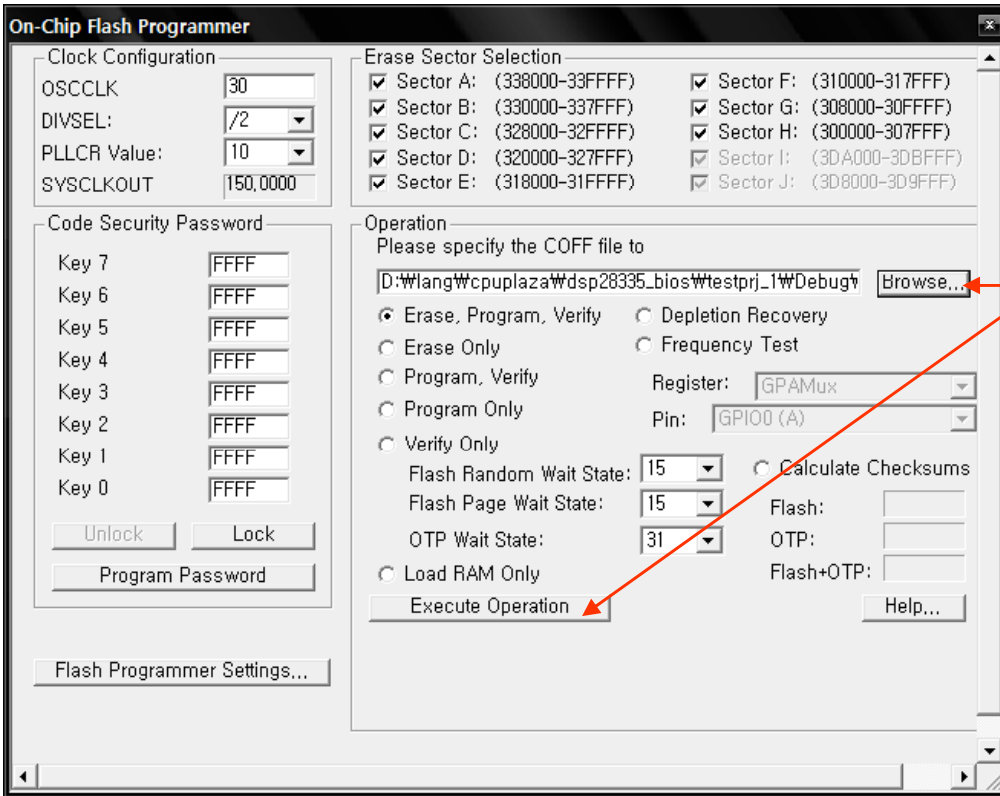


* 아래 CLOCK 설정 메뉴를 사용자에게 맞게 설정 합니다.



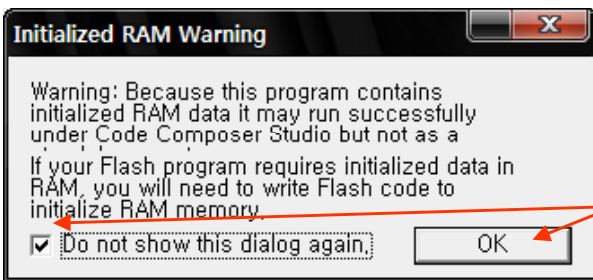
* API Interface 파일을 등록 합니다.



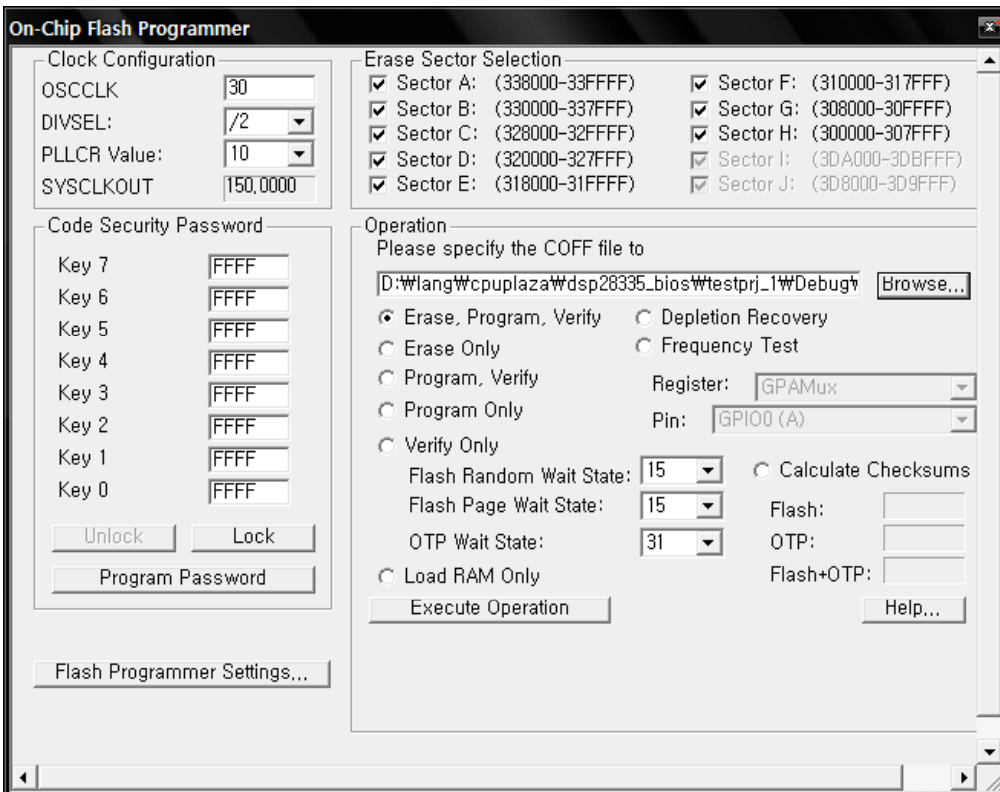


Browse.. 창에서 파일을 선택후 Execute Operation 탭을 실행합니다.

* TI 실행 파일은 *.OUT로 현재 작업 디렉토리 ..\Wdebug\ 에 있습니다.

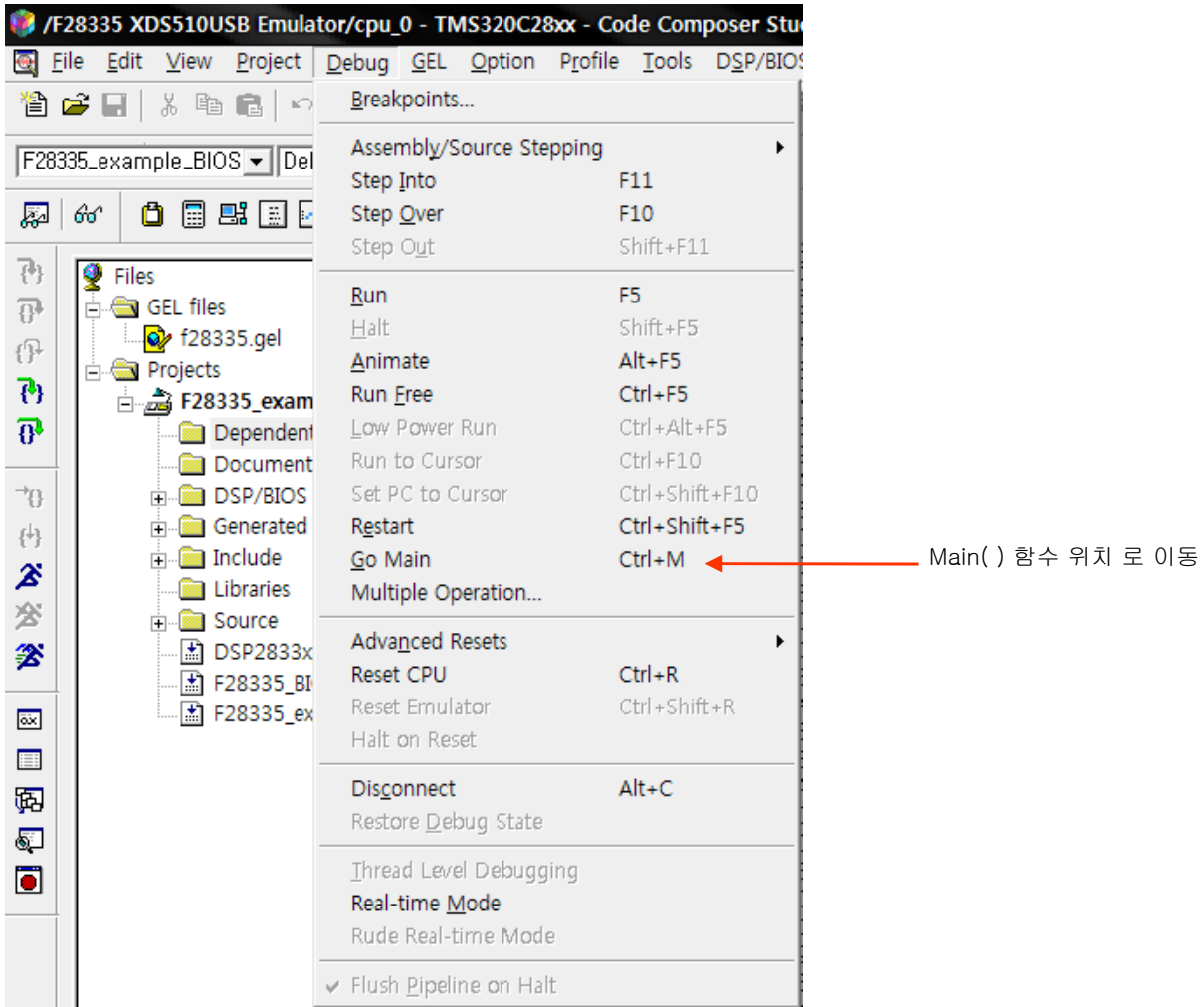


체크후 확인



다음

5. 프로그램을 로딩후 Debug 탭에서 Go Main 기능을 실행 합니다.



6. Main_Bios.c의 Task1_Proc() 에 break mode를 설정 합니다.

```
// the DSP/BIOS configuration file, System - Global Settings.
// -----
void UserInit(void)
{
#ifdef EXAMPLE_FLASH // EXAMPLE_FLASH, if defined, is in CCS project options
// Section .trcdata is generated by DSP/BIOS.
// It must be copied from its load to its run address BEFORE main().
memcpy(&trcdata_runstart, &trcdata_loadstart, &trcdata_loadend - &trcdata_loadstart);
#endif
}

// ===== 리얼 타임 타스크1 =====
// [인수] void
// -----
void task1_proc(void)
{
    while(1) {
        TSK_sleep(1);
    }
}
}
```

1. 커저를 위치 시킨후 F9키를 누른다.(한번더 누르면 삭제)
2. F5를 누르면 노란 화살표가 나오며 정지 한다.
그러면 현재 Task 스케줄링 정상.

- F5 : Debuf→Run
- F9 : Debuf→Break Point Toggle

7. 6번 항목을 확인후 F9(Break Point)를 눌러 삭제 한후 F5(RUN)를 실행 시킨 후 주메뉴의 DSP/BIOS 탭에서 BIOS TOOL을 사용해 확인 할수 있다.