

제 5 장 개발 환경

5.1 개발에 필요한 소프트웨어

- ☞ **컴파일러(Compiler)** : 고급언어로 쓰인 프로그램을 기계어로 번역하는 소프트웨어를 말한다. 일반적으로 컴파일러가 수행되는 컴퓨터의 기계어로 번역한다.
 예) C-컴파일러 : C-언어로 작성된 프로그램을 기계어로 번역
 Windows Visual C++: 윈도우즈 환경에서 C 또는 C++언어를 pentium 마이크로프로세서의 기계어로 번역
- ☞ **교차 컴파일러(Cross Compiler)** : 컴파일러가 수행되고 있는 컴퓨터의 마이크로프로세서가 아닌 다른 종류의 프로세서의 기계어로 번역하는 컴파일러 ==> 과목에서 사용하는 컴파일러는 PC(pentium 마이크로프로세서사용)에서 동작하며 ATmega128 마이크로컨트롤러의 기계어로 프로그램을 번역한다.
- ☞ **교차 어셈블리(Cross Assembler)** : 마이크로컨트롤러의 어셈블리를 기계어로 번역하는 프로그램
- ☞ **링커(Linker)** : 프로그램은 여러 개의 파일로 나누어 질 수 있다. 각 파일은 각각 컴파일 되며 링커가 컴파일 된 각 파일을 합쳐서 하나의 기계어 프로그램으로 만들어 준다.
- ☞ **Hex파일 컨버터** : 링커로 만들어진 기계어 프로그램을 ROM에 전송할 때는 intel-hex format을 사용한다. Hex파일 컨버터는 기계어를 intel-hex format으로 전환하여 준다.
- ☞ **디버거(Debugger)** : 프로그램 실행 중 여러 변수, 레지스터의 상태를 보여주고, 사용자가 문장별로 프로그램 수행을 제어할 수 있게 하여 프로그램 오류를 찾기 쉽게 도와주는 프로그램이다.
- ☞ **ISP(In-System-Programmer)** : ISP포트를 사용하여 Hex파일을 마이크로컨트롤러의 FLASH 메모리에 다운로드한다.

컴파일러는 소프트웨어이므로 여러 회사에서 제공을 한다. 각 제품마다 기계어 번역이 다르므로 번역된 실행파일의 성능은 제 각각이다.

ATmega128의 개발에 사용할 수 있는 교차 컴파일러 중 대표적인 것들은 다음과 같다. 모두 다 컴파일러 외에 링커, Hex파일 컨버터, 디버거를 동시에 사용할 수 있는 통합 환경을 제공한다.

(1) AVR Studio 4

ATmega128 제조사인 Atmel사의 홈페이지 <http://www.atmel.com> 에서 다운로드 받을 수 있는 무료 소프트웨어이다. 어셈블러, 디버거, ISP를 제공한다. C-컴파일러는 제공하지 않으므로 별도로 설치하여야 한다.

(2) CodeVisionAVR

HP Info Tech사의 교차 C-컴파일러이다. CodeVisionAVR은 ISP 다운로드를 내부에 탑재하고 있는 통합환경을 제공한다. AVR Studio 4와 함께 사용하여 프로그램 디버깅을 할 수 있다. 무료 평가판을 <http://www.hpinfotech.ro/html/download.htm>에서 다운로드 받을 수 있다. 평가판으로 프로그램 사이즈 2Kbyte내의 프로그램을 실행시킬 수 있다. 프로그램 사이즈 제한이 있지만 교재의 예제 프로그램은 충분히 수행시킬 수 있다.

(3) IAR EWAVR C

IAR사의 교차 C-컴파일러이다. ISP 다운로드에는 AVR Studio 4를 사용한다. <http://www.iar.com>에서 교육용을 무료로 다운로드 받을 수 있다. 이 역시 프로그램 사이즈가 2Kbyte로 제한되어 있다.

(4) WinAVR

<http://sourceforge.net/projects/winavr>에서 다운로드 받을 수 있는 무료 공개소프트웨어이다. gcc C-컴파일러를 내장하고 있으며 프로그램 사이즈 제한이 없다. 단 무료소프트웨어이므로 소프트웨어에 대한 보장이 없으며 사용 환경이 상용소프트웨어에 비해 조금 부실하다. 그러나 최근에 공개된 버전은 AVR Studio 4의 환경에서 직접 이 소프트웨어를 통합해서 사용할 수 있도록 기능이 보장되어 많이 편리해졌다. 2009년 12월 현재 최근 버전은 WinAVR 20090313(2009년 3월 13일 버전)이다.

본 교재에서는 무료 공개소프트웨어인 AVR Studio 4와 WinAVR C-컴파일러를 사용도록 한다.

5.2 프로그램 개발과정

일반 PC용 프로그램(Visual C++, Visual Basic 등)과 가장 큰 다른 점은 소스 프로그램의 작성은 PC에서 하지만 실행은 마이크로컨트롤러 보드에서 해야 한다는 점이다. 그러므로 소스 프로그램을 기계어로 번역할 때에 마이크로컨트롤러 ATmega128에 맞는 기계어로 번역해야만 하고, 실행 프로그램도 PC에서 마이크로컨트롤러 보드상의 프로그램 메모리인 FLASH 메모리로 옮겨 놓아야만 실행될 수가 있다. 기계어로 번역하는 것을 “컴파일(Compile)”이라 하고 프로그램을 PC에서 마이크로컨트롤러로 옮기는 것을 “다운로드(Download)”라 한다. 프로그램을 다운로드할 대상인 마이크로컨트롤러 보드를 타겟 보드(Target Board)라 부른다.

프로그램 개발과정을 요약하면 그림 5.1과 같다.

1단계 : PC에서 편집기를 사용해서 C나 어셈블리 소스파일을 작성한다.

☞ C 파일의 확장자는 'c'이고 어셈블리 파일의 확장자는 "asm"이다.

(예) ctest.c, asmtest.asm

2단계 : 크로스 컴파일러를 사용하여 소스파일을 컴파일한다.

☞ 컴파일 결과 생성되는 파일을 오브젝트파일이라 하며 확장자는 WinAVR은 ".o"이다 (예) ctest.o

☞ 오브젝트파일은 주소관련부분을 제외하고 모두 기계어로 번역된 파일이다. 주소관련부분은 링커가 정리하여 준다.

3단계 : 링커를 사용하여 실행파일을 만든다.

☞ 실행파일의 확장자는 "elf"이다.

4단계 : HEX파일 컨버터를 사용하여 HEX 파일 생성한다.

☞ HEX파일의 확장자는 "hex"이다.

5단계 : HEX 파일을 ISP를 사용하여 FLASH메모리에 다운로드한다.

※ WinAVR과 CodeVisionAVR 같은 통합 환경에서는 앞의 2,3,4단계를 "Build" 또는 "Make"라는 명령으로 한 번에 수행된다.

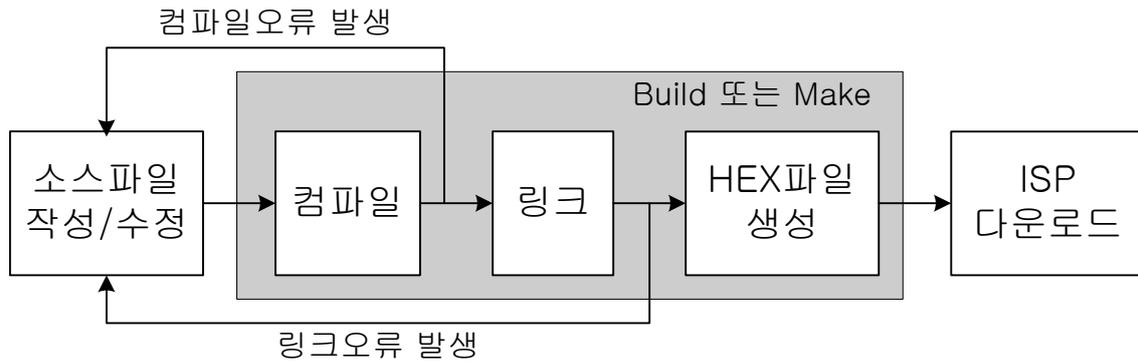


그림 5.1 소프트웨어 개발과정

5.3 WinAVR 컴파일러

최신 버전의 WinAVR 컴파일러는 AVR Studio 4에 Plug-in으로 설치되어 AVR Studio 4 환경에서 사용할 수 있다. 각 홈페이지에서 최신버전을 다운로드 받아 설치한다. 2009년 12월 현재 다운로드할 수 있는 최신 버전은 다음과 같다.

(1) Atmel사의 홈페이지 www.atmel.com :

AVR Studio 4 : aStudio4b623.exe

(2) WinAVR의 홈페이지 winavr.sourceforge.net :

WinAVR 20090313 : WinAVR-20090313-install.exe

위 순서대로 설치를 하면 AVR Studio 4의 통합 환경에서 C-소스파일의 편집, WinAVR C-컴파일러를 사용한 소스파일의 컴파일 및 Hex 파일을 타겟보드에 다운로드를 할 수 있다.

WinAVR C-컴파일러를 사용하는 자세한 방법은 사용자 매뉴얼을 참고하기 바람이며 여기서는 ATmega128 마이크로컨트롤러를 사용하기 위해 필요한 몇 가지만 언급한다.

5.3.1 데이터 형

제한된 메모리를 가지고 있는 마이크로컨트롤러의 프로그램에 있어서 메모리의 효율적인 사용은 매우 중요하며 이를 위해 C-언어에서 사용하는 변수의 데이터 형을 정확히 이해하여야 한다.

WinAVR에서 사용되는 데이터 형은 표 5.1과 같다.

표 5.1 WinAVR C-컴파일러의 데이터 형

데이터 형	비트수	바이트수	값 범위
char	8	1	-128 ~ 127
unsigned char	8	1	0 ~ 255
signed char	8	1	-128 ~ 127
int	16	2	-32768 ~ 32767
short int	16	2	-32768 ~ 32767
unsigned int	16	2	0 ~ 65535
signed int	16	2	-32768 ~ 32767
long int	32	4	$-2^{31} \sim 2^{31}-1$
unsigned long int	32	4	$0 \sim 2^{32}-1$
signed long int	32	4	$-2^{31} \sim 2^{31}-1$
float	32	4	$\pm 1.175e-38 \sim \pm 3.402e38$
double	32	4	$\pm 1.175e-38 \sim \pm 3.402e38$

- ☞ "int"형과 "double"형을 제외한 모든 데이터 형에 대해 대부분의 컴파일러가 동일한 바이트수를 사용한다. 그러나 "int"형과 "double"형은 컴파일러에 따라 사용 바이트수가 다를 수 있다. PC의 윈도우즈 프로그램에 사용되는 Visual C++ 컴파일러는 "int"형과 "double"형에 대해 각각 4바이트와 8바이트를 사용한다.
- ☞ "int"형은 컴파일러에 따라 바이트수의 차이를 보일 수 있으므로 2바이트 변수를 지정하려면 "int"를 사용하는 것보다는 "short int"형을 쓰는 것이 바람직하다.

5.3.2 I/O 레지스터 읽기/쓰기

Atmega128의 입출력 장치의 조작은 표 3.1의 I/O 레지스터와 확장 I/O 레지스터를 통하여 수행된다. 이들 레지스터에는 표 3.1과 같이 메모리 주소가 할당되어 있으므로 C-언어의 메모리 읽기/쓰기 조작으로 각 레지스터에 데이터를 읽기/쓰기 할 수 있다. 헤더파일 "avr/io.h"를 소스파일에 포함시키면 표 3.1에 정의된 레지스터 이름을 마치 변수인 것 처럼 사용할 수 있다.

(예) 표 3.1에서 메모리주소 22H에 있는 레지스터를 DDRE라 부른다. 이 레지스터에 데이터를 읽기/쓰기를 하려면 헤더파일 "avr/io.h"를 소스 파일에 포함시킨 후 DDRE를 마치 unsigned char형 변수인 것처럼 사용하면 된다. (DDRE는 1바이트 레지스터이고 부호가 없으므로 C-언어의 형으로는 unsigned char형에 해당한다.)

```
#include <avr/io.h>
. . .
unsigned char data;
DDRE = 0x01;          // 레지스터에 쓰기
data = DDRE;         // 레지스터 값 읽기
DDRE += 0x01;       // 레지스터 값 읽고/쓰기
```

5.4 AVR Studio 4에서 WinAVR 사용하기

AVR Studio 4은 WinAVR을 Plug-in 함으로써 파일 편집기, 어셈블러, C-컴파일러, 디버거, 다운로드작업을 모두 관리할 수 있는 통합개발환경을 제공한다.

 소프트웨어 버전에 따라 한글을 인식하지 못할 수 있으므로 파일명 및 파일이 위치하는 폴더의 이름에는 한글을 쓰지 않도록 한다.

c:/usr/프로그램/abc.c ==> 이와 같이 폴더명에 한글이 들어가지 않도록 한다.

5.4.1 프로젝트 만들기

하나의 실행파일을 만들기 위해서는 여러 개의 소스파일이 필요하며 여러 가지 속성을 설정해야 한다. 이를 프로젝트 별로 관리한다.

가. 새 WinAVR 프로젝트 만들기

1. [Project]->[New Project] 메뉴 선택하면 그림 5.2가 나타난다.
2. 창에서 다음에 기술한 사항을 선택한 후 "Finish" 버튼을 누르면 새 프로젝트가 만들어 진다.

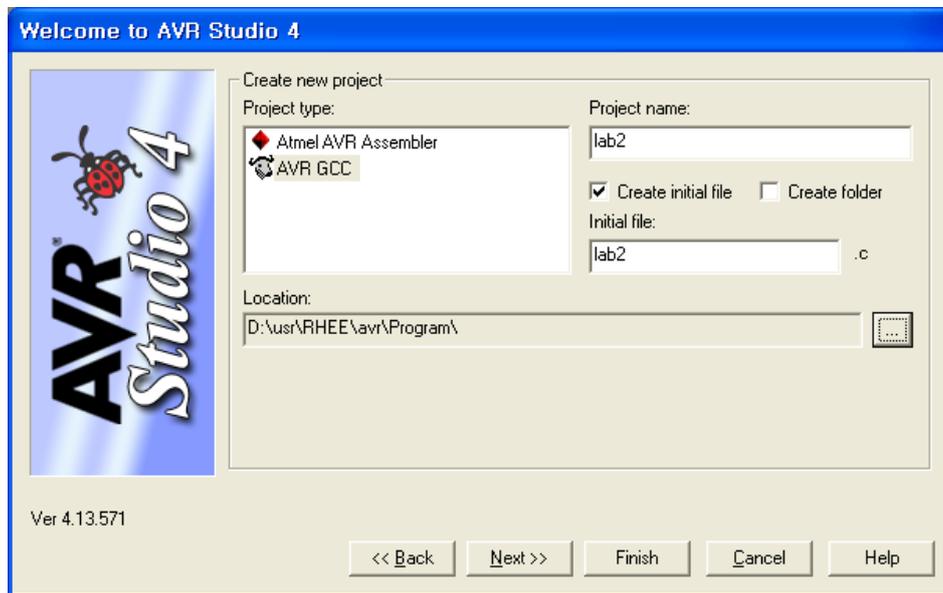


그림 5.2 새 프로젝트 생성 창

Project type : WinAVR을 사용하기 위해 "AVR GCC"를 선택한다.

WinAVR을 사용하는 프로젝트를 "AVR GCC" 프로젝트라 부른다.

Project name : 사용하고자 하는 프로젝트 명을 기입한다. 여기서는 설명을 위해 프로젝트 명으로 lab2를 사용하였다고 하자.(오동작할 수 있으므로 프로젝트 명으로 한글명을 사용하지 말 것)

Initial file : "Create initial file" 체크 박스가 선택되어 있으면 "Initial file"에 프로젝트명과 같은 이름인 lab2가 기입된다. 이 파일은 프로젝트가 만들어 지면 초기파일인 "lab2.c"가 자동적으로 생성되어 프로젝트에 포함된다. (초기파일 이름은 변경할 수도 있고 아니면 만들지 않을 수도 있다.)

Location : 프로젝트가 만들어 지면 "Location"창에 표시되어 있는 위치 아래에 프로젝트 명과 같은 폴더가 생성된다. 원하는 "Location"은 오른쪽 버튼을 눌러서 선택할 수 있다.

3. 새 프로젝트가 만들어지면 프로젝트의 속성을 설정하기 위해 [Project]->[Configuration Options] 메뉴를 선택한다. 그림 5.3에서 왼쪽의 "General" 아이콘을 선택하고 그림에서 사각형으로 표시된 부분을 그림과 같이 선택한다.

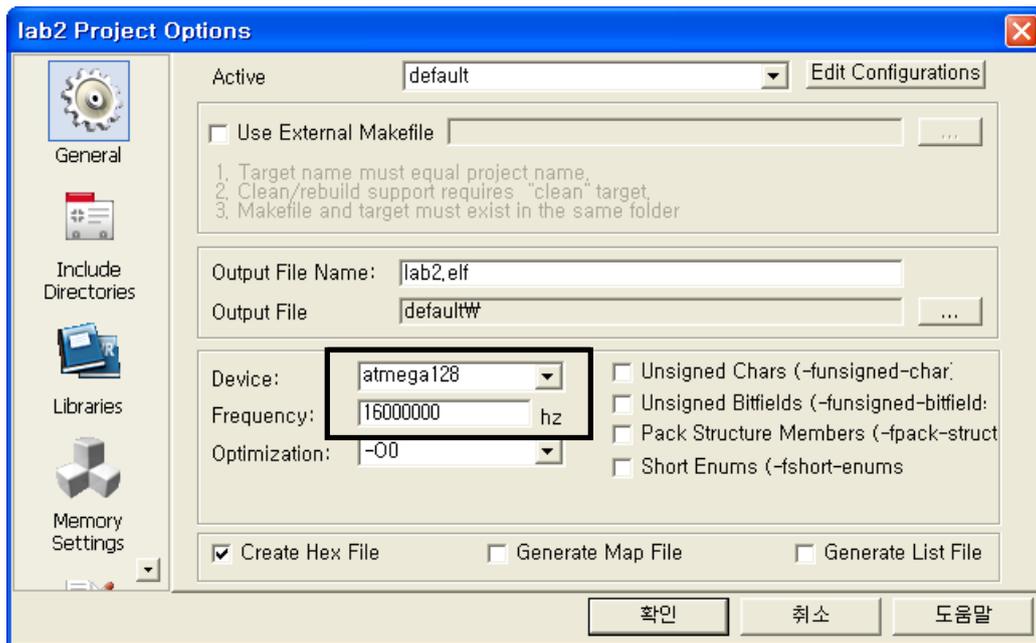


그림 5.3 프로젝트 Configuration Options 창

Active : 사용할 Configuration을 선택한다. 프로젝트 빌드 후 생성되는 오브젝트파일, 실행파일, Hex파일은 프로젝트 폴더 아래 Configure명의 폴더에 저장된다. Configuration "default"는 프로젝트생성 후 자동적으로 생성된다.

Output File Name : 실행파일 명을 나타낸다. 디폴트로 실행파일 명은 프로젝트 명과 같고, 저장 폴더는 예에서는 "lab2/default"이다.

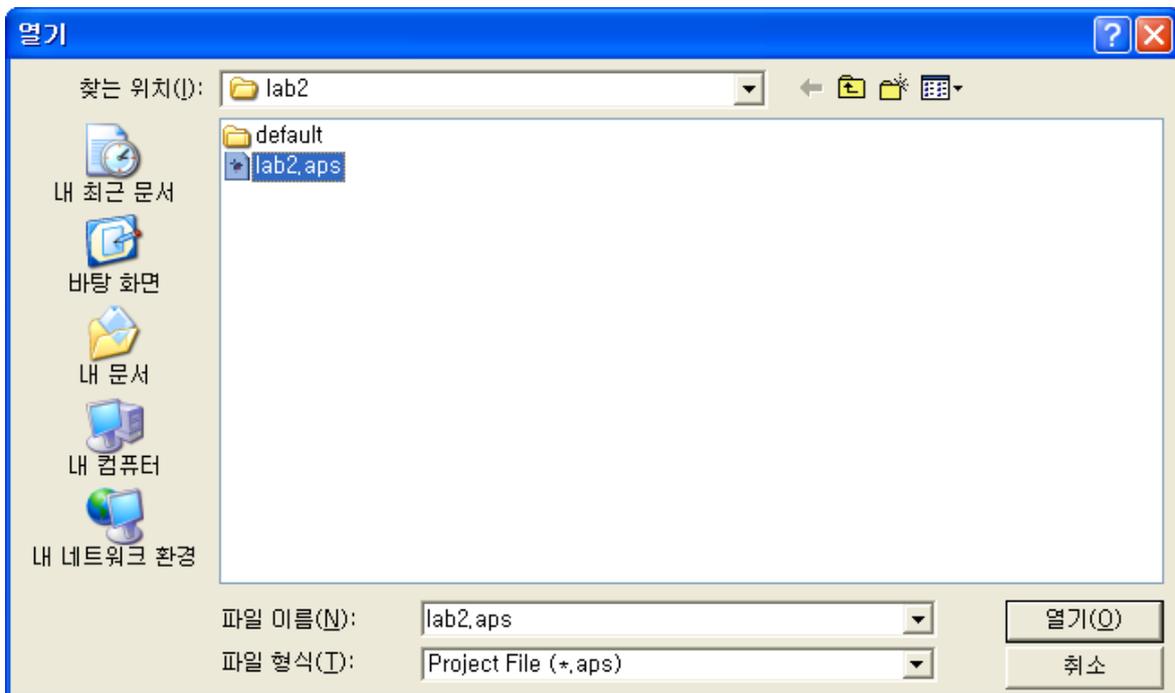
Device : 사용할 마이크로컨트롤러를 선택한다. 여기에 "atmega128"을 선택한다.

Frequency : 사용하는 마이크로컨트롤러의 클럭 주파수를 Hz단위로 기입한다. 사용하는 보드의 클럭 주파수는 16MHz이므로 16000000을 기입한다.

Create Hex File : Hex파일을 보드에 다운로드하여야 하므로 체크박스에 체크 표시를 하여 Hex파일이 만들어 지도록 한다.

나. 기존 프로젝트 열기

1. [Project]->[Open Project] 메뉴 선택한다.
2. 프로젝트 파일의 확장자는 "aps"이다. 열고자 하는 프로젝트를 열기 창에서 선택하고 [열기]를 선택한다.



다. 열려있는 프로젝트 닫기

1. [Project]->[Close Project] 메뉴 선택한다.

라. 프로젝트 마법사(Project Wizard) 사용하기

Project Wizard를 사용하면 프로젝트를 손쉽게 생성, 열기, 최근에 사용하였던 프로젝트 열기 등을 할 수 있다. AVR Studio 4가 실행되면 자동적으로 Project Wizard 창이 나타나거나, 다음과 같이 메뉴를 선택하면 그림 5.4가 나타난다.

1. [Project]->[Project Wizard] 메뉴 선택한다.

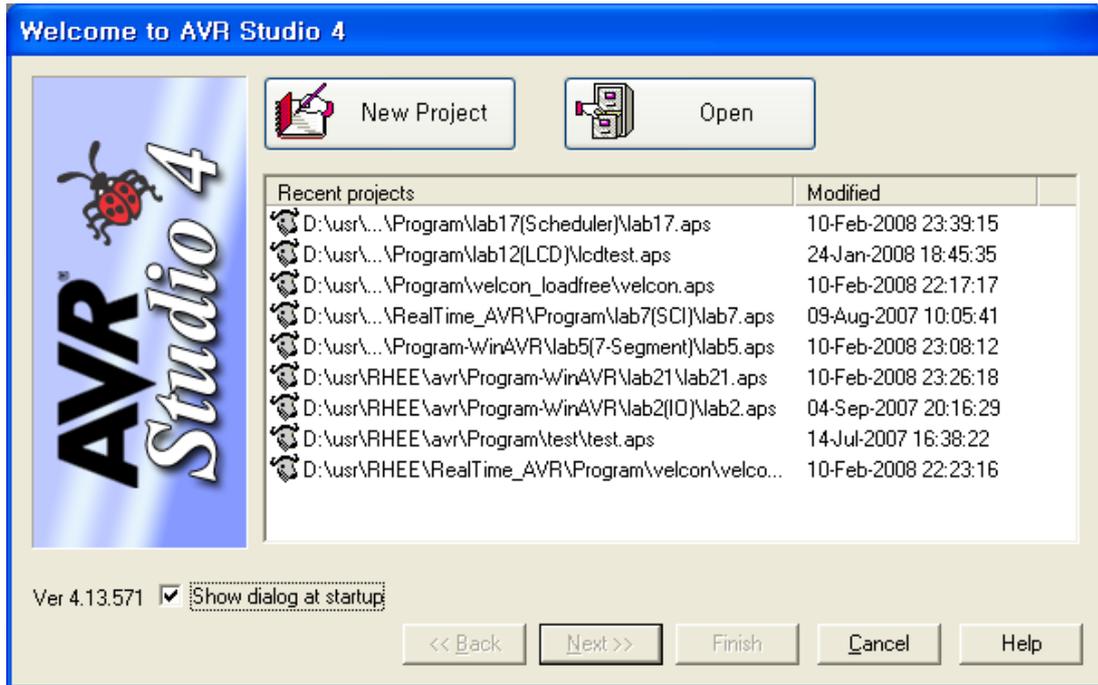


그림 5.4 Project Wizard 창

New Project Button : 새 프로젝트를 만든다.

Open Button : 기존 프로젝트를 연다.

Recent Projects 창 : 최근에 열었던 프로젝트 목록을 보여준다. 열고 싶은 프로젝트가 있으면 선택한 후 두 번 클릭하거나, 선택 후 Finish 버튼을 누르면 해당 프로젝트가 열린다.

5.4.2 소스파일 만들기

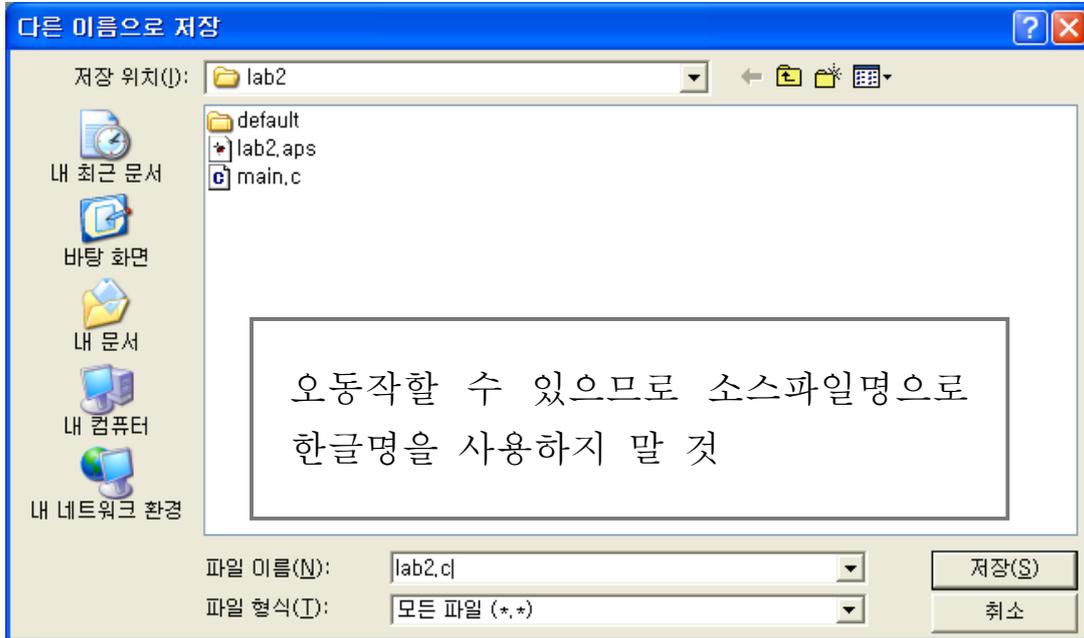
가. 새 소스파일 만들기

1. [File]->[New File] 메뉴를 선택하고 빈창이 나타나면 내용을 입력한다.

```
#include <avr/io.h>

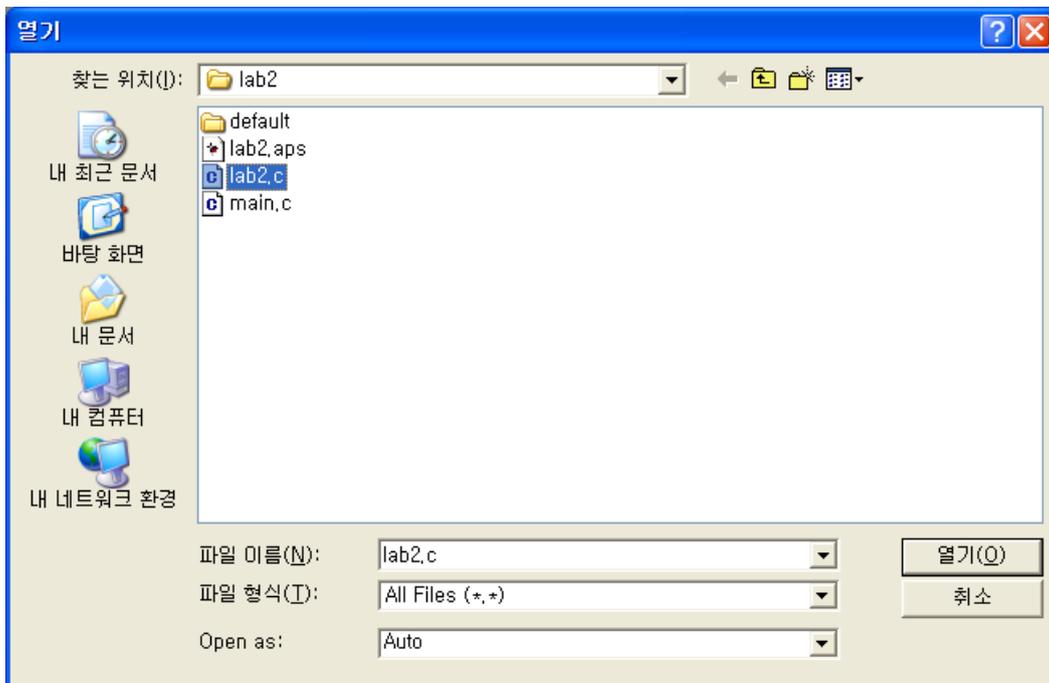
void main()
{
    DDRA = 0xFF;           // 포트A를 모두 출력포트로 설정
    while(1)              // 무한 루프
        PORTA = 0x5D;     // 패턴으로 LED를 점등
}
```

3. 입력이 완료되면 [File]->[Save] 메뉴를 선택하여 파일을 저장한다. 이 때 C-언어 소스파일이므로 확장자를 "c"로 하는 이름을 기입하고 [저장] 버튼을 누른다.(오동작할 수 있으므로 한글명을 파일명으로 사용하지 말 것)



나. 기존 소스파일 수정하기

1. [File] → [Open File] 메뉴 선택한다.



2. 파일을 선택하고 [열기] 버튼을 누른다. 소스파일이 나타나면 파일을 수정한다. 수정 후 [File]->[Save] 메뉴를 선택하여 저장한다.

5.4.3 프로젝트에 소스파일 추가/삭제 하기

앞에서와 같이 AVR-GCC 프로젝트를 생성하고 소스파일을 만들면 그림 5.5와 같은 AVR Studio 4의 실행화면을 볼 수 있다.

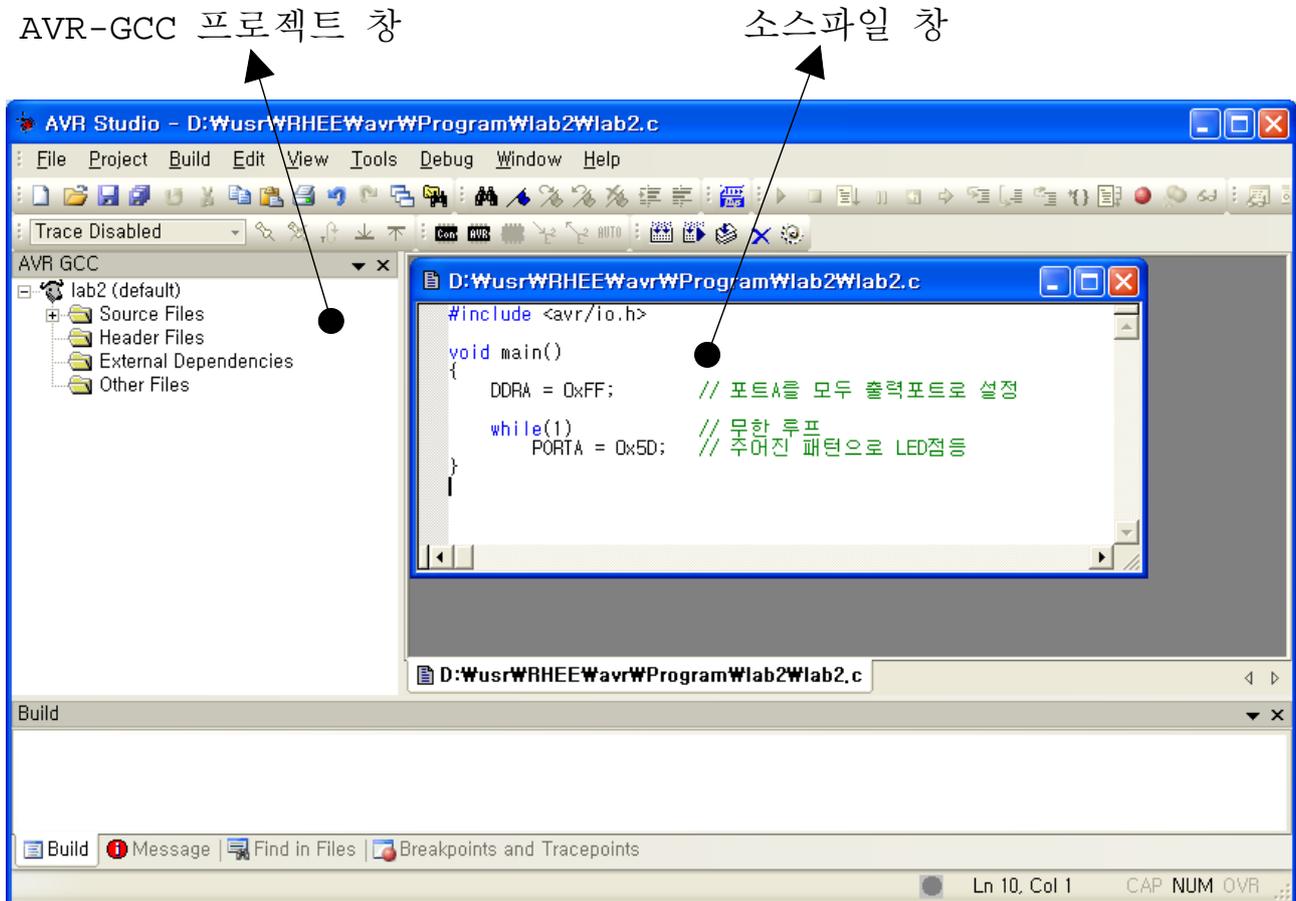


그림 5.5 AVR-GCC 프로젝트 수행 AVR Studio 4

AVR-GCC 프로젝트 창 : AVR-GCC 프로젝트의 내용을 보여주는 창이다. 이는 창 아래 [AVR GCC] 탭을 선택하면 나타난다. [I/O View] 탭을 선택하면 AVR 마이크로컨트롤러의 I/O 레지스터의 상태를 보여준다.

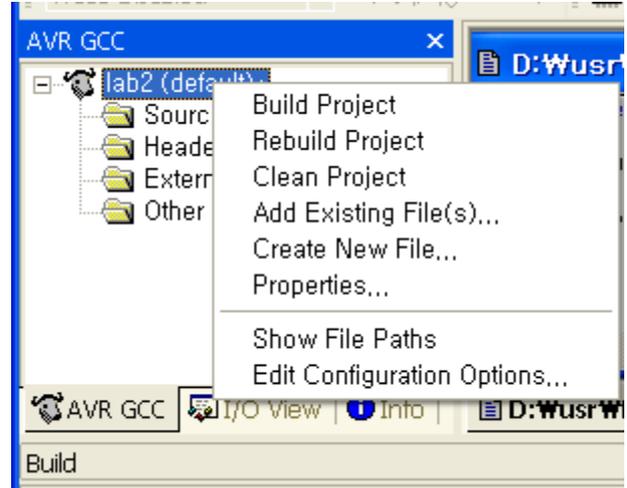
소스파일 창 : 소스파일의 내용을 보여준다.

Build 창 : 프로젝트를 빌드하여 실행파일과 Hex 파일을 만든 후 결과를 나타내는 창이다.

가. 프로젝트에 소스파일 추가하기

프로젝트는 여러 개의 소스파일로 구성이 된다. 이 때 필요한 소스파일을 프로젝트에 추가하여야 한다.

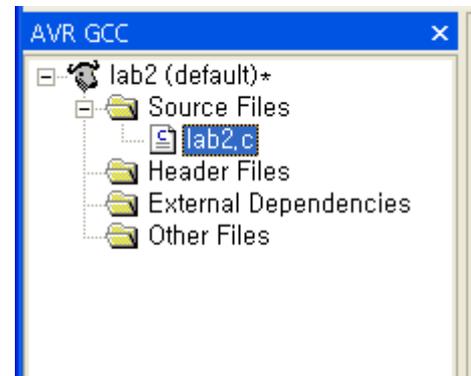
1. AVR-GCC창의 프로젝트(여기서는 lab2(default))에 마우스를 위치시키고 오른쪽 버튼을 누르면 리스트가 나타난다.
2. "Add Existing File(s)"를 선택하면 파일 선택 창이 나타난다. 추가할 파일을 선택하고 [열기]버튼을 누르면 파일이 프로젝트의 "Source Files"에 추가된다.



나. 프로젝트에서 소스파일 삭제하기

프로젝트의 소스파일 구성은 "Source Files"의 왼쪽 "+"를 누르면 "-"로 바뀌면서 구성소스파일이 나타난다.

1. 삭제하고자 하는 파일을 한 번 클릭하고 키보드의 "Delete" 키를 누르면 프로젝트에서 파일이 삭제된다.



5.4.4 프로젝트 컴파일/빌드 하기

실행파일을 생성하려면 소스파일의 컴파일/링크/HEX파일 변환을 거쳐야 한다. 빌드는 세 가지를 한꺼번에 수행하는 것을 말한다. 컴파일은 개개 소스파일에 대해 수행하고 빌드는 프로젝트 단위로 수행이 된다.

가. 컴파일(Compile) 하기

AVR-GCC 창에서 소스파일을 두 번 클릭하면 소스파일이 편집 창에 뜬다.

1. 컴파일 할 소스파일을 편집 창에서 선택한다.
2. [Build]->[Compile] 메뉴를 선택한다. 에러가 있으면 Build 창에 에러 메시지가 표시된다.

컴파일은 소스파일을 오브젝트 파일로만 만들기 때문에 실행파일을 생성하려면 빌드과정을 또 거쳐야 한다.

나. 빌드(Build) 하기

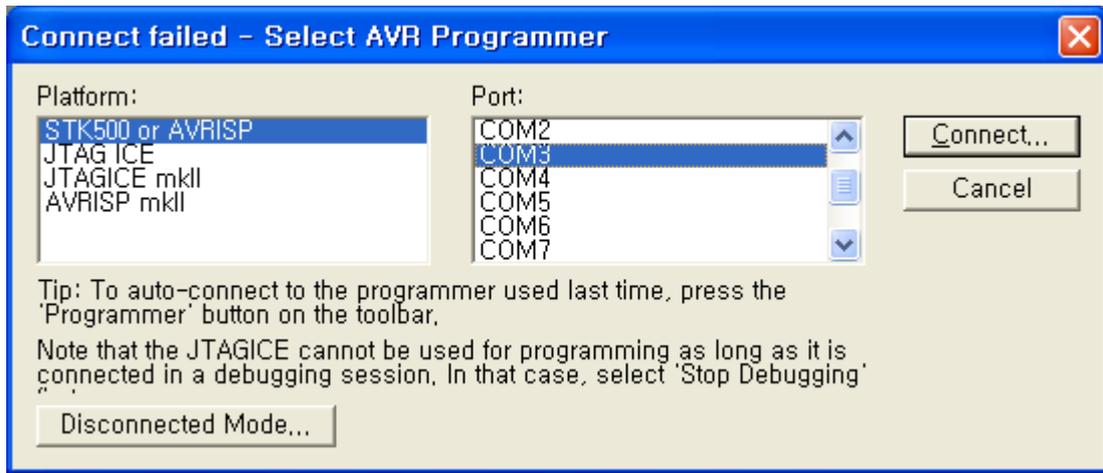
빌드는 프로젝트 내의 모든 소스파일을 컴파일/링크한 후 HEX파일을 한 명령으로 생성하는 과정이다. 빌드를 수행할 때는 앞의 컴파일과정을 할 필요가 없다.

1. 빌드할 프로젝트를 AVR-GCC 창에서 클릭하여 선택한다.
2. [Build]->[Build] 메뉴를 선택한다. 컴파일/링크 도중 에러가 있으면 Build 창에 표시된다.
3. 빌드 결과 생성되는 실행파일과 HEX파일의 이름은 프로젝트명과 같다.

5.4.5 타겟 보드에 프로그램 다운로드

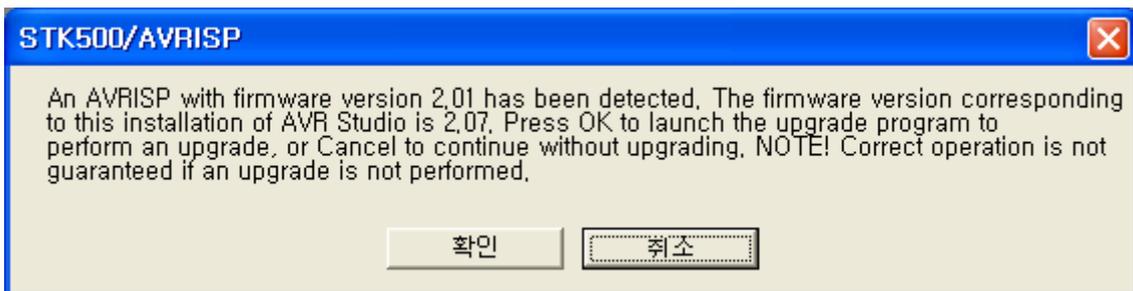
프로젝트의 빌드가 에러 없이 수행되면 타겟 보드에 프로그램을 ISP를 통하여 다운로드 할 수 있다. 우선 그림 4.4와 같이 타겟 보드와 USBISP를 연결하고 다음 순서에 따른다.

1. [Tools]->[Program AVR]->[Auto Connect] 메뉴를 선택한다. 접속이 성공되면 순서 3으로 넘어가고 접속이 실패하면 다음 창이 화면에 나타난다.



2. 접속이 실패하였으므로 다음 세 가지를 확인하고 [Connect] 버튼을 누른다.
 - [Platform] 창에서 "STK500 or AVRISP" 항목 선택확인
 - 4.3절의 USBISP 드라이버 설치 이후 USB Serial Port에 할당된 COM 포트 [Port] 창의 항목 선택
 - ATmega128 보드와 연결 확인 및 전원공급 확인

3. 사용하는 USBISP 버전에 따라 다음 창이 나타날 수 있다. 이 때 [취소] 버튼을 누른다.



4. 연결이 되면 그림 5.6의 AVRISP 창이 나타난다. 기본적으로 설정하여야 할 탭은 [Main]과 [Program] 탭이다. 설정을 마친 후 [Program] 탭에서 [Flash] 항목의 [Program] 버튼을 누르면 프로그램이 타겟보드에 다운로드되고 즉시 실행된다.

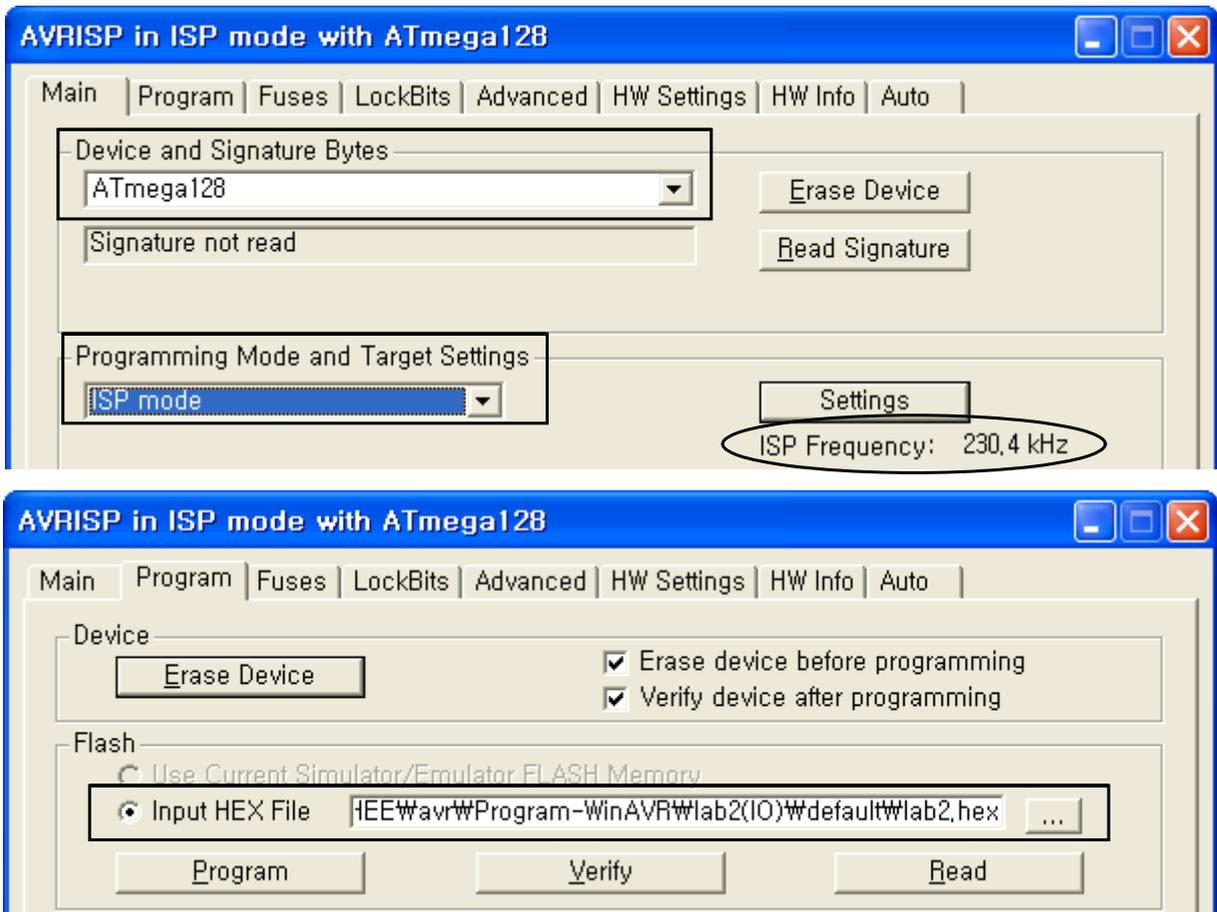


그림 5.6 AVRISP 프로그래머 창

Device and Signature Bytes : "ATmega128" 항목 선택확인

Programming Mode and Target Settings: "ISP mode"를 선택하고
ISP Frequency가 230.4kHz임을 확인한다. 이 보다 작은 주파수로
설정되어 있으면 다운로드 속도가 느리므로 "Setting"버튼을 눌러
주파수를 조정한다.

Flash : 플래쉬 메모리에 다운로드할 HEX파일은 선택하는 항목으로 선택
항목 오른 쪽 버튼을 눌러 다운로드할 HEX파일을 설정한다.

- ☞ HEX파일은 항상 가장 최근에 다운로드하였던 파일이 디폴트로 표시
된다. 프로젝트가 바뀌더라도 새로이 설정된 프로젝트의 HEX파일이
디폴트로 표시되지 않으므로 반드시 파일을 확인하여야 한다.
- ☞ HEX파일이 저장되는 위치는 프로젝트폴더 아래 현재 사용 중인
Configuration 폴더에 저장되어 있다. 그림 5.6에서는 사용 중인
Configuration은 "default"임을 나타낸다.

5.4.6 타겟 보드의 Fuse 세팅

ATmega128 마이크로컨트롤러는 Fuse와 Lock Bit라는 것이 있어 클럭, Security 모드, Bootloader 모드 등을 설정할 수 있다. 교재에서는 이들 중 몇 개를 제외하고 디폴트상태로 사용하고 있다. 그림 5.7에서는 퓨즈 중 예외로 설정 해제하여야 할 항목을 나타내고 있다.

1. 그림 5.6에서 [Fuses] 탭을 선택한다. 선택 후 창은 그림 5.7과 같다.
2. "M103" 항목은 ATmega128의 이전 버전인 ATmega103의 호환을 설정하는 항목으로 체크를 해제한다.
3. "JTAGEN" 항목은 입출력포트 F와 JTAG핀의 공유를 설정하는 것으로 교재에서는 포트 F를 사용하므로 해제하여야 한다.
4. "SUT_CKSEL" 항목은 사용 클럭을 설정하는 것으로 그림과 같이 외부 클럭으로 설정한다.
5. [Program] 버튼을 눌러 퓨즈를 프로그램한다.

☞ 퓨즈 프로그램은 단 한번만 하면 된다.

☞ Fuse와 Lock Bit 설정은 교재에서는 다루고 있지 않다.

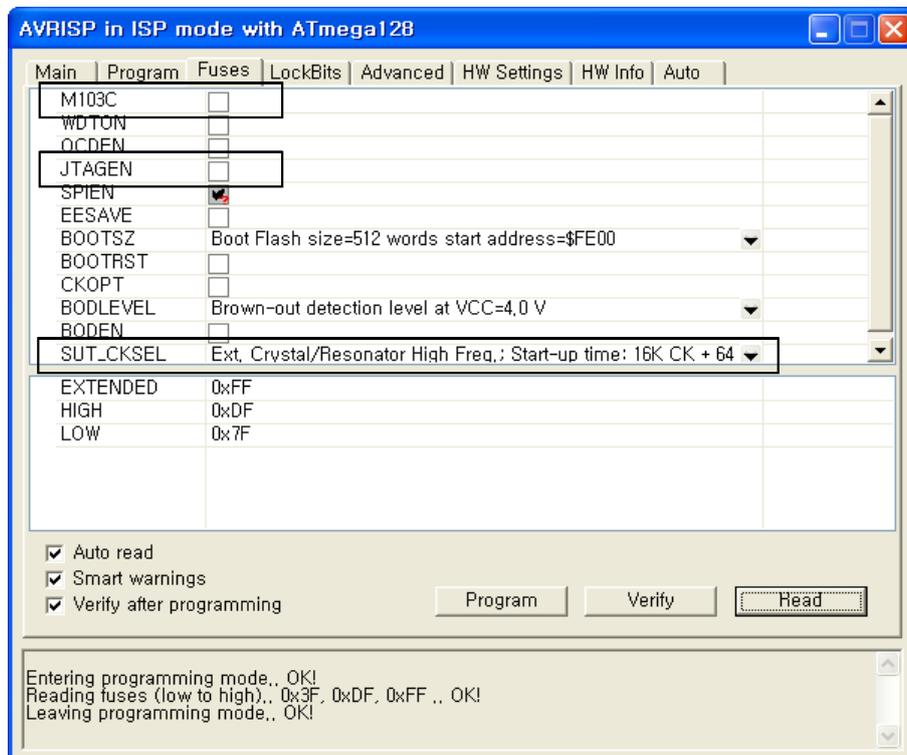


그림 5.7 퓨즈 설정

5.5 타겟보드 설치 및 실습회로구성

그림 4.1의 Atmega128 타겟보드는 빵판(Bread Board)에 설치할 수 있는 구조로 되어 있고 교재에서 수행하는 실습에 필요한 전자회로는 비교적 간단하여 손쉽게 빵판 위에 손쉽게 구성할 수 있다. 그림 5.8은 빵판에 타겟보드, 스위치 4개, 7-세그먼트 그리고 LCD를 설치한 것을 보여준다.

우선 Atmega128 타겟보드를 그림 5.8과 같이 설치하고 실습을 진행하면서 빵판에 실습회로를 구성하도록 한다.

