advanced digital chips inc

adStar

- *SDK Reference Manual* -

32bits EISC Microprocessor*adStar*

Ver 1.2
February 2, 2012

Advanced Digital Chips Inc.

History

2011-10-31 Created Preliminary Specification

2011-12-20 Ver1.02

- adStar SDK 구성, adStar 사용하기 부분이 adStar SDK 활용으로 통합 및 내용 수정.
- UART ➔ uart_putdata, uart_putstring 내용 추가.
- GRAPHIC ➔ loadtga, loadtgap, loadpng, loadpngp, drawsetclipwindow, drawsurfacescale drawsurfacescalerect 내용 추가.

2012-02-21 Ver1.1

- Bootloader 내용 추가
- lib_config.h 내용추가
- Font 내용 추가
- SOUND ➔ sound_vol_wav, sound_loadwavp, sound_loadmp3p 내용 추가.
- 회사 주소 수정.

2012-02-29 Ver1.11

- 4.lib_config.h 의 이미지 및 Nested Interrupt 내용변경.

2012-03-12 Ver 1.2

- GRAPHIC ➔ createsurface_from 내용 추가. (이미지 회전 관련)
- Font ➔ bitmapfont ➔ bmfont_setautokerning, bmfont_write_vleft, bmfont_write_vright 추가
- Font ➔ bitfont ➔bf_drawstring_vleft, bf_drawstring_vright 추가
- "2.3 Demo Program 실행"의 Linker Script 관련 내용 수정.
- "3.1 Bootloader" 의 Mode 관련 내용 수정, Linker Script 관련 내용 수정. Execute Mode 내용 수정. User Define Mode 내용 추가.
- "3.3 BootLoader 없이 사용하기" 내용 추가.
-

adStarSDK Reference Manual

Office

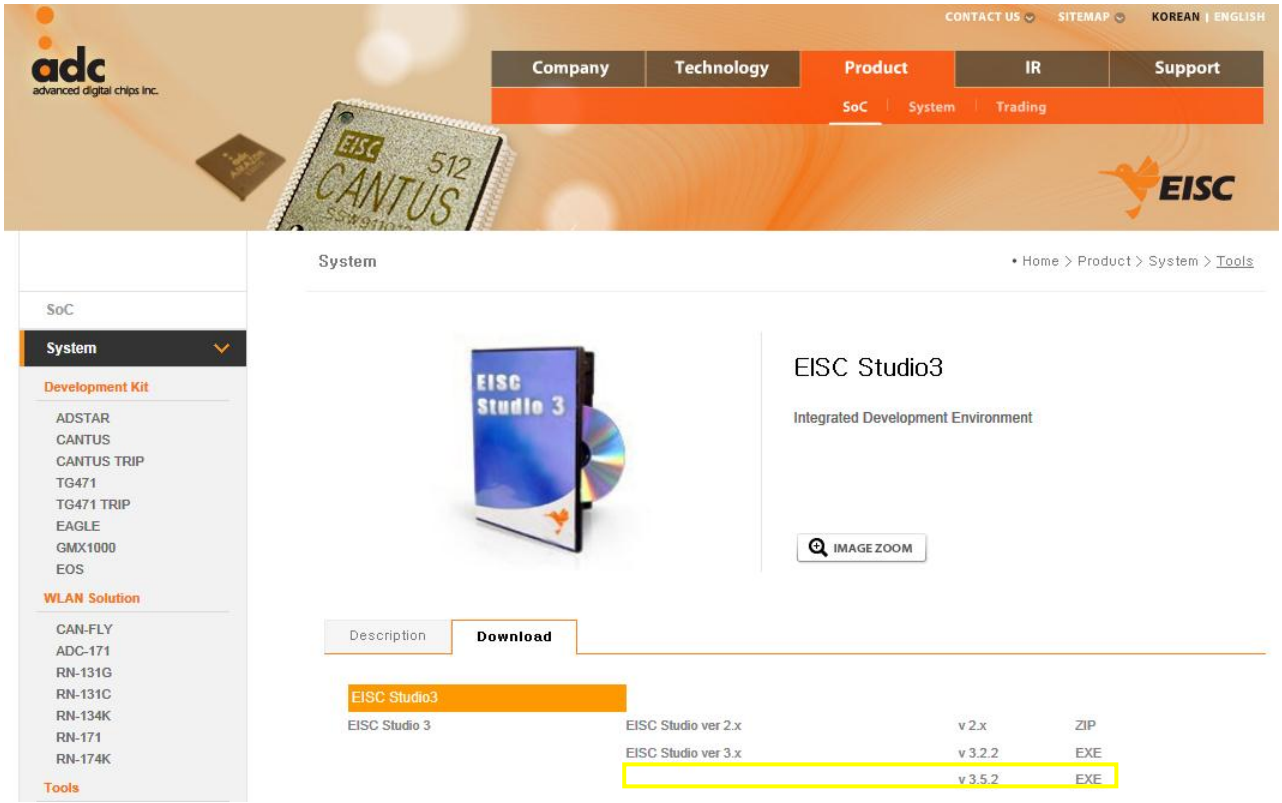
22F, Bldg A, KeumkangPenterium IT Tower,
810, Gwanyang_dong, Dongan-Gu, Anyang-si, Gyeonggi-do, 431-060, Korea
Tel : +82-31-463-7500
Fax : +82-31-463-7588
URL : http://www.adc.co.kr

— Table of Contents —

1. Software 개발환경

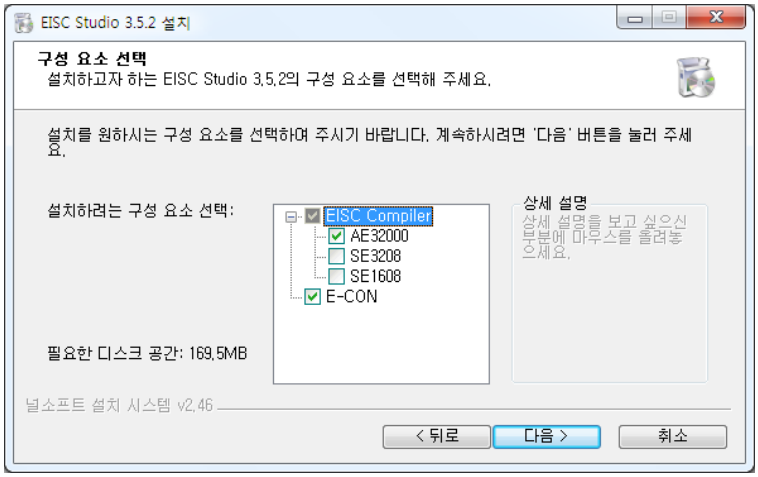
첫 번째 장으로 adStar 를 사용하기 위한 개발환경을 구축하는 방법부터 알아보도록 하겠다. 에이디칩스에서는 adStar Chip 을 위해 프로그램 코드 편집, compile, download, debugging 을 하나의 프로그램에서 수행할 수 있는 통합개발환경(IDE) EISC Studio3 를 제공하기 때문에 자료실에서 EISC Studio3 설치 파일을 내려 받아 설치만 해주면 간단하게 개발환경 구축을 마칠 수 있다. 참고로 EISC Studio3 는 Windows OS 만 지원하면 XP 이상에서 동작한다.

1. 에이디칩스 홈페이지(http://www.adc.co.kr)에 접속해서, Product → System → EISC Studio3 → Download 에서 EISC Studio ver 3.x 의 v3.5.2 를 내려 받는다.
   (설치 및 사용자 매뉴얼도 있으니 참고 하도록 한다.)

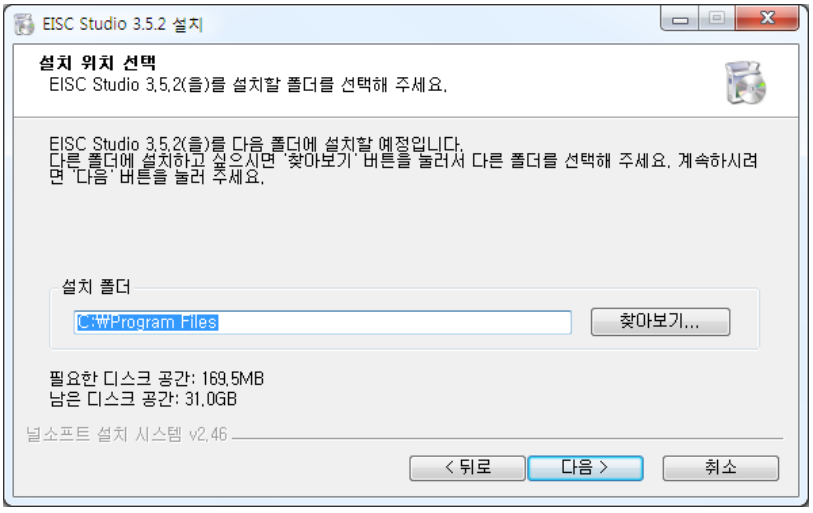


2. 내려 받은 ES3_setup_v3.5.2.exe 를 실행하여 설치를 시작하면, 설치를 시작한다는 창이 뜨는데 '다음'을 클릭하여 설치를 시작한다. '다음'을 클릭하면 아래와 같이 설치할 Compiler 와 개발장비인 E-CON 의 Driver 설치를 선택하는 창이 뜬다. adStar 는 AE32000 processor[1]를 내장한 Chip 으로 AE32000 compiler 를 선택하면 된다. 그리고 E-CON[2]장비를 처음 사용하는 사용자라면 E-CON 을 선택하여 E-Con Driver 도 설치한다.

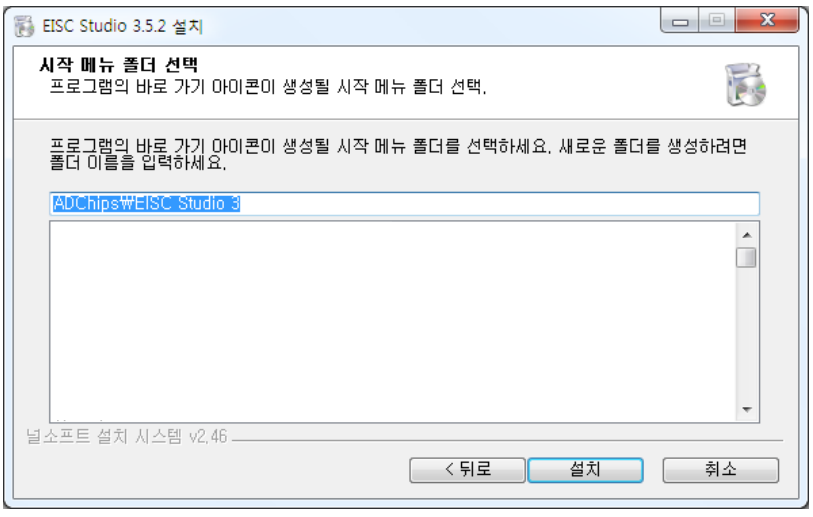
[1]에이디칩스에서개발한 32bit Processor. (EISC Architecture)
[2]adStar와연결하여program download및 debugging을수행할때필요한장비.

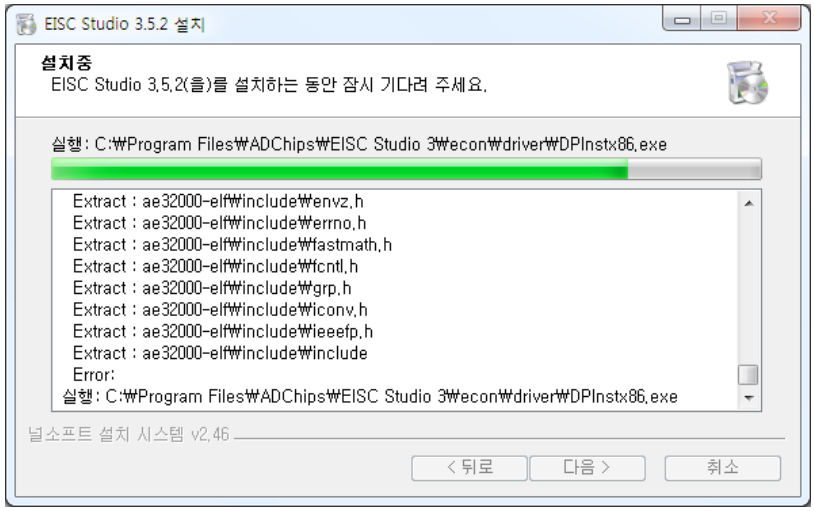


3. compiler 와 E-CON Driver 선택을 하고 다음을 클릭하면, 설치할 폴더를 선택하는 창이 나온다. 특별한 경우가 아니라면 default 로 되어있는 경로에 설치하면 된다.
(주의사항으로 64bit 운영체제를 사용할 경우 default 로 c:₩Program Files(x86) 이라고 되어 있는데, (x86)을 제거한 후 c:₩Program Files 에 설치하길 바란다.)

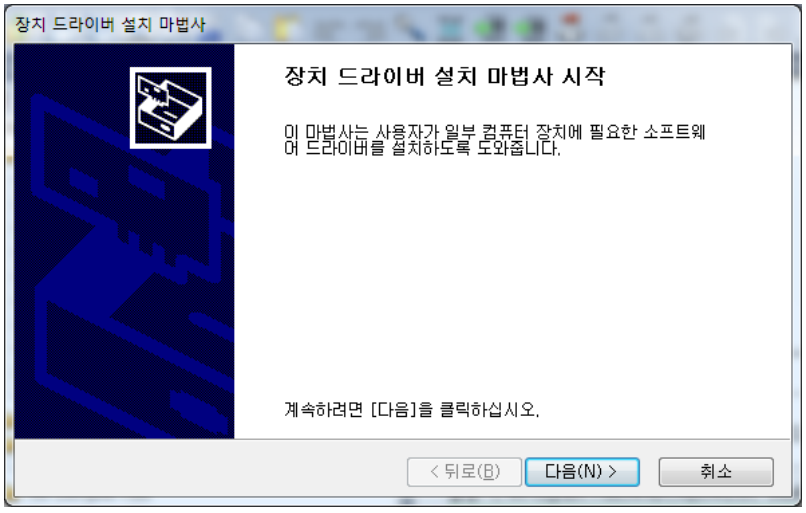


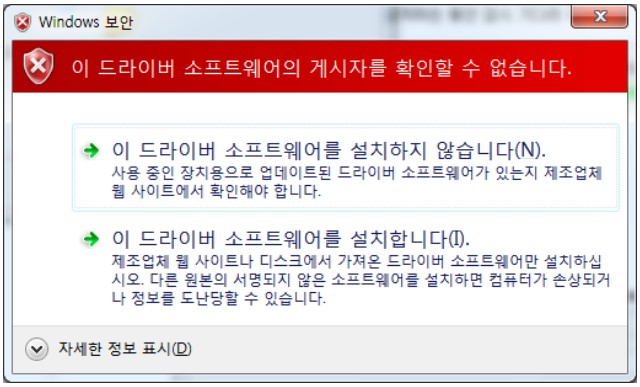
4. 다음을 클릭하면 아래와 같은 시작 메뉴 폴더를 선택하는 창이 뜨고, 설치를 누르면 설치가 시작된다.



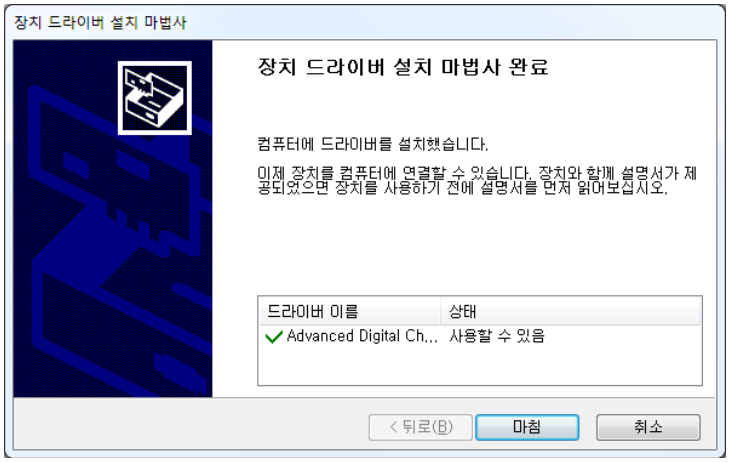


5. 구성요소 선택에서 E-CON 을 체크하였으면 설치가 끝날 무렵 E-Con Driver 설치를 위한 다음과 같은 창이 뜨고, 다음을 클릭하면 Driver 설치를 시작한다. 드라이버 설치 중 Windows 보안 경고 창이 뜨면 "이 드라이버 소프트웨어를 설치합니다."를 선택하도록 한다.
(주의사항으로는 컴퓨터와 E-CON 이 연결되어 있을 경우 설치에 문제가 생길 수도 있으므로, 컴퓨터와 E-CON 이 연결되지 않은 상태에서 설치하기 바란다.)

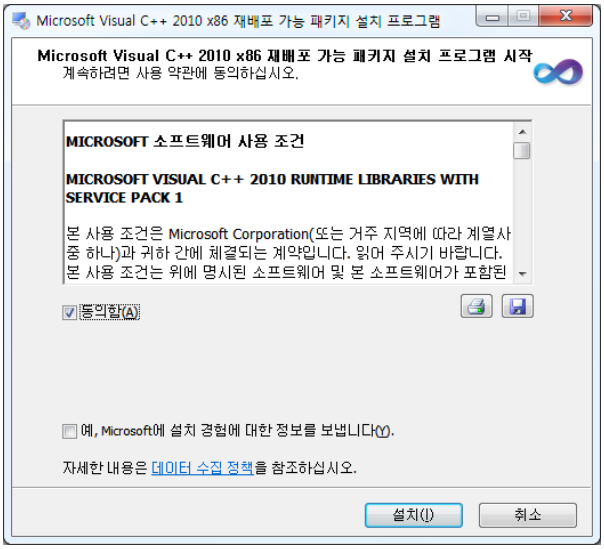




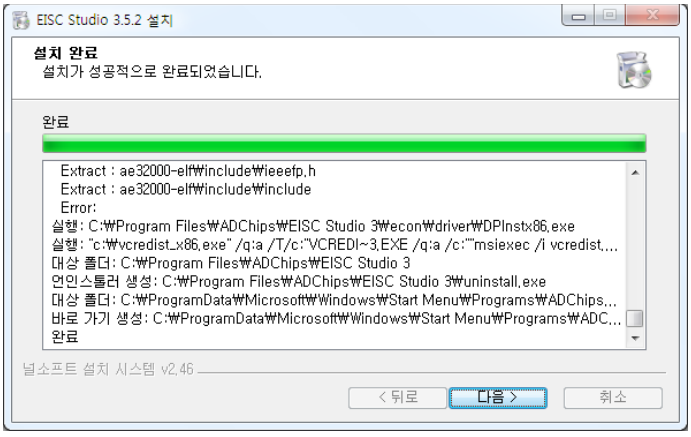
6. 장치 드라이버 설치가 완료되면 다음과 같은 창이 나오고, 마침을 눌러 EISC Studio3 의 남은 설치과정을 계속 진행한다.

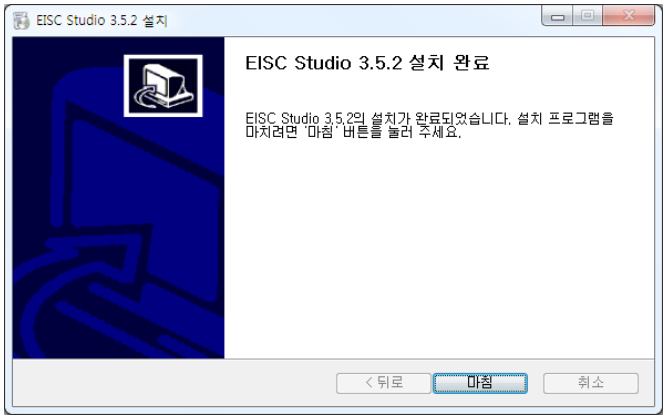


7. 다음과 같이 Microsoft Visual C++ 2010 재배포 가능 패키지를 설치하라는 창이 뜨면 동의함에 체크한 후 설치하면 된다. EISC Studio3 를 실행하는데 필요하므로 설치하고 넘어가도록 한다.



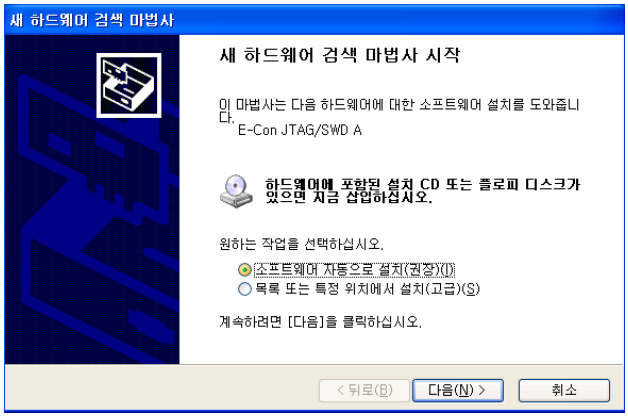
8. 설치가 완료되면 다음과 같은 창이 출력되고, 다음을 누르면 설치가 완료된다.



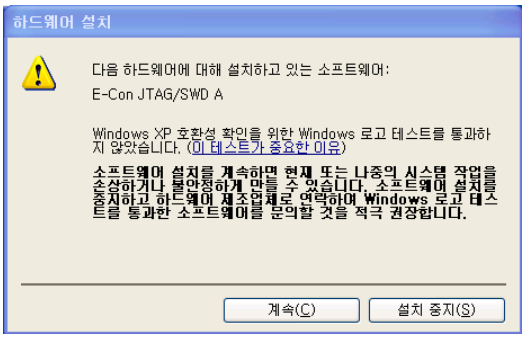


9. EISC Studio 3 설치가 완료 되면 다음으로 Download 장비인 E-Con driver 를 설치하여야 한다. EISC Studio 3 를 설치하면서 E-Con driver 파일을 복사하였다면 E-Con 을 컴퓨터와 연결하여 driver 설치를 진행하면 되고, E-Con driver 파일을 복사 하지 않았을 경우에는 EISC Studio 3 를 설치한 폴더에 econ₩driver 라는 폴더 안에 DPInstx86.exe(32bit) 또는 DPInstx64(64bit) 를 실행하여 드라이버 파일을 복사 하고, 설치를 진행하면 된다.
(DPInstx86.exe 또는 DPInstx64.exe 를 실행해서 설치 할 경우에도, 컴퓨터와 E-CON 을 연결하지 않은 상태에서 진행하도록 한다.)

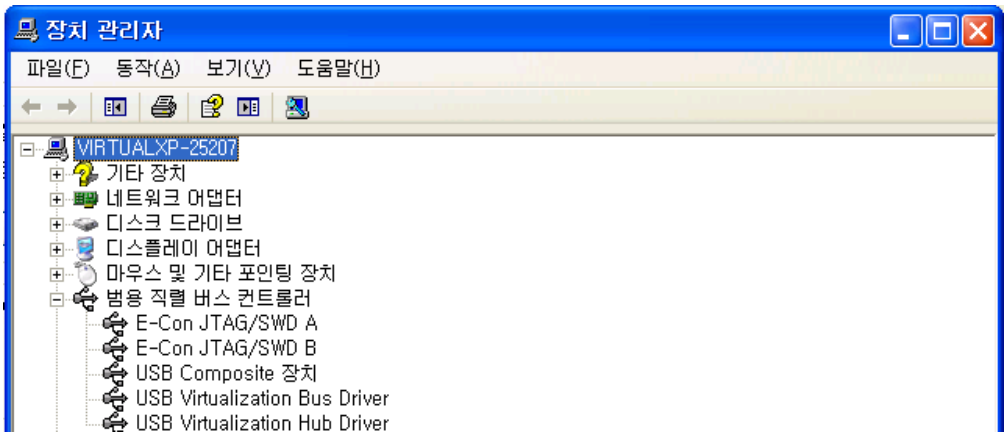
10. E-Con driver 파일 복사가 되었다면, 컴퓨터와 E-Con 을 연결하였을 때 Windows7 의 경우에는 자동으로 드라이버를 찾아 설치를 진행하게 된다. Windows XP 이하 버전에서는 다음과 같이 새 하드웨어 검색 마법사 창이 뜨게 되는데, "소프트웨어 자동으로 설치"를 체크하고 확인을 클릭하면 driver 설치가 진행된다.



설치 중 다음과 같은 경고 창이 뜨게 되는데, 계속을 클릭하면 된다.



11. E-Con 은 A,B 두 개의 채널로 되어 있어 위와 같은 방식으로 B 채널도 설치해주면 된다. 설치가 모두 완료 되면 장치관리자에서 다음과 같이 드라이버 설치가 완료 된 것을 확인할 수 있다. E-Con driver 설치와 관련된 좀 더 자세한 내용은 adchips 홈페이지(www.adc.co.kr) Product→System→E-CON→Download 의 "E-CON Driver Install Guide"를 참고하기 바란다.

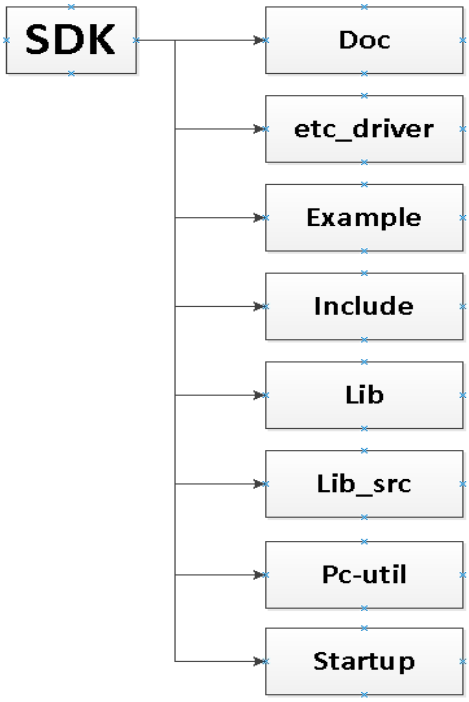


2. adStar SDK 활용

개발에 편의를 위해 adchips 에서는 adStar SDK 를 제공하고 있다. adStar SDK 는 adchips 의 홈페이지(www.adc.co.kr)의 Product →SoC→ ADSTAR → Tools & Software 에서 내려 받을 수 있다.

2.1 adStar SDK 구성 및 기본 정의

adStar SDK 는 다음과 같이 구성되어 있다.



Doc ➔ 현재 보고 있는 매뉴얼을 포함, adStar 와 관련된 문서가 들어있는 폴더.

Example ➔adStar STK Board 에서 동작하는 예제 프로그램 폴더.
(Example 폴더의 flash_data 에는 SDK 예제 프로그램이 동작하는데 있어 필요한 image, sound 파일이 들어있다.)

Include ➔adStar SDK 의 header file 폴더.

lib ➔adStar SDK 의 library file 폴더.

lib_src➔adStar SDK 의 library source file 폴더.
(최초에는 library 가 build 되지 않고 source 형태로만 존재하기 때문에, lib_src 폴더의 adStar.epx 프로젝트를 build 하여 lib 폴더에 libadstar.a 파일을 생성해 주어야 한다.
adStar.epx 를 open 한다음에 F7 또는 build menu 의 build project 를 실행하면 된다.)

Pc-util➔adStar 를 사용하는데 있어 필요한 Utility 폴더. adStar USB Driver 가 들어있다.

Startup ➔adStar 개발에 필요한 startup 코드와 link script 코드가 들어있는 폴더. 이 외에 STK board 기본 설정 코드도 들어있다.

adStar SDK 에는 사용하기 편하고 구분이 쉽게 변수형을 아래와 같이 정의해놓았다.
SDK Source 를 참조하는데 아래 표를 참고하기 바란다.

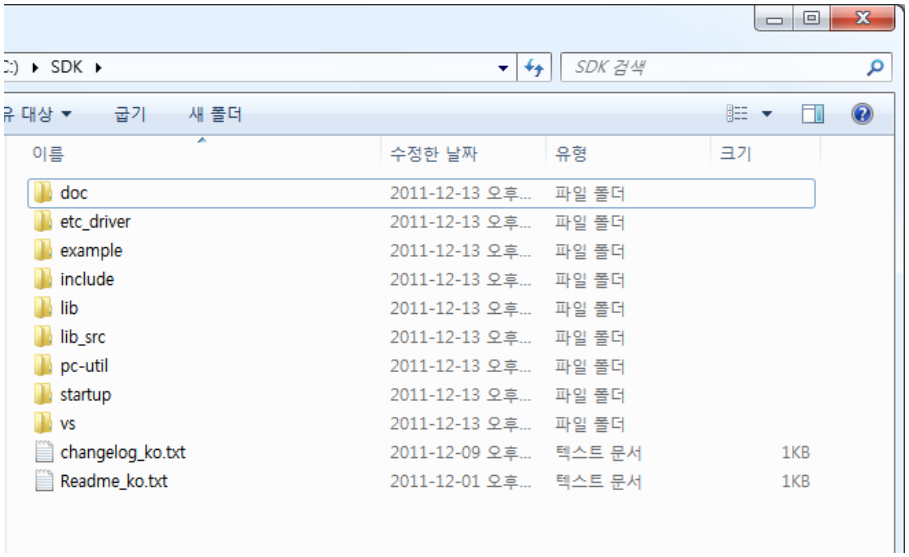
Signed char	S8, s8
Signed short	S16, s16
Signed int	S32, s32
Unsigned chr	U8, u8, __u8
Unsigned short	U16, u16, __u16, BYTE, uchar
Unsigned int	U32, u32, __u32, WORD, ushort
Unsigned long long	U64, u64, __u64, DWORD, ulong
Volatile unsigned char	vU8
Volatile unsigned short	vU16
Volatile unsigned int	vU32
Volatile unsigned long long	vU64
1	TRUE, true
0	FALSE, false

그리고 Register 의경우‘R_’ 접두어를 붙여 정의해놓았으므로 SDK Source 에서 R_이붙은명칭은 Register 로생각하면된다.
예를 들어 R_TM0CON 은채널 0 번 timer control register 이다.

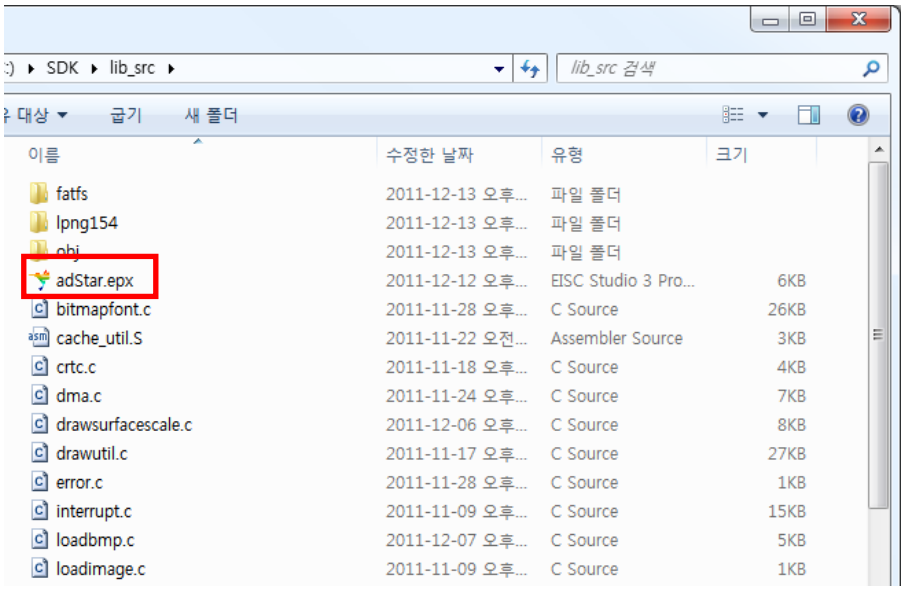
2.2 adStar SDK library build

adStar SDK 를 홈페이지에서 내려 받았으면, 제일 먼저 adStar SDK Library 를 build 해주어야 한다. adStar SDK Library source 는 lib_src 폴더에 있으므로, lib_src 폴더의 adStar.epx 파일을 열어서 build menu 의 build project 를 실행하거나, F7 을 눌러서 project build 를 진행하면 된다. Build 가 정상적으로 완료 되면, lib 폴더안에 libadStar.a 파일이 생성 된다.

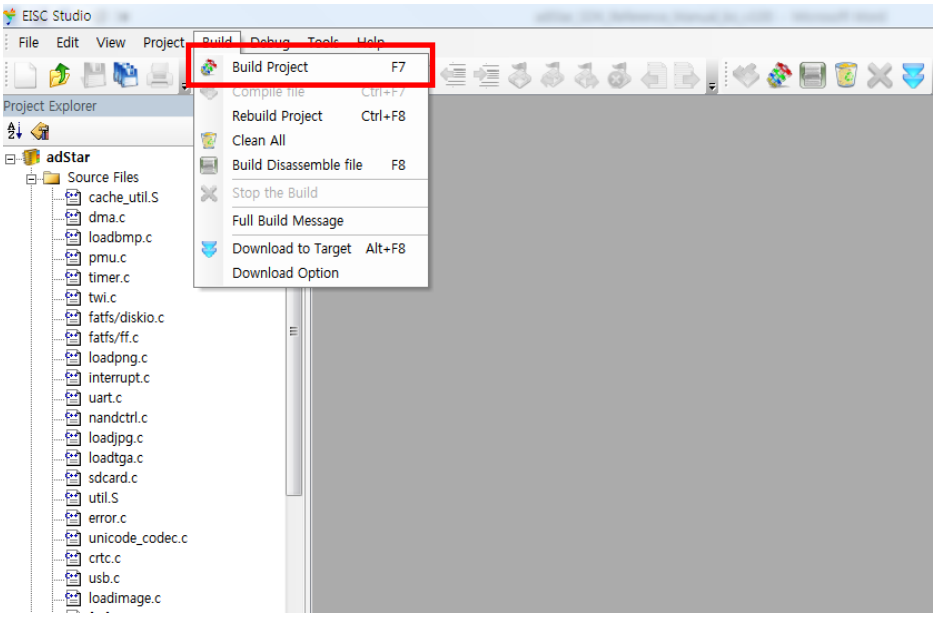
(adStar SDK 의 모든 예제들이 library 를 참조하므로 제일 먼저 해주어야 한다.)



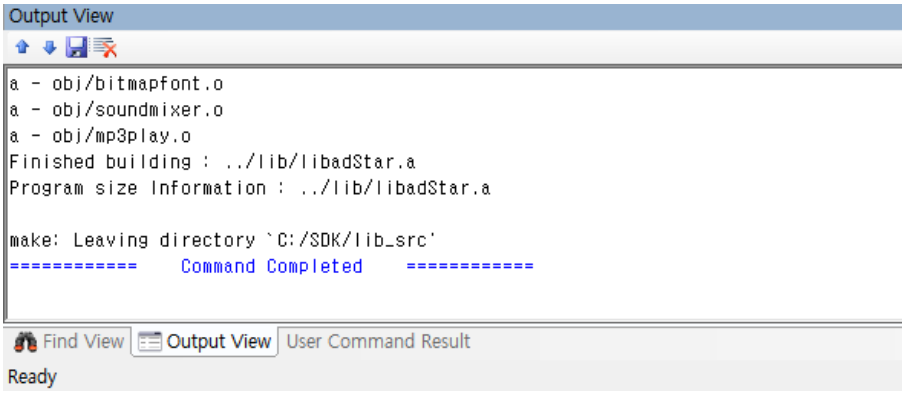
[lib_src 폴더]



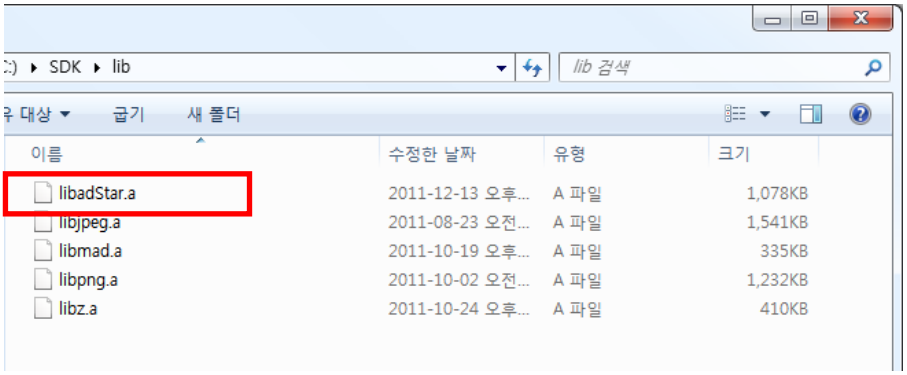
[adStar.epx 파일]



[Build Project]



[Build Project 정상 완료]



[libadStar.a 파일 생성]

만약에 adStar library 를 수정하게 되면 위처럼 다시 library 를 build 해주어야 한다.
그리고 adStar library 를 사용하는 program 도 다시 build 해주어야 수정 된 library 가 적용된다.

2.3 Demo Program 실행

adStar SDK example 폴더에는 adStar STK Board 에서 동작하는 예제들이 준비 되어 있어서, STK Board 에서 adStar 동작을 쉽게 확인 해 볼 수 있다.

이번 장에서는 SDK 의 example 중 Demo Project Source 를 build, download, 실행하는 방법에 대해 알아보도록 하겠다.

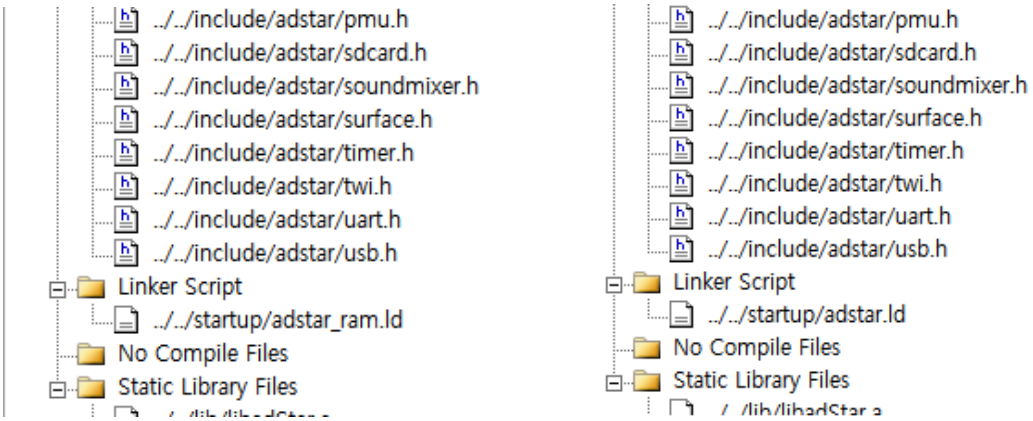
우선 Demo 예제를 동작 시키기 위해서는 Nand Flash 에 출력될 이미지 및 사운드 파일이 있어야 한다. Nand Flash 에 파일을 복사하기 위해서는 Example 폴더의 usb_mass_storage project 를 실행하거나 bootloader 의 mass storage mode 를 사용하면 되는데, bootloader 는 다음 장에서 설명하도록 하고, 이번 장에서는 usb_mass_storage project 를 사용하여 진행하는 방법에 대해 설명하도록 하겠다. usb_mass_storage 를 실행하기 위해 우선 project 를 build 하고, download 하는 방법에 대해서 알아보자.

2.3.1 Project build & download

example 폴더의 usb_mass_storage 폴더를 열어 usb_mass_storage.epx 파일을 마우스로 더블클릭하여 project 를 open 한다.

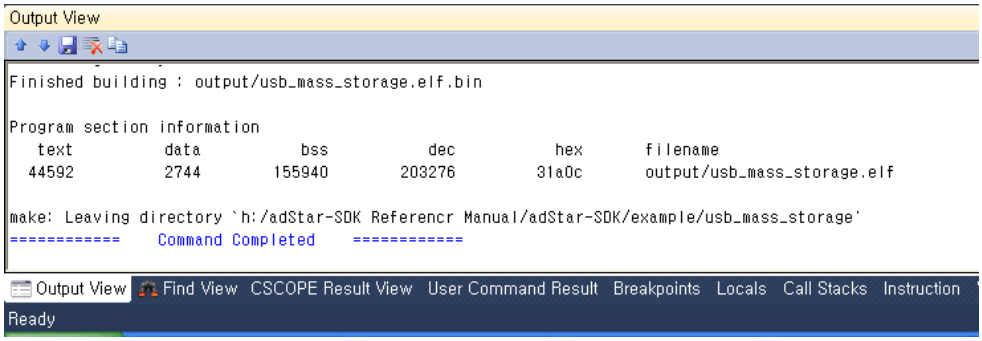
Project 를 open 한 후 build 하여 board 에서 실행되는 .bin(binary)파일을 만들기 전에 좌측 Project Explorer 창의 Linker Script 에 있는 파일을 변경 해 주어야 한다. SDK 에서 제공되는 Linker Script 파일은 두 가지가 있는데, 하나는 adstar.ld 파일이고, 다른 하나는 adstar_ram.ld 파일이다. SDK 에 example 에는 기본적으로 adstar_ram.ld 파일을 사용하도록 되어 있는데, adstar_ram.ld 파일은 프로그램을 SDRam 에서 동작시킬 때 사용하는 Linker Script 파일로 bootloader 가 올라간 상황에서 프로그램을 동작시킬 때 사용할 수 있다. 이번 장에서는 bootloader 없이 프로그램을 동작시키는 방법에 대한 설명으로, adstar_ram.ld 파일 대신 adstar.ld 파일을 사용한다.

따라서 Linker Script 에서 마우스 오른쪽을 클릭하여 Set Linker Script File 을 선택하여, startup 폴더의 adstar.ld 파일을 선택해 준다.



< adstar_ram.ld 파일을 adstar.ld 파일로 변경 >

Linker Script 를 변경 하였으면, build → Build Project 를 실행하거나 F7 키를 눌러서 project 를 build 한다. build 가 완료되면 Output View 에 다음과 같이 출력된다.

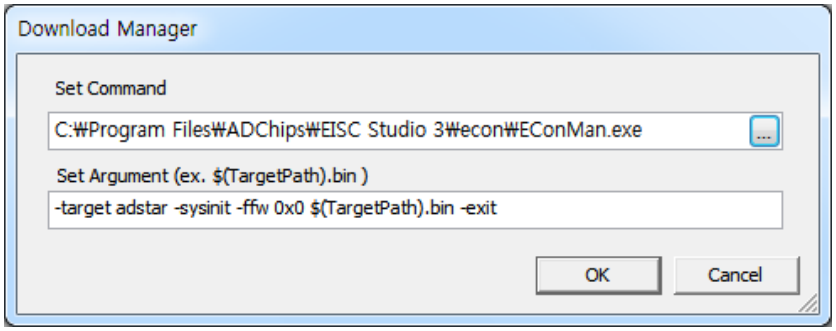


[Output View]

Build 가 정상적으로 완료 되면, output 폴더에 usb_mass_storage.elf.bin 파일이 생성 된 것을 확인할 수 있다.

다음으로 build 로 생성된 usb_mass_storage.elf.bin 파일을 STK board 로 download 하는 방법에 대해 알아보겠다. Download 는 download 할 project 를 EISC Studio3 에 열어 놓은 상태에서 E-CON 을 사용하여 다음의 순서대로 진행하면 된다.

1. Download 를 하기 위해 adStar board 와 E-CON 장비를 준비한다.
   (EISC Studio 3 가 설치되어 있어야 하고, E-CON Driver 가 설치되어 있어야 한다. 설치 방법은 1 장의 Software 개발환경을 참고하기 바란다.)
2. adStar board 와 E-CON 장비를 연결한 후 board 의 전원을 켠다.
3. Build 메뉴의 Download Option 을 클릭한다. 그러면 다음과 같이 Download Manager 창이 뜨는데, Set Command 에는 EISC Studio3 를 설치한 폴더에서 econ 이라는 폴더 안의 EConMan.exe 라는 E-CON 용 다운로드 프로그램을 선택해준다. (default 값인 EconMan.exe 로 하여도 된다. 그리고 Set Argument 에는 "–target adstar –sysinit –ffw 0x0 $(TargetPath).bin -exit" 이라고 입력하고 OK 를 클릭한다.



Argument 에 대해 설명하면,

-target 은 말 그대로 download 할 target 을 말하며 adStar 로 적어주면 된다.

-sysinit 는 download 를 위한 초기화를 해주는 command 이다.

-ffw 는 build 결과물로 나온 bin 파일을 다운로드 하는 command 로 0x0 은 download 할 주소이고, $(TargetPaht).bin 은 download 할 파일명으로 $(TargetPath).bin 은 현재 열려있는 project 의 bin 파일을 뜻한다.

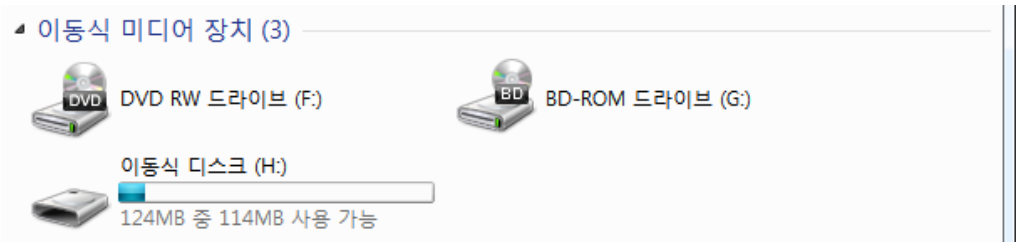
-exit 는 download 가 완료되면 자동으로 종료하라는 command 이다.

(Set Argument 에 대한 자세한 내용은 www.adc.co.kr→ Product → System → E-CON → Download → E-CON.pdf 파일을 참고하기 바란다.)

4. Build 메뉴의 Download to Target 을 클릭하면 download 가 진행이 된다.

2.3.2 Nand Flash(SD Card)에 파일 복사하기 (Mass Storage Mode 사용하기)

Download 가 완료 된 후 usb cable 을 컴퓨터와 STK board 의 device 단자에 연결을 한 후 board 에 전원을 켠다. 전원을 켜면 다음과 같이 새로운 저장 장치가 나타난다. 이 저장장치가 board 의 Nand Flash 가 잡힌 것으로 이곳에 파일을 복사하면 NandFlash 에 저장되어 adStar 프로그램에서 사용할 수 있다.



Demo Project 에서 사용하는 이미지 및 사운드 파일은 example 폴더의 flash_data 폴더 안의 내용을 복사하면 된다.

만약 Nand Flash 대신 SD Card 에 파일을 복사하기 위해서는 usb_mass_storage project 의 mass_stor.c 파일을 열어 상단에 주석으로 처리되어 있는 #define SDCARD_STORAGE 를 주석을 푼 다음 build 한 후 board 에 download 하면 위와 같은 방법으로 SD Card 에 파일을 복사할 수 있다.

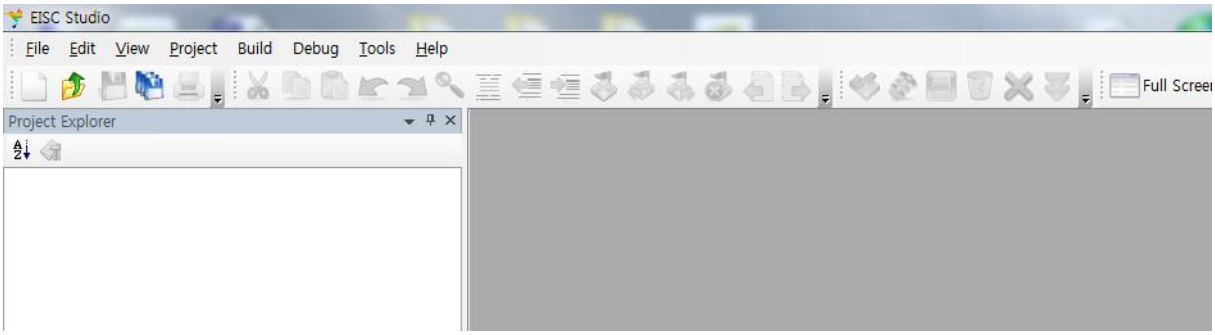
2.3.3 Demo Program build & download

Nand Flash 에 Demo Project 에서 사용하는 이미지 및 사운드를 복사하였으면 example 폴더의 Demo Project 를 열어서 이전 usb_mass_storage project 때와 마찬가지로 Linker Script 를 adstar.ld 파일로 변경한 후 build 하여 E-Con 으로 download 하면 Demo 동작을 확인할 수 있다.

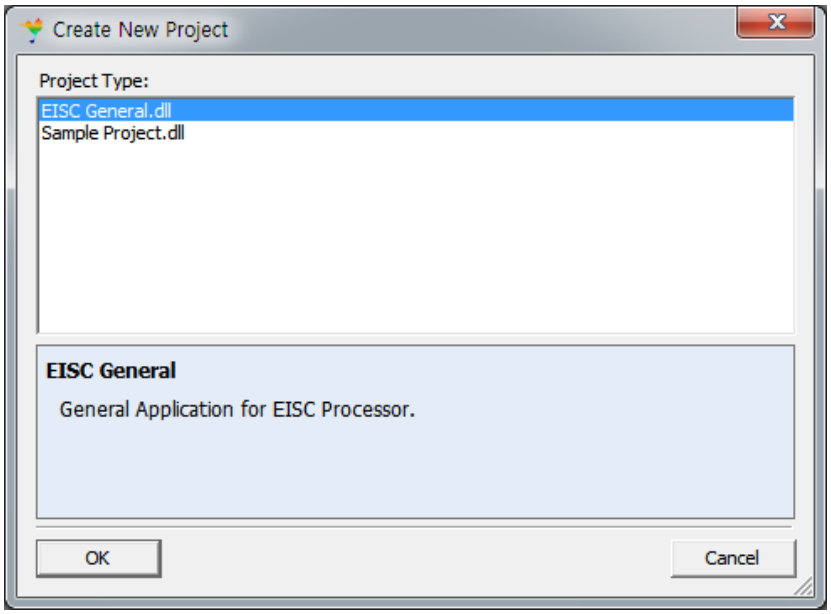
2.4 adStar새프로젝트만들기

이번 장에서는“Hello adchips!”라는 문장을
UART 를통해출력하는프로그램을작성하면서새프로젝트를만드는법에대해알아보겠다.

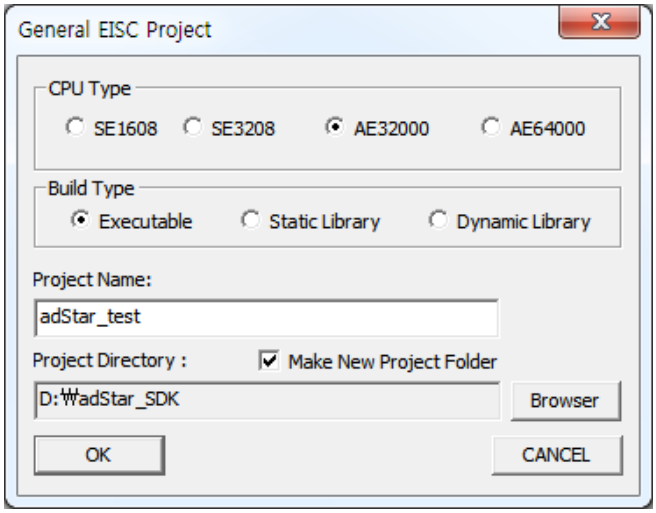
1. 먼저 EISC Studio3 를실행한다.



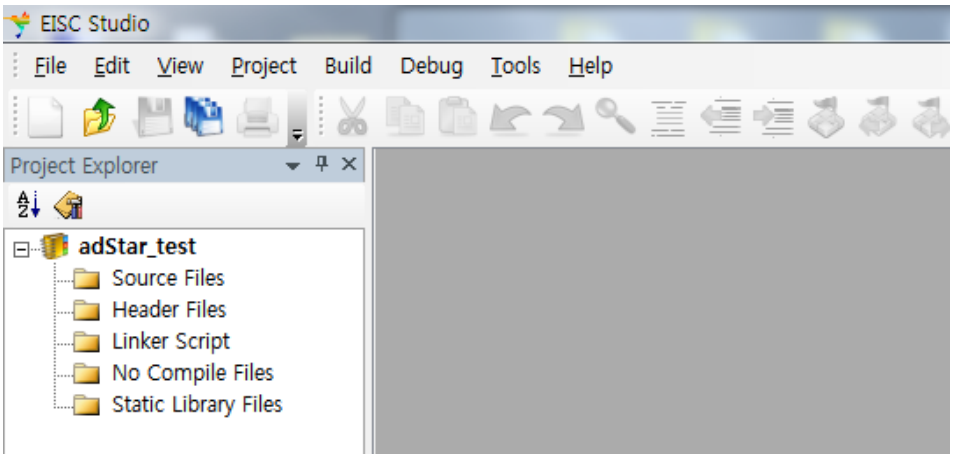
2. 메뉴의 File → New → Project 를 클릭하면 다음과 같은 창이 나오는데, 아래 그림과 같이 EISC General.dll 을선택한후 OK 버튼을 누른다.



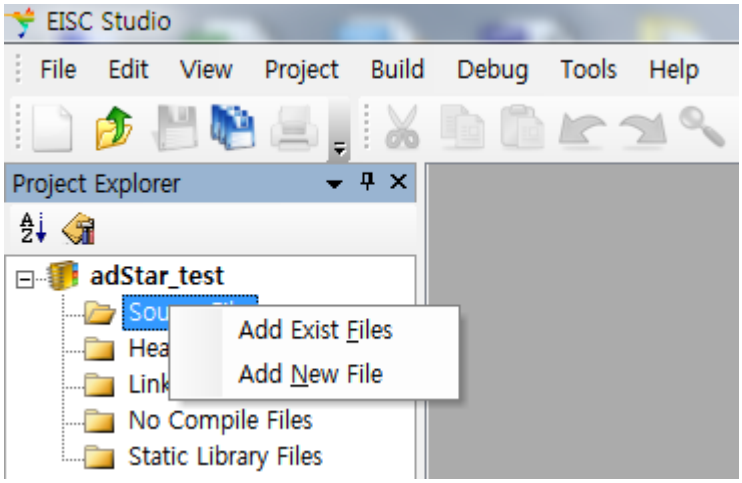
3. 다음과 같이, project 설정창이출력되는데, CPU Type, Build Type, Project Name, Project Directory 를결정해주어야한다.
   adStar 는 CPU Type 으로 AE32000 을선택해주면되고, 동작프로그램을 작성할 것이기에 Build Type 으로 Executable 을선택해준다. 다음으로 Project Name 과 Project 를생성할위치를정해주면된다. Project Directory 옆에 위치한 체크박스(Make New Project Folder)는 Project Name 과같은이름의폴더를생성유무를결정하는것으로, 체크하게 되면 Project Directory 로지정해놓은폴더에 Project Name 과같은이름의폴더를생성한후그안에 Project file 을생성해준다. 체크하지 않을 경우에는 Project Directory 로지정해놓은폴더에 Project file 을생성한다. 모든 설정이 끝나면 OK 를클릭하여새프로젝트생성을완료한다.



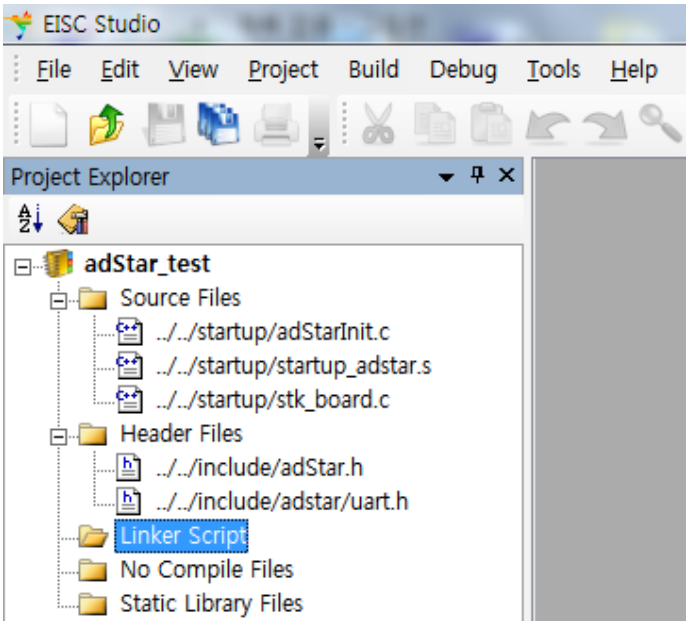
4. 새로운 프로젝트가 생성되면 다음과 같이 Project Explorer 창에 Projet 명과 Project tree 를 확인할 수 있는데, 이곳에 adStar 용 project 에 기본적으로 필요한 파일들을 추가해 주어야 한다.



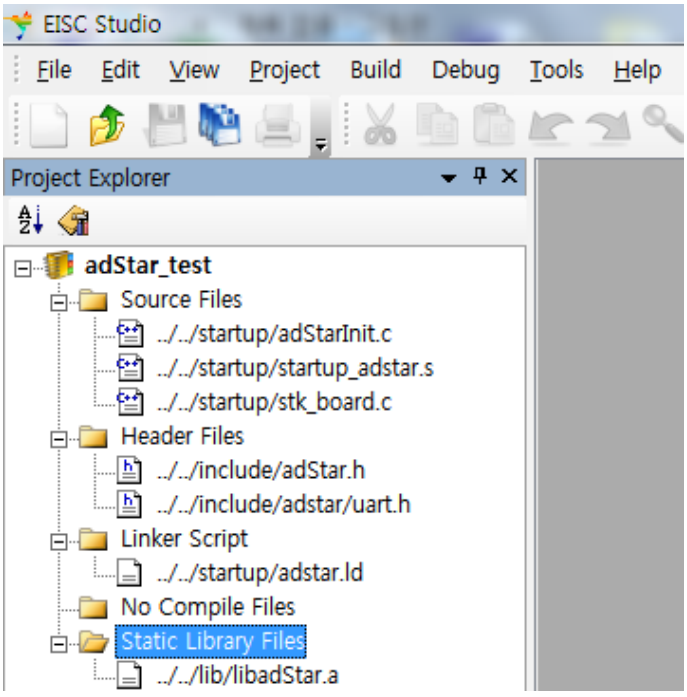
5. 먼저 Source Files 에서 마우스 오른쪽 키를 클릭하여 Add Exist Files 를 선택하여 파일을 추가한다. 추가할 파일은 SDK 의 startup 폴더에 있는 startup_adstart.s, adStarinit.c, stk_board.c 이다. (stk_board.c 파일은 adStar 용 stk board 를 사용할 경우 추가해주면 된다.)



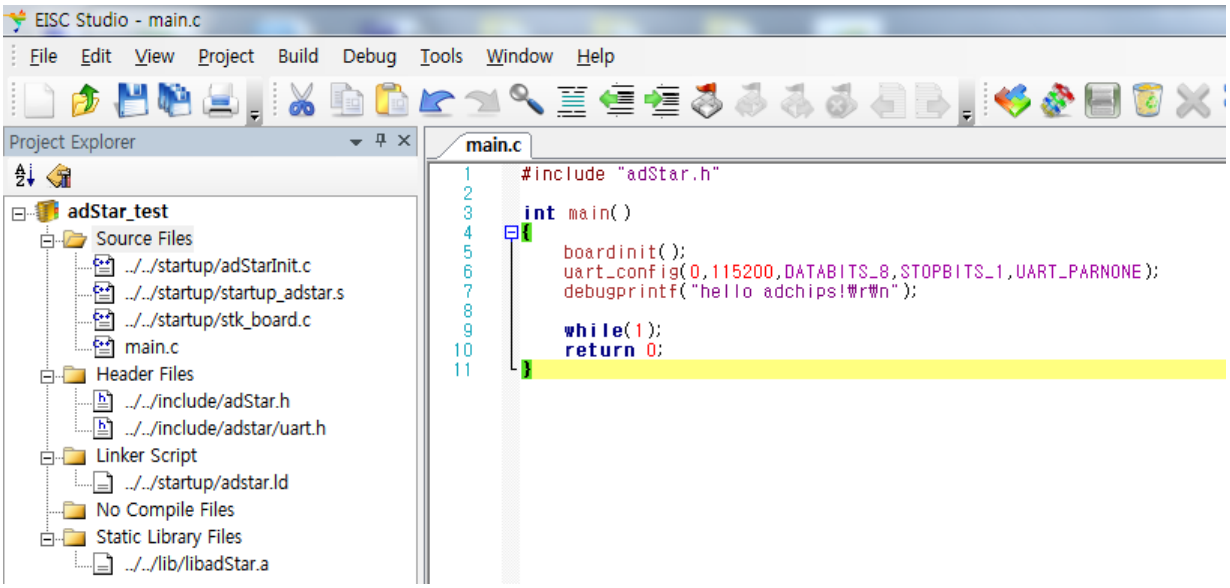
6. 다음으로 Source Files 와 마찬가지로 Header Files 에서도 마우스 오른쪽 키를 클릭하여 include 폴더에 있는 adStar.h 파일을 추가해준다. adStar.h 파일만 추가해줘도 되지만 프로그램 작성 시 함수 사용에 도움을 받기 위해서는 include/adstar 폴더에 있는 관련 header 파일도 같이 추가해주면 좋다. 여기서는 UART 를 사용하는 프로그램을 작성 해 볼것이므로, UART.h 파일도 추가하도록 하겠다.



7. 다음으로 Linker Script 에 startup 폴더에 있는 adstar.ld 파일을 추가하고, Static Library Files 에 lib 폴더에 있는 libadStar.a 파일을 추가한다. 그리고 지금 작성하고 있는 UART 예제프로그램에는 필요없지만, MP3 재생이나, JPG 이미지 파일을 출력하는 프로그램 작성시에는 libmad.a 파일과 libjpeg.a 파일도 추가해주어야 한다.

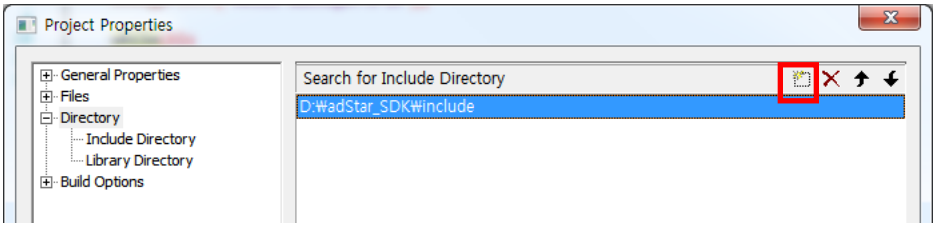


8. 위처럼 추가를 다 했으면 Source Files 에서 마우스 오른쪽 키를 누르고 Add New File 을 클릭 main 프로그램을 작성할 main.c 를 생성해주면 된다. 다음은 main.c 를 생성하고, UART 를 통해 Hello adchips!라는 문자열을 출력하는 코드를 작성 한 것이다.

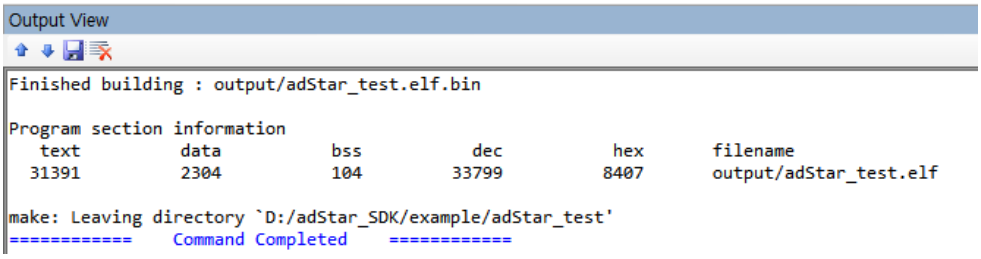


9. 코드를 살펴보면, 제일 먼저 boardinit() 함수를 호출하여 사용하는 board 에 맞게 pin 설정을 한다. (boardinit()함수는 사용하는 board 마다 다르므로 board 에 맞게 pin 설정을 해주어야 한다.) 다음으로 uart_config()함수를 호출하여 UART 를 설정한 값대로 초기화 해준다. 그리고 debugprintf()함수를 사용하여 "hello adchips!"를 출력한다.

10. 코드작성을 마쳤으면, project 를 build 해야하는데, 그 전에 먼저 project 설정을해주어야한다. Project Explorer 창에서 project 명에서마우스오른쪽을클릭하여 Properties 를클릭한다. 그러면 다음과 같은 창이 나오는데, Directory 에서 Include Directory 에 SDK 의 include 폴더를 추가해준다. (우측상단의 점선박스를 클릭하면 폴더를 추가할 수 있다.)



11. include 폴더까지 추가를 했으면 Build 메뉴에서 Build Project 를 클릭하거나, F7 을 누르면 Project Build 가 이루어진다. Build 가 정상적으로 완료 되었다면, Output View 창에 다음과 같이 출력되고, output 폴더에 project name.elf.bin 파일이 생성된다.



12. 생성된 bin 파일을 E-Con 을 사용하여 STK board 에 download 하면 Uart 0 번채널을 통해 hello adchips!가 출력되는 것을 확인 할 수 있다.

3. Bootloader

이번 장에서는 adStar Bootloader 에 대해 설명한다.

adStar 내부에 Flash 를 포함한 것과, Flash 를 포함하지 않은 칩을 위해, 내부 Flash Booting 과 Nand Booting 이 가능하다. 이에 맞춰 Bootloader 도 Flash Booting 을 위한 bootloader 와 Nand Booting 을 위한 Nand Boot Code 를 지원한다.

3.1 Bootloader

Bootloader 는 Serial Flash 를 사용한 Flash Boot Mode[3]일 경우에 사용 할 수 있도록 제작된 booting 프로그램으로, SDK Example 의 bootloader 폴더 안에 있으며, adStar Booting 기능 외에 다음의 기능을 제공한다.

가. Remote Communication Mode.
나. Mass Storage Mode.
다. Execute Mode.
라. User Define Mode

Remote Communication Mode 는 RemoteManCLI 프로그램을 사용하여 adStar 와 Communication 할 수 있는 mode 로, 주로 USB 를 사용하여 application 프로그램을 DRAM 으로 download 하여 실행할 때 사용한다. application 프로그램의 동작을 빠르게 확인해 볼 수 있는 장점이 있다. 단 DRAM 에 download 하는 것이기 때문에, 동작을 확인할 때마다 download 해주어야 한다.

Mass Storage Mode 는 example 의 usb_mass_storage 와 같은 동작을 수행하는 Mode 로 adStar board 의 Nand Flash 를 usb memory 와 같이 컴퓨터에서 저장 장치로 인식되도록 해준다. 따라서 bootloader 가 adStar 칩에 download 되어 있으면, Nand Flash 에 file 을 복사하기 위해 usb_mass_storage 예제를 download 하지 않고도, bootloader 의 Mode 만 변경하여 file 을 복사할 수 있다.

Execute Mode 는 실행 모드로, Flash 의 0x14000(sector number : 20)번지에 저장된 실행 프로그램을 Dram 으로 복사한 후 복사한 프로그램을 실행하는 Mode 이다.

마지막으로 User Define Mode 는 사용자가 원하는 동작을 코딩하여 사용하도록 해 놓은 Mode 인데 Bootloader 에서는 기본적으로 Execute Mode 와 비슷한 동작을 하도록 프로그램 되어 있다. Execute Mode 와 다른 점은 내부 Flash 에서 실행 프로그램을 복사하는 것이 아니라, Nand Flash(SD Card)에 boot.bin 이라는 파일을 Dram 으로 복사한 후 실행하도록 되어 있다.

자세한 동작 및 사용법에 대해서는 뒤에 설명하도록 하겠다.

[3] adStar의Boot Mode 중 하나로 Serial Flash를 사용하여 booting 하는 mode이다. 자세한 사항은 datasheet를 참고하기 바란다.

3.1.1 Bootloader 사용하기

Bootloader 는 SDK example 안에 들어있다. 다른 application 과 마찬가지로 project 를 열어서 build 후에 adStar 에 download 하면 동작을 확인할 수 있다.

Bootloader project 를 열었으면 build → Build Project 를 실행하여 프로그램을 build 한다. Build 가 완료되면, E-Con 을 board 와 연결한 후 전원을 켜고, build → Download to Target 을 실행하면 download 가 진행된다.

(build 및 download 의 좀더 자세한 설명은 본 매뉴얼의 2.3.1 장을 참고한다.)

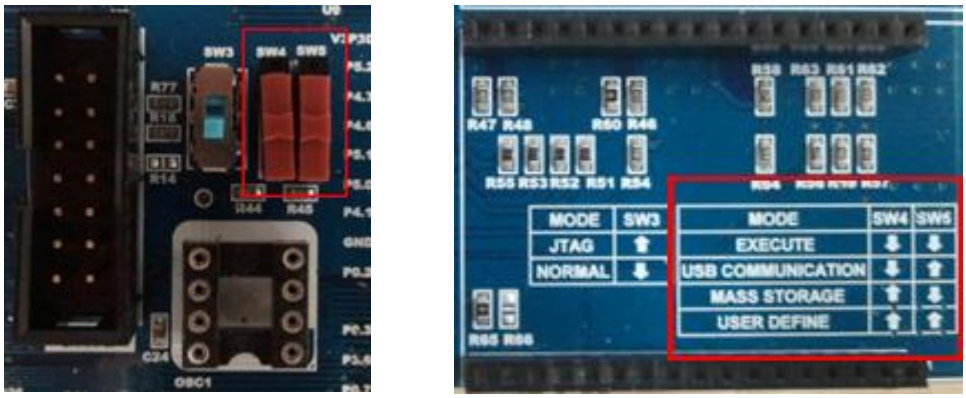
Bootloader 에 몇 가지의 모드가 있는데, 코드상에서 다음 부분이 모드를 설정하는 부분이다.

```
67      *R_GPIDIR(0)=0xc;//mode switch
68      int mode = ((*R_GPILEV(0))>>2)&3;
69
70      switch(mode)
71      {
72      case 0: //execute mode
73          bin_execute();
74          break;
75      case 1:
76          usb_clock_init();
77          mass_storage_main();
78          break;
79      case 2:
80          RSP_Run();
81          break;
82      case 3:
83          debugstring("User Define Mode \r\n");
84          bin_execute_fat();
85          break;
86      }
```

< bootloader project 의 main.c >

코드를 보면 P0.2 와 P0.3 PIO 를 입력으로 받아 mode 를 선택하게 되어 있는데, STK 보드상에서는 SW4 번과 SW5 번에 연결되어 있다.

(STK board 와 다른 PIO 를 사용하고 싶다면, 위 코드를 수정하여 사용자가 원하는 PIO 를 사용하면 된다.)



<SW4, SW5 과 SW setting 방법>

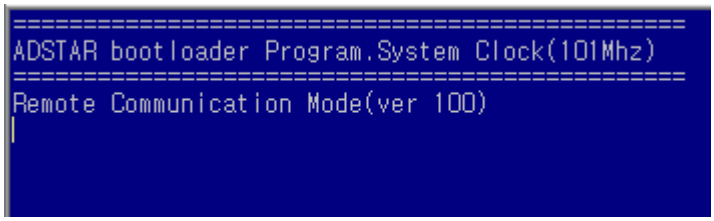
위의 사진에서 보듯이 booting mode 별 SW 설정은 다음과 같다.

MODE	SW4	SW5
Execute Mode	down	down
USB Communication Mode	down	up
Mass Storage Mode	up	down
User Define Mode	up	up

3.1.2 Bootloader Mode

가. Remote Communication Mode

Remote Communication Mode 는 RemoteManCLI 프로그램을 사용하여 adStar 와 Communication 할 수 있는 mode 로, 주로 USB 를 사용하여 application 프로그램을 DRAM 으로 download 한 후 바로 실행하여, 동작을 빠르게 확인해 볼 때 사용한다. Remote Communication Mode 로 부팅을 하면 다음과 같은 debug message 가 출력된다.



[Remote Communication Mode]

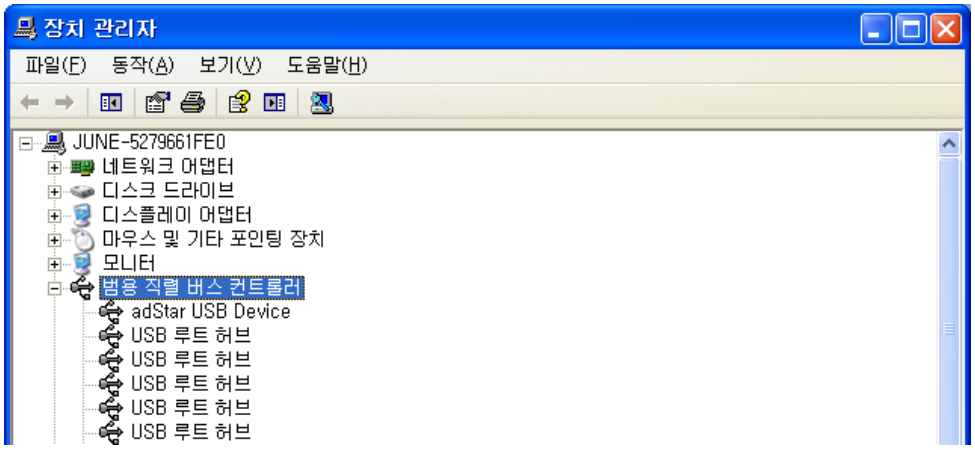
Remote Communication Mode 는 adStar USB device 를 사용하기 때문에 adStar USB Driver 를 설치해주어야 된다. USB Driver 는 pc-util/usb_driver 폴더에 들어있는데, 설치하는 방법에는 두 가지가 있다.

하나는 usb_driver 폴더에 있는 DPInstx86.exe(DPInstx64.exe)[4]파일을 이용하는 방법이고, 다른 하나는 windows 의 장치관리자에서 직접 usb_driver 폴더의 driver 파일을 직접 찾아 설치해주는 방법이다. 이 문서에서는 DPInstx86.exe(DPInstx64.exe)파일을 사용하기를 권장한다.

DPInstx86.exe(DPInstx64.exe)파일을 사용하여 usb driver 를 설치하는 방법은 다음과 같다.

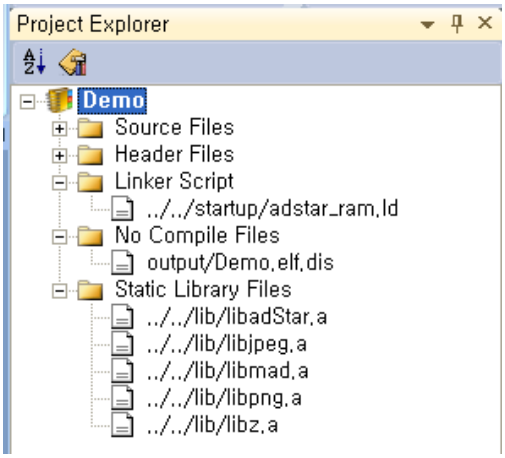
1. Usb_driver 폴더의 DPInstx86.exe(DPInstx64.exe)파일을 실행하여 설치를 진행한다.
2. 설치가 정상 적으로 완료되면, usb 를 연결하고, bootloader 의 Remote Communication Mode 로 부팅을 한다.
3. 새로운 장치를 찾았다고 화면의 우측 하단에 표시되면서, driver 설치를 시작하게 된다. Windows7 을 사용하는 사용자라면 dpinst.exe 에 의해서 driver 가 복사되었기 때문에 자동으로 설치가 이루어진다. Windows xp 를 사용하는 사용자라면 "새 하드웨어 검색 마법사"창이 나타나는데, "소프트웨어 자동으로 설치(권장)"을 선택한 후 다음을 누르면 역시 driver 파일을 찾아 자동으로 설치가 이루어진다.
4. 만약 자동으로 설치가 이루어지지 않는다면, "목록 또는 특정 위치에서 설치(고급)"을 선택해서 usb_driver 폴더를 선택해주거나, 장치관리자에서 driver update 를 해서 usb_driver 폴더를 선택해주어야 한다.
5. USB driver 설치가 완료되면, 장치관리자에서 adStar USB driver 가 설치된 것을 확인할 수 있다.

[4] 32 bit OS 사용시 DPInstx86.exe를 사용하고, 64 bit OS 사용시 DPInstx64.exe를 사용한다.



[장치관리자]

USB Driver 설치를 완료했으면 Remote Communication Mode 에서 example 폴더에 있는 application 프로그램을 Dram 에 download 하여 바로 확인을 해 볼 수 있는데, Dram 에서 application 을 동작시키기 전에 application project 를 열어 link script 파일을 확인하도록 한다.



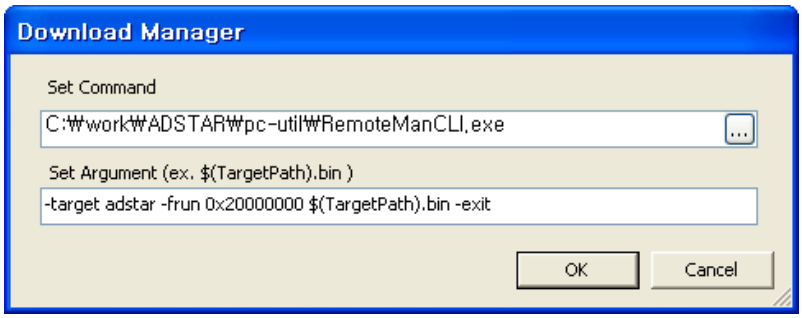
[Linker Script]

Demo Project 는 기본적으로 Linker Script 에 ../../startup/adstar_ram.ld 파일로 되어 있는데, 확인하였을 때 adstar_ram.ld 로 되어 있으면 프로그램이 Dram 에서 동작하도록 되어 있는 것이기 때문에 그대로 build 하면 된다. 만약 adstar.ld 파일로 되어 있다면, 이는 Dram 이 아닌 flash 에서 동작하도록 되어 있는 것으로 startup 폴더의 adstar_ram.ld 파일로 변경해야 한다.

정리하면 Flash 에서 동작하는 프로그램은 link script 에 adstar.ld 파일을 load 하면 되고, Dram 에서 동작하는 프로그램의 경우에는 link script 에 adstar_ram.ld 파일을 load 하면 된다.

같은 프로그램에서 link script 파일만 변경하여 Flash 또는 Dram 에서 동작하는 binary 를 생성할 수 있다. (bootloader 를 사용할 경우 Dram 에서 application 을 동작시키므로 Bootloader 를 사용할 경우 특별한 경우가 아니라면 application 의 link script 는 adstar_ram.ld 파일을 사용하면 된다.)

Link script 를 확인하였으면 다음으로 Dram 에 download 하여 실행하는 방법에 대해 알아보겠다. build 한 후에 adStar 를 Remote Communication Mode 로 부팅한다. 그런 후에 ECon 으로 download 할 때와 마찬가지로 build→download option 을 실행한다.



[build -> download option]

Dram 에 download 하기 위해서는 EConMan 프로그램이 아닌 RemoteMenCLI 라는 프로그램을 사용해야 하는데, pc_util 폴더에 있으므로 찾아서 선택하면 된다.

Argument 는 EConMan 과 비슷한데, 설명하면 다음과 같다.

-**target adstar**

➔ EConMan 과 동일하게 target 명을 명시하는 부분으로 adstar 로 지정한다.

-**frun 0x20000000 $(TargetPath).bin**

➔ binary file 을 downlaod 하고 실행하는 명령어로, Dram 영역인 0x20000000 에 현재 project 에서 생성된 binary 파일을 download 한 후 실행한다.

-**exit**

➔ 모든 동작이 정상적으로 이루어 졌으면 프로그램을 종료한다.

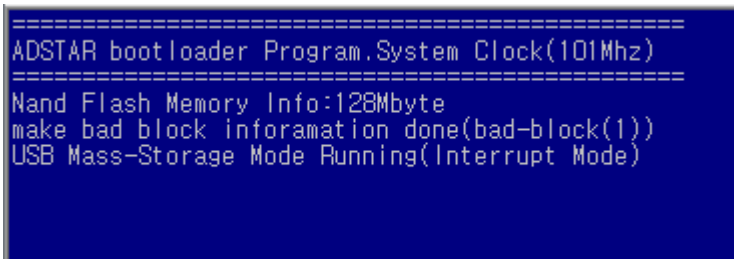
위처럼 download option 을 설정 한 후에, Remote Communication Mode 상태에서 build→download to target 을 실행하면 현재 열려져 있는 project 의 binary 파일 download 가 진행되고, 진행이 완료되면 바로 실행한다.

처음에 설명 했듯이, Remote Communication Mode 상태에서 application 을 빨리 확인을 해 볼 수 있지만, Dram 에 download 하여 실행하는 것으로, reset 시 다시 download 해야 한다는 점을 알고 있어야 한다. reset 시에도 항상 같은 동작을 하도록 하고 싶은 경우에는 Flash 에서 application 을 동작하도록 하거나, 뒤에 설명할 Execute Mode 를 사용하면 된다.

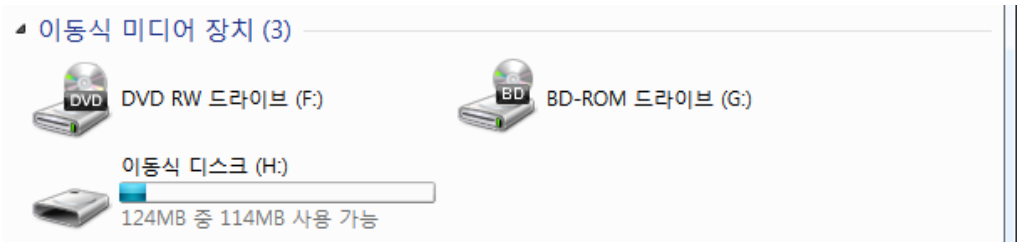
나. Mass Storage Mode

Mass Storage Mode 는 example 의 usb_mass_storage 와 같은 동작을 수행하는 Mode 로 adStar board 의 Nand Flash 를 usb memory 와 같이 컴퓨터에서 저장 장치로 인식되도록 해준다. 따라서 Mass Storage Mode 를 이용하여 usb_mass_storage 프로그램을 새로 download 할 필요 없이 Nand Flash 를 저장 장치로 사용할 수 있다. Mass Storage Mode 로 부팅을 하면 다음과 같이

debug message 를 출력하고, usb_mass_storage 와 같이 이동식 디스크 장치라는 저장장치가 생성된다.



[Mass Storage Mode]

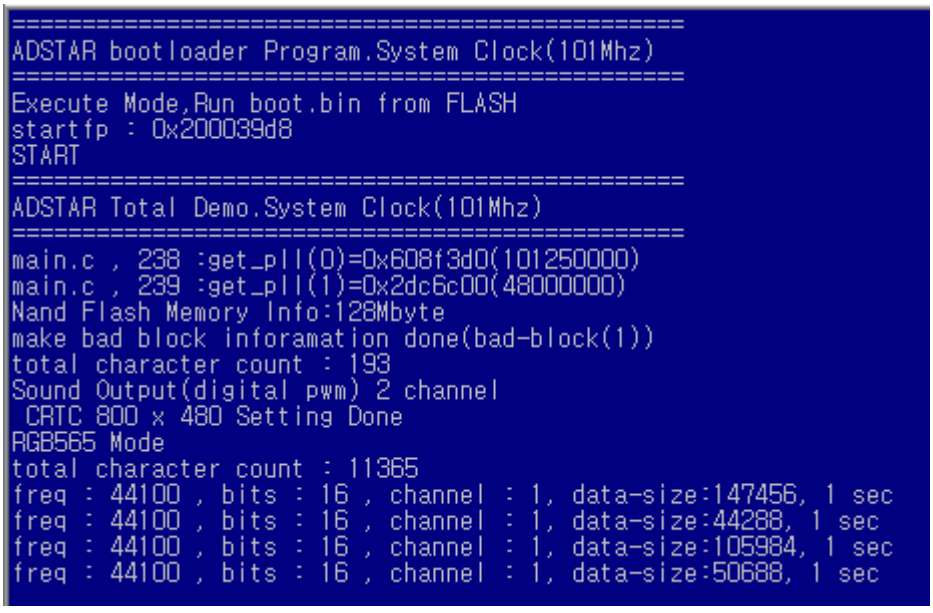


[이동식 디스크 저장 장치]

usb_mass_storage 와 마찬가지로 이 저장장치에 이미지 및 사운드 파일을 복사할 수 있다.

다. Execute Mode

Excute Mode 는 booting 과 동시에 application 이 동작하는 모드로, 주로 application 개발이 완료 되었을 때 사용하는 모드이다. Excute Mode 로 booting 을 하면 다음과 같이 터미널 창에 출력되는 것을 확인할 수 있다.



[Execute Mode]

Execute Mode 는 내부 Flash 의 data 를 Dram 으로 복사하여 실행하는 형태는 build 하여 생성된 project_name.elf.bin 파일을 지정된 내부 Flash 영역에 download 한 후에 Execute mode 로 부팅을 하면 application 이 동작을 한다. Dram 으로 복사되어 실행될 프로그램이기에 link script 는 adstar_dram.ld 파일을 사용한다.

지정된 내부 Flash 영역은 20 번째 섹터로 address 로 하면 0x14000 번지이다. bootloader 에서 0x14000 번지에서부터 data 를 dram 에 복사하여 실행하게 된다

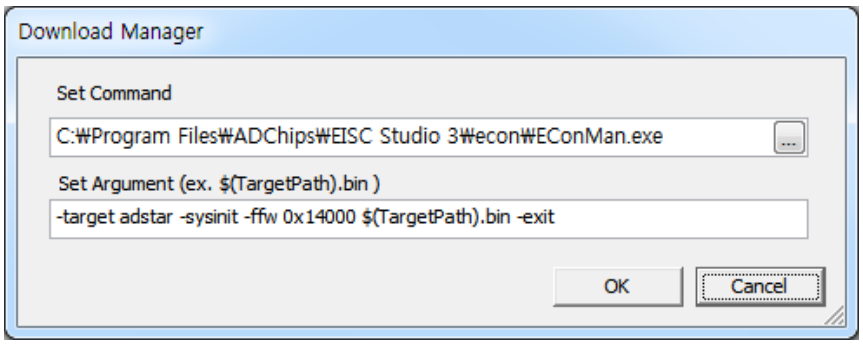
```
46  #define FLASH_APP_OFFSET  (1024*4*20)
47  static void bin_execute()
48  {
49      void (*entryfp)();
50      debugstring("Execute Mode,Run boot.bin from FLASH ₩r₩n");
51      memcpy((void*)0x20000000,(void*)FLASH_APP_OFFSET,(512*1024)-FLASH_APP_OFFSET);
52      dcache_invalidate_way();
53      entryfp = (void(*)())*(U32*)0x20000000;
54      debugprintf("startfp : %#x₩r₩n",entryfp);
55      entryfp();
56  }
```

[bootloader main.c]

Main.c 의 46 번재 줄을 보면 Flash 영역의 시작 주소가 정의 되어 있는데, bootloader 의 size 가 커져서 영역이 겹치게 되면 정의 값을 수정해주어야 한다.

내부 Flash 의 0x14000 번지에 download 하는 방법은 다음의 두 가지 방법 중 하나를 사용하면 된다.

1) EConMan 을 사용

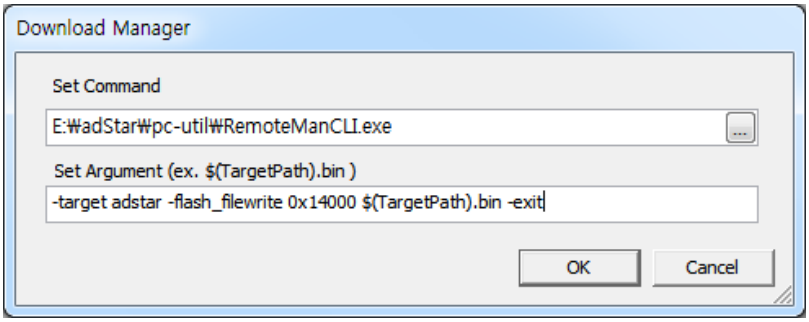


[download option]

기존에 EConMan 프로그램을 사용하여 download 하는 방법에서 address 를 0x14000 으로만 해주면 된다. 이런 식으로 원하는 address 에 binary 파일을 download 할 수 있다.

2) RemoteManager 사용

Bootloader 의 Remote Communication Mode 로 booting 하여 RemoteManCLI 를 사용하여 0x14000 에 download 할 수 있다. 방법은 Download Option 을 실행하여 다음과 같이 적어주면 된다.



-flash_filewrite 라는 command 를 사용하여 Flash 의 지정한 0x14000 번지에 application 을 download 한다.

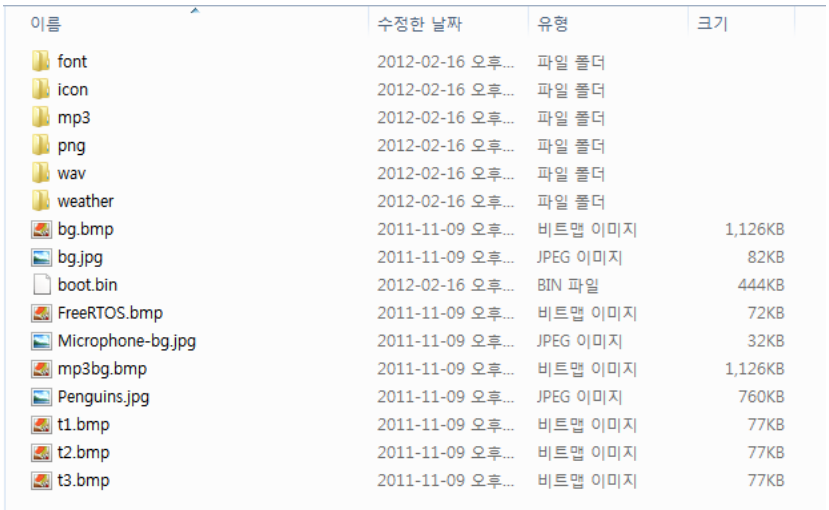
EConMan 또는 RemoteManCLI 를 사용하여 download 를 완료한 후에 Excute mode 로 booting 하면 application 이 동작하는 것을 확인할 수 있다.

라. User Define Mode (execute_fat)

User Define Mode 는 사용자가 원하는 동작을 코딩하여 사용할 수 있도록 만들어 놓은 Mode 이지만, bootloader 에서 기본적으로 Execute Mode 와 비슷한 동작을 하도록 프로그램 되어 있다. Booting 을 하면 Execute Mode 와 같이 바로 application 이 동작을 한다. 다만 Execute Mode 가 내부 Flash 에 저장되어 있는 프로그램을 Dram 에 복사하여 실행하는 것이었다면, User Define Mode(execute_fat)에서는 Nand Flash(SD Card)의 boot.bin 파일을 Dram 으로 복사한 후에 실행한다는 점이 다르다.

Nand Flash(SD Card)의 boot.bin 파일을 Dram 으로 복사하여 실행하는 형태에 대해 설명하면, build 하여 생성된 project_name.elf.bin 파일을 boot.bin 파일로 이름을 변경하여 Mass Storage 모드로 부팅, 저장장치 root 에 복사를 한 후, User Define Mode 로 부팅을 하면 저장한 application 이 동작을 한다.

여기서 주의할 점은 application 을 Dram 에서 동작시키기 때문에 link script 를 Dram 에서 동작하도록 설정 된 adstar_dram.ld 파일을 사용하여야 한다.



이름	수정한 날짜	유형	크기
font	2012-02-16 오후...	파일 폴더	
icon	2012-02-16 오후...	파일 폴더	
mp3	2012-02-16 오후...	파일 폴더	
png	2012-02-16 오후...	파일 폴더	
wav	2012-02-16 오후...	파일 폴더	
weather	2012-02-16 오후...	파일 폴더	
bg.bmp	2011-11-09 오후...	비트맵 이미지	1,126KB
bg.jpg	2011-11-09 오후...	JPEG 이미지	82KB
boot.bin	2012-02-16 오후...	BIN 파일	444KB
FreeRTOS.bmp	2011-11-09 오후...	비트맵 이미지	72KB
Microphone-bg.jpg	2011-11-09 오후...	JPEG 이미지	32KB
mp3bg.bmp	2011-11-09 오후...	비트맵 이미지	1,126KB
Penguins.jpg	2011-11-09 오후...	JPEG 이미지	760KB
t1.bmp	2011-11-09 오후...	비트맵 이미지	77KB
t2.bmp	2011-11-09 오후...	비트맵 이미지	77KB
t3.bmp	2011-11-09 오후...	비트맵 이미지	77KB

[Nand Flash 의 boot.bin 파일]

SD Card 에서 boot.bin 파일을 Dram 으로 복사하여 실행 할 수도 있는데, main.c 의 4 번째줄의 #define FAT_APP_TYPE DRIVE_NAND 를 주석처리 하면, SD Card 의 boot.bin 파일을 Dram 에 복사하여 실행한다.

3.2 Namd Boot Code

Nand Boot Code 는 내부 Flash 가 없는 adStar 를 사용할 경우 사용하는 Nand Booting 용 bootloader 로 Nand Flash Auto Boot Mode[5]로 설정이 되어 있어야 한다.

Nand Flash Auto Boot Mode 는 Nand Flash 0 번 블록의 2Kbyte data 를 내부 sram 에 복사하여 booting 하는 mode 이기 때문에, Nand Boot Code 의 크기는 2Kbyte 로 제한된다. 따라서 Nand Boot Code 는 수정 없이 제공되는 것을 그대로 사용하기를 권장한다.

Nand Boot Code 가 하는 역할은 Nand Flash 1 번 블록부터 7 개 블록의 data(application program)를 Dram 으로 복사하여 실행하는 역할을 수행한다.

```
282
283   #define BOOTLOADER_STARTADDR 0x20000000
284   #define COPY_BLOCK 7//128K*7Mbyte
285   int main()
286   {
287       void (*startfp)();
288       boardinit();
289       myuart_init();
290       *R_NFMCFG  = 0x2222;
291       nand_reset();
292       mydebugstring("Nand Booting.₩r₩n");
293       //U16 id = nand_id_read();
294       copydata_from_nand((U8*)BOOTLOADER_STARTADDR,1,COPY_BLOCK);
295       startfp = (void (*)())(*(volatile unsigned int*)BOOTLOADER_STARTADDR);
296       startfp();
297       while(1);
298       return 0;
299   }
300
```

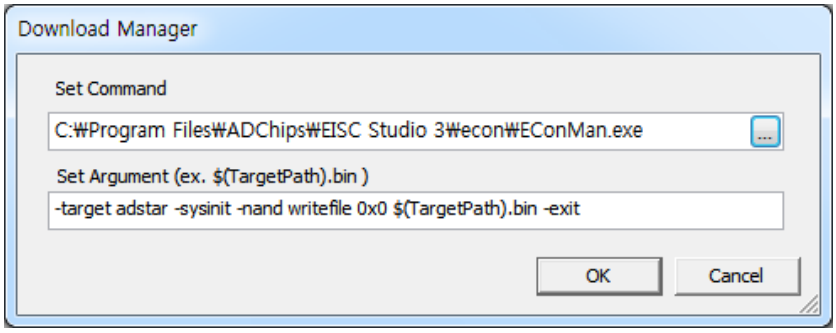
[nand_boot_code 의 main.c]

3.2.1 Namd Boot Code 사용하기

Nand Boot Code 는 SDK example 안에 들어있다. 다른 application 과 마찬가지로 project 를 열어서 build 후에 Nand Flash 의 0 번 블록에 download 하면 된다.

Build 는 이전에 설명한 방법대로 진행을 하면 되는데, download 는 기존 내부 Flash 에 download 하는 것과는 command 가 다르다. 다음은 Nand Flash 에 download 시 download option 이다. 참고하기 바란다.

[5] adStar의Boot Mode 중 하나로 Nand Flash를 사용하여 booting 하는 mode이다. 자세한 사항은 datasheet를 참고하기 바란다.

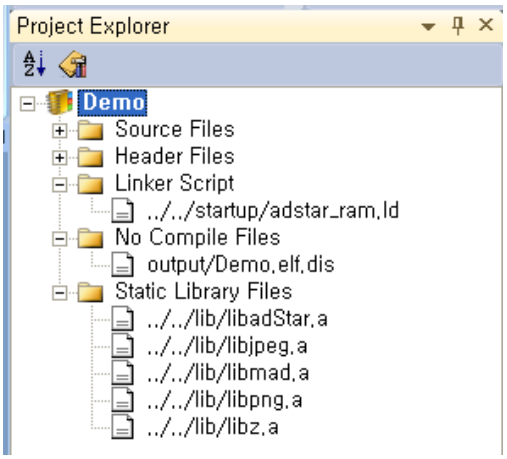


[Download Option]

'-nand writefile ' command 를 사용하여 Nand Flash 에 download 한다.

다음으로 Nand Boot Code 에서 Dram 에 복사하여 실행할 코드를 download 하는 방법에 대해 알아보겠다.

먼저 Nand Boot Code 에 의해 Dram 에 복사될 코드는 Dram 에서 수행되기 때문에 link script 가 adstar_ram.ld 인지 확인을 한다.

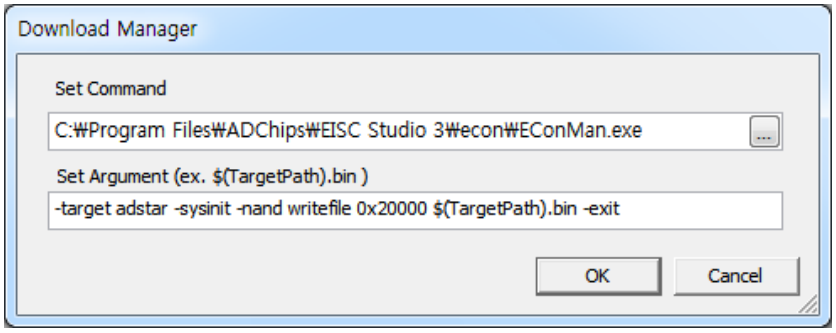


[Linker Script]

확인을 한 후 build 과정을 거쳐 download 를 진행하면 된다. 이 역시 Nand Flash 에 download 하므로 Nand Boot Code 를 download 할 때와 같은 command 를 사용한다. 다른 점은 download 하는 위치가 달라진다.

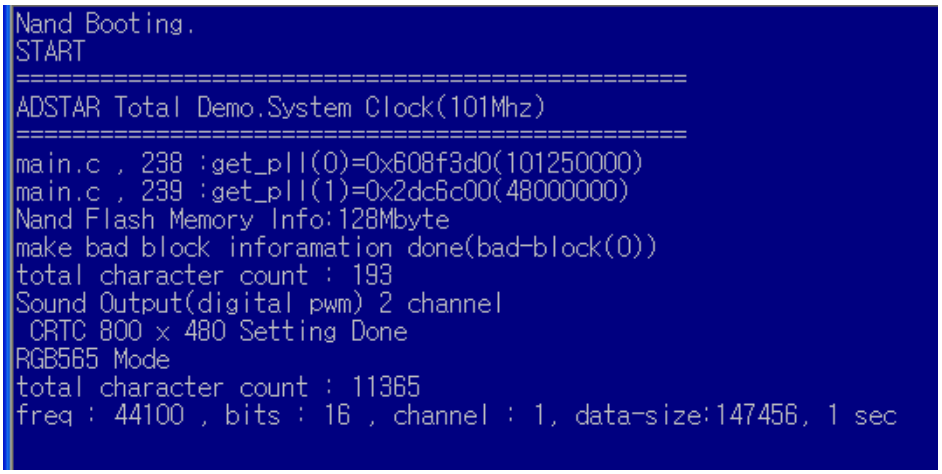
Nand Boot Code 에 의해 Dram 에 복사되어 실행할 코드는 Nand Flash 의 1 번 블록에 download 한다. Nand Flash 1 번 블록의 address 는 Nand Flash 마다 다르므로 주의하기 바란다.

(기본적으로 small page Flash 의 경우 1 번 블록은 0x4000 이고, large page Flash 의 경우 1 번 블록은 0x20000 이다.)



[Download Option]

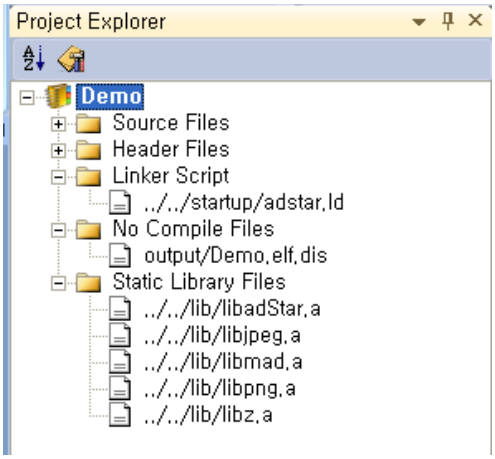
Download 완료 후 실행하면 다음과 같이 Nand Booting 라고 출력되고, 1 번 블록에 download 한 application 이 실행되는 것을 확인할 수 있다.



3.3 BootLoader 없이 사용하기

지금까지 Bootloader 와 Nand Boot Code 에 대해서 알아 보았는데, 이번에는 Bootloader 를 사용하지 않고 프로그램을 실행하는 방법에 대해 설명하도록 하겠다.

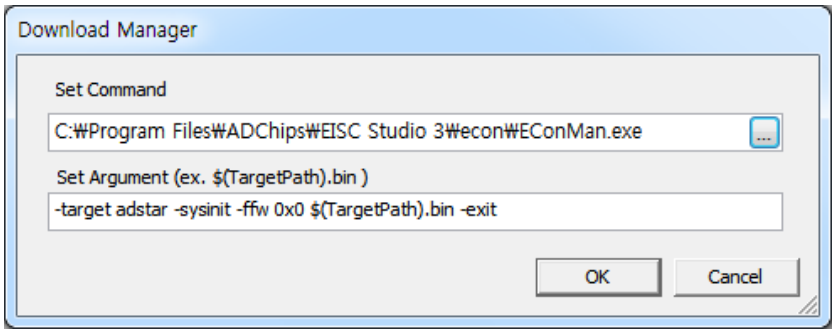
Bootloader 를 사용하지 않으면 Linker Script 파일과, 프로그램을 download 하는 위치가 달라지게 된다. Linker Script 는 adStar_ram.ld 파일 대신 adstar.ld 파일을 사용한다. 앞에서 몇 차례 설명하였지만, adStar_ram.ld 파일은 Dram 에서 프로그램이 동작하도록 설정되어 있는 파일이고, adstar.ld 파일은 flash 에서 프로그램이 동작하도록 설정되어 있는 파일이다. Bootloader 를 사용하지 않으면 실행 프로그램을 flash 에서 동작시켜야 하므로 adstar.ld 파일을 사용해야 한다. adstar.ld 파일은 startup 폴더에 있으므로 변경해주면 된다.



[Linker Script]

Linker Script 를 변경하였으면, build 후에 download 해주어야 하는데, bootloader 를 사용하지 않기 때문에, RemoteCLI 프로그램을 사용할 수 없고, EConMan 프로그램을 사용해서 download 해야 한다.

Build → Download Option 에 다음과 같이 적고, Download to Target 을 실행하면 된다.



[Download Option]

Bootloader 를 사용하지 않기 때문에 0 번지에 바로 download 하면 된다. Download 가 완료된 후 reset 을 하면 동작을 확인할 수 있다. 정상적으로 download 하였는데도 아무런 동작을 하지 않는다면, Linker Script 가 adstar.ld 로 되어있는지 확인하기 바란다.

4. lib_config.h

adStar SDK 의 lib_src 를 보면 Header Files 에 lib_config.h 파일이 있는데, 파일명에서도 알 수 있듯이 lib_config.h 파일은 adStar SDK library 의 설정 파일이라고 보면 된다. 이 장에서는 lib_config.h 파일에 대해 설명하도록 하겠다.

```
/*******************************************************************************
 Copyright (C) 2011      Advanced Digital Chips Inc.
                         http://www.adc.co.kr
 Author : Software Team.

 이 파일을 수정 함으로써 전체 실행파일 사이즈와 메모리 사용량을 줄일 수도 있다.
 하지만 줄어든 만큼 성능의 저하가 발생 할 경우도 있다.
 제거 하고자 하는 기능은 0 으로 설정 함으로써 제거된다.

*******************************************************************************/
```

[lib_config.h 요약]

lib_config.h 는 Nested Interrupt 설정에 대한 내용으로 시작한다.

```
/*******************************************************************************
    Nested Interrupt
    특정 인터럽트 서비스 함수(ISR)내에서 또다른 인터럽트를 받아 들일 것인지에 대한 설정
    특정 인터럽트를 1 로 설정 할 경우 해당 인터럽트 함수내에서 모든 다른 인터럽트를 받아 들인다.
    단,인터럽트 진입 속도가 그렇지 않은 경우 보다 느리다.
    ex)
        SDK 내의 MP3 decoding 은 DMA 인터럽트내에서 frmae 단위로 이루어 진다.
        따라서 이 구간내에서도 다른 인터럽트를 받아 들이고자 한다면
        모든 DMA 인터럽트를 1 로 설정 하면 된다.
*******************************************************************************/
#define SUPPORT_NESTED_INTNUM_EIRQ0 0
#define SUPPORT_NESTED_INTNUM_TIMER0 0
#define SUPPORT_NESTED_INTNUM_SOUNDMIXER 0
#define SUPPORT_NESTED_INTNUM_DMA0 1 //DMA is used by sound , 1 is recommended

#define SUPPORT_NESTED_INTNUM_FRAMESYNC 0
#define SUPPORT_NESTED_INTNUM_GPIOA 0
#define SUPPORT_NESTED_INTNUM_UART0 0
#define SUPPORT_NESTED_INTNUM_DMA1 1

#define SUPPORT_NESTED_INTNUM_EIRQ1 0
#define SUPPORT_NESTED_INTNUM_TIMER1 0
#define SUPPORT_NESTED_INTNUM_PMU 0
#define SUPPORT_NESTED_INTNUM_DMA2 1

#define SUPPORT_NESTED_INTNUM_SPI0 0
#define SUPPORT_NESTED_INTNUM_GPIOB 0
#define SUPPORT_NESTED_INTNUM_UART1 0
#define SUPPORT_NESTED_INTNUM_DMA3 1

#define SUPPORT_NESTED_INTNUM_RESV1 0
#define SUPPORT_NESTED_INTNUM_TIMER2 0
#define SUPPORT_NESTED_INTNUM_USB 0
#define SUPPORT_NESTED_INTNUM_DMA4 1

#define SUPPORT_NESTED_INTNUM_USBHOST 0
#define SUPPORT_NESTED_INTNUM_GPIOC 0
#define SUPPORT_NESTED_INTNUM_UART2 0
#define SUPPORT_NESTED_INTNUM_DMA5 1

#define SUPPORT_NESTED_INTNUM_RESV2 0
#define SUPPORT_NESTED_INTNUM_TIMER3 0
#define SUPPORT_NESTED_INTNUM_NAND 0
#define SUPPORT_NESTED_INTNUM_DMA6 1
```

```
55      #define SUPPORT_NESTED_INTNUM_RESV2 0
56      #define SUPPORT_NESTED_INTNUM_TIMER3 0
57      #define SUPPORT_NESTED_INTNUM_NAND 0
58      #define SUPPORT_NESTED_INTNUM_DMA6 1
59
60      #define SUPPORT_NESTED_INTNUM_SDHC 0
61      #define SUPPORT_NESTED_INTNUM_GPIOD 0
62      #define SUPPORT_NESTED_INTNUM_UART3 0
63      #define SUPPORT_NESTED_INTNUM_DMA7 1
64
65      #define SUPPORT_NESTED_INTNUM_RESV3 0
66      #define SUPPORT_NESTED_INTNUM_GPIOE 0
67      #define SUPPORT_NESTED_INTNUM_ADC 0
68      #define SUPPORT_NESTED_INTNUM_GPIOF 0
69
70      #define SUPPORT_NESTED_INTNUM_WATCHDOG 0
71      #define SUPPORT_NESTED_INTNUM_GPIOG 0
72      #define SUPPORT_NESTED_INTNUM_UART4 0
73      #define SUPPORT_NESTED_INTNUM_GPIOH 0
74
75      #define SUPPORT_NESTED_INTNUM_RESV4 0
76      #define SUPPORT_NESTED_INTNUM_GPIOI 0
77      #define SUPPORT_NESTED_INTNUM_TWI 0
78      #define SUPPORT_NESTED_INTNUM_GPIOJ 0
79
80      #define SUPPORT_NESTED_INTNUM_SPI1 0
81      #define SUPPORT_NESTED_INTNUM_CAPOVERFLOW 0
82      #define SUPPORT_NESTED_INTNUM_MOTORFAULTA 0
83      #define SUPPORT_NESTED_INTNUM_MOTORFAULTB 0
84
85      #define SUPPORT_NESTED_INTNUM_RESV5 0
86      #define SUPPORT_NESTED_INTNUM_MOTORQEI 0
87      #define SUPPORT_NESTED_INTNUM_PWM 0
88      #define SUPPORT_NESTED_INTNUM_RESV6 0
89  #endif
```

Nested Interrupt 사용을 설정하는 부분으로 1 로 설정하면 해당 Interrupt routine 에서 Nested Interrupt 를 허용한다는 의미이다. Nested Interrupt 설명에 적혀 있듯이, MP3 decoding 이 DMA Interrupt 내에서 frame 단위로 이루어지는데, 그 사이에 다른 Interrupt 를 받아 들여야 하므로 DMA 쪽 Interrupt 는 default 로 Nested Interrupt 를 허용해 놓는다. 이 외에도 Nested Interrupt 를 허용하고 싶은 경우에는 1 로 설정하면 된다. 중첩 허용은 최대 14 개까지 가능하므로, 꼭 필요한 경우만 사용하기 바란다.

```
90  /*****************************************************************************
91      INTERNAL TIME OUT
92  *****************************************************************************/
93  /*
94      TWI Function에서 Status를 기다리는 최대 시간을 설정한다.
95  */
96  #define TWI_RESP_TIME_OUT_COUNT (7200*100)  // About 100ms @ AHB 101MHz
97
98  /*****************************************************************************
99      External NAND/SD-Card/USB Memory Supports
100 *****************************************************************************/
101
102 #define CONFIG_NAND 1
103 #define CONFIG_SDCARD 1
104
105 #if (CONFIG_SDCARD != 0)
106 /*
107 SDCARD 는 두 군데 Port Alternate 가 되어 있다.
108 만약 NAND 와 SDCARD 를 같은 Port 를 사용 한다면
109 SDK 내부에서 Port 를 변환 하게 되어 있다.
110 따라서 사용자가 어떤 Port 를 사용하는지 여기서 선언 해야만 한다.
111
112 SDCARD_PIN_USE_67 를 1 로 선언 한다면 nand 와 sd card 가 서로 다른 Port 를
113 사용하므로 내부적으로 Port 변환을 하지 않는다.
114 따라서 사용자가 직접 최초 한번 Port alternative Function(0x800234xx) 를 초기화 해야 한다.
115 */
116 #define SDCARD_PIN_USE_67 0
117 #endif
```

다음은 TWI Timeout count 값을 설정 하는 부분이 있고, 사용할 저장 장치를 설정하는 부분이 있다. 이곳에서 사용할 저장장치를 설정하여 사용하지 않는 저장장치에 관련된 부분의 size 를 줄일

수 있다. default 로 모두 1 로 되어있다. 그리고 SD Card 의 경우에는 adStar chip 에서 사용할 수 있는 Pin 이 두 곳이 존재 하는데, 어디를 사용할지에 따라 설정을 해주면 된다. default 값은 STK board 와 같이 Nand Flash 와 같은 port 를 사용하도록 되어 있다.

```
120 /*******************************************************************************
121     Image file supports
122 *******************************************************************************/
123
124 #define CONFIG_SUPPORT_BMP 1
125 #define CONFIG_SUPPORT_TGA 1
126 #define CONFIG_SUPPORT_JPG 1
127 #define CONFIG_SUPPORT_PNG 1
```

다음은 library 에서 지원할 이미지 포맷을 결정하는 부분으로 default 로 모두 사용하도록 되어 있다. 하지만 굳이 사용하지 않을 이미지 포맷은 0 으로 설정하여 코드의 사이즈를 줄이는 것이 좋다.

```
128 /*******************************************************************************
129     FAT File System
130 *******************************************************************************/
131 //drive number
132 #define DRIVE_NAND 0
133 #define DRIVE_SDCARD 1
134
135 #define CONFIG_FAT_READONLY 0
136 #define CONFIG_FAT_VOLUMES 3
137
138 /*
139     아래 설정 값은 include/fatfs/ffconf.h 파일을 참조.
140     영문만을 사용 할 경우
141         CONFIG_CODE_PAGE 437 로 설정
142         library 프로젝트에 lib_src/fatfs/option/ccsbcs.c 파일을 추가 한 후 rebuild 한다
143     한글을 사용 할 경우
144         약 136,408 byte 가 늘어 난다.
145         CONFIG_CODE_PAGE 949 로 설정
146         library 프로젝트에 lib_src/fatfs/option/cc949.c 파일을 추가 한 후 rebuild 한다
147 */
148 #define CONFIG_CODE_PAGE 949
149
```

사용하는 저장장치의 driver number 를 설정하는 부분과 driver 의 개수를 설정하는 부분이다. 그리고 filesystem 에서 영문만을 사용할지 한글도 사용할지에 따라 설정을 해줄 수 있는데, 적힌 내용처럼 영문만 사용할 경우 변경을 해주면 filesystem 관련 코드 사이즈가 줄어든다.

```
153
154 /*******************************************************************************
155     UART Config
156 *******************************************************************************/
157 /*
158     인터럽트 모드일 경우    uart 송/수신 data 를 임시로 저장 할 버퍼의 사이즈이다.
159 */
160 #define UART_BUF_SIZE 512
161 #define CONFIG_UART_RX_INTERRUPT
162 //#define CONFIG_UART_TX_INTERRUPT
163
164 /*
165     debugstring, debugprintf 함수의  최종 출력 채널
166 */
167 #define DEBUG_CHANNEL 0
```

Uart Config 에서는 Uart library 에서 기본적으로 RX interrupt 를 사용하고, Ring buf 를 사용하여 uart 로 전송되는 data 를 저장을 하는데, data 를 저장하는 Ring buf 의 사이즈를 결정하고, 사용할 Interrupt 를 설정한다. 기본적으로 library 에서 RX interrupt 사용하도록 설정 되어 있고, Uart Interrupt Routine 도 작성되어 있다.

```
169 /*****************************************************************************
170     SYSTEM CLOCK
171 *****************************************************************************/
172 /*
173     외부 OSC 입력 값
174     STK 보드의 경우 10Mhz OSC 를 사용한다.
175     10Mhz 가 아닌 경우 adStarInit.c 의 PMUInit() 함수에서 PLL 설정 값을 수정 해야 한다.
176     사용가능한 주파수는 doc/adStar_PLL_Setting.xls 파일을 이용.
177 */
178  //#define OSC_CLOCK 8000000
179  #define OSC_CLOCK 10000000
180
181
182 /*****************************************************************************
183  *   SOUND Mixer
184 *****************************************************************************/
185 /*
186     버퍼 사이즈를 줄이면 메모리 사용량은 줄어 들지만
187     인터럽트 발생 횟수가 늘어 난다.
188     만약 wav 파일은 사용하지 않고 MP3 만 재생 한다면
189     1 프레임씩 디코딩 하므로 이 사이즈를 약 5kbyte 만 설정 해도 충분하다.
190 */
191  #define WAVE_BUF_MAX (512*1024) //if you use MP3, > 5Kbyte
192  #define DEFAULT_VOLUME 255//max 255
193 /*
194     0,1 번 채널은 I2S 출력이 2,3 번 채널은 digital modulator 출력
195     STK 보드의 경우 2번 채널을 이용한다.
196 */
197  #define SND_OUTPUT_CHANNEL  2
198 /*
199     최종 출력 sample rate
200 */
201  #define SND_OUTPUT_HZ   48000
202
```

lib_config.h 의 마지막으로 System clock 설정과 Sound mixer 에 대한 설정이 있다. Sound mixer 와 관련하여 buffer size, volume, output channel, sample rate 를 설정할 수 있다.

그리고 System clock 에서는 입력 clock 을 설정하는 부분이 있는데, 입력 clock 으로 10Mhz 가 아닌 다른 Clock 를 사용할 경우 이 부분을 수정 해주면 된다. 그리고 PMUInit()함수에서 입력 Clock 에 대한 PLL setting 을 해주어야 원하는 System clock 으로 adStar 를 동작시킬 수 있다.

lib_config.h 파일의 모든 설정이 완료 되었으면 library 를 다시 build 하여 libadStar.a 파일을 새로 생성하여 사용하여야 한다.

5. UART

5.1 uart_config

BOOL uart_config(**int** ch, **int** baud, **UART_DATABITS** databits, **UART_STOPBITS** stopbit,
UART_PARITY parity)

UART 사용을 위한 초기 설정을 하는 함수이다.

Parameter

ch UART channel 값. 초기 설정할 channel 을 입력한다.

baud UART baud rate 값. Baud rate 값을 설정한다.
SDK 는 기본적으로 115200 을 사용한다.

databits UART 전송 data bit 을 설정한다. SDK 에서는 data bit 를 다음과 같이 정의하고 있다.

```
typedefenum{
    DATABITS_5 = 5,
    DATABITS_6 = 6,
    DATABITS_7 = 7,
    DATABITS_8 = 8
}UART_DATABITS;
```

stopbit UART 의 stop bit 를 설정한다. SDK 에서는 stop bit 를 다음과 같이 정의하고 있다.

```
Typedefenum{
    STOPBITS_1 = 1,
    STOPBITS_2 = 2
}UART_STOPBITS;
```

parity UART 의 parity bit 를 설정한다. SDK 에서는 parity bit 를 다음과 같이 정의하고 있다.

```
Typedefenum{
    UART_PARNONE = 0,
    UART_PARODD,
    UART_PAREVEN
}UART_PARITY;
```

Return Value

TRUE(1) or FALSE(0)

5.2 uart_putch

BOOL uart_putch(**int** n, **char** ch)

UART n 채널을 통해 ch 값(한 문자)을 출력한다.

Parameter

n	값을 출력할 UART channel 값. adStar 는 UART 5 개 channel 을 가지고 있다.
Ch	UART 를 통해 출력할 값.

Return Value

TRUE(1) or FALSE(0)

5.3 uart_putdata

BOOL uart_putdata(**int** n, U8* buf, int len)

UART n 채널을 통해 buf 에 저장되어 있는 문자를 len 만큼 출력한다..

Parameter

n	값을 출력할 UART channel 값. adStar 는 UART 5 개 channel 을 가지고 있다.
buf	출력할 문자열이 저장되어 있는 buffer.
len	출력할 문자열의 개수.

Return Value

TRUE(1) or FALSE(0)

5.4 uart_putstring

BOOL uart_putstring(**int** n, U8* buf)

UART n 채널을 통해 buf 에 저장되어 있는 문자열을 출력한다.

Parameter

n	값을 출력할 UART channel 값. adStar 는 UART 5 개 channel 을 가지고 있다.
buf	출력할 문자열이 저장되어 있는 buffer.

Return Value

TRUE(1) or FALSE(0)

5.5 uart_getch

BOOL uart_getch(**int** n, **char*** ch)

UART n 채널을 통해 1Byte 값(한 문자)을 받아 ch 에 저장한다.

Parameter

n	값을 입력 받을 UART channel 값. adStar 는 UART 5 개 channel 을 가지고 있다.
ch	UART 를 통해 값을 입력 받을 변수.

Return Value

TRUE(1) or FALSE(0)

5.6 uart_getdata

int uart_getdata(**int** n, **U8*** buf, **int** bufmax)

UART n 채널을 통해 bufmax 만큼 값(문자)을 읽어서 buf 에 저장한다.

Parameter

n	값을 입력 받을 UART channel 값.
Buf	UART 를 통해 값을 입력 받을 buffer.
Bufmax	값(문자)을 읽을 개수. (Byte 단위)

Return Value

읽은 data byte 수. Bufmax 값과 비교하여 같으면 함수가 정상적으로 처리 한 것이고, 그렇지 않을 경우에는 정상 처리하지 못한 것이다.

5.7 uart_rx_flush

void uart_rx_flush(**int** ch)

UART n 채널의 rxfifo 를 초기화 시킨다.

Parameter

ch	초기화 시킬 UART channel 값.

Return value

없음.

5.8 uart_tx_flush

void uart_tx_flush(**int** ch)

UART n 채널의 txfifo 를 초기화 시킨다.

Parameter

ch 초기화 시킬 Uart channel 값.

Return value

없음.

5.9 set_debug_channel

Void set_debug_channel(**int** ch)

디버깅용으로 사용하는 **debugprintf**, **debugstring**, **PRINTVAR**, **PRINTLINE** 함수에 의해 디버깅 메시지가 출력 될 UART 채널을 결정한다.

Parameter

ch 디버깅용으로 사용할 UART 채널.

Return value

없음.

5.10 get_debug_channel

int get_debug_channel( )

디버깅용으로 사용하는 UART 채널 값을 return 한다.

Parameter

없음

Return Value

현재 설정된 디버깅용 UART 채널 값.

5.11 debugprintf

void debugprintf(**const char***const format, . . .)

c 언어의 printf 와 같은 역할을 하는 함수로, UART 를 통해, 숫자, 문자, 변수의 내용을 출력할 때 사용한다. 출력되는 UART 채널은 set_debug_channel 함수를 통해 설정해 놓은 디버깅 채널을 사용한다. 기본으로 0 번 채널이 설정되어 있다.

Usage

debugprintf("result number : %d₩r₩n",result);

→UART 를 통해 "result number :"라는 문자열과 result 변수 값을 10 진수로 출력한다. 그리고 줄 바꿈을 한다. printf 함수와 사용법이 같다. 단 줄 바꿈 시 ₩r₩n 을 모두 써주어야 한다.

5.12 debugstring

void debugstring(**const char*** str)

문자열을 출력하는 함수로, 문자열만 출력하기를 원할 때 사용하는 함수이다. 출력되는 UART 채널은 set_debug_channel 함수를 통해 설정해 놓은 디버깅 채널을 사용한다. 기본으로 0 번 채널이 설정되어 있다.
참고로 debugprintf 함수를 사용하여도 문자열만 출력 가능하다.

Usage

debugstring("=== adStar Start ===₩r₩n");

→""안의 문자열을 UART 를 통해 출력한다.

5.13 PRINTLINE

매크로 함수로 PRINTLINE 을 호출한 곳의 줄(line)값을 UART 통해 출력한다. 채널은 설정해 놓은 디버깅 채널을 사용한다. 출력되는 UART 채널은 set_debug_channel 함수를 통해 설정해 놓은 디버깅 채널을 사용한다. 기본으로 0 번 채널이 설정되어 있다.

Usage

PRINTLINE;

→ 함수를 호출한 곳의 line 값을 출력한다..

5.14 PRINTVAR(A)

매크로 함수로 PRINTVAR 를 호출한 곳의 줄(line)값과 A 값을 UART 를 통해 출력한다.출력되는 UART 채널은 set_debug_channel 함수를 통해 설정해 놓은 디버깅 채널을 사용한다. 기본으로 0 번 채널이 설정되어 있다.

Usage

int a = 10;

PRINTVAR(a);;

→ 변수 a 의 값을 출력한다. 레지스터 값을 출력 할 수도 있다.

5.15 UART Example

```
#include "adStar.h"

Int main()
{
    boardinit();
    uart_config(0, 115200, DATABITS_8, STOPBITS_1, UART_PARNONE );
                    // UART 0 번 채널을 다음과 같이 설정한다.
                    // Baud rate = 115200
                    // Data bit = 8bit
                    // Stop bit = 1
                    // Parity = none
    debugstring("==========================================₩r₩n");
    debugprintf("ADSTAR UART Example. System clock(%dMhz)₩r₩n",get_ahb_clock()/1000000);
    debugstring("==========================================₩r₩n");

    U8 ch;
    While(1)
    {
        if(uart_getch(0, &ch))      // UART 0 번에서 1Byte 데이터를 읽어 ch 에 저장한다.
        {
            uart_putch(0, ch);// UART 0 번에서 입력 받은 데이터를 UART 0 번으로 출력한다.
        }
    }
}
```

6. Interrupt

6.1 init_interrupt

void init_interrupt(**void**);

interrupt 를 초기화 하는 함수이다. 인터럽트 관련 함수를 사용하기 위해서 반드시 호출해주어야 하는 함수로 adStar SDK 의 startup_adstart.s 에서 호출하여 인터럽트를 초기화한다.

6.2 set_interrupt

BOOL set_interrupt(**INTERRUPT_TYPE** intnum, **void** (*fp)());

해당 interrupt 가 발생하였을 때 호출할 함수를 등록하는 함수이다.
Uart, Sound mixer 등의 Interrupt 함수는 SDK 에서 생성 및 설정 되어 있으므로 중복 등록하지 않도록 주의하기 바란다.

Parameter

intnum	interrupt type (interrupt type 에 대한 정보는 다음 장 참고)
(*fp)()	interrupt 가 발생했을 때 호출할 함수

Return value

TRUE or FALSE

6.3 enable_interrupt

void enable_interrupt(**INTERRUPT_TYPE** intnum, **BOOL** b);

해당 interrupt 를 enable 하는 함수이다.set_interrupt 함수로 interrupt 함수를 등록하고, enable_interrupt 함수로 활성화 시킨다.

Parameter

intnum	interrupt type (interrupt type 에 대한 정보는 다음 장 참고)
b	1 or TRUE => interrupt enable
	0 or FALSE => interrupt disable

Return value

TRUE or FALSE

INTERRUPT_TYPE

INTERRUPT_TYPE		INTERRUPT_TYPE	
INTNUM_EIRQ0	External interrupt 0	INTNUM_UART0	UART 0 interrupt
INTNUM_EIRQ1	External interrupt 1	INTNUM_UART1	UART 1 interrupt
INTNUM_TIMER0	Timer interrupt 0	INTNUM_UART2	UART 2 interrupt
INTNUM_TIMER1	Timer interrupt 1	INTNUM_UART3	UART 3 interrupt
INTNUM_TIMER2	Timer interrupt 2	INTNUM_UART4	UART 4 interrupt
INTNUM_TIMER3	Timer interrupt 3	INTNUM_PMU	PMU interrupt
INTNUM_SOUNDMIXER	Sound mixer interrupt	INTNUM_SPIO	SPI 0 interrupt
INTNUM_DMA0	DMA interrupt 0	INTNUM_SPI1	SPI 1 interrupt
INTNUM_DMA1	DMA interrupt 1	INTNUM_USB	USB device interrupt
INTNUM_DMA2	DMA interrupt 2	INTNUM_USBHOST	USB host interrupt
INTNUM_DMA3	DMA interrupt 3	INTNUM_NAND	NAND Flash interrupt
INTNUM_DMA4	DMA interrupt 4	INTNUM_SDHC	SDHC interrupt
INTNUM_DMA5	DMA interrupt 5	INTNUM_ADC	ADC interrupt
INTNUM_DMA6	DMA interrupt 6	INTNUM_WATCHDOG	Watch dog interrupt
INTNUM_DMA7	DMA interrupt 7	INTNUM_TWI	TWI interrupt
INTNUM_FRAMESYNC	Frame sync interrupt	INTNUM_CAPOVERFLOW	Capture
INTNUM_GPIOA	GPIO A interrupt		overflow interrupt
INTNUM_GPIOB	GPIO B interrupt	INTNUM_MOTORFAULTA	Motor
INTNUM_GPIOC	GPIO C interrupt		Fault A interrupt
INTNUM_GPIOD	GPIO D interrupt	INTNUM_MOTORFAULTB	Motor
INTNUM_GPIOE	GPIO E interrupt		Fault B Interrupt
INTNUM_GPIOF	GPIO F interrupt	INTNUM_MOTORQEI	Motor QEI interrupt
INTNUM_GPIOG	GPIO G interrupt	INTNUM_MOTORPWM	Motor PWM interrupt
INTNUM_GPIOH	GPIO H interrupt		
INTNUM_GPIOI	GPIO I interrupt		
INTNUM_GPIOJ	GPIO J interrupt		

6.4 Interrupt Example

```
#include "adStar.h"

void EIRQ0ISR( )
{
	debugprintf("EIRQ0 Interrupt₩r₩n");
}

int main()
{
	boardinit();
	uart_config(0, 115200, DATABITS_8, STOPBITS_1, UART_PARNONE );
	set_interrupt( INTNUM_EIRQ0, EIRQ0ISR );
			//External Interrupt 0 번이 발생하였을 때 호출 될 interrupt 함수를 등록한다.
	enable_interrupt( INTNUM_EIRQ0, TRUE );
			// External Interrupt 0 번을 enable 시킨다.

	while(1)
	return 0;
}
```

7. TIMER

7.1 set_timer

BOOL set_timer(**int** nCh, **U32** ms)

nCh 채널 timer 를 설정 및 동작시키는 함수이다. 사용할 timer 채널과 주기를 결정하고 timer control register 에서 timer 동작을 enable 한다. 그리고 마지막으로 timer interrupt 를 enable 한다.
set_interrupt 함수로 timer interrupt 등록한 후 이 함수를 호출하면 설정한 주기만큼 timer interrupt 가 발생한다.

Parameter

nCh	설정할 timer 채널 값.
ms	timer interrupt 주기. (ms 단위)

Return Value

TRUE(1) or FALSE (0)

7.2 stop_timer

BOOL stop_timer(**int** nCh)

nCh 채널 timer 를 정지시키는 함수이다. nCh 채널의 timer 동작을 disable 시킨다.

Parameter

nCh	정지시킬 timer 채널 값.

Return Value

TRUE(1) or FALSE (0)

7.3 delayms

BOOL delayms(**U32** ms)

ms 만큼 delay 가 걸린다. (단위는 ms 이다)

Parameter

ms	delay 걸리는 시간 단위는 ms.

Return Value

TRUE(1) or FALSE (0)

7.4 TIMER Example

```
#include "adStar.h"

void TIMER0ISR( )
{
        debugprintf("==TIMER0ISR==₩r₩n");
}

int main()
{
        boardinit();
        uart_config(0, 115200, DATABITS_8, STOPBITS_1, UART_PARNONE );
        set_interrupt( INTNUM_TIMER0, TIMER0ISR );
                        //Timer 0 번 Interrupt 가 발생하였을 때 호출 될 interrupt 함수를 등록한다.
        set_timer( 0, 1000);
                        // 1 초마다 Timer0 Interrupt 가 발생한다.
        delayms(5000);
                        // 5 초간 delay 가 발생.
        stop_timer( 0 );
                        // Timer 0 번 Interrupt 를 disable 시킨다.  더 이상 Timer 가 발생하지 않음.
        while(1)
        return 0;
}
```

8. Graphic

8.1 setscreen

void setscreen(**SCREENRES** size, **U32** screenmode)

LCD 해상도와 RGB 모드를 설정하는 함수. LCD 출력을 위해 처음으로 설정해주어야 한다.
설정한 해상도에 맞게 CRT Controller 를 설정한다.

Parameter

size LCD 의 해상도 값으로, 사용하려는 LCD 의 해상도로 설정을 해주면 해상도에 맞게 CRT Controller 를 설정한다. SCREENRES 는 다음과 같이 정의되어 있다. (LCD 마다 특성이 다르기 때문에 해상도와 맞게 설정하였는데도 LCD 가 정상적으로 나오지 않을 경우에는 crt.c 의 setscreen( ) 함수에서 LCD 특성에 맞게 값을 수정해주어야 한다.)

```
Typedefenum {
        SCREEN_480x272 = 0,
        SCREEN_640x480,
        SCREEN_800x480,
       } SCREENRES;
```

Screenmode RGB 모드를 설정하는 값으로, 다음과 같이 정의 되어 있다.

SCREENMODE_RGB888
SCREENMODE_RGB565

Return value

없음

8.2 createframe

SURFACE* createframe(**U32** w, **U32** h, **U32** bpp)

화면에 출력 될 frame(메모리 영역)을 생성하는 함수. 가로크기가 w 이고, 세로크기가 h 인 이미지 또는 도형을 그릴 수 있는 영역을 생성한다. 화면 출력을 위해서 한 개 이상의 frame 이 필요하다.

Parameter

w 생성될 frame 의 가로크기를 설정한다.

h 생성될 frame 의 세로 크기를 설정한다.
bpp 생성될 frame 의 bit per pixel 값을 설정한다.

Return value

frame 의 정보를 가지고 있는 SURFACE 구조체를 return 한다. SURFACE 는 다음과 같이 정의되어 있다.

```
typedefstruct {
        U32 w;
        U32 h;
        U32 bpp;
        void*  pixels;
        U32 pixtype;
        PALETTE* pal;
        U8 ashiftbit;
        U8 rshiftbit;
        U8 gshiftbit;
        U8 bshiftbit;
        void* reserve;
} SURFACE;
```

8.3 setframebuffer

void setframebuffer(**SURFACE*** surf)

화면에 보여지는 frame 을선택하는 함수. createframe()으로 생성한 frame 을 선택하면 그 frame 이 화면에 보이게 된다. (single frame)
화면 출력을 위해서 반드시 setframebuffer(), setdoubleframebuffer() 중 한가지를 호출하여 frame 을 선택해야 한다.

Parameter

surf createframe() 에 의해 생성된 SURFACE 구조체.

Return value

없음.

8.4 setdoubleframebuffer

void setdoubleframebuffer(**SURFACE*** surf, **SURFACE*** surf2)

화면에 보여지는 frame 을 두 개 설정하는 함수. frame 을 두 개 설정한 후, getbackframe()함수와, flip()함수를 사용하여 두 개의 frame 을 번갈아 가며 화면에 보여지게 할 수 있다. (double frame)
두 개의 frame 은 편의상 현재 보이는 frame 을 front frame, 현재 보여지지 않고 있는 frame 을 back frame 이라고 한다.
(setdoubleframebuffer() 함수를 사용하기 위해서는 frame 을 두 개 생성해야 한다.)
화면 출력을 위해서 반드시 setframebuffer(), setdoubleframebuffer() 중 한가지를 호출하여 frame 을 선택해야 한다.

Parameter

surf	createframe() 에 의해 생성된 SURFACE 구조체 화면 출력에 사용 될 첫 번째 frame 포인터이다.
surf2	createframe() 에 의해 생성된 SURFACE 구조체 화면 출력에 사용 될 두 번째 frame 포인터.

Return value

없음.

8.5 setframebufferxy

void setframebufferxy(**SURFACE*** surf, **U32** x, **U32** y);

화면에 보여지는 frame 의 시작 위치를 조정하여, frame 의 특정 지점부터 화면에 보여지도록 하는 함수이다. frame 크기가 화면보다 클 경우에 적용할 수 있으며, x 와 y 값은 frame 의 크기에서 화면 해상도의 차이 값을 Max 로 가질 수 있다.
(예: 800*480 해상도, frame size : 1024*512 면, Max x = 224, Max y = 32)
setframebufferxy() 함수를 사용하기 전에 setframebuffer() 또는 setdoublebuffer()함수로 화면에 출력할 frame 선택을 반드시 해주어야 한다.

Parameter

surf	createframe() 에 의해 생성된 SURFACE 구조체.
x	전체 frame 중 출력 할 부분의 시작 x 좌표.
y	전체 frame 중 출력 할 부분의 시작 y 좌표.

Return value

없음.

8.6 setdrawtarget

Void setdrawtarget(**SURFACE*** surf)

이미지를 출력하거나, 도형을 그릴 frame 을 선택하는 함수. createframe() 으로 생성된 frame 을 인자로 적어주면, draw 관련 함수를 호출했을 때 해당 frame 에 draw 하게 된다.

Parameter

surf frame 포인터. createframe()로 생성된 SURFACE 구조체

Return value

없음.

8.7 getdrawtarget

SURFACE* getdrawtarget( );

현재 drawtarget 으로 지정된 frame 의 포인터를 불러온다.

Parameter

없음.

Return value

frame 포인터. createframe()로 생성된 SURFACE 구조체

8.8 getbackframe

SURFACE*getbackframe()

setdoubleframebuffer()함수를 사용하여 frame 두 개를 사용할 경우, 현재 화면에 보여지지 않고 있는 frame 의 포인터를 반환하는 함수이다. 주로 setdrawtarget()함수와 함께 사용하여 현재 보여지지 않고 있는 frame 에 draw 하는 경우에 사용한다.

Parameter

없음.

Return value

createframe()에 의해 생성된 SURFACE 구조체를 return 한다. frame 의 정보를 가지고 있다.

8.9 getfrontframe

SURFACE*getfrontframe()

setdoubleframebuffer()함수를 사용하여 frame 두 개를 사용할 경우, 현재 화면에 보여지고 있는 frame 의 포인터를 반환하는 함수이다.

Parameter

없음.

Return value

createframe()에 의해 생성된 SURFACE 구조체를 return 한다. frame 의 정보를 가지고 있다.

8.10 flip

void flip()

setdoubleframebuffer()함수를 사용하여 frame 두 개를 사용할 경우, frame 간에 전환을 할 때 사용하는 함수이다.

Parameter

없음.

Return value

없음.

8.11 getscreenwidth

U32 getscreenwidth( )

현재 화면의 가로 크기 값을 반환하는 함수. setscreen()함수를 통해 설정 된 width 값이 반환된다.

Parameter

없음.

Return value

화면의 가로크기 값을 반환한다.

8.12 getscreenheight

U32 getscreenheight( )

현재 화면의 세로 크기 값을 반환하는 함수. setscreen()함수를 통해 설정 된 height 값이 반환된다.

Parameter

없음.

Return value

화면의 세로크기 값을 반환한다.

8.13 getscreenpitch

U32 getscreenpitch()

화면의 pitch 값을 반환하는 함수. Setscreen() 함수를 통해서 설정 된 pitch 값이 반환된다. pitch 값은 screen width ×bpp÷ 8 이다.

Parameter

없음.

Return value

화면의 pitch 값을 반환한다.

8.14 getscreenbpp

U32 getscreenbpp()

화면의 bpp(bit per pixel)값을 반환하는 함수. setscreen() 함수를 통해 설정 된 bpp 값이 반환된다.

.

Parameter

없음.

Return value

화면의 bpp 값을 반환한다.

8.15 drawsetrgb

Void drawsetrgb(**U8** sr, **U8** sg, **U8** sb)

선 및 도형의 색상을 설정한다. 설정 된 색으로 선 및 도형이 그려지게 된다.

Parameter

sr	RGB 중 R 의 값. (0 ~ 255)
sg	RGB 중 G 의 값. (0 ~ 255)
sb	RGB 중 B 의 값. (0 ~ 255)

Return value

없음.

8.16 drawgetrgb

void drawgetrgb(**U8*** dr, **U8*** dg, **U8*** db)

선 및 도형의 현재 설정되어 있는 색상을 읽어온다.

Parameter

dr	현재 설정되어 있는 R 값이 저장 될 변수.
dg	현재 설정되어 있는 G 값이 저장 될 변수.
db	현재 설정되어 있는 R 값이 저장 될 변수.

Return value

없음.

8.17 drawputpixel

void drawputpixel(**int** x, **int** y, **U8** r, **U8** g, **U8** b);

한 점을 찍는 함수. 지정한 위치에 지정된 색의 점을 찍는다. RGB 값을 직접 주는 함수로 drawsetrgb( )함수의 영향을 받지 않는다.

Parameter

x	점을 찍은 x 좌표
y	점을 찍은 y 좌표
r	점의 색상 중 r 값. (0 ~ 255)
g	점의 색상 중 g 값. (0 ~ 255)

b 점의 색상 중 b 값. (0 ~ 255)

Return value

없음.

8.18 drawline

void drawline(**U32** x1, **U32** y1, **U32** x2, **U32** y2)

x1, y1 위치에서 x2, y2 위치로 선을 그린다. 선의 색은 drawsetrgb( ) 함수에 의해 설정된 색을 사용한다.

Parameter

x1	선의 시작점의 x 좌표.
y1	선의 시작점의 y 좌표.
x2	선의 끝점의 x 좌표.
y2	선의 끝점의 y 좌표.

Return Value

없음.

8.19 drawrect

void drawrect(**S16** x, **S16** y, **U16** w, **U16** h)

x, y 좌표를 시작점으로 가로가 w, 세로가 h 인 사각형을 그린다. 사각형의 색은 drawsetrgb() 함수에 의해 설정된 색을 사용한다.

Parameter

x	사각형의 시작점의 x 좌표
y	사각형의 시작점의 y 좌표
w	사각형의 가로 길이.
h	사각형의 세로 길이.

Return value

없음.

8.20 drawrectfill

void drawrectfill(**U32** x, **U32** y, **U32** w, **U32** h)

x, y 좌표를 시작점으로 가로가 w, 세로가 h 인 사각형을 그린다. Drawrect 와 다른 점은 안을 채운 사각형을 그린다. 사각형의 색은 drawsetrgb()함수에 의해 설정된 색을 사용한다.

Parameter

x	사각형의 시작점의 x 좌표
y	사각형의 시작점의 y 좌표
w	사각형의 가로 길이.
h	사각형의 세로 길이.

Return value

없음.

8.21 drawround

void drawround(**S16** x0, **S16** y0, **U16** w, **U16** h, **U16** corner)

x, y 좌표를 시작점으로 가로가 w, 세로가 h 인 모서리가 부드러운 사각형을 그린다. corner 값에 따라 모서리의 부드러운 형태가 조정된다. 색은 drawsetrgb()함수에 의해 설정된 색을 사용한다.

Parameter

x0	사각형의 시작점의 x 좌표
y0	사각형의 시작점의 y 좌표
w	사각형의 가로 길이.
h	사각형의 세로 길이.
corner	모서리의 부드러운 형태 조정 값. 이 값이 클수록 부드러운 부분이 넓어진다.

Return value

없음.

8.22 drawroundfill

void drawroundfill(**S16** x0, **S16** y0, **U16** w, **U16** h, **U16** corner)

x, y 좌표를 시작점으로 가로가 w, 세로가 h 인 모서리가 부드러운 사각형을 그린다. corner 값에 따라 모서리의 부드러운 형태가 조정된다. drawround 와 다르게 안을 채운 사각형을 그린다. 색은 drawsetrgb()함수에 의해 설정 된 색을 사용한다.

Parameter

x0	사각형의 시작점의 x 좌표
y0	사각형의 시작점의 y 좌표
w	사각형의 가로 길이.
h	사각형의 세로 길이.
corner	모서리의 부드러운 형태 조정 값. 이 값이 클수록 부드러운 부분이 넓어진다.

Return value

없음.

8.23 drawcircle

void drawcircle(**U32** x, **U32** y, **U32** rad)

x, y 좌표를 원점으로 하고 반지름이 rad 인 원을 그린다. 원의 색은 drawsetrgb() 함수에 의해 설정된 색을 사용한다.

Parameter

x	원점의 x 좌표
y	원점의 y 좌표
rad	원의 반지름

Return value

없음.

8.24 drawcirclefill

void drawcirclefill(**U32** x0, **U32** y0, **U32** rad)

x, y 좌표를 원점으로 하고 반지름이 rad 인 안을 채운 원을 그린다. 색은 drawsetrgb()함수에 의해 설정된 색을 사용한다.

Parameter

X0	원점의 x 좌표
Y0	원점의 y 좌표
rad	원의 반지름

Return value

없음.

8.25 drawellipse

void drawellipse (**S16** x0, **S16** y0, **U16** Xradius, **U16** Yradius);

x0, y0 좌표를 원점으로 하고 x 축 반지름이 Xradius 이고 y 축 반지름이 Yradius 인 타원을 그린다. 색은 drawsetrgb()함수에 의해 설정 된 색을 사용한다.

Parameter

x0	원점의 x 좌표
x0	원점의 y 좌표
Xradius	타원의 x 축 반지름
Yradius	타원의 y 축 반지름

Return value

없음.

8.26 drawellipsefill

void drawellipsefill (**S16** x0, **S16** y0, **U16** Xradius, **U16** Yradius);

x0, y0 좌표를 원점으로 하고 x 축 반지름이 Xradius 이고 y 축 반지름이 Yradius 인 안을 채운 타원을 그린다. 색은 drawsetrgb()함수에 의해 설정 된 색을 사용한다.

Parameter

x0	원점의 x 좌표
x0	원점의 y 좌표
Xradius	타원의 x 축 반지름
Yradius	타원의 y 축 반지름

Return value

없음.

8.27 loadbmp

SURFACE* loadbmp(**char*** fname);

Bmp 파일을 load 하는 함수. bmp 파일을 load 하여 메모리 영역을 생성 drawsurface( )함수를 사용하여 이미지를 출력할 수 있도록 SURFACE 구조체를 생성 후 저장을 한다.

Parameter

fname bmp 파일 이름

Return value

이미지가 정보와 이미지 데이터가 저장된 메모리 영역의 포인터.

8.28 loadbmpp

SURFACE* loadbmpp(**U8*** startaddr)

BMP 이미지를 메모리에서 load 하는 함수. Loadbmp 와 다른 점은 파일을 load 하는 것이 아니고, 메모리에서 load 한다는 점이다.

Parameter

startaddr 이미지가 저장되어 있는 메모리 주소.

Return value

이미지가 정보와 이미지 데이터가 저장된 메모리 영역의 포인터.

8.29 loadjpg

SURFACE* loadjpg(**char*** fname);

jpg 파일을 load 하는 함수. jpg 파일을 load 하여 메모리 영역을 생성 drawsurface( )함수를 사용하여 이미지를 출력할 수 있도록 SURFACE 구조체를 생성 후 저장을 한다.
(참고로 jpg image 출력을 위해서는 libjpeg.a 파일을 project 에 추가해주어야 한다.)

Parameter

fname jpg 파일 이름

Return value

이미지가 정보와 이미지 데이터가 저장된 메모리 영역의 포인터.

8.30 loadjpgp

SURFACE* loadjpgp(**U8*** databuf, **U32** len)

JPG 이미지를 메모리에서 load 하는 함수. Loadjpg 와 다른 점은 파일을 load 하는 것이 아니고, 메모리에서 load 한다는 점이다.

Parameter

databuf	이미지가 저장되어 있는 메모리 주소.
len	이미지 데이터의 길이

Return value

이미지가 정보와 이미지 데이터가 저장된 메모리 영역의 포인터.

8.31 loadtga

SURFACE* loadtga(**char*** fname)

tga 파일을 load 하는 함수. tga 파일을 load 하여 메모리 영역을 생성 drawsurface( )함수를 사용하여 이미지를 출력할 수 있도록 SURFACE 구조체를 생성 후 저장을 한다.

Parameter

fnametga 파일 이름

Return value

이미지가 정보와 이미지 데이터가 저장된 메모리 영역의 포인터.

8.32 loadtgap

SURFACE* loadtga(**U8*** startaddr)

tga 이미지를 메모리에서 load 하는 함수. loadtga 와 다른 점은 파일을 load 하는 것이 아니고, 메모리에서 load 한다는 점이다.

Parameter

Startaddr	이미지가 저장되어 있는 메모리 주소.

Return value

이미지가 정보와 이미지 데이터가 저장된 메모리 영역의 포인터.

8.33 loadpng

SURFACE* loadpng(**char*** filename)

png 파일을 load 하는 함수. png 파일을 load 하여 메모리 영역을 생성 drawsurface( )함수를 사용하여 이미지를 출력할 수 있도록 SURFACE 구조체를 생성 후 저장을 한다.
(참고로 png image 출력을 위해서는 libpng.a 와 libz.a 파일을 project 에 추가해주어야 한다.)

Parameter

filenamepng 파일 이름

Return value

이미지가 정보와 이미지 데이터가 저장된 메모리 영역의 포인터.

8.34 loadpngp

SURFACE* loadpngp(**U8*** pngbuf, **U32** datalen)

png 이미지를 메모리에서 load 하는 함수. loadpng 와 다른 점은 파일을 load 하는 것이 아니고, 메모리에서 load 한다는 점이다.

Parameter

Pngbuf	이미지가 저장되어 있는 메모리 주소
Datalen	이미지 데이터 길이

Return value

이미지가 정보와 이미지 데이터가 저장된 메모리 영역의 포인터.

8.35 loadimage

SURFACE*loadimage(**char*** fname)

이미지를 load 하는 함수. bmp,jpg,tga,png 이미지에 구분 없이 이 함수로 모두 로드 가능하다. 함수내부에서 이미지를 구분하여 각각 이미지에 맞는 load 함수를 호출한다.

Parameter

fname	이미지 파일 이름

Return value

이미지가 정보와 이미지 데이터가 저장된 메모리 영역의 포인터.

8.36 drawsurface

BOOL drawsurface(**SURFACE*** src_surf, **int** dx, **int** dy)

이미지를 출력할 때 사용하는 함수.

Parameter

src_surf	load 함수에서 return 받은 이미지 데이터가 저장된 메모리 영역 포인터.
dx	이미지가 위치할 x 좌표.
dy	이미지가 위치할 y 좌표.

Return value

TRUE or FALSE

8.37 drawsurfaceRect

BOOL drawsurfaceRect(**SURFACE*** src_surf, **int** dx, **int** dy, **int** sx, **int** sy, **int** w, **int** h)

이미지 중 지정된 영역만 출력할 때 사용하는 함수. 이미지를 지정한 위치부터 지정한 사이즈만큼 dx, dy 에 출력한다. 이미지의 일부분만 출력할 때 사용하면 된다.

Parameter

src_surfload	함수에서 return 받은 이미지 데이터가 저장된 메모리 영역 포인터.
dx	이미지가 위치할 x 좌표.
dy	이미지가 위치할 y 좌표.
sx	이미지 중 출력할 부분의 시작 x 좌표.
sy	이미지 중 출력할 부분의 시작 y 좌표.
w	출력할 이미지의 가로 길이. 시작 sx 값부터 w 만큼 출력한다.
h	출력할 이미지의 세로 길이. 시작 sy 값부터 h 만큼 출력한다.

Return value

TRUE or FALSE

8.38 drawsetclipwinodw

void drawsetclipwindow(**int** x, **int** y, **int** w, **int** h)

이미지가 출력될 수 있는 영역을 제한한다. 좌표 x, y 부터 가로 w 세로 h 영역에만 이미지가 출력되고, 그 외 영역에는 출력되지 않는다.

Parameter

x	이미지 출력을 허용할 영역의 시작 x 좌표
y	이미지 출력을 허용할 영역의 시작 y 좌표
w	이미지 출력을 허용할 영역의 가로 길이
h	이미지 출력을 허용할 영역의 세로 길이

Return value

없음.

8.39 drawsurfacescale

BOOL drawsurfacescale(**SURFACE*** src_surf, **int** dx, **int** dy, **int** dw, **int** dh)

이미지를 확대 또는 축소해서 출력한다. 원본 이미지의 가로 세로 길이보다 dw, dh 값이 크면, 이미지가 확대 돼서 출력되고, dw, dh 값이 작으면, 축소 돼서 출력된다.

Parameter

src_surf	load 함수에서 return 받은 이미지 데이터가 저장된 메모리 영역 포인터.
dx	이미지가 출력 될 x 좌표.
dy	이미지가 출력 될 y 좌표.
dw	출력될 이미지의 가로 길이, 원본 이미지 가로길이보다 이 값이 크면 가로길이가 확대되어 출력되고, 작으면 가로길이가 축소되어 출력된다.
dh	출력될 이미지의 세로 길이, 원본 이미지 세로길이보다 이 값이 크면 세로길이가 확대되고, 작으면 세로길이가 축소되어 출력된다.

Return value

TRUE or FALSE

8.40 drawsurfacescalerect

BOOL drawsurfacescalerect(**SURFACE*** src_surf, **int** dx, **int** dy, **int** dw, **int** dh,
int sx, **int** sy, **int** sw, **int** sh)

drawsurfacescale() 함수와 drawrect()함수를 합쳐 놓은 함수이다. 이미지의 특정 부분을 확대 또는 축소하여 출력한다. 이미지의 (sx,sy)부터 sw,sh 영역의 이미지를 dx,dy 좌표에 dw, dh 크기로 출력한다. sw,sh 값보다 dw, dh 값이 크면 확대, 작으면 축소이다.

Parameter

src_surf	load 함수에서 return 받은 이미지 데이터가 저장된 메모리 영역 포인터.

dx	이미지가 출력될 x 좌표
dy	이미지가 출력 될 y 좌표
dw	출력될 이미지의 가로 길이, 출력할 부분 이미지의 가로길이보다 크면 가로로 확대되고, 작으면 가로로 축소된다.
dh	출력될 이미지의 세로 길이, 출력할 부분 이미지의 세로길이보다 크면 세로로 확대되고, 작으면 세로로 축소된다.
sx	원본 이미지 중 출력할 부분의 시작 x 좌표.
sy	원본 이미지 중 출력할 부분의 시작 y 좌표
sw	출력할 이미지의 가로 길이. 시작 sx 값부터 sw 만큼 출력한다
sh	출력할 이미지의 세로 길이. 시작 sy 값부터 sh 만큼 출력한다

Return value

TRUE or FALSE

8.41 releasesurface

void releasesurface(**SURFACE*** surf);

이미지 load 함수에 의해 생성된 메모리 영역을 해제한다. 메모리가 제한적이기 때문에 사용하지 않는 이미지에 대해서 releasesurface( )함수로 메모리 해제해주는 것이 좋다.

Parameter

surf	load 함수에서 return 받은 이미지 데이터가 저장된 메모리 영역 포인터..

Return value

없음.

8.42 createsurface_from

SURFACE* createsurface_from(**SURFACE*** src, **U32** option);

이미지의 복사본을 만든다. option 을 통해 좌/우 90 도 회전, 180 도 회전, 좌우/상하 뒤집은 복사본을 만들 수 있다. 이미지를 회전하여 출력할 경우 이미지를 load 한 후에 createsurface_from 함수를 사용하여 회전한 복사본을 만들어 출력하면 된다. 회전과 관련된 option 은 다음과 같으며, 원본 그대로의 복사본을 원하면, option 값으로 0 을 적어주면 된다. 복사본이 생성 되는 것으로 원본이미지는 그대로 사용할 수 있다.

```
#define PIVOT_RIGHT (1)
#define PIVOT_LEFT (1<<1)
```

```
#define PIVOT_180 (1<<2)
#define PIVOT_VFLIP (1<<4)
#define PIVOT_HFLIP (1<<5)
```

Parameter

src	복사할 surface 의 원본 source.
option	복사본을 만들면서 적용할 option.

Return value

복사본 이미지 정보와 데이터가 저장된 메모리 영역의 포인터.

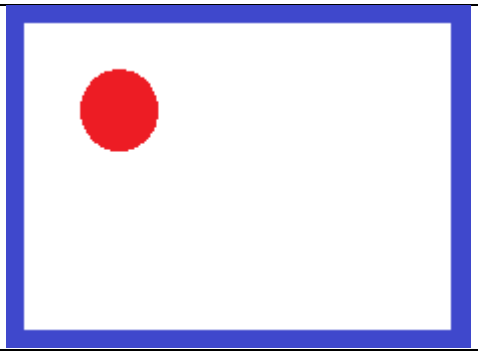
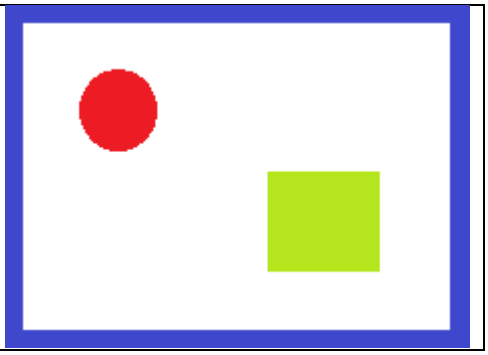
Example

```
SURFACE* surface = loadbmp("test.bmp");
SURFACE* surface_right =createsurface_from(surface, PIVOT_RIGHT)
                        // 원본이미지를 오른쪽 90 도 회전한 복사본 이미지를 만든다.
SURFACE* surface_left =createsurface_from(surface, PIVOT_LEFT)
                        // 원본이미지를 왼쪽 90 도 회전한 복사본 이미지를 만든다.
SURFACE* surface_right =createsurface_from(surface, PIVOT_180)
                        // 원본이미지를 180 도 회전한 복사본 이미지를 만든다.
SURFACE* surface_right =createsurface_from(surface, PIVOT_VFLIP)
                        // 원본이미지를 상하 뒤집은 복사본 이미지를 만든다.
SURFACE* surface_right =createsurface_from(surface, PIVOT_HFLIP)
                        // 원본이미지를 좌우 뒤집은 복사본 이미지를 만든다.

drawsurface(surface,0,0);
drawsurface(surface_right,200,0);
drawsurface(surface_left,0,200);
.
.
.
```

8.43 single frame & double frame사용 예

< SINGLE FRAME >

Single frame 은 화면에 보여지는 frame 을 하나만 생성한다.
그래서 현재 보여지는 화면에 이미지 및 도형을 그리는 동작을 수행한다.

현재 보여지는 frame		

위 그림처럼 frame 이 하나만 생성이 되고, 이미지 및 도형은 하나의 frame 에 계속 그려지게 된다.

Single frame 설정 및 사용법은 다음과 같다.

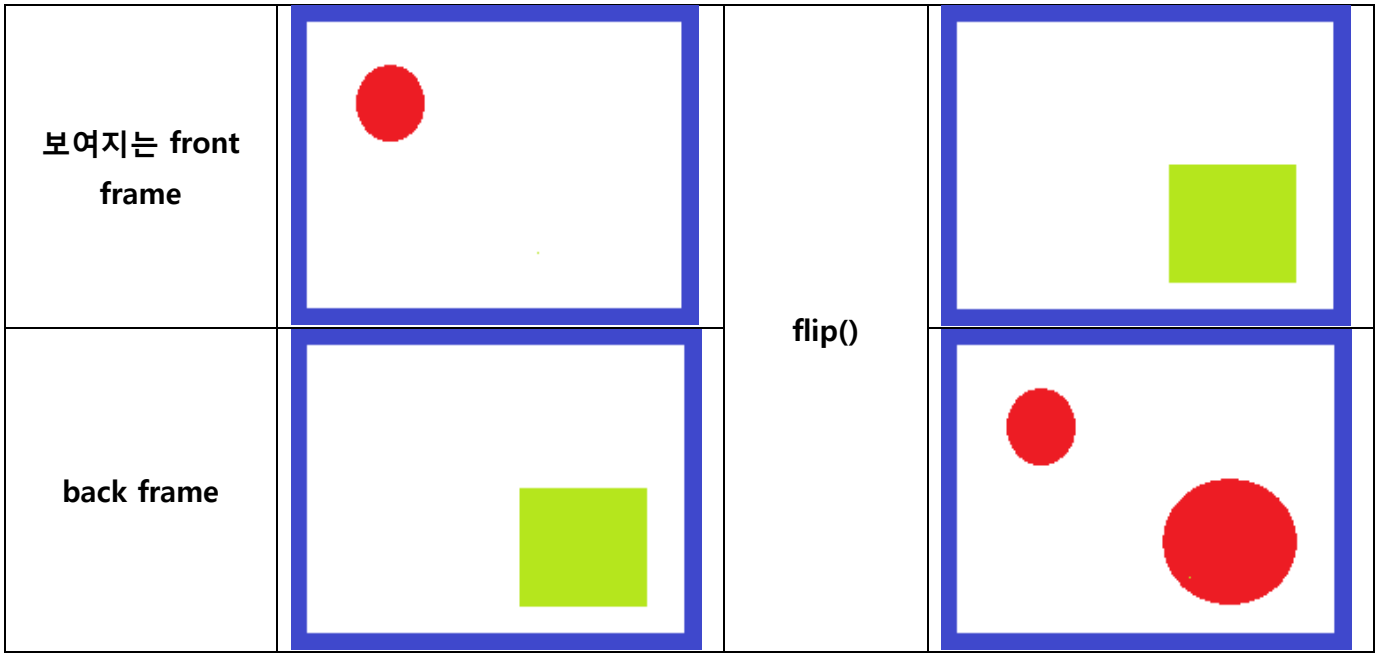
```
setscreen(SCREEN_800x480, SCREENMODE_RGB565 )
SURFACE* frame = createframe(800, 480, 16);
setframebuffer(frame);
setdrawtarget(frame);

setdrawrgb(255,0,0);
drawcirclefill(50,50,10);
setdrawrgb(0,255,0);
drawrect(100,100,100,100);
```

frame 하나를 생성하고, setframebuffer()함수와 setdrawtarget() 함수로 생성한 하나의 frame 을 설정한 후, 이미지 또는 도형을 그리면 된다.

<DOUBLE FRAME >

Double frame 은 frame 을 두 개 생성한다. 그래서 한 개의 frame 은 화면에 보여지고, 다른 한 개의 frame 은 화면에 보여지지 않는다. 편의상 화면에 보여지는 frame 을 front frame 이라 하고, 화면에 보이지 않는 frame 을 back frame 라 하겠다.

Double frame 을 사용하게 되면 back frame 에 이미지 및 도형을 그려서 현재 보여지는 front frame 에 영향을 주지 않고, 이미지 및 도형을 그릴 수 있다.



위 그림에서 보듯이 back frame 에 도형을 그려도 front frame 에는 영향을 미치지 않아, 화면에 변화를 주지 않는다. Back frame 에 도형을 그린 후 flip()함수를 호출하면 frame 간에 전환이 이루어져 back frame 이 front frame 이 되면서 화면에 보여지게 되고, front frame 이 back frame 이 되면서 화면에 보이지 않게 된다.

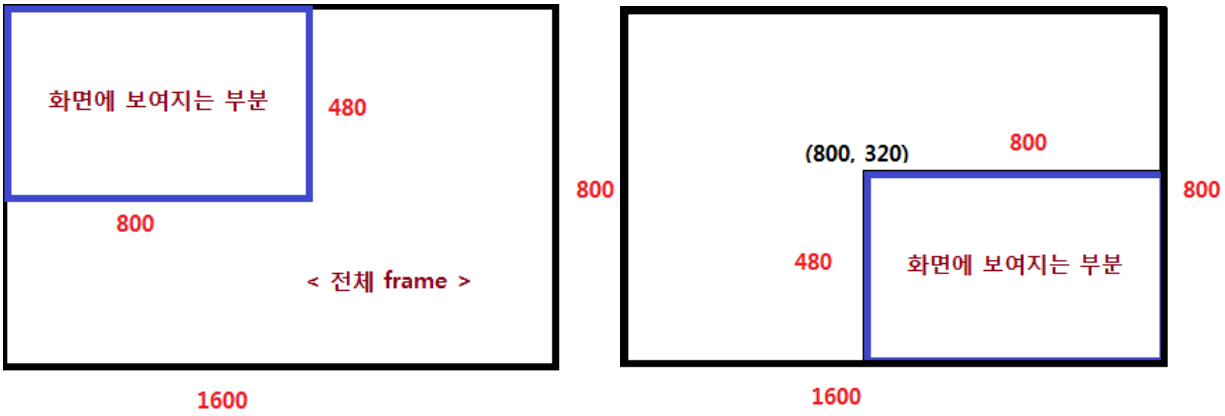
double frame 의 사용법은 다음과 같다.

```
setscreen(SCREEN_800x480, SCREENMODE_RGB565 )
SURFACE* frame1 = createframe(800, 480, 16);
SURFACE* frame2 = createframe(800, 480, 16);

setdoubleframebuffer(frame1, frame2);

setdrawtarget(getbackframe());
setdrawrgb(255,0,0);
drawcirclefill(50,50,10);
flip();
```

```
setdrawtarget(getbackframe());
setdrawrgb(0,255,0);
drawrect(600,300,100,100);
flip();

setdrawtarget(getbackframe());
setdrawrgb(255,0,0);
drawrect(600,300,20);
```

frame 을 두 개를 생성하고, setdrawtarget() 함수를 사용하여 backframe 을 draw target 으로 설정 한 후 이미지 및 도형을 그린다. 다 그린 후에 flip()함수를 호출하여 back frame 을 front frame 으로 만들어 화면에 보이게 한다. Front frame 은 back frame 이 되고, setdrawtarget() 함수로 draw target 이 되어 이미지 및 도형이 그려지게 된다.

8.44 set frame buffer xy사용 예



setframebufferxy 함수를 사용하여, 화면 크기보다 큰 frame 을 생성하였을 때 위 그림처럼 특정 위치만 화면에 보여지게 할 수 있다.

```
setscreen(SCREEN_800x480, SCREENMODE_RGB565);
SURFACE *frame = createframe(1600,800,16);
setframebuffer(frame);
setdrawtarget(frame);
.....
.....
setframebufferxy(frame,800,320);    // 위의 두 번째 그림처럼 전체 frame 중 800,320 부터
                                    화면에 보여지게 된다.
```

위 코드에서 처럼 frame 의 크기를 화면 해상도보다 크게 설정하고, setframebuffer( )함수로 화면 해상도보다 큰 frame 을 선택한다. 그런 후에 setframebufferxt()함수로 화면에 보여지는 부분의 시작점을 설정하면 frame 의 그 지점부터 화면에 보여지게 된다.

8.45 Graphic Example

< SINGLE FRAME >

```
#include "adStar.h"

int main()
{
        boardinit();
        uart_config(0,115200,DATABITS_8,STOPBITS_1,UART_PARNONE);
        debugstring("================================================₩r₩n");
        debugprintf("ADSTAR Graphic.System Clock(%dMhz)₩r₩n",get_ahb_clock()/1000000);
        debugstring("================================================₩r₩n");

         crtc_clock_init(); // CRT clock enable.
         setscreen(SCREEN_800x480,SCREENMODE_RGB565);// 해상도 및 draw mode 를 설정한다.
         SURFACE *frame = createframe(800,480,16);// frame 을 생성한다.
         setframebuffer(frame);   // 화면에 보여질 frame 을 선택한다.
         setfrawtarget(frame);// 이미지 및 도형이그려질 frame 을 선택한다.

         lcdon();
         drawsetrgb(255,255,255);// 도형의 색을 흰색으로 설정한다.
         drawrectfill(0,0,getscreenwidth(),getscreenheight());
                        // 0,0 부터 800,480 크기의 흰색 채운 사각형을 기린다.  화면 clear.
         drawsetrgb(255,0,0);

         drawline(30,30,100,100);          // 30,30 에서 100,100 까지 선을 그린다.
         drawrect(10,10,100,100);          // 10,10 부터 가로 세로 크기가 100 인 사각형을 그린다.
         drawcircle(100,100,10);          // 원점이 100,100 이고 반지름이 10 인 원을 그린다.
         drawcirclefill(200,200,20);          // 원점이 200,200 이고 반지름이 20 인 채운 원을 그린다.
         drawround(50,50,100,100,5);      // 50,50 부터 가로 세로 크기가 100 인 모서리가 부드러운
                                           //사각형을 그린다.
         drawellipse (300, 100, 20, 40);// 원점이 300,100 이고, x 축 반지름이 20, y 축 반지름이 40 인
                                           타원을 그린다.

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

< DOUBLE FRAME >

```
#include "adStar.h"

int main()
{
	boardinit();
	 uart_config(0,115200,DATABITS_8,STOPBITS_1,UART_PARNONE);
	 debugstring("==============================================₩r₩n");
	 debugprintf("ADSTAR Graphic.System Clock(%dMhz)₩r₩n",GetAHBclock()/1000000);
	 debugstring("==============================================₩r₩n");

	  FATFS fs;
	  F_mount(DRIVE_NAND, &fs);
	  crtc_clock_init(); // CRT clock enable.
	  setscreen(SCREEN_800x480,SCREENMODE_RGB565);// 해상도 및 draw mode 를 설정한다.
	  SURFACE *frame1 = createframe(800,480,16);          // frame1 을 생성한다.
	  SURFACE *frame2 = createframe(800,480,16);          // frame2 을 생성한다.
	                //double frame 사용을 위해 frame 을 2 개 생성한다.
	  setdoubleframebuffer(frame1,frame2);              //double frame 으로 설정을 한다..
	  lcdon();
	  setdrawtarget(getbackframe());// 이미지 및 도형이그려질 frame 을
	  // back frame 으로 선택한다.
	  SURFACE* image1 = loadbmp("test1.bmp");// Nand 에서 test1.bmp 파일을 load 한다.
	  drawsurface(image1,0,0);// bmp 이미지를 0,0 에 출력한다. (back frame)
	  flip();// frame 을 전환하여 back frame 을 화면에 보여준다.

	  setdrawtarget(getbackframe());// 이미지 및 도형이 그려질 frame 을
	  // back frame 으로 선택한다.

	  SURFACE* image2 = loadjpg("test2.jpg");// Nand 에서 test2.jpg 파일을 load 한다.
	  drawsurfaceRect(image2,0,0,100,100,200,200);// jpg 이미지를 0,0 에 출력하는데,
	    // jpg 이미지 중 100,100 부터 가로 세로 200 사이즈만큼 출력한다.
	  flip();// frame 을 전환하여 back frame 을 화면에 보여준다. (jpg image).
	  releasesurface(image1);// 사용하지 않는 image1 을 메모리에서 제거한다.
	  while(1);
	   return 0;
}
```

9. Sound

이번 장에서는 adStar sound 출력에 대해 알아보겠다.

adStar 는 8bit/16bit, signed/unsigned, mono/stereo WAV 와 MP3 를 지원한다.

9.1 sount_init()

BOOLsount_init

Sound 출력을 위해 Soundmixer 를 초기화 하는 함수.

Parameter

없음.

Return value

없음.

9.2 sound_loadwav

WAVE* sound_loadwav(**char*** filename);

WAV 파일을 load 하는 함수. WAV 파일을 load 하여 WAVE 구조체를 생성 메모리에 저장한다.

Parameter

filename WAV 파일

Return value

WAV 파일의 정보와 데이터가 저장된 WAVE 구조체.

9.3 sound_loadwavp

WAVE* sound_loadwavp(**U8*** buf, **U32** len);

WAV data 를 메모리에서 load 하는 함수.

Parameter

buf WAV data 가 저장되어 있는 buffer.

len WAV data 의 길이.

Return value

WAV 파일의 정보와 데이터가 저장된 WAVE 구조체.

9.4 sound_loadmp3

WAVE* sound_loadmp3(**char*** filename);

MP3 파일을 load 하는 함수. MP3 파일을 load 하여 WAVE 구조체를 생성 메모리에 저장한다.

Parameter

filename MP3 파일

Return value

MP3 파일의 정보와 데이터가 저장된 WAVE 구조체.

9.5 sound_loadmp3p

WAVE* sound_loadmp3p(**void*** buf, **U32** mp3databufsize);;

MP3 data 를 메모리에서 load 하는 함수.

Parameter

buf MP3 data 가 저장되어 있는 buffer.

Mp3databufsize mp3 data 의 길이.

Return value

MP3 파일의 정보와 데이터가 저장된 WAVE 구조체.

9.4 sound_release

BOOL sound_release(**WAVE*** pWave);

load_soundwav() 또는 load_soundmp3() 함수에 의해 생성된 메모리를 해제한다. 사용할 수 있는 메모리가 제한적이기 때문에 사용하지 않는 sound 대해 sound_release( )함수를 호출하여 사용하지 않는 메모리를 반환하는 것이 좋다.

Parameter

pWavesound_loadwav( )함수 또는 sound_loadmp3( )함수에 의해 생성된 WAVE 구조체.

Return value

TRUE(1) or FALSE(0)

9.5 sound_play

BOOL sound_play(**WAVE*** pWave);

load_soundwav() 또는 load_soundmp3() 함수로 load 한 sound 를 play 한다.

Parameter

pWavesound_loadwav( )함수 또는 sound_loadmp3( )함수에 의해 생성된 WAVE 구조체.

Return value

TRUE(1) or FALSE(0)

➔adStar 는 sound 출력을 4 개채널 중 하나를 선택해서 할 수 있다. 0 번 1 번 채널은 I2S 를 통한 출력이고, 2 번 3 번 채널은 digital modulator 를 통한 출력이다. SDK 에서는 2 번 채널을 default 출력으로 정하고 있는데, 이를 변경하기 위해서는 include 안의 soundmixer.h 파일의 "#define SND_OUTPUT_CHANNEL 2"에서 2 가 채널을 나타내는 값으로 이 값을 수정하면 다른 채널로 출력이 가능하다.

9.6 sound_stop

BOOL sound_stop(**WAVE*** pWave)

play 중인 sound 를 stop 시킨다.

Parameter

pWave 현재 play 중인 sound 의 WAVE 구조체.

Return value

TRUE(1) or FALSE(0)

9.7 sound_vol

void sound_vol(**U8** vol);

sound volume 을 조정한다.

Parameter

vol volume 값. 범위는 0 부터 255 이다.

Return value

없음

9.7 sound_vol_wav

void sound_vol_wav(**WAVE*** pWav, **U8** vol);

음원 파일 각각의 volume 을 조정한다. sound_vol() 함수와 별도로 volume 을 설정한다. 두 개의 음원을 동시에 출력할 때 음원 마다 volume 크기를 다르게 하고 싶을 경우에 사용하면 된다.

Parameter

pWave volume 을 조정할 sound 구조체

vol volume 값. 범위는 0 부터 255 이다.

9.8 sound_pause

BOOL sound_pause(**WAVE*** pWave);

Play 중인 sound 를 pause 시킨다.

Parameter

pWave 현재 play 중인 sound 의 WAVE 구조체.

Return value

TRUE(1) or FALSE(0)

9.9 sound_resume

BOOL sound_resume(**WAVE*** pWave);

Pause 상태의 sound 를 resume 시킨다.

Parameter

pWave 현재 pause 중인 sound 의 WAVE 구조체.

Return value

TRUE(1) or FALSE(0)

9.10 sound_isplay

BOOL sound_isplay(**WAVE*** pWave);

현재 pWave sound 가 play 중인지 확인할 수 있다.

Parameter

pWave 현재 play 중인지 확인하려는 sound 의 WAVE 구조체.

Return value

TRUE(1 = play) or FALSE(0 = stop)

9.11 sound_ispause

BOOL sound_ispause(**WAVE*** pWave);

현재 pWave sound 가 pause 중인지 확인할 수 있다.

Parameter

pWave 현재 pause 중인지 확인하려는 sound 의 WAVE 구조체.

Return value

TRUE(1 = pause) or FALSE(0 = none pause)

9.12 Sound Example

< WAV File Play >

```
#include "adStar.h"

Intmain()
{
    boardinit();
    uart_config(0,115200,DATABITS_8,STOPBITS_1,UART_PARNONE);
    debugstring("==============================================₩r₩n");
    debugprintf("ADSTAR Wave Play.System Clock(%dMhz)₩r₩n",get_ahb_clock()/1000000);
    debugstring("==============================================₩r₩n");

    FATFS fs;
    f_mount(DRIVE_NAND, &fs);

    sount_init();                    // Sound 출력을 위한 sound mixer 초기화
    WAVE* sound = sount_loadwav("test.wav");    // wav 파일 load

    sound_play(sound);                          // wav 파일 재생

    delayms(1000);

    If(sound_isplay(sound))                     // 1 초후에 wav 파일 재생 중인지 확인
    {
        sound_stop(sound);                      // 재생 중이면 재생 중지
    }

    sound_release(sound);                   // 더 이상 사용하지 않을 sound 라 메모리에서 제거.

      …
      …

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

< MP3 File Play >

```
#include “adStar.h“

Intmain()
{
    uart_config(0,115200,DATABITS_8,STOPBITS_1,UART_PARNONE);
    debugstring("==========================================₩r₩n");
    debugprintf("ADSTAR MP3Play.System Clock(%dMhz)₩r₩n",get_ahb_clock()/1000000);
    debugstring("==========================================₩r₩n");

    FATFS fs;
    f_mount(DRIVE_NAND, &fs);

    sound_init();                       // Sound 출력을 위한 sound mixer 초기화
    WAVE* sound = sount_loadmp3(“test.mp3”);    // mp3 파일 load

    sound_play(sound);                          // mp3 파일 재생

    delayms(1000);

    sound_pause(sound);                     // 1 초 후에 Sound pause

    delayms(1000);

    If(sound_ispause(sound))                    //  1 초 후에 Sound 가 pause 상태인지 확인
    {
        sound_resume(sound);                        // pasuse 중이면 resume
    }

    …
    …

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

10. File System

adStar SDK 는 Nand Flash 와 SD Card 에 file system 을 적용하여 file 저장 장치로 사용할 수 있다. 이번 장에서는 Nand Flash 와 SD Card 를 file 저장장치로 사용하기 위해 초기 설정하는 법에 대해 설명을 하고, 자세한 file system 사용에 관한 내용은 adStar SDK 중 doc 폴더안의 fatfs 매뉴얼을 참고하기 바란다.(SDK/doc/fatfs/doc/00index_e.html)

10.1 f_mount

FRESULT f_mount(BYTE, FATFS*);

Nand Flash 또는 SD Card 를 mount 하여 file 저장 장치로 사용하도록 설정한다.

Parameter

BYTE — Nand Flash 또는 SD Card 의 장치 번호. 이에 대해서는 다음과 같이 정의 되어 있다.

```
#define DRIVE_NAND      0
#define DRIVE_SDCARD    1
```

FATFS — 장치 정보를 가지고 있는 구조체

Result value

동작의 결과.

0 인 경우 정상 동작 한 것이다. 이 외에 결과값은 fatfs 매뉴얼을 참조하기 바란다.

10.2 f_chdrive

FRESULT f_chdrive(BYTE);

Mount 된 driver 간에 drive change 할 때 사용한다.

Parameter

BYTE — Nand Flash 또는 SD Card 의 장치 번호.

```
#define DRIVE_NAND      0
#define DRIVE_SDCARD    1
```

Return value

동작의 결과.

0 인 경우 정상 동작 한 것이다. 이 외에 결과값은 fatfs 매뉴얼을 참조하기 바란다.

10.3 f_chdir

FRESULT f_chdir(const TCHAR*);

Mount 된 driver 에서 directory 이동할 때 사용한다.

Parameter

TCHAR* 이동 할 directory 명

Return value

동작의 결과.

0 인 경우 정상 동작 한 것이다. 이 외에 결과값은 fatfs 매뉴얼을 참조하기 바란다.

10.4 FILE System Example

```
int main()
{
        init_interrupt();    // Interrupt 초기화
        uart_config(0, 115200, DATABITS_8, STOPBITS_1, UART_PARNONE );

       FATFS nand_fs, sdcard_fs;
        f_mount(DRIVE_NAND, &nand_fs);// Nand Flash mount
        f_mount(DRIVE_SDCARD, &sdcard_fs);// SD Card mount

       SURFACE* bmp = loadbmp("test.bmp");  // Nand Flash 에서 이미지 파일 Load

        f_chdrive(DRIVE_SDCARD);                  // SD Card 로 drive change

       WAVE* wav = sound_loadwav("test.wav");  // SD Card 에서 사운드 파일 Load

        f_chdir("font");                                // SD Card 의 font directory 로 이동.

        …
        …

      while(1)
      return 0;
}
```

11. Font

adStar SDK 에서는 두 종류의 font 를 사용할 수 있다. 이미지 폰트(편의상 bm font 라 하겠다)와 bit 타입의 폰트(편의상 bit font 라 하겠다)가 있는데, 이번 장에서는 이 두 폰트를 사용하기 위해 필요한 함수에 대해 설명하도록 하겠다.
bm font 는 프로그램을 사용하여 폰트 이미지를 제작해야 하지만 여러 종류의 폰트를 쉽게 사용할 수 있다. bf font 는 따로 이미지 제작 없이 제공되는 파일을 가지고 사용할 수 있지만 폰트의 종류는 변경할 수 없다.

bmfont 의 제작 방법은 추가로 제공되는 문서를 참고하기 바란다.

<< bm font >>

11.1 bmfont_init

BMFont* bmfont_init(**const char** *fontFile)

폰트 파일을 load 하여 bmfont 를 사용할 수 있도록 준비한다.

Parameter

fontFile 폰트 파일 이름

Result value

bmfont 포인터.

11.2 bmfont_setfont

Bool bmfont_setfont(**BMFont*** pFont);

init 한 bmfont 중 사용할 폰트를 결정한다.

Parameter

pFont bmfont 포인터

Return value

TRUE(1) or FALSE(0)

11.3 bmfont_write

int bmfont_write(**int** x, **int** y, **const char** *text)

bmfont 를 사용하여 특정 위치에 문자를 출력한다.

Parameter

x	문자가 출력될 x 좌표
y	문자가 출력될 y 좌표
text	출력될 문자(열)

Return value

마지막 문자가 출력된 x 좌표

11.4 bmfont_setrgb

bool bmfont_setrgb(**BMFont*** pFont, **U8** r, **U8** g, **U8** b)

폰트의 색을 변경한다.

Parameter

pFont	색을 변경하려는 bmfont 포인터.
r	RGB 중 R 값
g	RGB 중 G 값
b	RGB 중 B 값

Return value

TRUE(1) or FALSE(0)

11.5 bmfont_close

i**nt** bmfont_close(**BMFont*** pFont)

더 이상 사용하지 않는 bmfont 를 release 한다.

Parameter

pFont	더 이상 사용하지 않는 bmfont 포인터.

Return value

TRUE(1) or FALSE(0)

11.6 bmfont_makesurface

SURFACE* bmfont_makesurface(**char*** text)

자주 사용하는 문자를 이미지화 시켜서 필요할 때마다 draw 함수를 사용하여 출력할 수 있도록 만들어 준다. surface 를 생성하기 때문에 더 이상 사용하지 않을 경우에는 releasesurface() 함수로 release 해주어야 한다.

Parameter

Text	이미지로 만들 문자(열)

Return value

SURFACE 구조체

11.7 bmfont_setkerning

bool bmfont_setkerning(**VMFont*** pFont, **int** k)

bmfont 의 자간을 조정한다.

Parameter

pFont	bmfont 포인터
k	자간 간격

Return value

TRUE(1) or FALSE(0)

11.8 bmfont_setautokerning

bool bmfont_setautokerning(**bool** b)

폰트 한 개가 차지하는 폭을 일정하게 유지여부를 지정한다.

Parameter

b	폭을 일정하게 유지할지 여부, TRUE or FALSE.

Return value

TRUE(1) or FALSE(0)

11.9 bmfont_write_vleft

int bmfont_write_vleft(**int** x, **int** y, **const char** *test)

bmfont 를 사용하여 특정 위치에 문자를 출력하는데, 좌측으로 90 도 회전한 문자를 출력한다.

Parameter

x	문자가 출력될 x 좌표
y	문자가 출력될 y 좌표
text	출력될 문자(열)

Return value

마지막 문자가 출력된 y 좌표

11.10 bmfont_write_vright

int bmfont_write_vleft(**int** x, **int** y, **const char** *test)

bmfont 를 사용하여 특정 위치에 문자를 출력하는데, 우측으로 90 도 회전한 문자를 출력한다.

Parameter

x	문자가 출력될 x 좌표
y	문자가 출력될 y 좌표
text	출력될 문자(열)

Return value

마지막 문자가 출력된 y 좌표

<< bit font >>

Bit font 를 사용하기 위해서는 project 에 etc_driver 폴더에 있는 bit_font.c 파일을 추가해 주어야 한다.

11.11 bf_init

bool bf_init( )

bit font 를 사용할 수 있도록 초기화 한다.

Return value

TRUE(1) or FALSE(0)

11.12 bf_setrgb

void bf_setrgb(**U8** r, **U8** g, **U8** b)

Bit font 의 색을 변경한다.

Parameter

r	RGB 중 R 값
g	RGB 중 G 값
b	RGB 중 B 값

11.13 bf_setsize

void bf_setsize(**int** eng_w, **int** eng_h, **int** kor_w, **int** kor_h)

Bit font 의 크기를 변경한다. 크기가 너무 작거나 크게 되면 정상적으로 출력되지 않을 수 있다.

Parameter

eng_w	영어 font 의 가로크기
eng_h	영어 font 의 세로크기
kor_w	한글 font 의 가로크기
kor_h	힌글 font 의 세로크기

11.14 bf_drawstring

int bf_drawstring(**int** x, **int** y, **char*** str)

bit font 를 사용하여 문자를 출력한다.

Parameter

x	문자를 출력할 x 좌표
y	문자를 출력할 y 좌표
str	출력할 문자(열)

11.15 bf_drawstring_vleft

int bf_drawstring_vleft(**int** x, **int** y, **char*** str)

bit font 를 사용하여 문자를 출력하는데, 좌측으로 90 도 회전하여 출력한다.

Parameter

x	문자를 출력할 x 좌표
y	문자를 출력할 y 좌표
str	출력할 문자(열)

11.16 bf_drawstring_vleft

int bf_drawstring_vright(**int** x, **int** y, **char*** str)

bit font 를 사용하여 문자를 출력하는데, 우측으로 90 도 회전하여 출력한다.

Parameter

x	문자를 출력할 x 좌표
y	문자를 출력할 y 좌표
str	출력할 문자(열)

11.17 bf_makesurface

SURFACE* bf_makesurface(**char*** str)

자주 사용하는 문자를 이미지화 시켜서 필요할 때마다 draw 함수를 사용하여 출력할 수 있도록 만들어 준다. surface 를 생성하기 때문에 더 이상 사용하지 않을 경우에는 releasesurface() 함수로 release 해주어야 한다.

Parameter

str 이미지로 만들 문자(열)

Return value

SURFACE 구조체

11.18 FONT Example

< bm font example >

```
#include "adStar.h"

int main()
{
	boardinit();
	uart_config(0,115200,DATABITS_8,STOPBITS_1,UART_PARNONE);
	debugstring("==========================================₩r₩n");
	debugprintf("ADSTAR FONT Example. System Clock(%dMhz)₩r₩n",get_ahb_clock()/1000000);
	debugstring("==========================================₩r₩n");

	FATFS fs;
	f_mount(DRIVE_NAND,&fs);
	f_chdrive(DRIVE_NAND);

	crtc_clock_init();
	SURFACE*  frame = createframe(800,480,16);
	setscreen(SCREEN_800x480,SCREENMODE_RGB565);
	lcdon();

	setframebuffer(frame);
	setdrawtarget(frame);
	drawsetrgb(255,255,255);
	drawrectfill(0,0,getscreenwidth(),getscreenheight());

	f_chdir("font");		//font 파일이 있는 font 폴더로 이동.
	BMFont*  pFont = bmfont_init("nanum_8bit_16px_han.fnt");
```

```
                        // nanum_8bit_16px_han.fnt 파일을 load 하여, 사용할 수 있도록 setting.
    bmfont_setrgb(pFont,0,0,255);
                        // nanum 체의 font 색을 파란색으로 변경.
    BMFont*  pFont2 = bmfont_init("batang_24.fnt");
                        // batang_24.fnt 파일을 load 하여, 사용할 수 있도록 setting.
    bmfont_setrgb(pFont2,255,0,0);
                        // batang 체의 font 색을 빨간색으로 변경.
     bmfont_setfont(pFont);      // nanum 체를 font 로 사용하도록 설정.
     bmfont_wrtie(100,100,"나눔체 테스트");
                        // 현재 선택된 font(nanum 체)를 사용하여 문자열을 100,100 위치에 출력.
     bmfont_setfont(pFont2);     // batang 체를 font 로 사용하도록 설정.
     bmfont_write(100,200,"바탕체 테스트");
                      // 현재 선택된 font(batang 체)를 사용하여 문자열을 100,200 위치에 출력.

     SURFACE*  fontsurf = bmfont_makesurface("bmfont_makesurface 로 만든 surface");
     // 문자열을 하나의 image 로 만듦. drawsurface 함수를 사용하여 image 처럼 사용할 수 있다.
     drawsurface(fontsurf,100,300);
           // image 로 만든 font 를 100,300 위치에 출력.
     while(1);
     return 0;
}
```

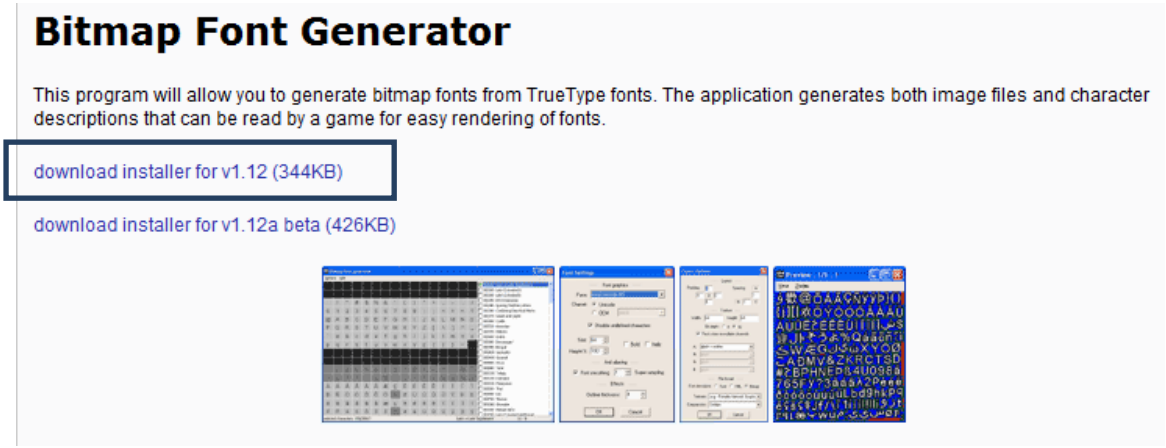
< bit font example >

```
#include "adStar.h"

int main()
{
    boardinit();
    uart_config(0,115200,DATABITS_8,STOPBITS_1,UART_PARNONE);
    debugstring("=============================================₩r₩n");
    debugprintf("ADSTAR FONT Example. System Clock(%dMhz)₩r₩n",get_ahb_clock()/1000000);
    debugstring("=============================================₩r₩n");

     FATFS fs;
     f_mount(DRIVE_NAND,&fs);
     f_chdrive(DRIVE_NAND);
```

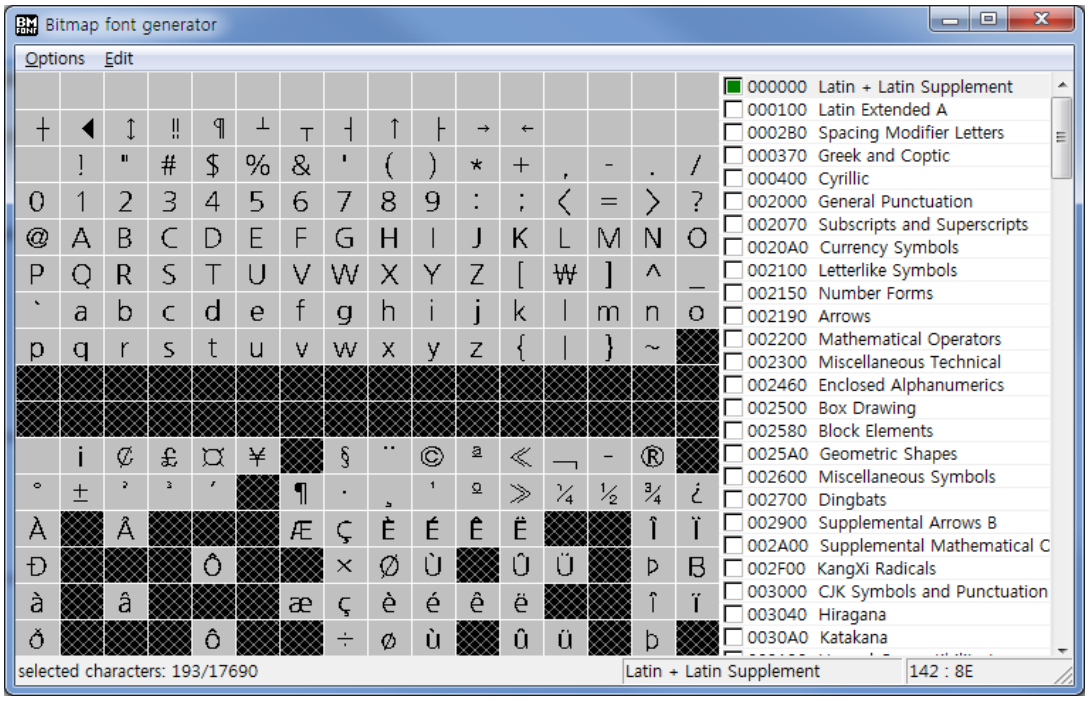
```
    crtc_clock_init();
    SURFACE*  frame = createframe(800,480,16);
    setscreen(SCREEN_800x480,SCREENMODE_RGB565);
    lcdon();

    setframebuffer(frame);
    setdrawtarget(frame);
    drawsetrgb(255,255,255);
    drawrectfill(0,0,getscreenwidth(),getscreenheight());

    bf_init();                    // bit type font 를 사용하도록 setting.
    bf_setrgb(255,0,0);           // bit type font 색을 빨간색으로 변경.
    bf_drawstring(100,100,"bit type font test");
                                  // bit type font 를 사용하여 100,100 위치에 문자열을 출력.
    bf_setsize(30,30,30,30);      // bit type font 크기를 변경.
    bf_setrgb(0,0,255);           // bit type font 색을 파란색을 변경
    bf_drawstring(100,200,"bit type font test");
                                  // bit type font 를 사용하여 100,200 위치에 문자열을 출력.
    SURFACE*  fontsurf = bf_makesurface("bf_makesurface 로 만들어진 surface");
    // 문자열을 하나의 image 로 만듦. drawsurface 함수를 사용하여 image 처럼 사용할 수 있다.
    drawsurface(fontsurf,100,300);   // image 로 만든 font 를 100,300 위치에 출력.

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

< font image 제작 방법 >

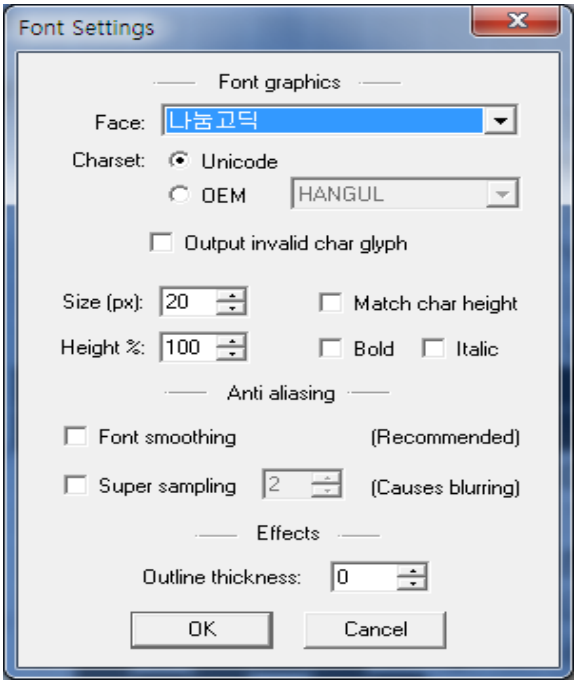
1. http://www.angelcode.com/products/bmfont 에서 Bitmap Font Generator v1.12 를 다운받아 설치한다.



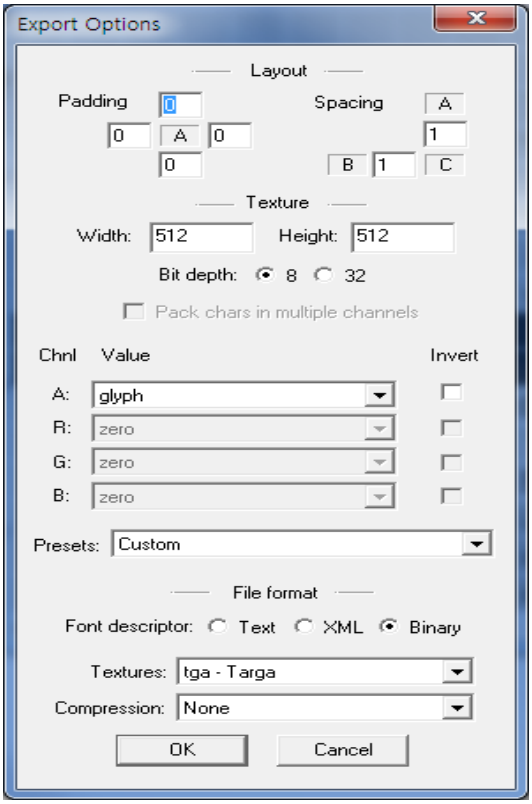
2. 설치한 후 실행하면 아래 그림과 같은 창이 나온다.



3. Options → Font settings 을 실행하여 원하는 Font 와 Font 의 size 를 결정한다.

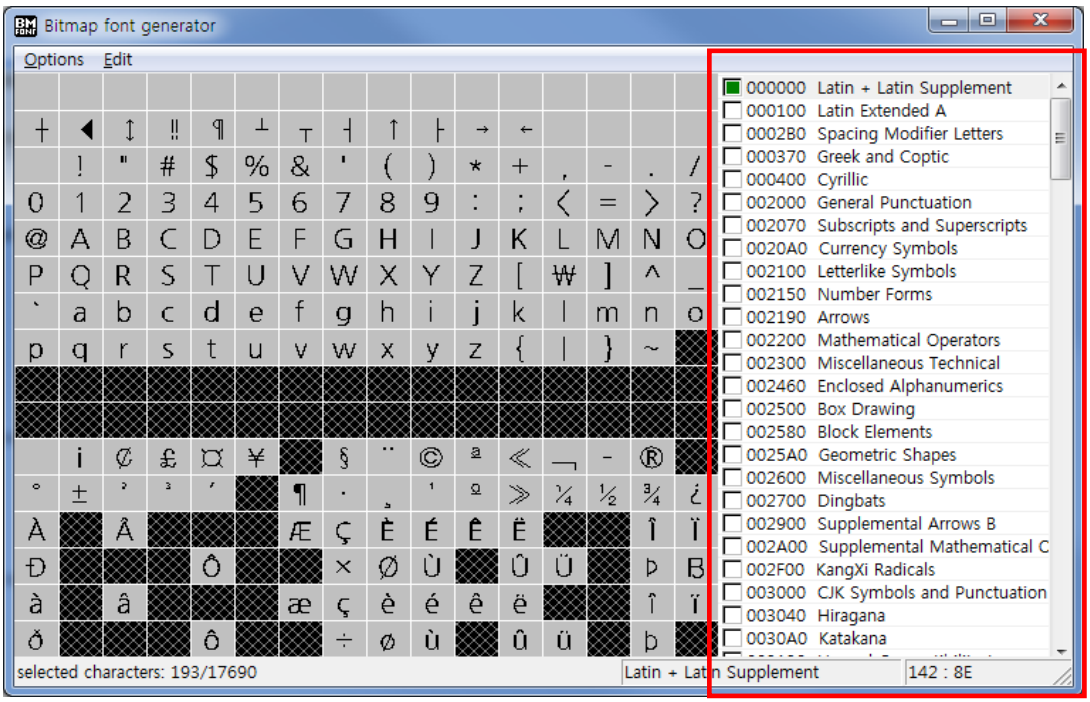


4. Options → Export options 를 실행하여 아래 그림과 같이 설정한다.



Width 나 Height 는 하나의 bitmap 파일의 가로 세로 크기로, 크기가 커지면 총 bitmap 파일의 수는 줄어드나 낭비되는 영역이 존재할 수 있다.

5. 처음 실행했을 때 나온 main 창에서 사용하고 싶은 font 를 체크한다. 좌측 화면에서 font 를 마우스로 선택하여 선택된 font 는 image 생성이 되지 않게도 할 수 있다.



6. Option → Save bitmap font as 를 실행하여 저장하면, FNAME.fnt 파일과 FNAME_xx.tga 파일이 생성된다. 생성된 파일 중 FNAME.fnt 파일을 bmfont_init 함수로 load 하면 된다.

7. fnt → font 의 위치 정보와 출력에 필요한 기타 정보를 저장
   tga → 실제 font image