adc
advanced digital chips inc

AMAZONⅡ

- SDK Reference Manual-

Ver 1.06

February 9, 2015

Advanced Digital Chips Inc.

History

2013-07-24	Created
2014-07-04	Ver 1.01 - Correct typos.
2014-09-18	Ver 1.02 - Change ECON or ECon → E-Con
2014-10-20	Ver 1.03 - Add 8.49 loadjpg_hw, 8.50 loadbmp, 8.51 loadjpg - Add 9. MOVIE FILE PLAY
2014-11-05	Ver 1.04 - Add egl_visible_object / egl_window_delete_object function in EGL Library - Add release function for object. (window, button, checkbutton, etc …) - 3. SDK활용 내용 변경.
2014-12-11	Ver 1.05 - Add 10.15 sound_playex, 10.16 sound_current_time.
2015-02-09	Ver 1.06 - Add description about nand booting at 2. Download.

— Table of Contents —

1. SOFTWARE 개발환경 9

2. DOWNLOAD 17

3. SDK 활용 24

3.1 AMAZON II SDK 구성 24
3.2 AMAZON II Board Switch 25
3.3 Serial Flash boot 30
3.4 NAND Flash boot 25
3.5 Dram download 32

4. MEMORY 설정 33

5. UART 35

5.1 uart_config 36
5.2 uart_putch 37
5.3 uart_putdata 37
5.4 uart_putstring 38
5.5 uart_getch 38
5.6 uart_getdata 39
5.7 uart_rx_flush 39
5.8 uart_tx_flush 40
5.9 set_debug_channel 40
5.10 get_debug_channel 41
5.11 debugprintf 41
5.12 debugstring 42
5.13 PRINTLINE 42
5.14 PRINTVAR(A) 42
5.15 UART Example 43
< UART Interrupt > 44

6. INTERRUPT 45

6.1 init_interrupt 46
6.2 set_interrupt 46
6.3 enable_interrupt 47
6.4 Interrupt Example 49

7. TIMER 50

7.1 set_timer 51
7.2 stop_timer 51
7.3 delayms 52
7.4 TIMER Example 53

8. GRAPHIC 54

8.1 ge_set_texture_mem 57
8.2 get_ge_target_buffer 57
8.3 setscreen 58
8.4 get_front_buffer 59
8.5 get_back_buffer 59
8.6 get_screen_width 60
8.7 get_screen_height 60
8.8 get_screen_pitch 60
8.9 get_screen_bpp 61
8.10 flip 61
8.11 flip_wait 62
8.12 async_flip 62
8.13 async_flip_wait 62

8.14 ge_wait_cmd_complete *63*
8.15 ge_draw_rectfill *63*
8.16 ge_draw_rectfillalpha *64*
8.17 surface_set_alpha *64*
8.18 draw_line *65*
8.19 draw_rect *66*
8.20 draw_rectfill *67*
8.21 draw_roundrect *68*
8.22 draw_roundrectfill *69*
8.23 draw_circle *70*
8.24 draw_circlefill *71*
8.25 draw_ellipse *72*
6.26 draw_ellipsefill *73*
8.27 loadbmp *74*
8.28 loadbmpp *74*
8.29 loadjpg *75*
8.30 loadjpgp *75*
8.31 loadtga *76*
8.32 loadtgap *76*
8.33 loadpng *77*
8.34 loadpngp *77*
8.35 loadsurf *78*
8.36 draw_surface *78*
8.37 draw_surface_rect *79*
8.38 draw_surface_scale *80*
8.39 draw_surface_scalerect *81*
8.40 release_surface *82*
8.41 draw_set_clip_winodw *82*
8.42 osd_set_surface *83*
8.43 osd_set_surface_with_alpha *84*
8.44 osd_set_pal *85*
8.45 osd_set_off *85*
8.46 OSD_MOVE *86*
8.47 vga_mode_on *86*
8.48 vce_on_surface *87*
8.49 loadjpg_hw *88*
8.50 savebmp *88*
8.51 savejpg *89*
8.52 Graphic Example *90*

9. MOVIE FILE PLAY **92**

9.1 loadmovie *93*
9.2 release_movie *93*
9.3 movie_play *94*
9.4 movie_drawnext *95*
9.5 movie_drawnextex *96*
9.6 movie_seek *97*
9.7 movie_current_time *97*
9.8 example *98*

10. SOUND **99**

10.1 sound_init *100*
10.2 sound_loadwav *100*
10.3 sound_loadwavp *101*
10.4 sound_loadmp3 ..... *101*
10.5 sound_loadmp3p *102*
10.6 sound_release *102*
10.7 sound_play *103*
10.8 sound_stop *103*
10.9 sound_vol *104*
10.10 sound_vol_wav *104*
10.11 sound_pause *105*

10.12 sound_resume *105*
10.13 sound_isplay *106*
10.14 sound_ispause *106*
10.15 sound_playex *107*
10.16 sound_current_time *107*
10.17 Sound Example *108*

11. FILE SYSTEM **110**

11.1 f_mount *111*
11.2 f_chdrive *112*
11.3 f_chdir *113*
11.4 FILE System Example *114*

12. FONT **115**

12.1 create_bmpfont *116*
12.2 release_bmpfont *116*
12.3 bmpfont_draw *117*
12.4 bmpfont_draw_vleft *118*
12.5 bmpfont_draw_vright *119*
12.6 egl_font_set_color *119*
12.7 bmpfont_makesurface *120*
12.8 bmpfont_setkerning *120*
12.9 bmpfont_setautokerning *121*
12.10 create_bitfont *122*
12.11 release_bitfont *122*
12.12 bitfont_draw *123*
12.13 bitfont_draw_vleft *124*
12.14 bitfont_draw_vright *125*
12.15 bitfont_makesurface *125*
< font image 제작 방법 > *126*

13. EGL LIBRARY **129**

WINDOW 131
▶*egl_create_window function* *132*
▶*egl_window_show function* *133*
▶*egl_window_isshow function* *134*
▶*egl_window_invalidate function* *135*
▶*egl_window_invalidate_rect function* *136*
▶*egl_window_redraw_rect function* *137*
▶*egl_window_set_bg function* *138*
▶*egl_window_get_bg function* *139*
▶*egl_window_add_object function* *140*
▶*egl_window_set_callback function* *141*
▶*egl_window_get_active function* *142*
▶*egl_user_touch_input function* *143*
▶*egl_draw function* *144*
▶*egl_visible_object function* *145*
▶*egl_window_delete_object function* *146*
▶*egl_release_window function* *147*
▶ *Window Example.* *148*
BUTTON 150
▶*egl_create_button function* *151*
▶*egl_button_callback function* *152*
▶*egl_release_button function* *153*
▶ *Button Example.* *154*
IMAGE BUTTON 155
▶*egl_create_image_button function* *156*
▶*egl_image_button_callback function* *157*

▶*egl_release_image_button function* *158*
▶ *Image Button Example.* *159*
TOGGLE IMAGE BUTTON 160
▶*egl_create_toggle_image function* *161*
▶*egl_toggle_image_callback function* *162*
▶*egl_toggle_image_set_on function* *163*
▶*egl_release_toggle_image function* *164*
▶ *Toggle Image Button Example.* *165*
LABEL 166
▶*egl_create_label function* *167*
▶*egl_label_callback function* *168*
▶*egl_label_set_text function* *169*
▶*egl_label_set_redraw_bg function* *170*
▶*egl_label_set_color function* *171*
▶*egl_release_label function* *172*
▶ *Label Example* *173*
CHECK BUTTON / RADIO BUTTON 174
▶*egl_create_checkbutton function* *175*
▶*egl_checkbutton_callback function* *176*
▶*egl_checkbutton_set_check function* *177*
▶*egl_checkbutton_get_check function* *178*
▶*egl_checkbutton_set_style function* *179*
▶*egl_release_checkbutton function* *180*
▶ *Check / Radio button Example.* *181*
PROGRESS BAR 183
▶*egl_create_progressbar function* *184*
▶*egl_progressbar_set_barcolor function* *185*
▶*egl_progressbar_set_bgcolor function* *186*
▶*egl_progressbar_set_textcolor function* *187*
▶*egl_progressbar_set_text function* *188*
▶*egl_progressbar_set_pos function* *189*
▶*egl_progressbar_get_pos function* *190*
▶*egl_release_progressbar function* *191*
▶ *Progress Bar Example.* *192*
SCROLL BAR 193
▶*egl_create_scrollbar function* *195*
▶*egl_scrollbar_callback function* *196*
▶*egl_scroll_set_position function* *197*
▶*egl_scroll_get_position function* *198*
▶*egl_scroll_set_totalcount function* *199*
▶*egl_scroll_get_totalcount function* *200*
▶*egl_scroll_set_viewcount function* *201*
▶*egl_scroll_get_viewcount function* *202*
▶*egl_scroll_set_upcount function* *203*
▶*egl_scroll_get_upcount function* *204*
▶*egl_scroll_set_bgcolorbar function* *205*
▶*egl_scroll_set_size function* *206*
▶*egl_release_scrollbar function* *207*
▶ *Scroll Bar Example.* *208*
SLIDER 209
▶*egl_create_slider function* *211*
▶*egl_slider_callback function* *212*
▶*egl_slider_set_pos function* *213*
▶*egl_slider_get_pos function* *214*
▶*egl_slider_set_range function* *215*

▶*egl_slider_get_range function* *216*
▶*egl_slider_stepit function* *217*
▶*egl_slider_set_tick_frequency function* *218*
▶*egl_slider_set_tick_style function* *219*
▶*egl_slider_set_thumb_size function* *220*
▶*egl_slider_get_thumb_size function* *221*
▶*egl_slider_set_barcolor function* *222*
▶*egl_slider_set_tickcolor function* *223*
▶*egl_slider_set_transparent function* *224*
▶*egl_release_slider function* *225*
▶ *Slider Example.* *226*
LIST BOX 227
▶*egl_create_listbox function* *229*
▶*egl_listbox_callback function* *230*
▶*egl_listbox_additem function* *231*
▶*egl_listbox_delitem function* *232*
▶*egl_listbox_delitem_text function* *233*
▶*egl_listbox_delitem_num function* *234*
▶*egl_listbox_alldelitem function* *235*
▶*egl_listbox_ get_all_itemlist function* *236*
▶*egl_listbox_ get_sel_item function* *237*
▶*egl_listbox_ get_ multiple_sel_itemlist function* *238*
▶*egl_listbox_ set_bgcolor function* *239*
▶*egl_listbox_ set_selbgcolor function* *240*
▶*egl_listbox_ set_textcolor function* *241*
▶*egl_listbox_ set_seltextcolor function* *242*
▶*egl_listbox_ set_textalign function* *243*
▶*egl_listbox_ set_scrollwidth function* *244*
▶*egl_listbox_change_item_text function* *245*
▶*egl_listbox_change_item_num function* *246*
▶*egl_release_listbox function* *247*
▶ *List box Example.* *248*
CIRCLE GAUGE 251
▶*egl_create_circle_gauge function* *252*
▶*egl_circle_gauge_set_value function* *254*
▶*egl_circle_gauge_get_value function* *255*
▶*egl_release_circle_gauge function* *256*
▶ *Circle Gauge Example* *257*
BAR GAUGE 259
▶*egl_create_bar_gauge function* *260*
▶*egl_bar_gauge_set_value function* *262*
▶*egl_bar_gauge_get_value function* *263*
▶*egl_release_bar_gauge function* *264*
▶ *Bar Gauge Example* *265*
PICTURE 267
▶*egl_create_picture function* *268*
▶*egl_picture_callback function* *269*
▶*egl_picture_set function* *270*
▶*egl_release_picture function* *271*
▶ *Picture Example.* *272*
ANIMATION 273
▶*egl_create_animation function* *274*
▶*egl_release_animation function* *276*
▶ *Animation Example* *277*
CUSTOM OBJECT 278

▶*egl_create_custom_object function* *279*
▶ *Custom Example* *281*
MESSAGEBOX 283
▶*egl_show_messagebox function* *284*
▶ *MessageBox Example* *286*
EGL FONT 288
▶*egl_get_font function* *289*
▶*egl_set_font function* *290*
▶*egl_font_set_bkmode function* *291*
▶*egl_font_get_bk_color function* *292*
▶*egl_font_set_bk_color function* *293*
▶*egl_font_get_color function* *294*
▶*egl_font_set_color function* *295*
▶*create_bitfont function* *296*
▶*release_bitfont function* *297*
▶*create_bmpfont function* *298*
▶*bmfont_release function* *299*
▶*draw_text function* *300*
▶*draw_text_pivot function* *301*
▶*draw_text_len function* *302*
▶*draw_text_in_box function* *303*
▶*text_width function* *305*
EGL PRIMITIVES 306
▶*draw_line function* *308*
▶*draw_hline function* *309*
▶*draw_vline function* *310*
▶*draw_thickline function* *311*
▶*draw_rect function* *312*
▶*draw_rectfill function* *313*
▶*draw_rectfill_gradient function* *314*
▶*draw_rectfill_h_gradient function* *315*
▶*draw_rectfill_v_gradient function* *316*
▶*draw_roundrect function* *317*
▶*draw_roundrectfill function* *318*
▶*draw_arc function* *319*
▶*draw_pie function* *320*
▶*draw_piefill function* *321*
▶*draw_ellipse function* *322*
▶*draw_ellipsefill function* *323*
▶*draw_circle function* *324*
▶*draw_circlefill function* *325*
▶*draw_bezier function* *326*
▶*draw_polyline function* *328*
▶*draw_polygon function* *329*
▶*draw_polygonfill function* *330*
EGL ETC. 331
▶*egl_init function* *332*
▶*egl_show_object function* *333*
▶*egl_object_set_redraw function* *334*

14. DEBUGGING 335

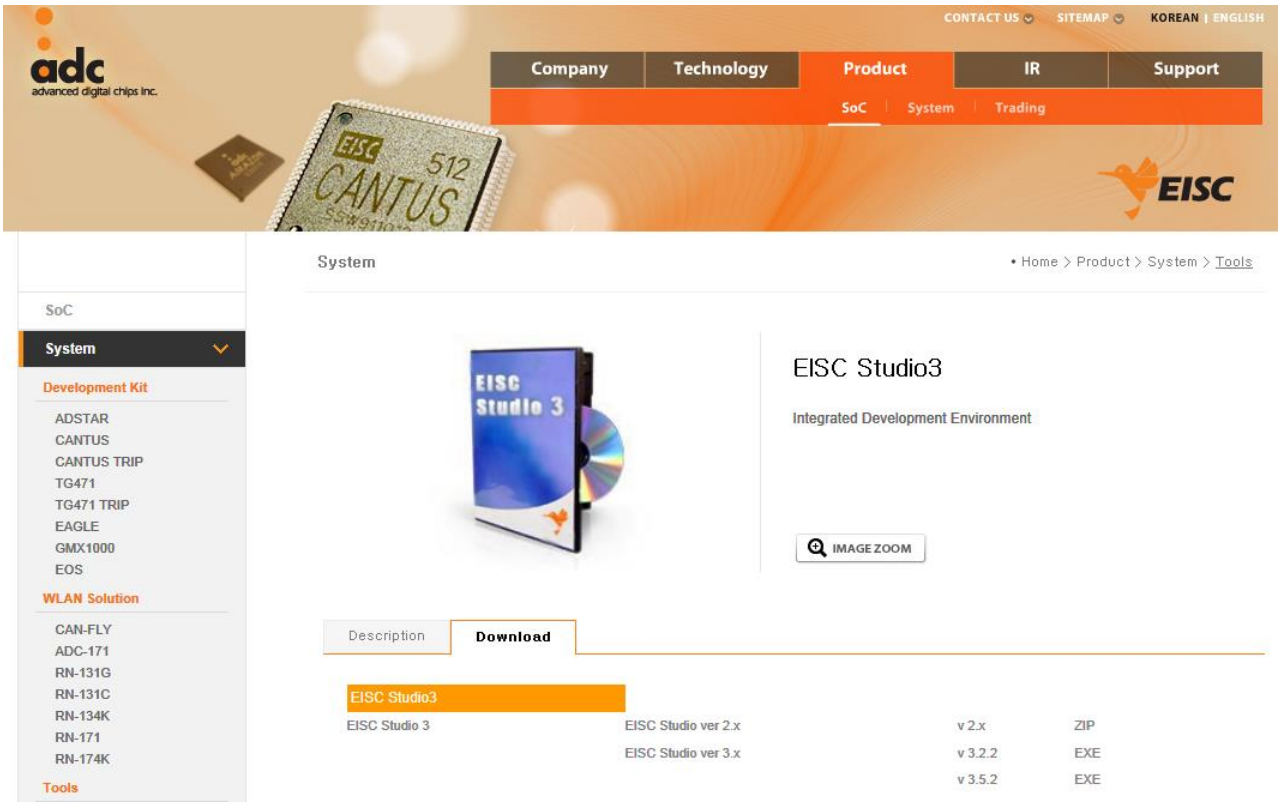
14.1 Debugging 준비 *335*
14.2 Debugging *336*

1. Software 개발환경

첫 번째 장으로 개발환경을 구축하는 방법부터 알아보도록 하겠다.

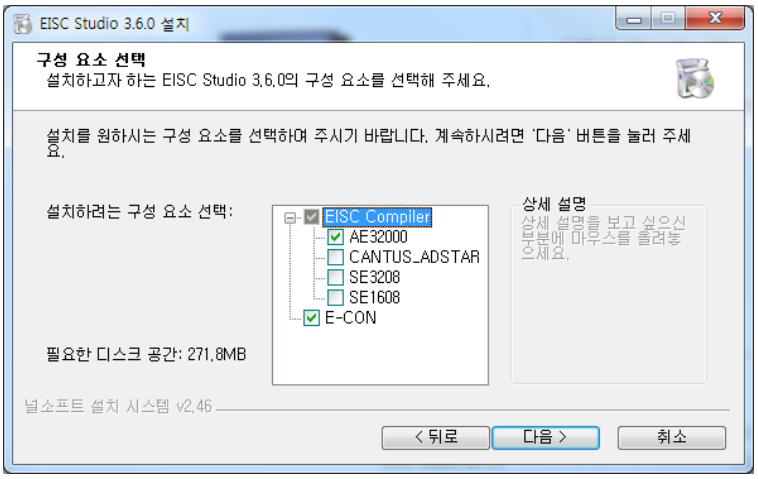
에이디칩스에서는 프로그램 코드 편집, compile, download, debugging 을 하나의 프로그램에서 수행할 수 있는 통합개발환경(IDE) EISC Studio3 를 제공하기 때문에 자료실에서 EISC Studio3 설치 파일을 내려 받아 설치만 해주면 간단하게 개발환경 구축을 마칠 수 있다. 참고로 EISC Studio3 는 Windows OS 만 지원하면 XP 이상에서 동작한다.

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

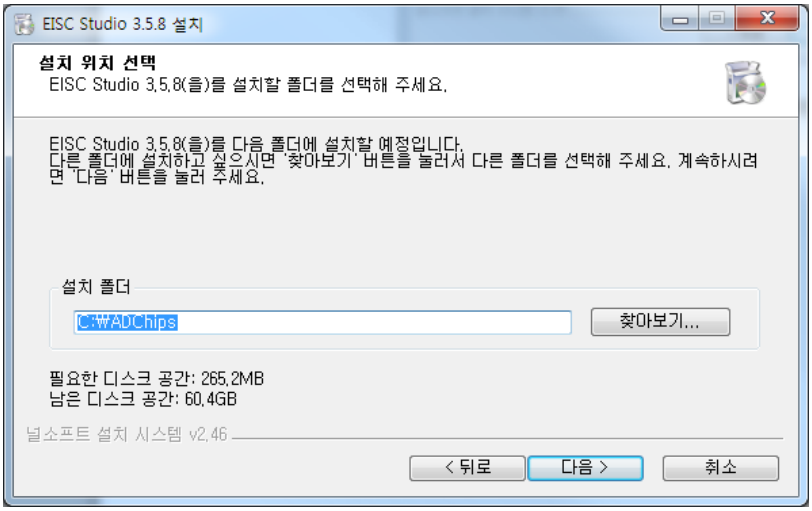


2. 내려 받은 설치파일을 실행하여 설치를 시작하면, 설치를 시작한다는 창이 뜨는데 '다음'을 클릭하여 설치를 시작한다. '다음'을 클릭하면 아래와 같이 설치할 Compiler 와 개발장비인 E-Con 의 Driver 설치를 선택하는 창이 뜬다. EISC Compiler 중 AE32000 compiler 를 선택하면 된다. 그리고 E-Con[1]장비를 처음 사용하는 사용자라면 E-Con 을 선택하여 E-Con Driver 도 설치한다. (Window 8 이상 version 을 사용하는 경우, E-Con 을 선택하지 말고 pc-util 폴더의 Window8_64bit_device_driver.pdf 파일을 참고하여 driver 를 설치해야 한다.)

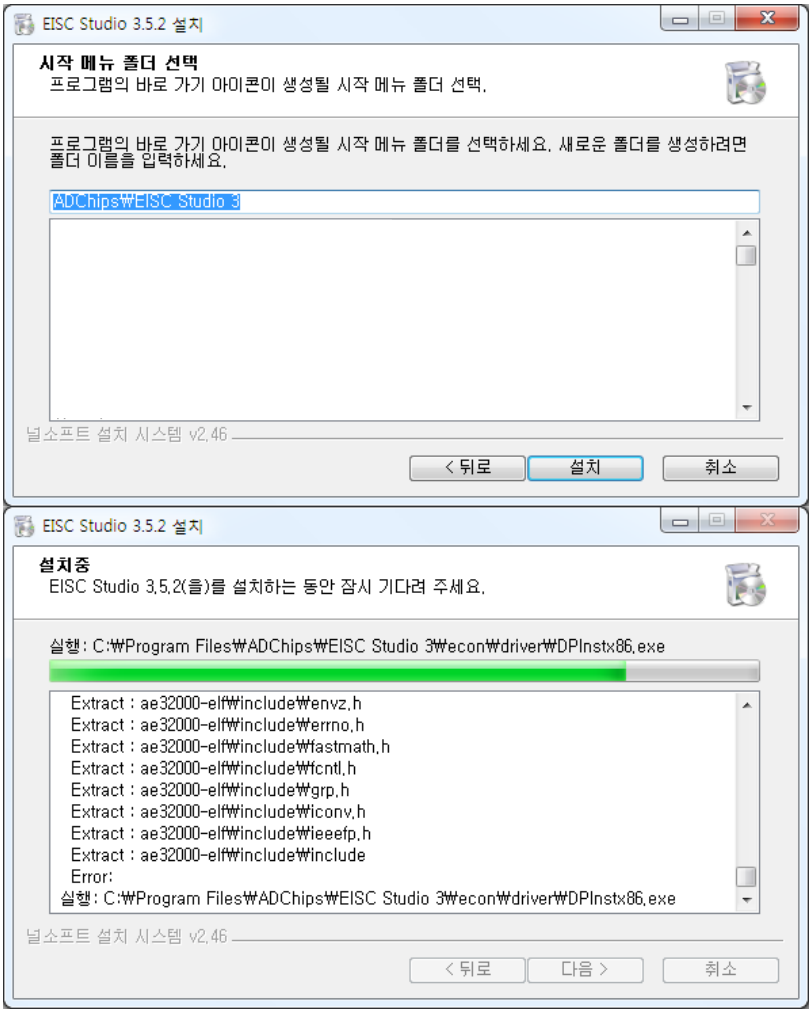
[1] adStar와연결하여program download및 debugging을수행할때필요한장비.



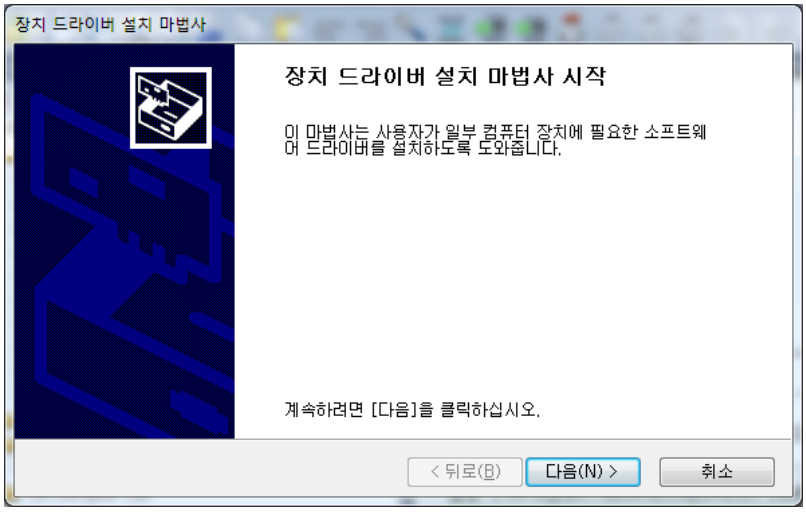
3. 구성 요소 선택을 완료하고 다음을 클릭하면, 설치할 폴더를 선택하는 창이 나온다. 특별한 경우가 아니라면 default 로 되어있는 경로에 설치하면 된다.
   (주의사항으로는 경로상에 괄호 ( ) 가 들어가면 안된다.)

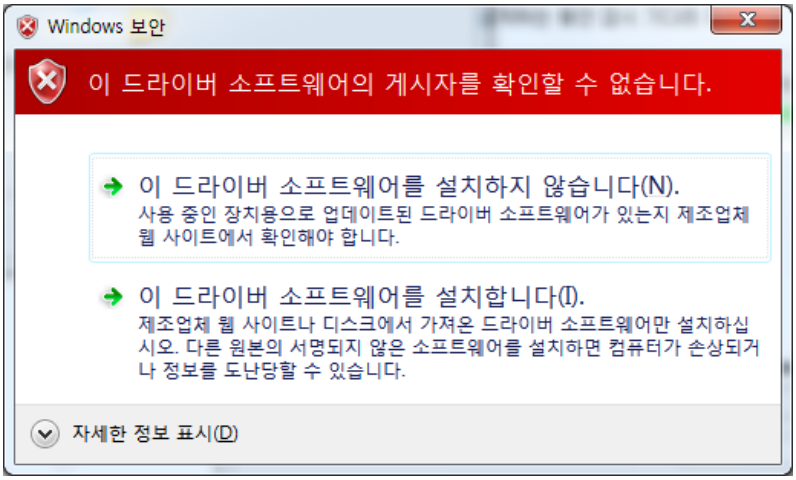


4. 다음을 클릭하면 아래와 같은 시작 메뉴 폴더를 선택하는 창이 뜨고, 설치를 누르면 설치가 시작된다.

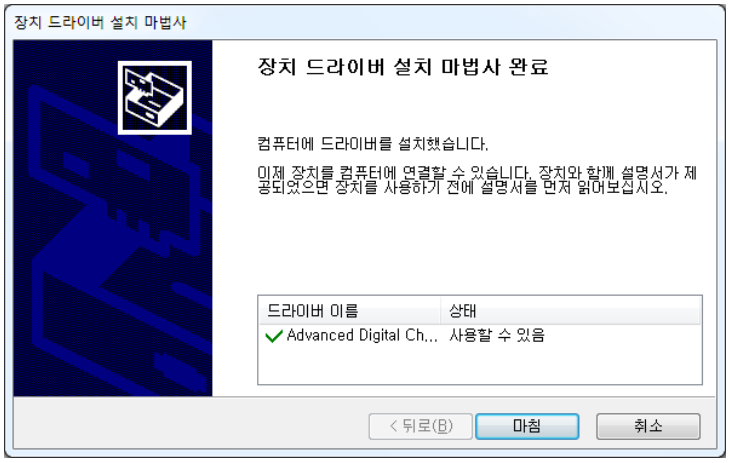


5. 구성요소 선택에서 E-Con 을 체크하였으면 설치가 끝날 무렵 E-Con Driver 설치를 위한 다음과 같은 창이 뜨고, 다음을 클릭하면 Driver 설치를 시작한다. 드라이버 설치 중 Windows 보안 경고 창이 뜨면 "이 드라이버 소프트웨어를 설치합니다."를 선택하도록 한다.
(컴퓨터와 E-Con 이 연결되어 있을 경우 설치에 문제가 생길 수도 있으므로, 컴퓨터와 E-Con 이 연결되지 않은 상태에서 설치하기 바란다.)

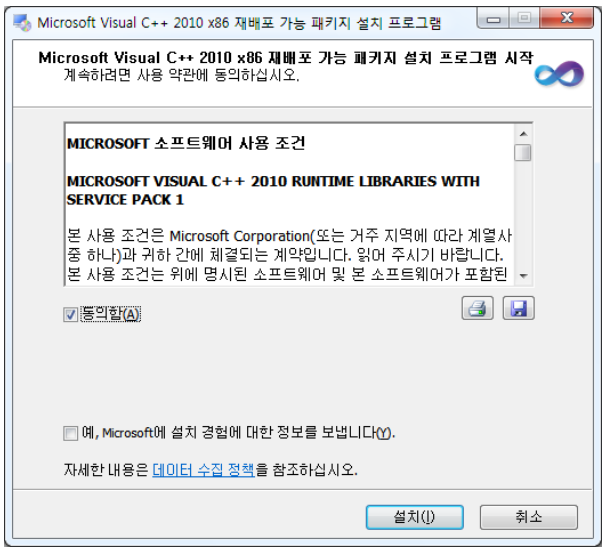




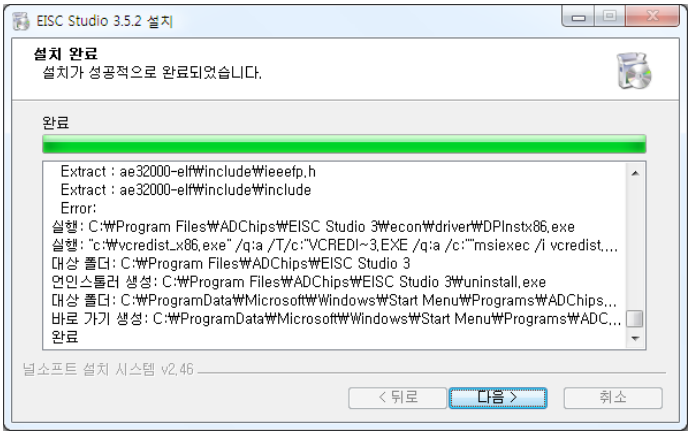
6. 장치 드라이버 설치가 완료되면 다음과 같은 창이 나오고, 마침을 눌러 EISC Studio3 의 남은 설치과정을 계속 진행한다.

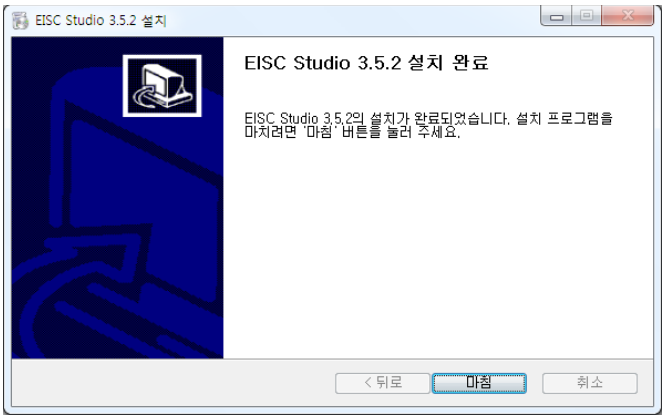


7. 다음과 같이 Microsoft Visual C++ 재배포 가능 패키지를 설치하라는 창이 뜨면 동의함에 체크한 후 설치하면 된다. EISC Studio3 를 실행하는데 필요하므로 설치하고 넘어가도록 한다.



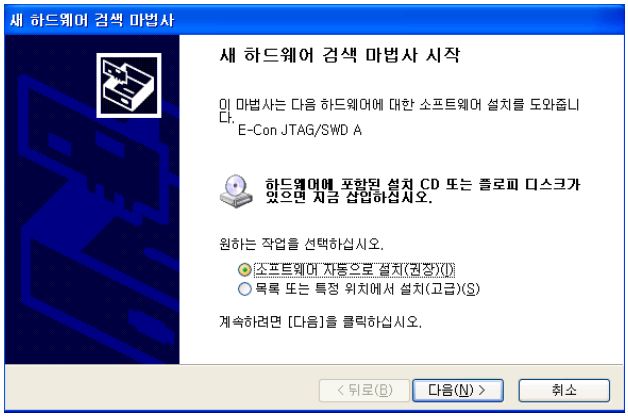
8. 설치가 완료되면 다음과 같은 창이 출력되고, 다음을 누르면 설치가 완료된다.



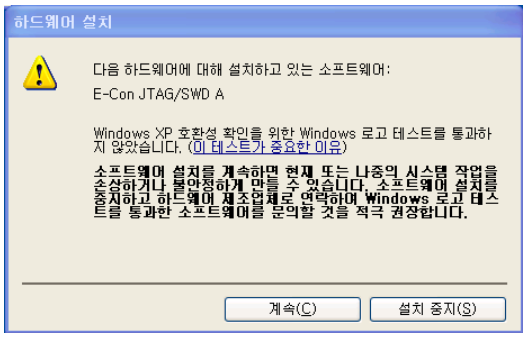


9. EISC Studio 3 설치가 완료 되면 다음으로 Download 장비인 E-Con driver 를 설치하여야 한다. EISC Studio 3 를 설치하면서 E-Con driver 파일을 복사하였다면 E-Con 을 컴퓨터와 연결하여 driver 설치를 진행하면 되고, E-Con driver 파일을 복사 하지 않았을 경우에는 EISC Studio 3 를 설치한 폴더에 econ₩driver 라는 폴더 안에 DPInstx86.exe(32bit) 또는 DPInstx64(64bit) 를 실행하여 드라이버 파일을 복사 하고, 설치를 진행하면 된다. 만약 Window 8 이상의 버전을 사용한다면 pc-util 의 Windows8_64bit_device_driver.pdf 파일의 내용대로 진행 한 후 DPInstx64.exe 를 실행해서 설치를 진행하면 된다.
(DPInstx86.exe 또는 DPInstx64.exe 를 실행해서 설치 할 경우에도, 컴퓨터와 E-Con 을 연결하지 않은 상태에서 진행하도록 한다.)

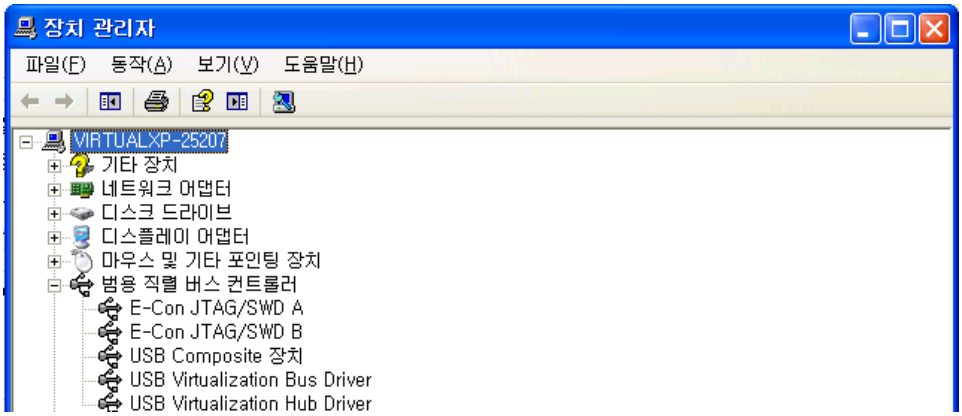
10. E-Con driver 파일 복사가 되었다면, 컴퓨터와 E-Con 을 연결하였을 때 Windows7 의 경우에는 자동으로 드라이버를 찾아 설치를 진행하게 된다. Windows XP 이하 버전에서는 다음과 같이 새 하드웨어 검색 마법사 창이 뜨게 되는데, "소프트웨어 자동으로 설치"를 체크하고 확인을 클릭하면 driver 설치가 진행된다.



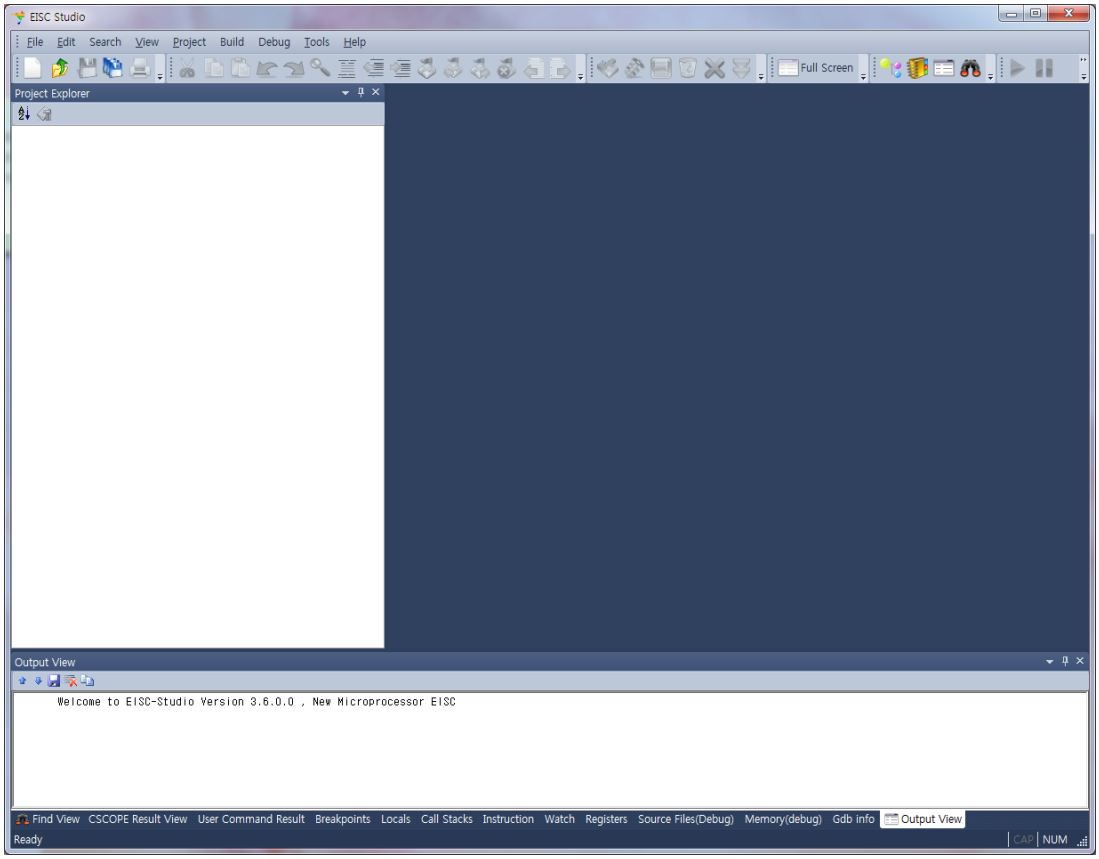
설치 중 다음과 같은 경고 창이 뜨게 되는데, 계속을 클릭하면 된다.



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.
(Windows 8 을 사용하는 경우에는 pc-util 폴더의 Window8_64bit_device_driver.pdf 파일을 참고하기 바란다.)



12. E-Con 설치까지 완료되면 개발환경 설치는 끝이난다.
시작 메뉴의 ADChips 에 보면 EISC Studio 3 가 설치된 것을 확인할 수 있다.
EISC Studio 3 를 실행하면 최근에 사용하였던 project 항목이 출력된다. 설치 후 처음 실행 한 것이라면 Recent Project 에 아무것도 나와있지 않을 것이다. 우측 상단의 x 를 클릭하여 창을 닫는다.



< EISC Studio 3 실행화면 >

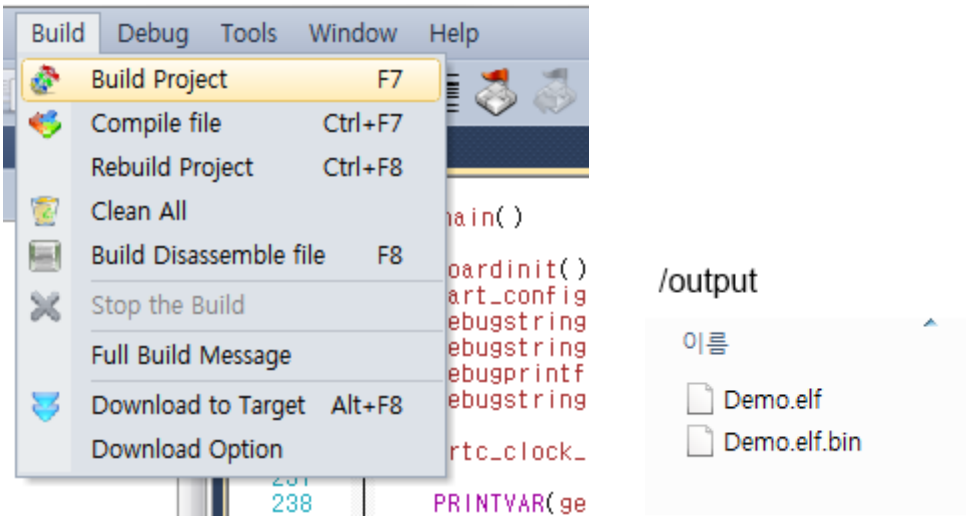
13. EISC Studio 3 는 이전에도 설명하였지만, 프로그램 내에서 편집, build (compile), download, debugging 을 수행할 수 있다.

<Project open>

Project 를 여는 방법은 File → Open Project 를 실행하여 Project 파일인 확장자가 .epx 파일을 선택하면 해당 Project 를 열어 작업을 진행할 수 있다. 또는 직접 Project 파일을 더블클릭하여 열수도 있다.

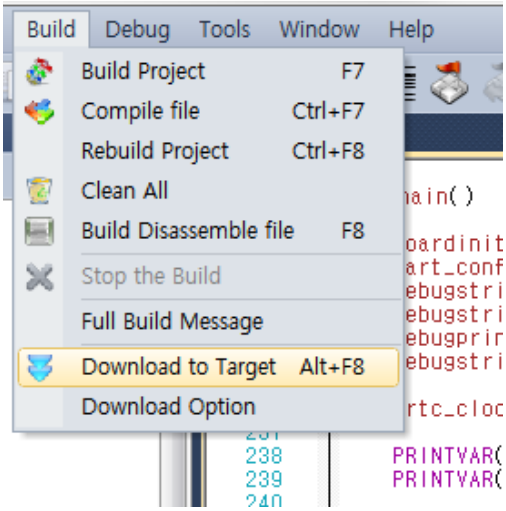
<Project build>

Project 편집이 완료되면 결과를 보기 위하여 build 를 해주어야 한다. build 는 Build → Build Porject 를 실행하면 된다. build 가 완료되면 output 폴더에 project name.elf.bin 파일이 생성된다. 이 생성된 파일을 board 에 download 하여 동작을 확인해 볼 수 있다.



<Download>

Build 과정을 통해 만들어진 bin 파일을 download 할 수 있다. Build → Download Option 과 Download to Target 을 사용하여 download 를 수행한다. 이 장에서는 Download 기능에 대한 내용을 설명한 것으로 읽고만 넘어가고, 실제 Download 방법에 대한 내용은 2.Download 와 3.SDK 활용을 참고하기 바란다.



< Debugging >

Debugging 은 Debug 메뉴를 통해 수행할 수 있다. Debug Options 로 option 을 설정할 수 있고, Start Debugger 로 debugger 를 실행할 수 있다. 자세한 내용은 마지막장의 Debugging 장을 참고하기 바란다.

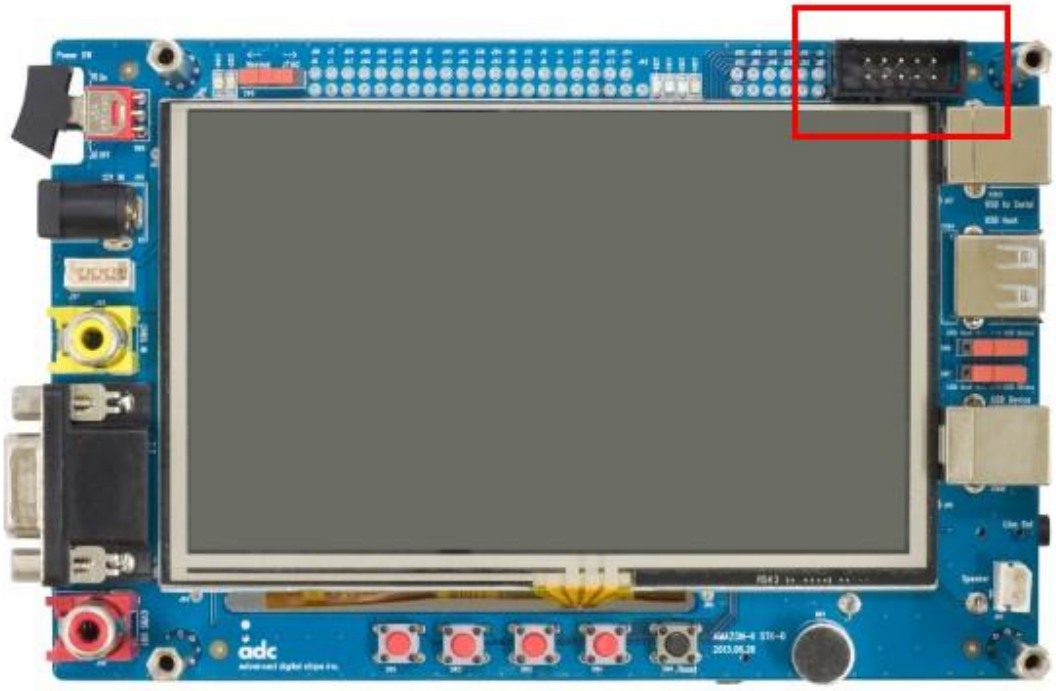
2. Download

이번 장에서는 프로그램을 Download 하는 방법에 대해 설명하겠다.

Download 방법에는 JTAG 장비인 E-Con 을 사용하는 것과 bootloader 가 실행 된 상태에서 USB 를 통한 download 방법 두 가지가 있다. 이 곳에서는 E-Con 을 사용한 download 방법에 대해 설명을 하고, USB 를 통한 download 방법은 bootloader 를 설명하는 "3. SDK 활용"의 bootloader 장에서 설명하도록 하겠다.

E-Con 을 사용하여 download 하기 위해서는 E-Con driver 설치가 필요하다. Driver 는 EISC Studio 3 설치 시에 설치하거나, E-Con driver 폴더의 파일을 사용하여 직접 설치하면 된다. (설치 방법은 "1.Software 개발환경"을 참고하기 바란다.) 그리고 E-Con driver 외에 download 를 하기 위해서는 EConMan 이라는 download program 이 필요하다. EConMan Program 은 EISC Studio 3 설치시에 같이 설치 되므로 별도로 설치할 필요는 없다. (EConMan 경로는 c:₩ADChips₩EISC Studio 3₩econ 이다.)

E-Con 은 다음 board 사진의 빨간색 박스로 표시된 부분에 연결을 하면 된다. E-Con 의 반대편은 USB 케이블을 사용하여 컴퓨터와 연결한다.



E-Con 을 사용해서 Download 하는 위치는 크게 3 가지로 나뉜다.

1. NAND Flash
2. Serial Flash
3. Ram

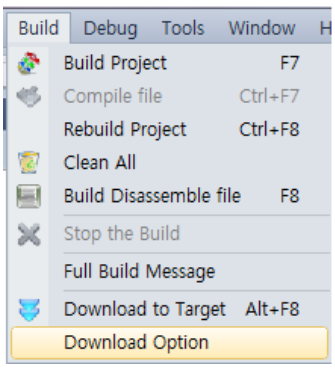
1) Nand Flash Download

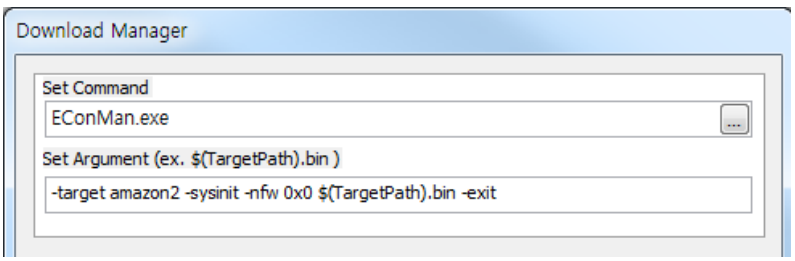
Serial Flash 없이 Nand Flash 만을 사용하여 동작시킬 경우 Nand Booting 을 사용해야 하고, E-Con 을 사용하여 NandBootCode 와 bootloader 를 다운로드 해야 한다. 참고로 Nand Booting 을 사용하기 위해서는 사용하려는 NAND Flash type 에 맞게 config pin 을 설정해주어야 한다. AMAZONII STK board 는 2G08 NAND Flash 에 맞도록 boot mode 가 고정 되어있다. AMAZONII는 config pin 을 설정하여 boot mode 를 결정하는데, config pin 에 대한 내용은 이 장의 마지막에서 설명하겠다.

NAND Booting 을 사용하기 위해서는 무조건 NandBootCode 가 필요하다. 그 이유는 NAND Flash 에서 코드를 실행할 수 없기에 NAND Booting 을 하게 되면, 제일 먼저 NAND Flash 의 0 번 block 에서 2Kbyte 의 프로그램 코드를 내부 SRam 으로 복사하여 복사한 코드를 내부 SRam 에서 동작시킨다. 복사하는 코드의 사이즈가 2Kbyte 로 제한되기때문에, NandBootCode 에서는 NAND Flash 의 1 번 block 부터 저장되어 있는 프로그램을 Ram 영역으로 복사하여 실행하도록 구성되어 있다.

NandBootCode 를 download 하는 방법은 다음과 같다.

1. 다운로드 할 프로젝트를 build 하여 binary 파일을 생성한다. (example/NandBootCode)
2. 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 amazon2 –sysinit –nfw 0x0 $(TargetPath).bin -exit" 이라고 입력하고 OK 를 클릭한다.
   Argument 를 위처럼 적어주면 현재 Project 에서 생성된 binary 파일을 Nand Flash 0 번지(0 번 block)에 download 하게 된다.





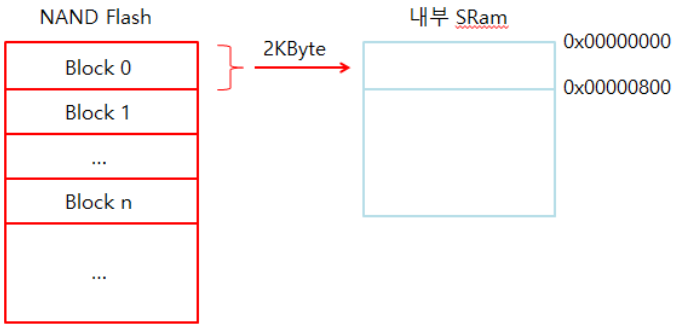
4. Build 메뉴의 Download to Target 을 실행하면 download 를 시작한다.
   Argument 에 대해 설명하면,

-target 은 말 그대로 download 할 target 을 말하며 amazon2 로 적어주면 된다.
-sysinit 는 download 를 위한 초기화를 해주는 command 이다.
-nfw 는 build 결과물로 나온 bin 파일을 NAND Flash 에 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. 다운로드 할 주소는 binary 를 다운로드 하고 싶은 주소로 변경할 수 있고 파일명도 $(TargetPath).bin 대신 직접 지정해주어도 된다.
-exit 는 download 가 완료되면 자동으로 종료하라는 command 이다.

지금까지 설명은 NAND 의 0 번 block 에 NandBootCode 를 download 할 때 사용하는 option 이다. NAND Bootloader 의 경우에는 1 번 block 에 download 를 해야 하는데, NAND 의 경우 block 단위로 write 가 이루어 지기 때문에 address 로 block 의 시작 address 를 적어주어야 한다. block 의 address 는 NAND 한 페이지의 크기에 따라 다른데, small page(512byte)의 경우 1 번 block 의 address 는 0x4000 이고, large page (2048byte)의 경우 1 번 block 의 address 는 0x20000 이다. (0 번 block 은 모두 0x0 이다.) 1 번 block 에 download 할 경우에는 NAND 의 page size 에 따라 맞는 address 에 download 해야 한다. Bootloader download 에 대한 좀더 자세한 설명은 3.3 NAND Flash boot 를 참고하기 바란다.

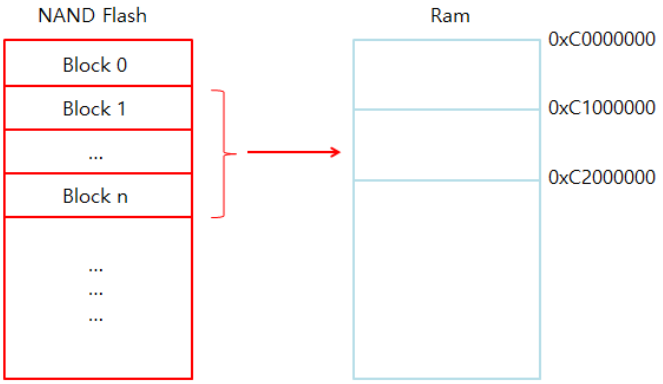
NAND Booting 에 대한 이야기가 나와 NAND Boot mode 의 booting 순서에 대해 설명을 하면 다음과 같다.

1. NAND Booting 을 시작하면 NAND Flash 의 0 block 에서 2Kbyte 의 data 를 읽어서 내부 SRam 에 복사한 후 복사된 내용을 SRam 상에서 실행한다. 그래서 NANDBootCode size 는 2Kbyte 를 넘어선 안된다.



2. 다음으로 NandBootCode 가 실행되면서, NandBootCode 에서는 NAND Flash 1 block 부터 download 된 bootloader 를 Ram 영역의 0xc1000000 번지로 복사한 후 Ram 상에서 동작을 한다. bootloader 사용 시 NAND Flash data 를 0xc1000000 번지에 복사하는 것은 bootloader 에서 application 프로그램을 download 하는 위치가 0xc0000000 번지이기 때문이다.

다음 그림은 NandBootCode 가 NAND Flash 1blockd 의 bootloader 를 복사하는 것을 그림으로 표현한 것이다.



요약하면 NAND Booting 을 하게 되면, NAND Flash 0block 의 2Kbyte 의 NandBootCode 가 내부 SRam 으로 복사되어 실행되고, NandBootCode 에 의해 1block 의 Bootloader 가 Dram 의 0xc1000000 번지로 복사되어 실행을 하게 된다.

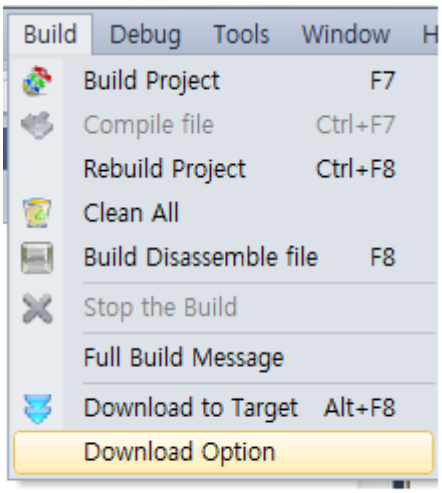
2) Serial Flash Download

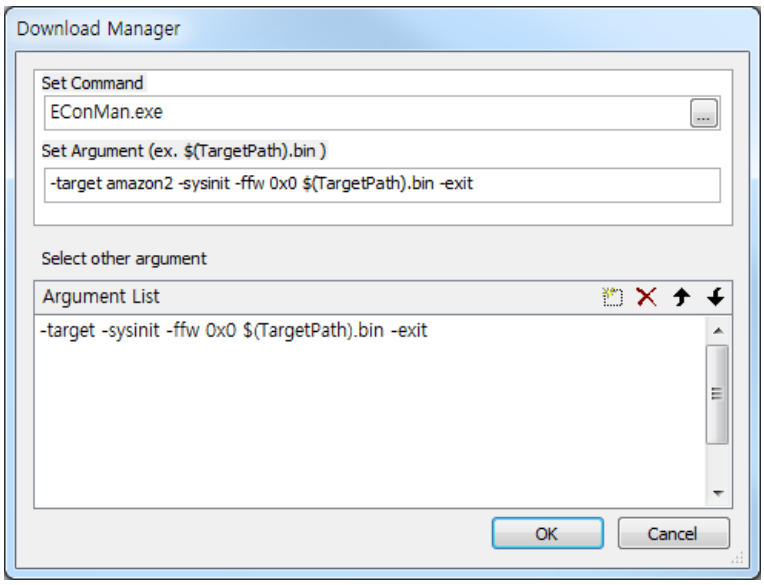
Serial Flash 를 사용하여 bootloader 또는 application 을 실행시킬 경우 해당 프로그램을 Serial Flash 에 download 하는 방법에 대해 설명한다. 참고로 AMAZONII STK board 는 NAND booting 을 사용하도록 설정이 고정되어 있어서 Serial Flash Download 를 사용할 일이 없지만, AMAZONII chip 에서는 지원이 되기에 설명하고 넘어가겠다.

(AMAZONII STK board 사용자 또는 NAND Booting 을 사용하는 개발자는 이전장인 1) NAND Flash Download 와 3.3 NAND Flash boot 를 참고 하기 바란다.)

Download 방법은 다음과 같다.

1. 다운로드 할 프로젝트를 build 하여 binary 파일을 생성한다.
2. 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 amazon2 –sysinit –ffw 0x0 $(TargetPath).bin -exit" 이라고 입력하고 OK 를 클릭한다.
   Argument 를 위처럼 적어주면 현재 Project 에서 생성된 binary 파일을 0 번지(serial flash or rom)에 download 하게 된다.
4. Build 메뉴의 Download to Target 을 실행하면 download 를 시작한다.





Argument 에 대해 설명하면,

-target 은 말 그대로 download 할 target 을 말하며 amazon2 로 적어주면 된다.

-sysinit 는 download 를 위한 초기화를 해주는 command 이다.

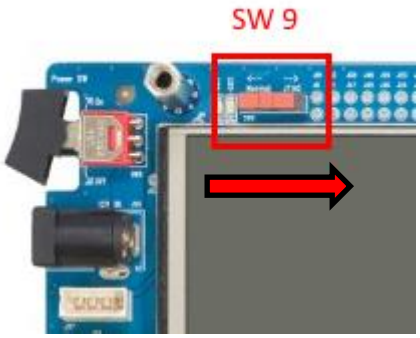
-ffw 는 build 결과물로 나온 bin 파일을 다운로드 하는 command 로 0x0 은 download 할 주소이고, $(TargetPaht).bin 은 download 할 파일명으로 $(TargetPath).bin 은 현재 열려있는 project 의 bin 파일을 뜻한다. 다운로드 할 주소는 binary 를 다운로드 하고 싶은 주소로 변경할 수 있고 파일명도 $(TargetPath).bin 대신 직접 지정해주어도 된다.

-exit 는 download 가 완료되면 자동으로 종료하라는 command 이다.

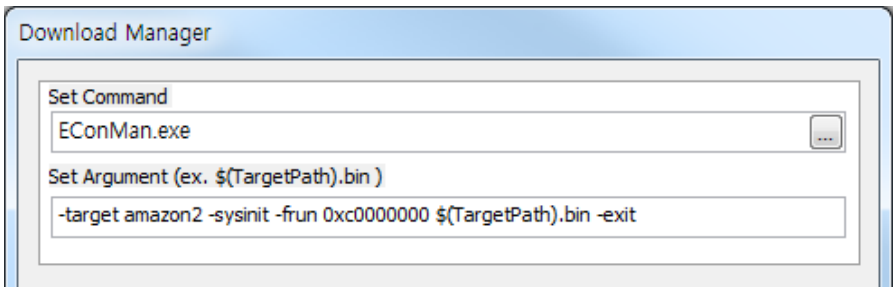
(Set Argument 에 대한 자세한 내용은 www.adc.co.kr→ Product → System → E-Con → Download → E-Con.pdf 파일을 참고하기 바란다.)

3) RAM Download

이번에는 ECon 을 사용하여 RAM 에 프로그램을 download 하여 실행하는 방법에 대해 설명하도록 하겠다. 이 방법을 사용하면 Bootloader 없이도 RAM 에서 프로그램을 바로 동작시켜 볼 수 있다. RAM 에 프로그램을 download 하여 실행할 경우에는 SW9 을 JTAG 방향으로 하여 JTAG DEBUG mode 로 전원을 켠다. 그리고 download 할 프로그램의 Linker Script 가 amazon2_ram.ld 파일로 되어 있는지 확인한다. RAM 에서 동작하는 프로그램은 amazon2_ram.ld 파일을 사용해야 한다. 그런 후에 다음 순서대로 진행을 하면 된다.



1. 다운로드 할 프로젝트를 build 하여 binary 파일을 생성한다. (Linker Script 파일 확인 필요)
2. board 와 E-Con 장비를 연결한 후 SW9 를 JTAG 방향에 놓고 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 amazon2 –sysinit –frun 0xc0000000 $(TargetPath).bin -exit" 이라고 입력하고 OK 를 클릭한다.
   Argument 를 위처럼 적어주면 현재 Project 에서 생성된 binary 파일을 0xc0000000 번지 (RAM 영역)에 download 한 후 실행을 한다.



4. Build 메뉴의 Download to Target 을 실행하면 download 를 시작한다. Download 가 완료되면 downlod 한 프로그램이 DRam 에서 동작하는 것을 확인할 수 있다.

   Argument 에 대해 설명하면,

   -target 은 말 그대로 download 할 target 을 말하며 amazon2 로 적어주면 된다.

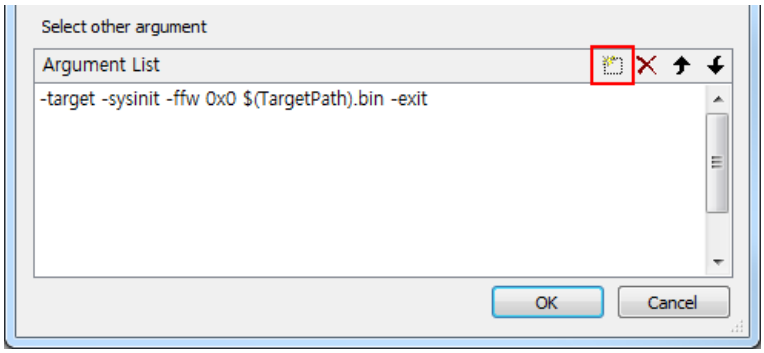
   -sysinit 는 download 를 위한 초기화를 해주는 command 이다.

   -frun 은 build 결과물로 나온 bin 파일을 다운로드 한 후 실행하는 command 로 Dram 에 download 후 실행할 때 사용한다. 0xC0000000 은 download 할 주소로 Dram 의 시작 번지이고, $(TargetPaht).bin 은 download 할 파일명으로 $(TargetPath).bin 은 현재 열려있는 project 의 bin 파일을 뜻한다. 다운로드 할 주소는 binary 를 다운로드 하고 싶은 주소로 변경할 수 있으나 SDK 의 example 은 0xC0000000 에서 실행되도록 되어 있으므로 그대로 사용하기를 권장한다. 파일명은 $(TargetPath).bin 대신 직접 지정해주어도 된다.

-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.)

Ram download 의 경우에는 바로 Ram 영역에 download 하여 동작을 확인해 보는 것으로 Reset 시 프로그램을 다시 download 해주어야 한다.

4) Select other argument



아래의 Select other argument 라고 되어 있는 부분은 argument 를 변경해야 할 경우 직접 변경하여도 되지만 자주 사용하는 argument 를 미리 저장 해 놓고 간단히 클릭만으로 argument 를 변경할 수 있도록 도와준다. 빨간 박스로 된 부분을 클릭하면 argument 를 추가할 수 있다.

◈ Config pin 과 Boot mode.

AMAZONII의 Boot mode 는 config pin 0~3 에 의해 결정된다.

Config pin 0 은 JTAG mode / Nomal mode 를 결정하고, Config pin 1,2,3 에 의해 어떤 메모리를 사용하여 booting 할지를 결정한다.

Config pin 0 가 high 일 경우에는 Nomal mode 로 일반적으로 동작하는 모드이다. Config pin 0 가 low 일 경우에는 JTAG mode 라고 해서, chip 이 프로그램 시작번지에 정지한 상태를 유지하게 된다. Debugging 을 하거나 안정적인 download 를 하기 위해 이 모드를 사용한다. AMAZONII STK board 에서는 SW 9 번으로 JTAG mode 와 Nomal mode 를 결정할 수 있다.

Config pin 1, 2, 3 은 booting 시 어떤 메모리를 사용하는지 결정하는 것으로 다음의 표를 참고하기 바란다.

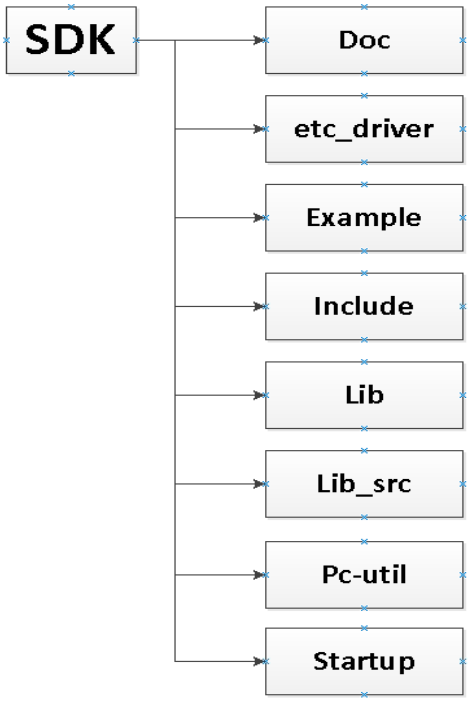
boot Mode	Config 1	Config 2	Config 3
NAND Small 3cycle	Low	Low	Low
NAND Small 4cycle	High	Low	Low
NAND Large 4cycle	Low	High	Low
NAND Large 5cycle	High	High	Low
NAND MLC 4bit	Low	Low	High
NAND MLC 24bit	High	Low	High
Serial Flash	Low	High	High

3. SDK 활용

개발에 편의를 위해 adchips 에서는 AMAZONII SDK 를 제공하고 있다. SDK 는 achips 홈페이지(www.adc.co.kr)의 Product → SoC → AMAZONII → Tools & Software 에서 내려받을 수 있다.

3.1 AMAZONII SDK 구성

AMAZONII SDK 는 다음과 같이 구성되어 있다.



Doc → 현재 보고 있는 매뉴얼을 포함, AMAZONII와 관련된 문서가 들어있는 폴더.

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

Include → AMAZONII SDK 의 header file 폴더.

lib → AMAZONII SDK 의 library file 폴더.

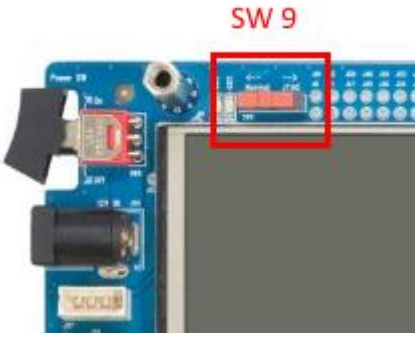
lib_src→ AMAZONII SDK ㅈ의 library source file 폴더.

Pc-util→ AMAZONII를 사용하는데 있어 필요한 Utility 폴더. USB Driver 가 들어있다.

Startup → AMAZONII개발에 필요한 startup 코드와 link script 코드가 들어있는 폴더. 이 외에 board 기본 설정 코드도 들어있다.

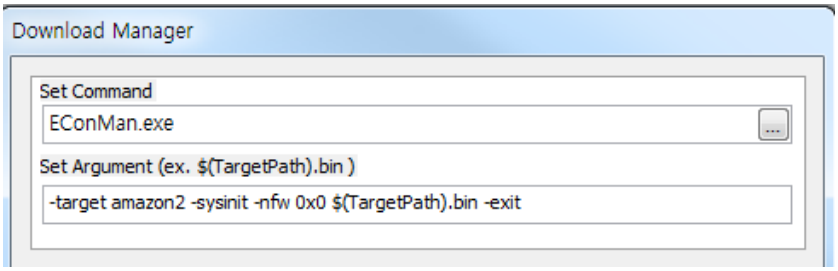
3.2 AMAZONII Board Switch

AMAZONII board 에서 SW 9 로 JTAG Debug mode 를 설정한다. JTAG Debug 모드는 program 이 실행되지 않고 시작지점에서 멈춰있는 모드로 debugging 시에 사용되고, 다른 동작에 영향을 받지 않고 download 해야 할 경우 사용한다. E-Con 으로 download 가 되지 않을 경우에는 JTAG Debug 모드에서 download 해보기 바란다.



3.3 NAND Flash boot

AMAZONII STK board 는 boot mode 가 NAND Booting 으로 고정되어 있기 때문에 NAND Boot Code 와 편의에 따라 Bootloader 를 사용해야 한다. example 에 보면 NandBootCode(example/NandBootCode)가 있다. Project 를 open 하여 build 한 후 Download Option 을 실행하여 다음과 같이 적은 후 Download to Target 을 실행하면 E-Con 으로 NAND Flash 에 NandBootCode 가 download 된다.



Argument 에 대해 알아보면,

-target amazon2

→ Serial Flash 에 download 할 때와 마찬가지로 target 을 지정하는 것이다.

-sysinit

→ Serial Flash 에 download 할 때와 마찬가지로 초기화를 진행하는 command 이다.

-nfw 0x0 $(TargetPath).bin

→ NAND Flash 에 download 하는 명령어로 0x0 번지 (0 번 블록)에 파일을 download 한다.

-exit

→ Download 가 정상적으로 완료되었으면 program 을 종료한다.

Download 후에 Board 의 전원을 켜면 NandBootCode 가 실행되면서 UART ch0 로 다음과 같이 Nand Boot Code 메시지가 출력된다.

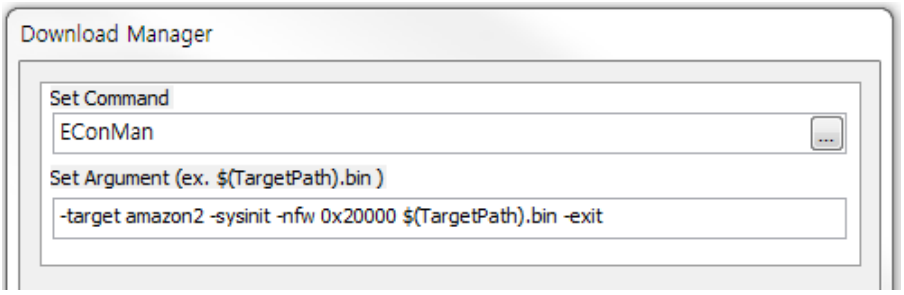
```
Nand Boot Code
```

하지만 NandBootCode 는 NAND Flash 의 1 번 블록에 저장되어 있는 binary data 를 읽어와 Dram 에 복사한 후 실행하는 동작을 하기 때문에 NAND Flash 의 1 번 블록에 아무것도 download 하지 않은 상태에서는 다른 동작을 하지 않는다.

(Bootloader 와 application 동작시에도 UART ch 0 로 메시지가 출력되므로 프로그램의 동작 여부를 UART ch 0 를 통해 확인할 수 있다.)

따라서 지금부터는 NAND Flash 의 1 번 블록에 프로그램을 download 하는 방법에 대해 설명하도록 하겠다. NAND Flash 의 1 번 블록에 download 할 project(bootloader)를 open 한 다음 Build Project 를 수행하여 binary 파일을 생성한다. 그런 다음 Download Option 을 실행하여 다음과 같이 작성한다.

(NAND Flash 1block 에는 bootloader 또는 application 프로그램을 download 하여 실행해볼 수 있다.)



이전 0 번 블록에 download 할 때와 다른 것은 0x0 이 아닌 1 번 블록의 0x20000 번지에 download 하는 것이다.

NAND Flash 는 page size 에 따라 1 번 블록의 주소가 달라지는데, small page 의 경우 0x4000 번지이고, 개발보드에 붙어 있는 large page NAND Flash 의 경우 0x20000 번지이다. 1Gbit(1G08)이상의 NAND Flash 는 large page 이다. 위처럼 Download Option 을 지정한 후 Download to Target 을 수행하면 E-Con 을 통해 NAND Flash 1 번 블록에 program 이 download 되고, download 완료 후 NAND Flash boot mode 로 부팅을 하면 download 한 program 이 실행되는 것을 확인할 수 있다.

NAND Flash Boot 모드에서도 Serial Flash boot 모드와 같이 bootloader 를 사용할지 바로 application 을 실행할지를 NAND Boot Code 에서 결정하여 사용할 수 있다.

우선 NAND Boot Code 에서 main.c 의 상단에 보면 다음과 같이 APP load address 를 정의한 부분이 있다.

main.c

```
1   #include "sdk.h"
2
3
4
5   #define APP_LOAD_ADDRESS 0xc0000000
6   //#define APP_LOAD_ADDRESS 0xc1000000
7   #define COPY_BLOCK_SIZE 4
8
9   #define UART_BAUD 115200
10
```

바로 application 을 실행시킬 경우에는 0xc0000000 으로 APP_LOAD_ADDRESS 를 사용하면 되고, Bootloader 를 실행시킬 경우에는 0xc1000000 으로 APP_LOAD_ADDRESS 를 사용하면된다. APP_LOAD_ADDRESS 를 결정 한 후 build 하여 이전과 마찬가지로 0 번지에 download 한다.

다음으로 Application program 의 경우에는 상관 없지만, Bootloader 를 사용할 경우에는 Bootloader 를 열어서, Linker Script 파일을 확인해야 한다. Linker Script 파일로 amazon2-bootloader_ram.ld 를 사용해야 하는데, amazon2-bootloader.ld 파일로 되어 있다면 변경한다. 그런 후에 build 하여 NAND Flash 의 0x20000 번지에 download 하면 된다. (application 의 경우에는 Linker Script 가 amazon2_ram.ld 로 되어 있는지 확인한 후 download 한다.)

이렇게 download 를 완료하고 전원을 켜면, NAND Flash Boot 모드에서 bootloader 또는 application 이 실행되는 것을 확인할 수 있다.

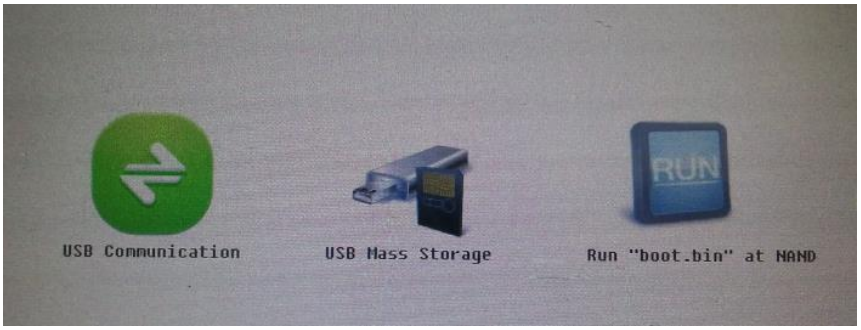
정리하면, NAND Booting 에서 bootloader 를 사용할 경우

1. NandBootCode 의 APP_LOAD_ADDRESS 를 0xc1000000 을 사용
2. Bootloader 의 Linker Script 를 amazon2-bootloader_ram.ld 를 사용
3. NandBootCode 를 0block 에 Bootloader 를 1block 에 download 하면 된다.

NAND Booting 에서 application 을 실행할 경우

1. NandBootCode 의 APP_LOAD_ADDRESS 를 0xc0000000 을 사용
2. Application 의 Linker Script 를 amazon2_ram.ld 를 사용.
3. NandBootCode 를 0block 에 Bootloader 를 1block 에 download 하면 된다.

Bootloader 를 1block 에 download 하고 실행하면 다음 그림과 같이 LCD 에 출력된다.

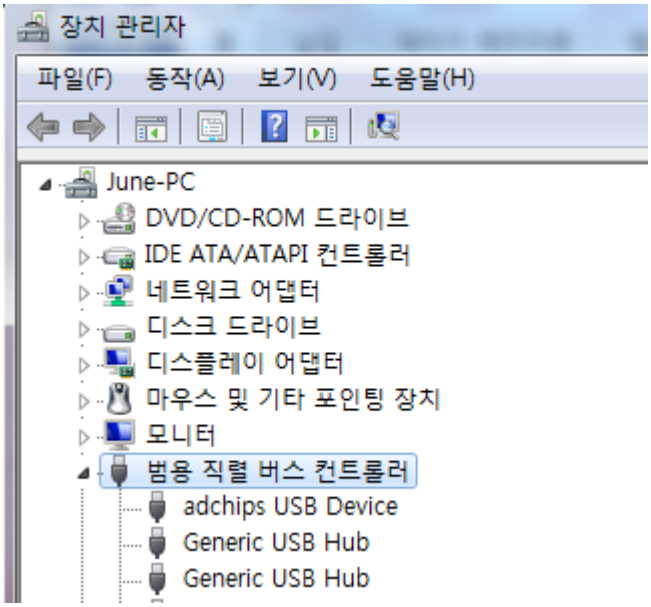


각 아이콘은 각각의 기능을 실행하는데, USB Communication 은 USB device 를 사용하여 application program 을 Dram 에 download 하여 바로 실행해 볼 수 있는 mode 이다. USB Mass Storage 는 개발보드의 NAND Flash 를 컴퓨터상에 USB memory 같이 이동식 디스크로 인식시켜 application 에서 사용할 image 와 sound file 을 복사할 때 사용한다. 마지막으로 Run "boot.bin" at NAND 는 NAND Flash 에 있는 boot.bin 파일의 내용을 Dram 에 복사한 후 실행하는 모드 이다.

3.3.1 USB Communication

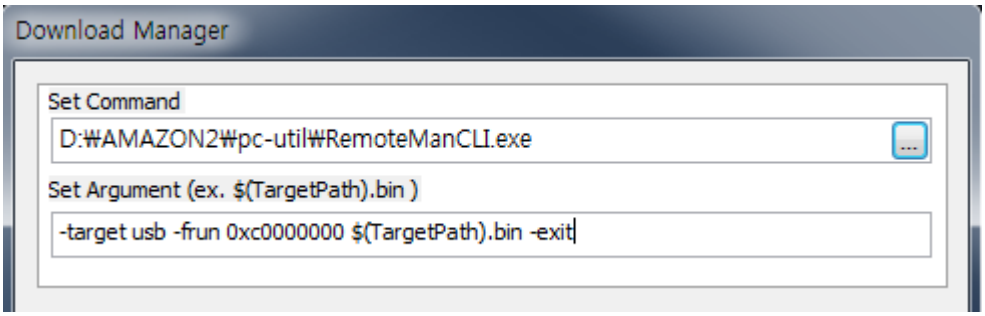
USB Communication 을 실행하고, USB Device connector 에 USB 케이블을 컴퓨터와 연결하면 USB 장치가 인식이 된다. 최초 시도시에는 USB driver 를 설치해주어야 한다. USB driver 는 pc-util/usb_driver 폴더 안에 ADChips USB DriverInstaller.exe 파일을 실행하여 설치할 수 있다.

1. Usb_driver 폴더의 ADChips USB Driver Installer.exe 파일을 실행하여 설치를 진행한다.
2. 설치가 정상적으로 완료되면, USB 를 연결하고, bootloader 의 USB Communication 을 실행한다.
3. 새로운 장치를 찾았다고 화면의 우측 하단에 표시되면서, driver 설치를 시작하게 된다. Windows7 이상 사용자라면 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. USB driver 설치가 완료되면, 장치관리자에서 adchips USB device 가 설치된 것을 확인할 수 있다.



USB Driver 설치를 완료 했으면 USB Communication 에서 example 폴더에 있는 application 프로그램을 Dram 에 download 하여 바로 동작을 확인해 볼 수 있다. example 의 Demo project 를 open 한다. DRam 에서 program 을 동작시키려는 것이기에 Linker Script 파일이 amazon2_ram.ld 파일을 사용하고 있는지 확인한다. (amazon2.ld 파일과 amazon2_ram.ld 파일을 startup 폴더에 있다. 필요에 따라 변경하여 사용하면 된다.)

amazon2_ram.ld 파일로 되어 있으면 Build→Build Project 를 실행하여 binary 파일을 생성 한 후 Build → Download Option 을 실행한다. 그런 후에 다음과 같이 적어주고, Download to Target 을 실행하면 USB 를 통해 program 이 Dram 에 download 되고 실행된다.



USB 로 download 하기 위해서는 EConMan program 이 아닌 RemoteManCLI 라는 프로그램을 사용해야 하는데 pc-util 폴더에 있다. Argument 는 EConMan 과 유사한데, 설명하면 다음과 같다.

-target usb

→ target 을 명시하는 부분으로 usb 로 지정한다.

-frun 0xc0000000 $(TargetPath).bin

→ binary 파일을 download 하고 실행하는 명령어로, Dram 영역인 0xc0000000 에 현재 project 에서 생성된 binary 파일을 download 한 후 실행한다.

-exit

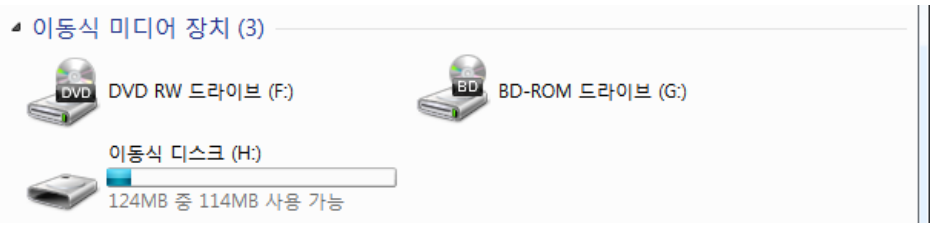
→ 모든 동작이 정상적으로 이루어 졌으면 program 을 종료한다.

위처럼 download option 을 설정 한 후에, USB communication 상태에서 Build → Download to Target 을 실행하면 현재 열려져 있는 project 의 binary 파일이 download 가 되고 완료되면 실행된다.

USB Communication 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.
(program 이 정상적으로 실행되지 않는 경우 NAND Flash 에 image 및 sound file 이 없어서 그럴 수 있으므로, 다음에 설명할 USB Mass Storage 에서 Flahs 에 file 을 복사한 후 실행해보기 바란다.)

3.3.2 USB Mass Storage

USB Mass Storage 는 example 의 usb_mass_storage 와 같은 동작을 수행하는 모드로 AMAZONII board 의 Nand Flash 를 usb memory 와 같이 컴퓨터에서 저장 장치로 인식되도록 해준다. USB Mass Storage 가 실행되면 내컴퓨터에 다음 그림과 같이 이동식 디스크 라는 저장장치가 생성된다. 이렇게 생성된 저장 장치에 application 에서 사용할 파일을 복사하면 된다. example 에서 사용하는 파일을 example 폴더의 flash_data 라는 곳에 있다. example 의 예제를 동작시키기 전에 flash_data 폴더의 파일을 미리 복사해놓아야 한다. 기본적으로 개발보드에는 NAND Flash 에 example 에서 사용하는 파일이 복사되어 있다.



[이동식 디스크 저장 장치]

3.3.3 Run "boot.bin" at NAND

Run "boot.bin" at NAND 는 NAND Flash 에 boot.bin 이라는 파일을 Dram 에 복사한 후 Dram 에서 Program 을 동작시키는 모드이다. build 하여 생성된 binary 파일을 boot.bin 으로 이름을 변경한 후 USB Mass Storage 를 사용하여 NAND Flash 에 복사한 후 이 모드로 부팅하면 복사한 binary 가 실행이 된다. Dram 에서 동작하는 program 이기 때문에 build 시에 Linker Script 는 amazon2_ram.ld 파일로 되어 있어야 한다.

3.4 Serial Flash boot

Serial Flash boot 를 사용할 경우 serial falsh 에 download 하는 방법에 대해 설명한다. AMAZONII STK board 는 boot mode 가 NAND Booting 으로 고정되어 있어 사용할 수 없지만, AMAZONII chip 에서 지원되므로 참고하기 바란다.

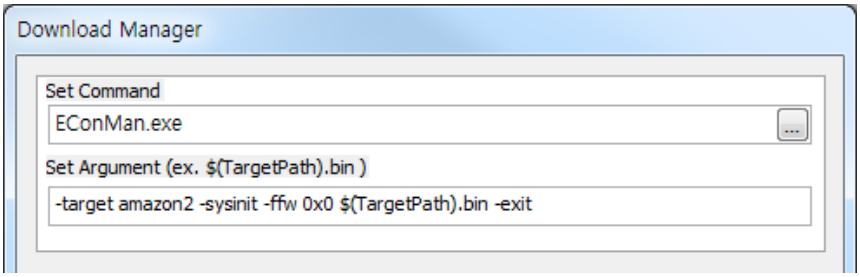
(AMAZON2 STK board 사용자는 이전 장인 3.3 Nand Flash boot 를 참고하기 바란다.)

Serial Flash boot 를 하기 위해서는 이전 장의 Config pin 설정을 참고하여 boot mode 를 맞게 설정 해주어야 한다. Serial Flash 를 사용하여 booting 을 할 경우에 Bootloader 를 사용하여 application program 을 동작시킬 수 있고, Bootloader 없이 바로 application program 을 동작 시킬 수 있다. 이는 application program 의 Linker Script 의 LD 파일을 어떤 것으로 사용하는지에 따라 달라진다.

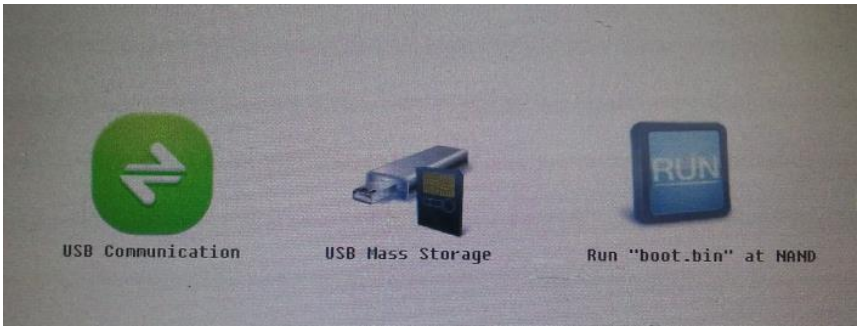
그림과 같이 2 개의 LD 파일이 있는데, amazon2.ld 파일은 Bootloader 없이 0 번지부터 application 을 동작시킬 경우 사용하는 LD 파일이고, amazon2_ram.ld 파일은 Bootloader 를 사용하여 Ram 에서 동작하는 application 을 만들 때 사용하는 LD 파일이다. SDK 의 example 은 bootloader / bootcode 를 제외하고 Ram 에서 동작하도록 amazon2_ram.ld 파일을 사용한다. 지금부터는 Serial Flash boot 모드에서 bootloader 를 사용하는 경우와 그렇지 않은 경우를 각각 설명할 것이다.

3.4.1 Bootloader 사용

Bootloader 를 사용하려면 SDK example 폴더의 bootloader 를 download 하면 된다.
example 폴더의 bootloader project 폴더로 들어가 확장자가 .epx 파일을 더블클릭하면 project 를 열 수 있다. 그런 다음 Build → Build Project 를 실행하여 project 를 build 한다. 다음으로 E-Con 을 board 에 연결한 후 board 의 전원을 켠다. 그리고 Build → Download Option 을 실행하여 다음 그림과 같이 download 관련 command 를 입력하고, Build → Download to Target 을 실행하여 Serial Flash 0 번지에 program 을 download 한다.



Program 을 정상적으로 download 한 후 Serial Flash boot 모드 (cfg 1 : L, cfg 2 : H, cfg 3 : H)로 놓은 후 전원을 켜면 다음과 같은 화면이 LCD 에 출력되는 것을 확인할 수 있다.

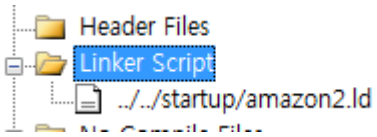


각 아이콘은 각각의 기능을 실행하는데, USB Communication 은 USB device 를 사용하여 application program 을 Dram 에 download 하여 바로 실행해 볼 수 있는 mode 이다. USB Mass Storage 는 개발보드의 NAND Flash 를 컴퓨터상에 USB memory 같이 이동식 디스크로 인식시켜 application 에서 사용할 image 와 sound file 을 복사할 때 사용한다. 마지막으로 Run "boot.bin" at NAND 는 NAND Flash 에 있는 boot.bin 파일의 내용을 Dram 에 복사한 후 실행하는 모드 이다. 각각의 기능에 대한 내용은 3.4 NAND Flash boot 장에서 설명하겠다.

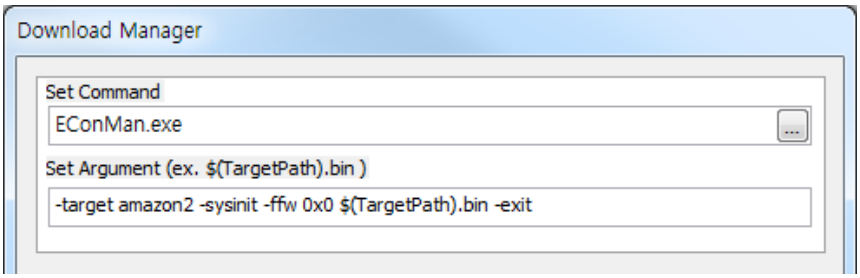
3.4.2 Bootloader 없이 Serial Flash에서 동작시키기

Bootloader 를 사용하지 않고 바로 application program 을 실행할 수 있다. Bootloader 가 있으면 USB Mass Storage 사용이 편리하므로 Bootloader 를 사용하기를 권장하지만 Bootloader 가 필요 없는 경우에는 여기서 설명하는 방식으로 바로 program 을 실행할 수 있다.

실행할 program 을 open 한다. 여기서는 example 의 Demo project 를 예로 들겠다. program 을 open 하였으면, Linker Script 파일을 amazon2.ld 파일로 변경한다. example 의 예제는 모두 amazon2_ram.ld 파일을 사용하고 있다. amazon2.ld 파일을 startup 폴더에 있다. Linker Script 에서 마우스 오른쪽키를 눌러 ld 파일을 변경할 수 있다.



Linker Scrip 파일을 변경하였으면 Build Project 를 실행하여 binary 파일을 생성한 다음, ECon 을 보드와 연결하고, 전원을 켠 후 Build → Download Option 을 실행하여 다음과 같이 적어준다.

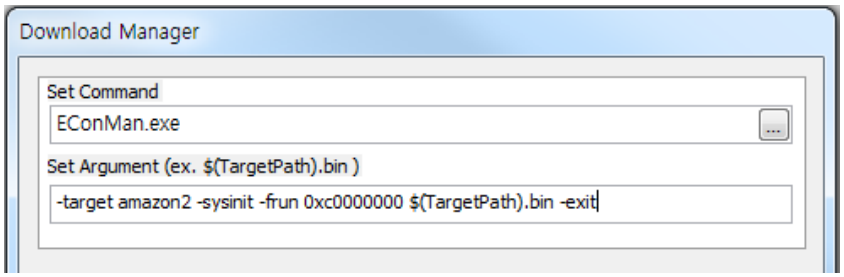


그런 다음 Build 의 Download to Target 을 실행하여 download 하고 download 가 완료 된 후 reset 을 하면 download 한 program 이 동작하는 것을 확인할 수 있다.
(Download 가 실패한 경우 JTAG Debug 모드에서 download 를 다시 해보기 바란다.)

3.5 Dram download

Bootloader 와 NAND Boot Code 없이 바로 Dram 에 program 을 download 하여 실행해 볼 수 있는 방법이 있어 이번장에서 설명하도록 하겠다. E-Con 을 사용하여 바로 Dram 에 program 을 download 하여 실행할 수 있는데 방법은 다음과 간다.

우선 E-Con 이 연결 된 상태에서 개발보드를 JTAG Debug 모드로 놓고 전원을 켠다. 그런 후에 Download 하여 실행할 application program 을 open 한다. Dram 에서 동작하는 것이므로 Linker Script 파일이 amazon2_ram.ld 로 되어 있는지 확인하고, Build Project 를 실행하여 binary 파일을 생성한다. 다음으로 Build 메뉴의 Download Option 을 실행하여 다음과 같이 적어준다.



-frun 0xc0000000 $(TargetPath).bin

→ Dram 0xc0000000 번지에 binary 파일을 download 후 실행하는 명령이다.

위처럼 Download Option 을 작성하고, Download to Target 으로 program 을 download 하면, download 가 완료된 후 바로 동작하는 것을 확인할 수 있다. 바로 Dram 에 download 하여 동작을 확인할 수 있기 때문에, 개발시에 Board 를 JTAG Debug 모드로 놓은 상태에서 위 방법으로 바로 download 하여 동작을 확인해 보기 좋다.

4. Memory 설정

AMAZON2 개발보드에는 128MByte DDR2 가 붙어 있는데, SDK 에서는 128MByte 의 RAM 영역을 다음과 같이 나누어 사용하고 있다.

0xC0000000 ~ 0xC1FFFFFF (32MByte)	text (Program code) data bss heap stack
0xC2100000 ~ 0xC2900000 (8MByte)	Frame buffer
0xC2900000 ~ 0xC7FFFFFF (87MByte)	Texture memory

0xC0000000 영역에는 data, bss, heap, stack 등 Program 동작에 필요로 하는 RAM 영역으로 사용을 하고, RAM 에서 동작하도록 Program 을 설정한 경우 Program code 도 0xC0000000 영역에 복사되어 실행된다. (Linker Script 로 amazon2_ram.ld 파일 사용 시).

이 영역은 Linker Script 파일에서 stack address 를 설정함으로써 크기를 조정하고 있다.

```
8   MEMORY
9   {
10      rom (rx)    : ORIGIN = 0x00000000,  LENGTH = 512K
11      /* vector table will stored at 0x18000000 by InitInterrupt()*/
12      spm(rwx)    : ORIGIN = 0x20000200,  LENGTH = (16*1024-0x200)
13      ram (!rx)   : ORIGIN = 0xC0000000,  LENGTH = 32M
14  }
```

```
79      ._stack 0xC1FFFFFC :
80      {
81        _stack = .;
82        /*PROVIDE (__stack = .);*/
83        *(._stack)
84      }
85  }
```

< amazon2_ram.ld 파일 >

위 파일은 startup 폴더에 있는 amazon2_ram.ld 파일로 ram 의 시작주소를 0xC0000000 으로 설정하고 LENGTH 를 32M 로 설정, 그리고 stack pointer 의 주소를 0xC1FFFFFC 로 해서 32MByte 를 위의 표처럼 Program 이 사용하는 영역으로 설정한 것이다. stack pointer 의 위치를 조정하여 이 영역의 크기를 조정할 수 있다.

Frame buffer 는 화면에 출력할 이미지 데이터가 저장되는 공간으로 single buf mode 인지 double buf mode 인지에 따라 생성되는 Frame buffer 의 개수가 결정되고, 해상도와 RGB565/RGB888 방식에 따라 Frame buffer 의 size 가 결정된다. 기본적으로 single buf mode 의 경우 Frame buffer 가 한 개 생성되고, double buf mode 의 경우 Frame buffer 가 두 개 생성된다.

Frame buffer 의 size 는 해상도와 한 픽셀이 차지하는 byte 수에 의해서 결정 되어 지는데, 계산 시에 가로/세로 사이즈가 512 보다 작으면 512 를 곱하고, 512 보다 크고 1024 보다 작으면 1024 를 2048 보다 작으면 2048 을 곱한다. 예를 들면, 1024x600 해상도의 RGB565 mode 를 사용할 경우 size 는 1024 x 1024 x 2 = 2MByte 가 된다. (RGB888 일 경우에는 픽셀당 byte 수가 4 로 x4 를 해주면 된다.)

이렇게 정해진 개수 크기만큼 setscreen 함수에서 설정한 주소 (default 0xc2100000)부터 생성되게 된다.

single buf / 1024 x 600 / RGB565 => Frame buf 1 개, 한 개 Frame buf 당 size 2MByte, 총 2MByte.
double buf / 1024 x 600 / RGB888 => Frame buf 2 개, 한 개 Frame buf 당 size 4MByte, 총 8MByte

Texture memory 는 graphic engine 에서 접근하는 memory 로 load 된 image 가 저장되는 영역이다. ge_set_texture_mem( ) 함수에 의해 시작주소와 크기가 결정된다. 이 함수를 사용하여 가능한 크기 내에서 설정하여 사용할 수 있다. default 로는 0xc2900000 부터 87MByte 를 사용하고 있다.

load 함수에 의해 이 영역에 메모리가 할당이 되고, release_surface 함수에 의해 메모리가 해제된다.

5. UART

Function	Description
uart_config	UART 초기 설정을 한다.
uart_putch	문자를 출력한다..
uart_putdata	문자를 지정한 개수만큼 출력한다.
uart_putstring	문자열을 출력한다.
uart_getch	문자를 지정한 변수에 저장한다.
uart_getdata	지정한 개수만큼 문자를 저장한다.
uart_rx_flush	UART Receiver FIFO를비운다.
uart_tx_flush	UART Transmitter FIFO를 비운다.
set_debug_channel	Debugging용으로 사용할 UART 채널을 결정한다.
get_debug_channel	Debugging용으로 사용하는 UART채널 값을 읽어온다.
debugprintf	문자, 숫자, 변수의 내용을 출력한다.
debugstring	문자열을 출력한다.
PRINTLINE	호출한 곳의 LINE을 출력한다.
PRINTVAR	호출한 곳의 LINE값과 변수 값을 출력한다.

5.1 uart_config

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

Overview

UART 사용을 위한 초기 설정을 하는 함수이다.

Parameter

int ch	UART channel 값. 초기 설정 할 channel 을 입력한다.
int baud	UART baud rate 값. SDK 는 기본적으로 115200 을 사용한다.
UART_DATABITS 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**;
UART_STOPBITS 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**;
UART_PARITY 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 or FALSE

5.2 uart_putch

```
BOOL  uart_putch (
      int  n,
      char  ch
);
```

Overview

UART n 채널을 통해 ch 값(한 문자)을 출력한다.

Parameter

int n	값을 출력 할 UART channel 값.
char ch	UART 를 통해 출력할 값.

Return Value

TRUE or FALSE

5.3 uart_putdata

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

Overview

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

Parameter

int n	값을 출력 할 UART channel 값.
U8* buf	출력 할 문자열이 저장되어 있는 buffer.
int len	출력 할 문자열의 개수

Return Value

TRUE or FALSE

5.4 uart_putstring

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

Overview

UART n 채널을 통해 buf 에 저장되어 있는 문자열을 출력한다.

Parameter

int n	값을 출력 할 UART channel 값.
U8* buf	출력 할 문자열이 저장되어 있는 buffer.

Return Value

TRUE or FALSE

5.5 uart_getch

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

Overview

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

Parameter

int n	값을 입력 받을 UART channel 값.
char* ch	UART 를 통해 값을 입력 받을 변수.

Return Value

TRUE or FALSE

5.6 uart_getdata

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

Overview

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

Parameter

int n	값을 입력 받을 UART channel 값.
U8* buf	UART 를 통해 값을 입력 받을 변수.
int bufmax	값(문자)을 읽을 개수. (Byte 단위)

Return Value

읽은 data byte 수. Bufmax 값과 비교하여 원하는 만큼 다 읽어 왔는지를 확인한다.

5.7 uart_rx_flush

```
void  uart_rx_flush (
     int  ch
);
```

Overview

UART n 채널의 receiver FIFO 를 비운다.

Parameter

int ch	FIFO 를 비울 UART channel 값.

Return Value

없음.

5.8 uart_tx_flush

```
void  uart_tx_flush (
      int  ch
);
```

Overview

UART n 채널의 transmitter FIFO 를 비운다.

Parameter

int ch	FIFO 를 비울 UART channel 값.

Return Value

없음.

5.9 set_debug_channel

```
void  set_debug_channel (
      int  ch
);
```

Overview

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

Parameter

int ch	디버깅용으로 사용 할 UART 채널.

Return Value

없음.

5.10 get_debug_channel

```
void  get_debug_channel ( );
```

Overview

디버깅용으로 사용하는 UART 채널 값을 return 한다.

Parameter

없음

Return Value

없음.

5.11 debugprintf

```
void  debugprintf (
        const char*  const format,
);
```

Overview

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
);
```

Overview

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("=== AMAZON2 Start ===₩r₩n");

→""안의 문자열을 UART 를 통해 출력한다.

5.13 PRINTLINE

Overview

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)

Overview

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 "sdk.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("AMAZON2 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.
        }
    }
}
```

< UART Interrupt >

AMAZONII SDK library 는 lib_config.h 파일의 정의에 따라 UART Interrupt 를 사용하도록 되어 있다.

```
148
149 /*****************************************************************************
150     UART Config
151 *****************************************************************************/
152 /*
153     인터럽트 모드일 경우    uart 송/수신 data 를 임시로 저장 할 버퍼의 사이즈이다.
154 */
155  #define UART_BUF_SIZE 512
156  #define CONFIG_UART_RX_INTERRUPT
157  //#define CONFIG_UART_TX_INTERRUPT
```

< lib_config.h >

6. INTERRUPT

Function	Description
init_interrupt	Interrupt 초기화 함수.
set_interrupt	Interrupt발생 시 호출 할 함수를 등록한다.
enable_interrupt	Interrupt enable 함수.

6.1 init_interrupt

```
void  init_interrupt ( );
```

Overview

interrupt 를 초기화 하는 함수이다. 인터럽트 관련 함수를 사용하기 위해서 반드시 호출해주어야 하는 함수로 AMAZONII SDK 의 startup_amazon2.S 에서 호출하여 인터럽트를 초기화한다.

6.2 set_interrupt

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

Overview

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

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

Return Value

TRUE or FALSE

6.3 enable_interrupt

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

Overview

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

Parameter

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

Return Value

TRUE or FALSE

INTERRUPT_TYPE

INTERRUPT_TYPE
INTNUM_EIRQ0
INTNUM_TIMER0
INTNUM_SEIP
INTNUM_DMA0
INTNUM_FRAMESYNC
INTNUM_GPIO0
INTNUM_UART0
INTNUM_XDMA0
INTNUM_EIRQ1
INTNUM_TIMER1
INTNUM_PMU
INTNUM_DMA1
INTNUM_ICE
INTNUM_GPIO1
INTNUM_UART1
INTNUM_XDMA1
INTNUM_TIMER2
INTNUM_USBDEV
INTNUM_TWI
INTNUM_USBHOST
INTNUM_I2SRX1
INTNUM_UART2
INTNUM_SEIPRX
INTNUM_WATCHDOG
INTNUM_NAND
INTNUM_DMA2
INTNUM_SDHC
INTNUM_I2STX1

INTERRUPT_TYPE
INTNUM_SPI0
INTNUM_SOUNDMIXER
INTNUM_GPIO2
INTNUM_SPI1
INTNUM_TIMER3
INTNUM_UART3
INTNUM_GPIO3
INTNUM_MJPEG_FULL
INTNUM_DMA3
INTNUM_XDMA2
INTNUM_GPIO4
INTNUM_MJPEG_DEC_END
INTNUM_GPIO5
INTNUM_XDMA3
INTNUM_GPIO6
INTNUM_XDMA4
INTNUM_XDMA5
INTNUM_GPIO7
INTNUM_XDMA6
INTNUM_TIMER_CAP_OVER
INTNUM_GPIO8
INTNUM_XDMA7
INTNUM_GPIO9
INTNUM_GPIO10
INTNUM_GPIO11
INTNUM_GPIO12
INTNUM_GPIO13
INTNUM_GPIO14
INTNUM_GPIO15

6.4 Interrupt Example

```
#include "sdk.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

Function	Description
set_timer	Timer 설정 및 동작시킨다.
stop_timer	Timer 를 정지 시킨다.
delayms	Delay 함수.

7.1 set_timer

```
BOOL  set_timer (
      int  nCh,
      U32  ms
);
```

Overview

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

int nCh	설정 할 timer 채널 값.
U32 ms	timer interrupt 주기. (ms 단위)

Return Value

TRUE or FALSE

7.2 stop_timer

```
BOOL  stop_timer (
      int  nCh
);
```

Overview

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

Parameter

int nCh	정지시킬 timer 채널 값.

Return Value

TRUE or FALSE

7.3 delayms

```
BOOL  delayms (
    U32  ms
);
```

Overview

ms 만큼 delay 가 걸린다. (단위는 ms 이다)

Parameter

U32 ms	delay 걸리는 시간. 단위는 ms.

Return Value

TRUE or FALSE

7.4 TIMER Example

```
#include "sdk.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

Function	Description
ge_set_texture_mem	Texture memory 영역과 크기를 설정한다.
get_ge_target_buffer	현재 graphic engine이 접근하는 frame buffer를 반환한다.
setscreen	LCD 출력을 위한 기본 설정을 수행한다.
get_front_buffer	현재 화면에 보여지는 frame buffer를 반환한다.
get_back_buffer	현재 화면에 보여지지 않는 back buffer를 반환한다.
get_screen_width	현재 화면의 가로 크기 값을 가져온다.
get_screen_height	현재 화면의 세로 크기 값을 가져온다.
get_screen_pitch	현재 화면의 pitch 값을 가져온다.
get_screen_bpp	현재 화면의 bpp(bit per pixel)값을 가져온다.
flip	Double frame buffer를 사용하는 경우 buffer간에 전환을 한다.
flip_wait	Double frame buffer를 사용하는 경우 buffer간에 전환을 한다. flip()과 다른 점은 전환이 완료 될 때까지 기다린다.
async_flip	Double frame buffer를 사용하는 경우 buffer간에 전환을 한다. flip()과 다른 점은 Frame sync 신호와 비 동기로 전환을 수행한다.
async_flip_wait	Double frame buffer를 사용하는 경우 buffer간에 전환을 한다. async_flip()과 다른 점은 전환이 완료 될 때까지 기다린다.
ge_wait_cmd_complete	graphic engine의 동작의 완료를 기다린다.
ge_draw_rectfill	graphic engine을 사용하여 채워진 사각형을 그린다.
ge_draw_rectfillalpha	graphic engine을 사용하여 alpha효과의 채워진 사각형을 그린다.
surface_set_alpha	지정한 surface의 alpha값을 설정한다.
draw_line	선을 그린다.

Function	Description
draw_rect	사각형을 그린다.
draw_rectfill	안이 채워진 사각형을 그린다.
draw_roundrect	모서리가 둥근 사각형을 그린다.
draw_roundrectfill	모서리가 둥근 안이 채워진 사각형을 그린다.
draw_circle	원을 그린다.
draw_circlefill	안이 채워진 원을 그린다.
draw_ellipse	타원을 그린다.
draw_ellipsefill	안이 채워진 타원을 그린다.
loadbmp	BMP 이미지 파일을 load한다.
loadbmpp	BMP 이미지를 메모리에서 load한다.
loadjpg	JPG 이미지 파일을 load한다.
loadjpgp	JPG 이미지를 메모리에서 load한다.
loadpng	PNG 이미지 파일을 load한다.
loadpngp	PNG 이미지를 메모리에서 load한다.
loadsurf	SURF 이미지 파일을 load한다.
draw_surface	load 된 image를 출력한다.
draw_surface_rect	image 중 지정된 영역만 출력한다.
draw_surface_scale	image를 확대 또는 축소해서 출력한다.
draw_surface_scalerect	image의 특정 부분을 확대 또는 축소하여 출력한다.

Function	Description
release_surface	image load 함수에 의해 할당 된 메모리 영역을 해제한다.
draw_set_clip_window	화면에서 이미지가 출력될 수 있는 제한 영역을 설정한다.
osd_set_surface	OSD가 사용할 영역, 좌표를 설정한다.
osd_set_surface_with_alpha	OSD가 사용할 영역, 좌표를 설정한다. alpha값을 적용할 수 있다.
osd_set_pal	palette를 설정한다.
osd_set_off	OSD를 disable 시킨다.
OSD_MOVE	OSD 좌표를 이동한다.
vga_mode_on	VGA (analog RGB) 출력모드를 설정한다. VGA출력 시 반드시 호출해주어야 한다.
vce_on_surface	외부 영상을 사용할 때 호출하는 함수로, 지정한 SURFACE로 외부 영상 데이터를 저장한다.
loadjpg_hw	H/W decoder를 사용하여 JPG이미지를 load한다.
savebmp	특정 화면을 bmp이미지로 저장한다.
savejpg	특정 화면을 jpg이미지로 저장한다.

8.1 ge_set_texture_mem

```
void  ge_set_texture_mem (
      U32  addr,
      U32  size
);
```

Overview

graphic engine 이 사용하는 texture memory 영역과 크기를 설정한다.

Parameter

U32 addr	Texture memory 시작 주소.
U32 size	Texture memory 크기.

Return Value

없음

8.2 get_ge_target_buffer

```
SURFACE*  get_ge_target_buffer ( );
```

Overview

현재 graphic 엔진이 draw 를 위해 접근하는 buffer 의 SURFACE 포인터를 반환한다.
DOUBLEBUF 의 경우에는 get_back_buffer()와 동일한 동작을 수행하고, SINGLEBUF 의 경우에는 get_front_buffer()와 동일한 동작을 수행한다.

Parameter

없음.

Return Value

현재 graphic 엔진이 접근하는 buffer 의 SURFACE 포인터.

8.3 setscreen

```
void  setscreen (
      U32  framebufferAddr,
      SCREENRES  size,
      U32  scmode
);
```

Overview

LCD 해상도와 RGB 모드와 framebuffer 시작 주소를 설정한다.
LCD 출력을 위해 설정해주어야 한다.

Parameter

U32 framebufferAddr — frame buffer 시작 주소.

SCREENRES 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,
            SCREEN_800x600,
            SCREEN_1024x768,
            SCREEN_TV_HD,
            SCREEN_TV_NTSC,
            SCREEN_TV_PAL,
           } SCREENRES;
```

U32 scmode — screen 모드를 설정하는 값으로, 다음과 같이 정의 되어 있다.

SCREENMODE_RGB888
SCREENMODE_RGB565
SCREENMODE_SINGLEBUF
SCREENMODE_DOUBLEBUF

RGB888/RGB565 중 하나를 선택해야 하고,
SINGLEBUF/DOUBLEBUF 중 하나를 선택해야 한다.

Return Value

없음

8.4 get_front_buffer

```
SURFACE*  setscreen (
    U32  framebufferAddr,
    SCREENRES  size,
    U32  scmode
);
```

Overview

현재 화면에 보여지는 front buffer 의 SURFACE 포인터를 반환한다.

Parameter

없음.

Return Value

현재 보여지고 있는 buffer 의 SURFACE 포인터.

8.5 get_back_buffer

```
SURFACE*  get_back_buffer ( );
```

Overview

Back buffer 의 SURFACE 포인터를 반환한다.

Parameter

없음.

Return Value

Back buffer 의 SURFACE 포인터.

8.6 get_screen_width

U32 get_screen_width ( );

Overview

현재 화면의 가로 크기 값을 가져온다.

Parameter

없음.

Return Value

화면의 가로크기 값.

8.7 get_screen_height

U32 get_screen_height ( );

Overview

현재 화면의 세로 크기 값을 가져온다.

Parameter

없음.

Return Value

화면의 세로크기 값.

8.8 get_screen_pitch

U32 get_screen_pitch ( );

Overview

화면의 pitch 값을 가져온다. pitch 값은 가로 크기와 bpp 값에 따라 다른데, 다음과 같다.

가로 크기 <= 512 → 512 * bpp / 8

가로 크기 <= 1024 → 1024 * bpp / 8

가로 크기 <= 2048 → 2048 * bpp / 8

Parameter

없음.

Return Value

화면의 pitch 값.

8.9 get_screen_bpp

```
U32  get_screen_bpp ( );
```

Overview

화면의 bpp(bit per pixel)값을 가져온다.

Parameter

없음.

Return Value

화면의 bpp 값.

8.10 flip

```
void  flip ( );
```

Overview

Double frame buffer 를 사용하는 경우 buffer 간에 전환을 한다.
Double frame buffer 사용을 원한다면 setscreen 함수에서 scmode 인자 값을 SCREENMODE_DOUBLEBUF 로 해야 한다.

Parameter

없음.

Return Value

없음.

8.11 flip_wait

```
void  flip_wait ( );
```

Overview

Double frame buffer 를 사용하는 경우 buffer 간의 전환을 한다.
flip()과 다른 점은 전환이 완료 될 때까지 기다렸다가 함수를 빠져 나온다.

Parameter

없음.

Return Value

없음.

8.12 async_flip

```
void  async_flip ( );
```

Overview

Double frame buffer 를 사용하는 경우 buffer 간의 전환을 한다.
flip()과 다른 점은 Frame sync 신호와 비 동기로 전환을 수행한다.

Parameter

없음.

Return Value

없음.

8.13 async_flip_wait

```
void  async_flip_wait ( );
```

Overview

Double frame buffer 를 사용하는 경우 buffer 간의 전환을 한다.
async_flip()과 다른 점은 완료 될 때까지 기다렸다가 함수를 빠져 나온다..

Parameter

없음.

Return Value

없음.

8.14 ge_wait_cmd_complete

```
void  ge_wait_cmd_complete ( );
```

Overview

Graphic engine 의 동작의 완료를 기다린다. Graphic engine 동작에 영향을 줄 수 있는 동작을 수행 할 때 이 함수를 사용하여 graphic engine 동작 완료를 체크한다.

Parameter

없음.

Return Value

없음.

8.15 ge_draw_rectfill

```
void  ge_draw_rectfill (
    int  x,
    int  y,
    int  w,
    int  h,
    EGL_COLOR  c
 );
```

Overview

Graphic engine 을 사용하여 채워진 사각형을 그린다.

Parameter

int x	사각형의 x 좌표.
int y	사각형의 y 좌표.
int w	사각형의 가로 크기.
int h	사각형의 세로 크기.
EGL_COLOR c	사각형의 color. MAKE_COLORREF(r,g,b) 함수를 사용하면 color 지정을 쉽게 할 수 있다.

Return Value

없음.

8.16 ge_draw_rectfillalpha

```
void  ge_draw_rectfillalpha (
    int  x,
    int  y,
    int  w,
    int  h,
    EGL_COLOR  acolor
 );
```

Overview

Graphic engine 을 사용하여 alpha 효과의 채워진 사각형을 그린다.

Parameter

int x	사각형의 x 좌표.
int y	사각형의 y 좌표.
int w	사각형의 가로 크기.
int h	사각형의 세로 크기.
EGL_COLOR c	사각형의 color. MAKE_COLORREF(r,g,b) 함수를 사용하면 color 지정을 쉽게 할 수 있다.

Return Value

없음.

8.17 surface_set_alpha

```
BOOL  surface_set_alpha (
    SURFACE*  surf,
    U8  a
);
```

Overview

지정한 surface 의 alpha 값을 설정한다.

Parameter

SURFACE* surf	Alpha 값을 지정 할 surface.
U8 a	설정 할 alpha 값.

Return Value

TRUE or FALSE

8.18 draw_line

```
void  draw_line (
    int  x1,
    int  y1,
    int  x2,
    int  y2,
    EGL_COLOR  color
);
```

Overview

x1, y1 위치에서 x2, y2 위치로 선을 그린다. 선의 색은 MAKE_COLORREF( ) MACRO 에 의해 색을 설정하여 사용한다.

Parameter

int x1	선의 시작점의 x 좌표.
int y1	선의 시작점의 y 좌표.
int x2	선의 끝점의 x 좌표.
int y2	선의 끝점의 y 좌표.
EGL_COLOR color	선의 color. MAKE_COLORREF(r,g,b) 함수를 사용하면 color 지정을 쉽게 할 수 있다.

Return Value

없음

8.19 draw_rect

```
void  draw_rect (
    int  x,
    int  y,
    int  w,
    int  h,
    EGL_COLOR  c
);
```

Overview

x, y 좌표를 시작점으로 가로가 w, 세로가 h 인 사각형을 그린다.

Parameter

int x	사각형 시작점의 x 좌표.
int y	사각형 시작점의 y 좌표.
int w	사각형의 가로 길이.
int h	사각형의 세로 길이.
EGL_COLOR c	사각형의 color. MAKE_COLORREF(r,g,b) 함수를 사용하면 color 지정을 쉽게 할 수 있다.

Return Value

없음

8.20 draw_rectfill

```
void  draw_rectfill (
    int  x,
    int  y,
    int  w,
    int  h,
    EGL_COLOR  c
);
```

Overview

x, y 좌표를 시작점으로 가로가 w, 세로가 h 인 사각형을 그린다. draw_rect 와 다른 점은 안을 채운 사각형을 그린다.

Parameter

int x	사각형 시작점의 x 좌표.
int y	사각형 시작점의 y 좌표.
int w	사각형의 가로 길이.
int h	사각형의 세로 길이.
EGL_COLOR c	사각형의 color. MAKE_COLORREF(r,g,b) 함수를 사용하면 color 지정을 쉽게 할 수 있다.

Return Value

없음

8.21 draw_roundrect

```
void  draw_roundrect (
    int  x0,
    int  y0,
    int  w,
    int  h,
    int  corner,
    EGL_COLOR  c
);
```

Overview

x, y 좌표를 시작점으로 가로가 w, 세로가 h 인 모서리가 둥근 사각형을 그린다. corner 값에 따라 모서리의 둥근 형태가 조정된다.

Parameter

int x0	사각형 시작점의 x 좌표.
int y0	사각형 시작점의 y 좌표.
int w	사각형의 가로 길이.
int h	사각형의 세로 길이.
int corner	모서리의 둥근 형태 조정 값. 이 값이 클수록 둥근 부분이 넓어진다.
EGL_COLOR c	사각형의 color. MAKE_COLORREF(r,g,b) 함수를 사용하면 color 지정을 쉽게 할 수 있다.

Return Value

없음

8.22 draw_roundrectfill

```
void  draw_roundrectfill (
    int  x0,
    int  y0,
    int  w,
    int  h,
    int  corner,
    EGL_COLOR  c
);
```

Overview

x, y 좌표를 시작점으로 가로가 w, 세로가 h 인 모서리가 둥근 사각형을 그린다. corner 값에 따라 모서리의 둥근 형태가 조정된다. draw_roundrect 와 다르게 안을 채운 사각형을 그린다.

Parameter

int x0	사각형 시작점의 x 좌표.
int y0	사각형 시작점의 y 좌표.
int w	사각형의 가로 길이.
int h	사각형의 세로 길이.
int corner	모서리의 둥근 형태 조정 값. 이 값이 클수록 둥근 부분이 넓어진다.
EGL_COLOR c	사각형의 color. MAKE_COLORREF(r,g,b) 함수를 사용하면 color 지정을 쉽게 할 수 있다.

Return Value

없음

8.23 draw_circle

```
void  draw_circle (
    int  x,
    int  y,
    int  r,
    EGL_COLOR  color
);
```

Overview

x, y 좌표를 원점으로 하고 반지름이 r 인 원을 그린다.

Parameter

int x	원점의 x 좌표.
int y	원점의 y 좌표.
int r	원의 반지름.
EGL_COLOR color	원의 color. MAKE_COLORREF(r,g,b) 함수를 사용하면 color 지정을 쉽게 할 수 있다.

Return Value

없음

8.24 draw_circlefill

```
void  draw_circlefill (
    int  x,
    int  y,
    int  r,
    EGL_COLOR  color
);
```

Overview

x, y 좌표를 원점으로 하고 반지름이 r 인 안을 채운 원을 그린다.

Parameter

int x	원점의 x 좌표.
int y	원점의 y 좌표.
int r	원의 반지름.
EGL_COLOR color	원의 color. MAKE_COLORREF(r,g,b) 함수를 사용하면 color 지정을 쉽게 할 수 있다.

Return Value

없음

8.25 draw_ellipse

```
void  draw_ellipse (
    int  x,
    int  y,
    int  rx,
    int  ry,
    EGL_COLOR  color
);
```

Overview

x, y 좌표를 원점으로 하고 x 축 반지름이 rx 이고 y 축 반지름이 ry 인 타원을 그린다.

Parameter

int x	원점의 x 좌표.
int y	원점의 y 좌표.
int rx	타원의 x 축 반지름.
int ry	타원의 y 축 반지름.
EGL_COLOR color	타원의 color. MAKE_COLORREF(r,g,b) 함수를 사용하면 color 지정을 쉽게 할 수 있다.

Return Value

없음

6.26 draw_ellipsefill

```
void  draw_ellipsefill (
    int  x,
    int  y,
    int  rx,
    int  ry,
    EGL_COLOR  color
);
```

Overview

x, y 좌표를 원점으로 하고 x 축 반지름이 rx 이고 y 축 반지름이 ry 인 안을 채운 타원을 그린다.

Parameter

int x	원점의 x 좌표.
int y	원점의 y 좌표.
int rx	타원의 x 축 반지름.
int ry	타원의 y 축 반지름.
EGL_COLOR color	타원의 color. MAKE_COLORREF(r,g,b) 함수를 사용하면 color 지정을 쉽게 할 수 있다.

Return Value

없음

8.27 loadbmp

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

Overview

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

Parameter

char* fname	BMP 파일 이름.

Return Value

이미지 정보와 이미지 데이터가 저장된 SURFACE 포인터.

8.28 loadbmpp

```
SURFACE*  loadbmpp (
    U8*  startaddr
);
```

Overview

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

Parameter

U8* startaddr	이미지가 저장되어 있는 메모리 주소.

Return Value

이미지 정보와 이미지 데이터가 저장된 SURFACE 포인터.

8.29 loadjpg

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

Overview

JPG 이미지 파일을 load 한다. 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

char* fname	JPG 파일 이름

Return Value

이미지 정보와 이미지 데이터가 저장된 SURFACE 포인터.

8.30 loadjpgp

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

Overview

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

Parameter

U8* databuf	이미지가 저장되어 있는 메모리 주소.
U32 len	이미지 데이터의 길이.

Return Value

이미지 정보와 이미지 데이터가 저장된 SURFACE 포인터.

8.31 loadtga

```
SURFACE*  loadtga (
    char*  fname
);
```

Overview

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

Parameter

char* fname	tga 파일 이름

Return Value

이미지 정보와 이미지 데이터가 저장된 SURFACE 포인터.

8.32 loadtgap

```
SURFACE*  loadtgap (
    U8*  startaddr
);
```

Overview

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

Parameter

U8* startaddr	이미지가 저장되어 있는 메모리 주소.

Return Value

이미지 정보와 이미지 데이터가 저장된 SURFACE 포인터.

8.33 loadpng

```
SURFACE*  loadpng (
    char*  filename
);
```

Overview

PNG 이미지 파일을 load 한다. 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

char* filename	PNG 파일 이름

Return Value

이미지 정보와 이미지 데이터가 저장된 SURFACE 포인터.

8.34 loadpngp

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

Overview

PNG 이미지를 메모리에서 load 한다. loadpng 와 다른 점은 파일을 load 하는 것이 아니고, 메모리에서 load 한다는 점이다.

Parameter

U8* pngbuf	이미지가 저장되어 있는 메모리 주소
U32 datalen	이미지 데이터 길이

Return Value

이미지 정보와 이미지 데이터가 저장된 SURFACE 포인터.

8.35 loadsurf

```
SURFACE*  loadsurf (
    char*  fname
);
```

Overview

Suf 이미지 파일을 load 하는 함수. suf 파일을 load 하여 메모리 영역을 생성 drawsurface( ) 함수를 사용하여 이미지를 출력할 수 있도록 SURFACE 구조체를 생성 후 저장한다.
Suf 파일은 RGB 모드에 맞게 변환한 파일로 load 시간이 지원하는 이미지 중 가장 적게 소요된다.
사용방법은 다른 이미지 load 함수와 동일하며, 이미지 변환은 SDK pc-util 의 MakeSurface2.exe 프로그램을 사용하면 된다.

Parameter

char* fname	surf 파일 이름.

Return Value

이미지 정보와 이미지 데이터가 저장된 SURFACE 포인터.

8.36 draw_surface

```
BOOL  draw_surface (
    SURFACE*  src_surf,
    int  dx,
    int  dy
);
```

Overview

이미지를 출력한다.

Parameter

SURFACE* src_surf	Load 된 이미지의 SURFACE 포인터.
int dx	이미지가 위치 할 x 좌표.
int dy	이미지가 위치 할 y 좌표.

Return Value

TRUE or FALSE

8.37 draw_surface_rect

```
BOOL  draw_surface_rect (
    SURFACE*  src_surf,
    int  dx,
    int  dy,
    int  sx,
    int  sy,
    int  w,
    int  h
);
```

Overview

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

Parameter

SURFACE* src_surf	Load 된 이미지의 SURFACE 포인터.
int dx	이미지가 위치 할 x 좌표.
int dy	이미지가 위치 할 y 좌표.
int sx	이미지 중 출력 할 부분의 시작 x 좌표.
int sy	이미지 중 출력 할 부분의 시작 y 좌표.
w	출력 할 이미지의 가로 길이. 시작 sx 부터 w 만큼 출력한다.
h	출력 할 이미지의 세로 길이. 시작 sy 부터 h 만큼 출력한다.

Return Value

TRUE or FALSE

8.38 draw_surface_scale

```
BOOL  draw_surface_scale (
    SURFACE*  src_surf,
    int  dx,
    int  dy,
    int  dw,
    int  dh
);
```

Overview

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

SURFACE* src_surf	Load 된 이미지의 SURFACE 포인터.
int dx	이미지가 위치 할 x 좌표.
int dy	이미지가 위치 할 y 좌표.
int dw	출력 될 이미지의 가로 길이. 원본 이미지 가로길이보다 이 값이 크면, 가로길이가 확대되어 출력되고, 작으면 가로길이가 축소되어 출력된다.
int dh	출력 될 이미지의 세로 길이. 원본 이미지 세로길이보다 이 값이 크면, 세로길이가 확대되고, 작으면 세로길이가 축소되어 출력된다.

Return Value

TRUE or FALSE

8.39 draw_surface_scalerect

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

Overview

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

SURFACE* src_surf	Load 된 이미지의 SURFACE 포인터.
int dx	이미지가 출력 될 x 좌표.
int dy	이미지가 출력 될 y 좌표.
int dw	출력 될 이미지의 가로 길이. 원본 이미지 가로길이보다 이 값이 크면, 가로길이가 확대되어 출력되고, 작으면 가로길이가 축소되어 출력된다.
int dh	출력 될 이미지의 세로 길이. 원본 이미지 세로길이보다 이 값이 크면, 세로길이가 확대되고, 작으면 세로길이가 축소되어 출력된다.
int sx	원본 이미지 중 출력 할 부분의 시작 x 좌표.
int sy	원본 이미지 중 출력 할 부분의 시작 y 좌표.
int sw	출력 할 이미지의 가로 길이. 시작 sx 부터 sw 만큼 출력한다.
int sh	출력 할 이미지의 세로 길이. 시작 sy 부터 sh 만큼 출력한다.

Return Value

TRUE or FALSE

8.40 release_surface

```
void  release_surfacet (
    SURFACE*  surf
);
```

Overview

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

Parameter

SURFACE* surf	Load 된 이미지의 SURFACE 포인터.

Return Value

없음

8.41 draw_set_clip_winodw

```
void  draw_set_clip_window (
    SURFACE*  dst,
    CLIP_RECT*  pRect
);
```

Overview

화면에서 이미지가 출력될 수 있는 제한 영역을 설정한다. pRect 안의 영역에만 이미지가 출력된다.

Parameter

SURFACE* dst	대상 frame.
CLIP_RECT* pRect	이미지 출력을 허용할 영역. typedef struct { int x; int y; int endx; int endy; }CLIP_RECT;

Return Value

없음

8.42 osd_set_surface

```
BOOL  osd_set_surface (
    SURFACE*  surf,
    int  x,
    int  y
);
```

Overview

OSD 가 사용할 영역(surface)과 OSD 가 출력될 좌표를 설정하고 OSD 를 enable 시킨다.

Parameter

SURFACE* surf	OSD 영역으로 사용 할 surface.
int x	OSD 출력 x 좌표.
int y	OSD 출력 y 좌표.

Return Value

TRUE or FALSE

8.43 osd_set_surface_with_alpha

```
BOOL  osd_set_surface_with_alpha (
    SURFACE*  surf,
    int  x,
    int  y,
    U8  alpha
);
```

Overview

OSD 가 사용할 영역(surface)과 OSD 가 출력될 좌표를 설정하고, OSD 를 enable 시킨다.
alpha 값을 적용할 수 있다.

Parameter

SURFACE* surf	OSD 영역으로 사용 할 surface.
int x	OSD 출력 x 좌표.
int y	OSD 출력 y 좌표.
U8 alpha	OSD 영역의 alpha 값.

Return Value

TRUE or FALSE

8.44 osd_set_pal

```
void  osd_set_pal (
    EGL_COLOR*  pal,
    int  num
);
```

Overview

palette 를 설정한다.

Parameter

EGL_COLOR* pal	palette color buffer.
int num	palette color 개수.

Return Value

없음

8.45 osd_set_off

```
void  osd_set_off ( );
```

Overview

OSD 를 disable 시킨다..

Parameter

없음

Return Value

없음

8.46 OSD_MOVE

```
OSD_MOVE ( x , y );
```

Overview

OSD 좌표를 이동한다.

Parameter

x	이동 할 x 좌표.
y	이동 할 y 좌표.

Return Value

없음

8.47 vga_mode_on

```
void vga_mode_on( void );
```

Overview

VGA (analog RGB) 출력모드를 설정한다. VGA 출력 시 반드시 호출해주어야 한다.

Parameter

없음.

Return Value

없음

8.48 vce_on_surface

```
BOOL vce_on_surface (
   SURFACE* dst,
   int x,
   int y,
   bool halfmode
);
```

Overview

외부 영상을 화면에 출력할 때 사용하는 함수이다. 지정한 SURFACE 로 외부 영상 데이터가 저장되어 image 를 출력하 듯이 draw_surface 함수를 사용하여 원하는 곳에 외부 영상을 출력할 수 있다. SURFACE 를 지정해 주어야 하기 때문에 create_surface 함수로 surface 를 생성해주어야 한다.

Parameter

SURFACE* dst.	외부 영상이 저장될 SURFACE
int x	SURFACE 에서 외부 영상의 x 좌표
int y	SURFACE 에서 외부 영상의 y 좌표
bool halfmode	half mode 를 사용 여부를 결정

Return Value

TRUE or FALSE

Example

```
SURFACE* ext_surf =  create_surface(720,480,16)
vce_on_surface(ext_surf,0,0,false);

draw_surface(ext_surf,24,24);
flip();
```

8.49 loadjpg_hw

```
SURFACE*  loadjpg_hw (
    char*  filename
);
```

Overview

H/W decoder 를 사용하여 JPG 이미지 파일을 load 한다. jpg 파일을 load 하여 메모리 영역을 생성 drawsurface( )함수를 사용하여 이미지를 출력할 수 있도록 SURFACE 구조체를 생성 후 저장을 한다. S/W 로 load 하는 것보다 decoding 속도가 빠르고, H/W decoder 는 standard format 만 지원하기 때문에 pc-util 의 makesurface2.exe 파일을 사용하여 standard format jpg 이미지로 변환해주어야 한다.

Parameter

char* fname	JPG 파일 이름

Return Value

이미지 정보와 이미지 데이터가 저장된 SURFACE 포인터.

8.50 savebmp

```
BOOL  savebmp (
    char*  savefname,
    SURFACE*  src
);
```

Overview

SURFACE 의 내용을 BMP 이미지로 저장한다. 현재화면을 이미지파일로 저장하거나, 특정 SURFACE 를 이미지로 저장할 때 사용한다.

Parameter

char* savefname	저장하려는 BMP 파일 이름
SURFACE* src	저장하려는 화면의 SURFACE

Return Value

TRUE or FALSE.

8.51 savejpg

```
BOOL  savejpg (
    char*  savefname,
    SURFACE*  src
);
```

Overview

SURFACE 의 내용을 JPG 이미지로 저장한다. 현재화면을 이미지파일로 저장하거나, 특정 SURFACE 를 이미지로 저장할 때 사용한다.

Parameter

char* savefname	저장하려는 JPG 파일 이름
SURFACE* src	저장하려는 화면의 SURFACE

Return Value

TRUE or FALSE.

< savebmp/savejpg example >

```
SURFACE* current_frame = get_front_buffer();
res = savebmp("image1.bmp", current_frame); // 현재 보여지는 화면을 image1.bmp 파일로 저장
res = savejpg("image2.jpg", current_frame);  // 현재 보여지는 화면을 image2.jpg 파일로 저장
```

8.52 Graphic Example

<Single Frame>

```
#include "sdk.h"

int main()
{
        boardinit();
        uart_config(0,115200,DATABITS_8,STOPBITS_1,UART_PARNONE);

        display_clock_init();  // set LCD clock
        ge_set_texture_mem(0xc3000000, 0x50000000);
              // set texture memory. start address : 0xc3000000, size : 80MByte
       setscreen(0xc2000000,SCREEN_1024x600,SCREENMODE_RGB565|SCREENNMODE_SINGLEBUF);
              // 1024x600 LCD, frame buffer start address : 0xc2000000, RGB565, single buf mode.

        lcdon();  // turn on the lcd back light.

        draw_rectfill(0,0,get_screen_width(),get_screen_height(),MAKE_COLORREF(255,255,255));
              // draw 1024x600 white rectangle. ( screen cleared as white color )
        draw_line(0,0,100,100,MAKE_COLORREF(255,0,0));  // draw red line.
        draw_rect(100,100,100,100,MAKE_COLORREF(255,0,0));  // draw red rectangle.
        draw_circle(200,200,80,MAKE_COLORREF(0,0,255));  // draw blue circle.
        draw_ellipse(200,50,30,20,MAKE_COLORREF(0,255,0));  // draw green ellipse.
       draw_roundrect(100,270,100,100,10,MAKE_COLORREF(128,128,128));  // draw round rectangle.
       draw_roundrectfill(110,280,80,80,10,MAKE_COLORREF(255,0,0)); // draw round rectangle filled.

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

<Double Frame>

```
#include "sdk.h"

int main()
{
        boardinit();
        uart_config(0,115200,DATABITS_8,STOPBITS_1,UART_PARNONE);

        FATFS fs;
        f_mount(DRIVE_NAND, &fs);

        display_clock_init();  // set LCD clock
        ge_set_texture_mem(0xc3000000, 0x50000000);
              // set texture memory. start address : 0xc3000000, size : 80MByte
      setscreen(0xc2000000,SCREEN_1024x600,SCREENMODE_RGB565|SCREENNMODE_DOUBLEBUF);
              // 1024x600 LCD, frame buffer start address : 0xc2000000, RGB565, double buf mode.

        lcdon();  // turn on the lcd back light.

       SURFACE* bg = loadbmp("background.bmp");  // load bmp image.
        drawsurface(bg,0,0);  // draw image.
        flip();  // switch frame buffer.

        SURFACE* bg2 = loadbmp("background2.bmp");  // load bmp image.
        SURFACE* icon = loadjpg("icon.jpg");    // load jpg image.
        drawsurface(bg2,0,0);  // draw back ground image.
        drawsurface(icon,100,100);  // draw icon image.
        flip(); // switch frame buffer.

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

9. Movie file play

Movie 파일은 SDK 의 pc-util 폴더 내에 있는 MovieGenerator™ 을 통해서 생성 할 수 있다.
MovieGenerator™ 에 대한 상세한 내용은 별도의 문서가 있으므로 본 장에서는 다루지 않는다.

Function	Description
loadmovie	Movie 파일을 load한다.
release_movie	loadmovie 함수에서 생성된 메모리를 해제 한다.
movie_play	Video의 첫번째 frame을 출력한다.
movie_drawnext	다음 video frame을 원하는 위치에 출력한다.
movie_drawnextex	다음 video frame을 원하는 위치에 출력한다. Size 조절 가능.
movie_seek	파일의 현재 위치를 특정 시간으로 변경한다.
movie_current_time	현재 출력중인 frame의 시간을 반환한다.

9.1 loadmovie

```
MOVIE*  loadmovie (
    char*  fname,
    BOOL  bAll
 );
```

Overview

Movie 파일을 저장장치에서 load 하여 필요한 정보를 얻어 온다.
이때 audio data 는 전체가 주 메모리에 저장된다. 따라서 audio data 가 있는 경우 충분한 메모리가 확보되어야 한다.
11025Hz * 2 (16bit mono) * 30 sec = 661,500 bytes
loadmovie 에 의해서 사용된 메모리는 release_movie 를 이용하여 메모리 해제할 수 있다.

Parameter

char* fname	Movie 파일 이름.
BOOL bAll	Video data 도 주 메모리에 로딩 할 것인지 여부 false 인 경우 video 를 매 frame 마다 저장장치에서 읽어 오게 된다.

Return Value

Pointer to MOVIE

9.2 release_movie

```
void  release_movie (
      MOVIE*  mov
);
```

Overview

loadmovie 함수에서 생성된 메모리를 해제한다.

Parameter

MOVIE* mov	Pointer to MOVIE.

Return Value

없음.

9.3 movie_play

```
BOOL  movie_play (
      MOVIE*  mov,
      int  x,
      int  y,
      BOOL  bAudio
);
```

Overview

Video 의 첫 번째 frame 을 해당 영역에 출력한다. bAudio 가 true 이고 movie 파일에 audio 가 존재 할 경우 audio play 를 시작한다.

Parameter

MOVIE* mov	Pointer to MOVIE.
int x	x 좌표
int y	y 좌표
BOOL bAudio	audio 출력 여부, false 일 경우 audio 를 출력하지 않는다.

Return Value

True or False.

9.4 movie_drawnext

```
MOVIE_DRAW_RESULT  movie_drawnext (
    MOVIE*  mov,
    int  x,
    int  y,
    BOOL  b_audiosync
);
```

Overview

Mov 의 다음 video frame 을 특정 좌표에 출력한다. b_audiosync 가 disable 될 경우 video 를 audio 와 동기화 하지 않고 차례대로 출력하게 된다.

Parameter

MOVIE* mov	Pointer to MOVIE.
int x	x 좌표
int y	y 좌표
BOOL b_audiosync	출력할 video frame 과 audio 에 동기화 여부. True 일 경우 현재 출력 중인 audio 에 동기화를 맞춘 video 를 출력한다.

Return Value

MOVIE_DRAW_RESULT_FAIL – if mov is null, or mov is not running.

MOVIE_DRAW_RESULT_OK – if success.

MOVIE_DRAW_RESULT_EOF – if end of file.

9.5 movie_drawnextex

```
MOVIE_DRAW_RESULT  movie_drawnextex (
    MOVIE*  mov,
    int  x,
    int  y,
    int  w,
    int  h,
    BOOL  b_audiosync
);
```

Overview

Mov 의 다음 video frame 을 특정 좌표에 설정한 크기로 출력한다. b_audiosync 가 disable 될 경우 video 를 audio 와 동기화 하지 않고 차례대로 출력하게 된다.

Parameter

MOVIE* mov	Pointer to MOVIE.
int x	x 좌표
int y	y 좌표
int w	Video frame 의 가로크기 (video 크기보다 작을 경우 축소, 클 경우 확대)
int h	Video frame 의 세로크기 (video 크기보다 작을 경우 축소, 클 경우 확대)
BOOL b_audiosync	출력할 video frame 과 audio 에 동기화 여부. True 일 경우 현재 출력 중인 audio 에 동기화를 맞춘 video 를 출력한다.

Return Value

MOVIE_DRAW_RESULT_FAIL – if mov is null, or mov is not running.

MOVIE_DRAW_RESULT_OK – if success.

MOVIE_DRAW_RESULT_EOF – if end of file.

9.6 movie_seek

```
U32  movie_seek (
     MOVIE*  mov,
     U32  ms
);
```

Overview

Mov 파일의 현재 위치를 특정 시간으로 변경 한다.

Parameter

MOVIE* mov	Pointer to MOVIE.
U32 ms	Milliseconds

Return Value

Number of video frame.

9.7 movie_current_time

```
U32  movie_current_time (
     MOVIE*  mov
);
```

Overview

현재 출력중인 frame 의 시간을 반환한다.

Parameter

MOVIE* mov	Pointer to MOVIE.

Return Value

Current milliseconds.

9.8 example

```
Int main()
{
        …
        …
        MOVIE* mov = loadmovie(fname, FALSE);
        while(1)
        {
                if(movie_drawnext(mov, x, y, true) == MOVIE_DRAW_RESULT_EOF)
                        break;
                flip();
        }
        …
        …
}
```

10. SOUND

Function	Description
sound_init	Sound mixer를 초기화 한다.
sound_loadwav	WAV 파일을 load 한다.
sound_loadwavp	WAV data를 메모리에서 load 한다.
sound_loadmp3	MP3 파일을 load 한다.
sound_loadmp3p	MP3 data를 메모리에서 load 한다.
sound_release	Load된 sound파일이 사용한 메모리를 해제한다.
sound_play	Load 된 sound를 play한다.
sound_stop	Sound를 stop 한다.
sound_vol	Sound mixer의 volume을 조정한다.
sound_vol_wav	Sound 파일 각각의 volume을 조정한다.
sound_pause	Sound를 pause 시킨다.
sound_resume	Pause 상태의 sound를 resume 시킨다.
sound_isplay	Sound가 play 중인지 확인한다.
sound_ispause	Sound가 pause 중인지 확인한다.
sound_playex	Sound를 특정 위치부터 play한다.
sound_current_time	현재 play 중인 위치를 반환한다.

10.1 sound_init

```
BOOL  sound_init ( );
```

Overview

Sound 출력을 위해 Sound mixer 를 초기화 한다.

Parameter

없음

Return Value

없음

10.2 sound_loadwav

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

Overview

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

Parameter

char* filename	WAV 파일 이름.

Return Value

WAV 파일의 정보와 데이터가 저장 된 WAVE 구조체.

10.3 sound_loadwavp

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

Overview

WAV data 를 메모리에서 load 한다.

Parameter

U8* buf	WAV data 가 저장되어 있는 buffer.
U32 len	WAV data 의 길이.

Return Value

WAV 파일의 정보와 데이터가 저장 된 WAVE 구조체.

10.4 sound_loadmp3

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

Overview

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

Parameter

char* filename	MP3 파일.

Return Value

MP3 파일의 정보와 데이터가 저장된 WAVE 구조체.

10.5 sound_loadmp3p

```
WAVE*  sound_loadmp3p (
    U8*  buf,
    U32  mp3databufsize
);
```

Overview

MP3 data 를 메모리에서 load 하는 함수.

Parameter

U8* buf	MP3 data 가 저장되어 있는 buffer.
U32 mp3databufsize	MP3 data 의 길이.

Return Value

WAV 파일의 정보와 데이터가 저장 된 WAVE 구조체.

10.6 sound_release

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

Overview

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

Parameter

WAVE* pWave	load 함수에 의해 생성 된 WAVE 구조체.

Return Value

TRUE or FALSE

10.7 sound_play

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

Overview

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

Parameter

WAVE* pWave	load 함수에 의해 생성 된 WAVE 구조체.

Return Value

TRUE or FALSE

10.8 sound_stop

```
BOOL  sound_stop (
    WAVE*  pWave
);
```

Overview

Play 중인 sound 를 stop 시킨다.

Parameter

WAVE* pWave	load 함수에 의해 생성 된 WAVE 구조체.

Return Value

TRUE or FALSE

10.9 sound_vol

```
void  sound_vol (
      U8  vol
);
```

Overview

sound volume 을 조정한다.

Parameter

U8 vol	volume 값. 최대 값은 255 이다.

Return Value

없음

10.10 sound_vol_wav

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

Overview

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

WAVE* pWav	volume 을 조정 할 음원의 WAVE 구조체.
U8 vol	volume 값. 최대 값은 255 이다.

Return Value

없음

10.11 sound_pause

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

Overview

Play 중인 sound 를 pause 시킨다.

Parameter

WAVE* pWave	현재 play 중인 sound 의 WAVE 구조체.

Return Value

TRUE or FALSE

10.12 sound_resume

```
BOOL  sound_resume (
    WAVE*  pWav
);
```

Overview

Pause 상태의 sound 를 resume 시킨다.

Parameter

WAVE* pWave	현재 pause 중인 sound 의 WAVE 구조체.

Return Value

TRUE or FALSE

10.13 sound_isplay

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

Overview

현재 pWave sound 가 play 중인지 확인할 수 있다.

Parameter

WAVE* pWave	현재 play 중인지 확인하려는 sound 의 WAVE 구조체.

Return Value

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

10.14 sound_ispause

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

Overview

현재 pWave sound 가 pause 중인지 확인할 수 있다.

Parameter

WAVE* pWave	현재 pause 중인지 확인하려는 sound 의 WAVE 구조체.

Return Value

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

10.15 sound_playex

```
BOOL  sound_playex (
      WAVE*  pWave,
      U32 start_ms
);
```

Overview

Sound 를 특정 위치부터 play 한다.

Parameter

WAVE* pWave	특정 위치부터 play 하려는 sound 의 WAVE 구조체.
U32 start_ms	play 하려는 위치. ms 단위.

Return Value

TRUE or FALSE

10.16 sound_current_time

```
U32  sound_current_time (
      WAVE*  pWave
);
```

Overview

현재 play 중인 위치를 반환한다.

Parameter

WAVE* pWave	Play 중인 위치를 확인하려는 sound WAVE 구조체.

Return Value

현재 play 중인 위치 (ms 단위)

10.17 Sound Example

< WAV File Play >

```
#include "sdk.h"

Intmain()
{
    boardinit();
    uart_config(0,115200,DATABITS_8,STOPBITS_1,UART_PARNONE);

    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 "sdk.h"

Intmain()
{
    uart_config(0,115200,DATABITS_8,STOPBITS_1,UART_PARNONE);

    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;
}
```

11. FILE System

이번 장에서는 Nand Flash 와 SD Card 를 file 저장장치로 사용하기 위해 초기 설정하는 법에 대해 설명을 하고, 자세한 file system 사용에 관한 내용은 FATFS 매뉴얼을 참고하기 바란다.
(http://elm-chan.org/fsw/ff/00index_e.html)

Function	Description
f_mount	Nand Flash 또는 SD Card를 mount하여 file 저장 장치로 사용하도록 설정한다.
f_chdrive	Mount 된 driver 간에 driver change 할 때 사용한다.
f_chdir	Mount 된 driver에서 directory 이동할 때 사용한다.

11.1 f_mount

```
FRESULT  f_mount (
      BYTE,
      FATFS*
);
```

Overview

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

Parameter

BYTE	NAND FLASH 또는 SD Card 의 장치 번호. 다음과 같이 정의되어 있다. #define DRIVE_NAND 0 #define DRIVE_SDCARD 1
FATFS	장치 정보를 가지고 있는 구조체.

Return Value

동작의 결과.

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

11.2 f_chdrive

```
FRESULT  f_chdrive (
    BYTE
);
```

Overview

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 참고하기 바란다.

11.3 f_chdir

```
FRESULT  f_chdir (
    const TCHAR*
);
```

Overview

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 참고하기 바란다.

Ex) f_chdir("mp3"); // mp3 directory 로 이동.
f_chdir(".."); // 한 단계 상위 디렉토리로 이동.
f_chdir("/"); // root 디렉토리로 이동.

11.4 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;
}
```

12. Font

두 종류의 font 이미지 폰트(편의상 bmp 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. bmp font 는 프로그램을 사용하여 폰트 이미지를 제작해야 하지만 여러 종류의 폰트를 쉽게 사용할 수 있다. bit font 는 따로 이미지 제작 없이 제공되는 파일을 가지고 사용할 수 있지만 폰트의 종류는 변경할 수 없다.

Function	Description
create_bmpfont	폰트 파일을 load하여 bmp font를 사용할 수 있도록 준비한다.
release_bmpfont	생성한 bmp font 중 더 이상 사용하지 않을 font를 release한다.
bmpfont_draw	bmp font를 사용하여 특정 위치에 문자를 출력한다.
bmpfont_draw_vleft	bmp font를 사용하여 좌측으로 90도 회전한 문자를 출력한다.
bmpfont_draw_vright	bmp font를 사용하여 우측으로 90도 회전한 문자를 출력한다.
egl_font_set_color	font의 색을 변경한다.
bmpfont_makesurface	bmp font를 사용한 문자열을 이미지로 만든다.
bmpfont_setkerning	bmp font의 자간을 조정한다.
bmpfont_setautokerning	bmp font의 글자 간격을 동일하게 할지를 결정한다.
create_bitfont	bit font를 사용할 수 있도록 생성한다.
release_bitfont	생성한 bit font 중 더 이상 사용하지 않을 font를 release한다.
bitfont_draw	bit font를 사용하여 문자를 출력한다.
bitfont_draw_vleft	bit font를 사용하여 좌측으로 90도 회전한 문자를 출력한다.
bitfont_draw_vright	bit font를 사용하여 우측으로 90도 회전한 문자를 출력한다.
bitfont_makesurface	bit font를 사용한 문자열을 이미지로 만든다.

<< bmp font >>

12.1 create_bmpfont

```
EGL_FONT*  create_bmpfont (
    const char  fname
);
```

Overview

폰트 파일을 load 하여 bmp font 를 사용할 수 있도록 준비한다.

Parameter

fname	폰트 파일 이름

Return Value

bmp font 포인터

12.2 release_bmpfont

```
void  release_bmpfont (
    EGL_FONT*  pFont
);
```

Overview

생성한 bmp font 중 더 이상 사용하지 않을 font 를 release 한다.

Parameter

pFont	bmp font 포인터

Return Value

없음

12.3 bmpfont_draw

```
int  bmpfont_draw (
     EGL_FONT*  pFont,
     int  x,
     int  y,
     const char* str
);
```

Overview

bmp font 를 사용하여 특정 위치에 문자를 출력한다.

Parameter

EGL_FONT* pFont	bmp font 포인터.
int x	문자가 출력 될 x 좌표.
int y	문자가 출력 될 y 좌표.
const char* str	출력 될 문자(열)

Return Value

출력 된 문자열의 width.

12.4 bmpfont_draw_vleft

```
int  bmpfont_draw_vleft (
     EGL_FONT*  pFont,
     int  x,
     int  y,
     const char* str
);
```

Overview

bmp font 를 사용하여 특정 위치에 문자를 출력하는데, 좌측으로 90 도 회전하여 출력한다.

Parameter

EGL_FONT* pFont	bmp font 포인터.
int x	문자가 출력 될 x 좌표.
int y	문자가 출력 될 y 좌표.
const char* str	출력 될 문자(열)

Return Value

출력 된 문자열의 width.

12.5 bmpfont_draw_vright

```
int  bmpfont_draw_vright (
     EGL_FONT*  pFont,
     int  x,
     int  y,
     const char* str
);
```

Overview

bmp font 를 사용하여 특정 위치에 문자를 출력하는데, 우측으로 90 도 회전하여 출력한다.

Parameter

EGL_FONT* pFont	bmp font 포인터.
int x	문자가 출력 될 x 좌표.
int y	문자가 출력 될 y 좌표.
const char* str	출력 될 문자(열)

Return Value

출력 된 문자열의 width.

12.6 egl_font_set_color

```
EGL_COLOR  egl_font_set_color (
     EGL_FONT*  pFont,
     EGL_COLOR  clr
);
```

Overview

폰트의 색을 변경한다.

Parameter

EGL_FONT* pFont	색을 변경하려는 bmp font 포인터
EGL_COLOR clr	font color. MAKE_COLORREF(r,g,b) 를 사용한다.

Return Value

이전에 설정되어 있던 color.

12.7 bmpfont_makesurface

```
SURFACE*  egl_font_set_color (
    EGL_FONT*  pFont,
    char*  text
);
```

Overview

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

Parameter

EGL_FONT* pFont	출력에 사용할 bmp font 포인터.
EGL_COLOR text	이미지로 만들 문자(열)

Return Value

SURFACE 구조체

12.8 bmpfont_setkerning

```
BOOL  bmpfont_setkerning (
    EGL_FONT*  pFont,
    int  k
);
```

Overview

bmp font 의 자간을 조정한다.

Parameter

EGL_FONT* pFont	출력에 사용할 bmp font 포인터.
int k	자간 간격

Return Value

TRUE or FALSE

12.9 bmpfont_setautokerning

```
BOOL  bmpfont_setautokerning (
      EGL_FONT*  pFont,
      bool  b
);
```

Overview

bmp font 의 글자 간격을 동일하게 할지를 결정한다.

Parameter

EGL_FONT* pFont	출력에 사용할 bmp font 포인터.
bool b	TRUE : 글자의 간격을 동일하게 맞춘다. 각 글자마다 width 가 다르기 때문에, 문자열의 전에 width 가 글자에 따라 다르다. FALSE : 글자의 간격을 글자의 width 에 따라 다르게 맞춘다. 문자열의 전체 width 는 동일하다.

Return Value

TRUE or FALSE

<< bit font >>

12.10 create_bitfont

```
EGL_FONT*  create_bitfont ( );
```

Overview

bit font 를 사용할 수 있도록 생성한다.

Parameter

없음

Return Value

bit font 포인터.

12.11 release_bitfont

```
void  release_bitfont ( );
```

Overview

생성한 bit font 중 더 이상 사용하지 않을 font 를 release 한다.

Parameter

pFont	bit font 포인터

Return Value

없음

12.12 bitfont_draw

```
int  bitfont_draw (
     EGL_FONT*  pFont,
     int  x,
     int  y,
     const char* str
);
```

Overview

bit font 를 사용하여 특정 위치에 문자를 출력한다.

Parameter

EGL_FONT* pFont	출력에 사용 할 bit font 포인터.
int x	문자가 출력 될 x 좌표.
int y	문자가 출력 될 y 좌표.
const char* str	출력 될 문자(열)

Return Value

출력 된 문자열의 width.

12.13 bitfont_draw_vleft

```
int  bitfont_draw_vleft (
     EGL_FONT*  pFont,
     int  x,
     int  y,
     const char* str
);
```

Overview

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

Parameter

EGL_FONT* pFont	출력에 사용 할 bit font 포인터.
int x	문자가 출력 될 x 좌표.
int y	문자가 출력 될 y 좌표.
const char* str	출력 될 문자(열)

Return Value

출력 된 문자열의 width.

12.14 bitfont_draw_vright

```
int  bitfont_draw_vright (
     EGL_FONT*  pFont,
     int  x,
     int  y,
     const char* str
);
```

Overview

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

Parameter

EGL_FONT* pFont	출력에 사용 할 bit font 포인터.
int x	문자가 출력 될 x 좌표.
int y	문자가 출력 될 y 좌표.
const char* str	출력 될 문자(열)

Return Value

출력 된 문자열의 width.

12.15 bitfont_makesurface

```
int  bitfont_makesurface (
     EGL_FONT*  pFont,
     char* str
);
```

Overview

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

Parameter

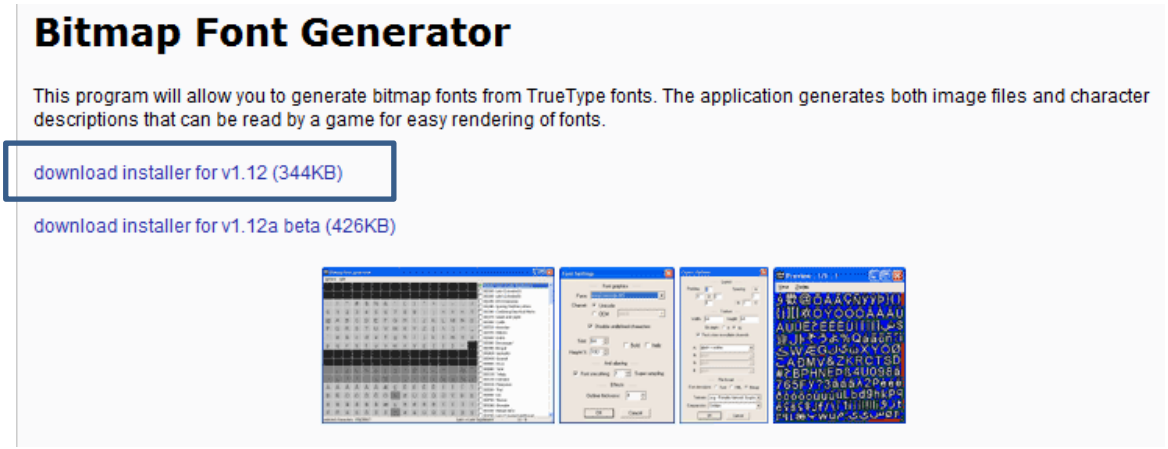
EGL_FONT* pFont	출력에 사용 할 bit font 포인터.
const char* str	출력 될 문자(열)

Return Value

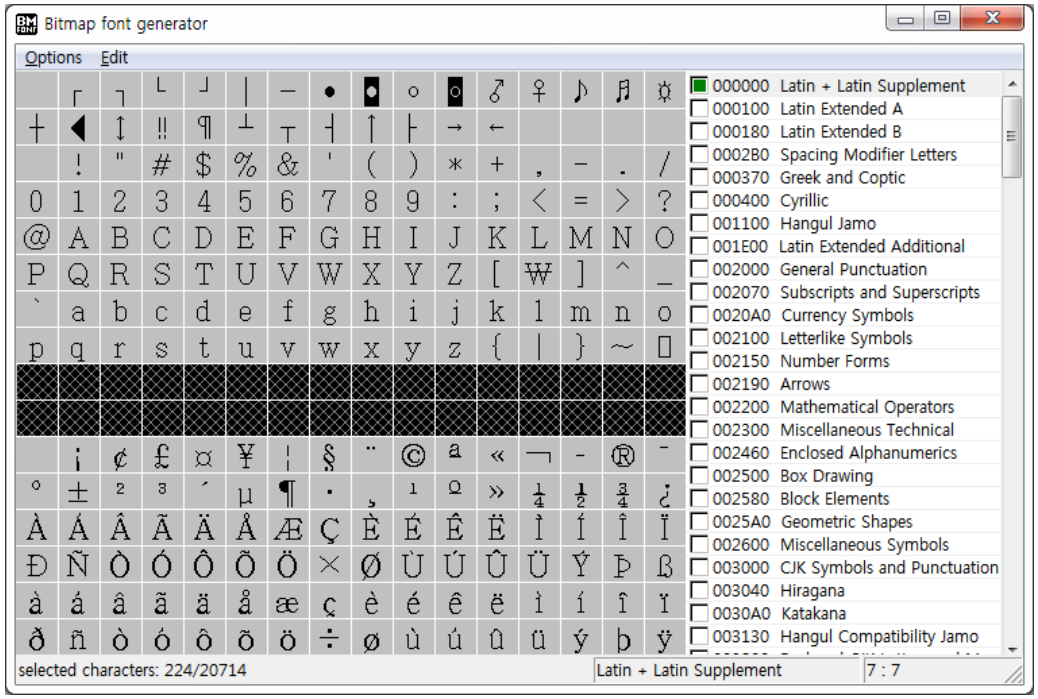
SURFACE 구조체

< font image 제작 방법 >

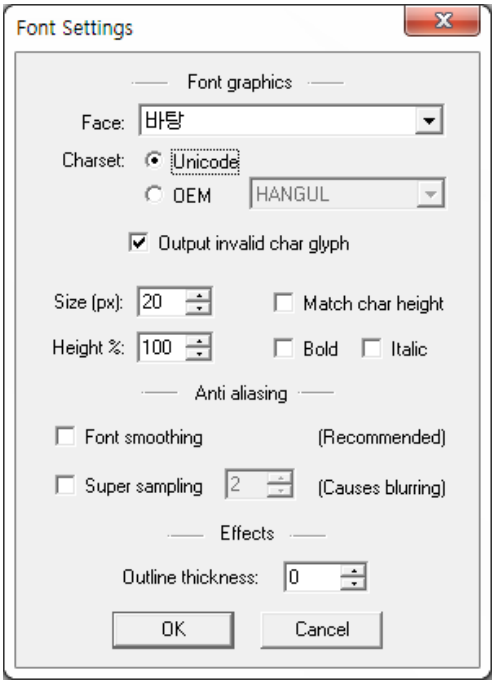
1. http://www.angelcode.com/products/bmfont 에서 Bitmap Font Generator v1.12 를 다운받아 설치한다.



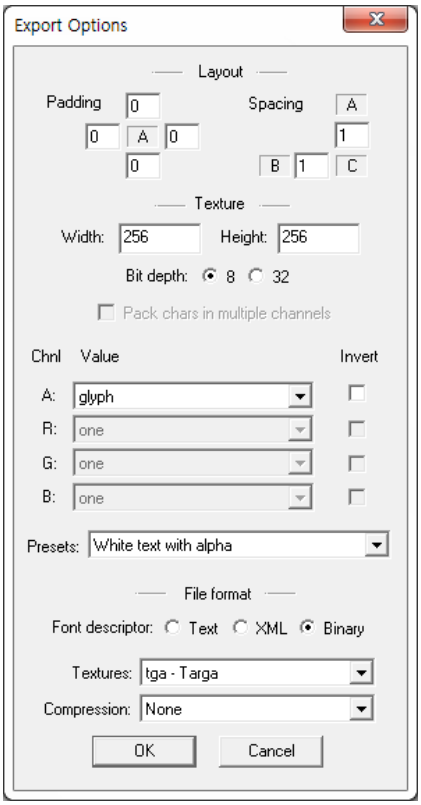
2. 설치한 후 실행하면 아래 그림과 같은 창이 나온다.



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

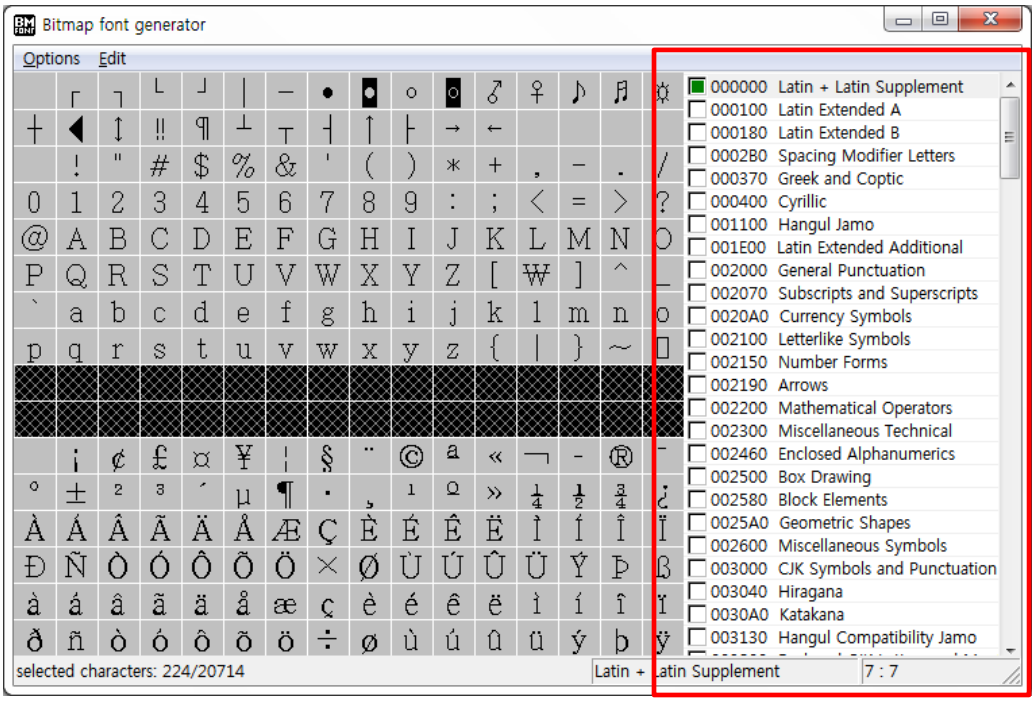


4. Options → Export options 를 실행하여 아래 그림과 같이 설정한다.



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 create_bmpfont 함수로 load 하면 된다.

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

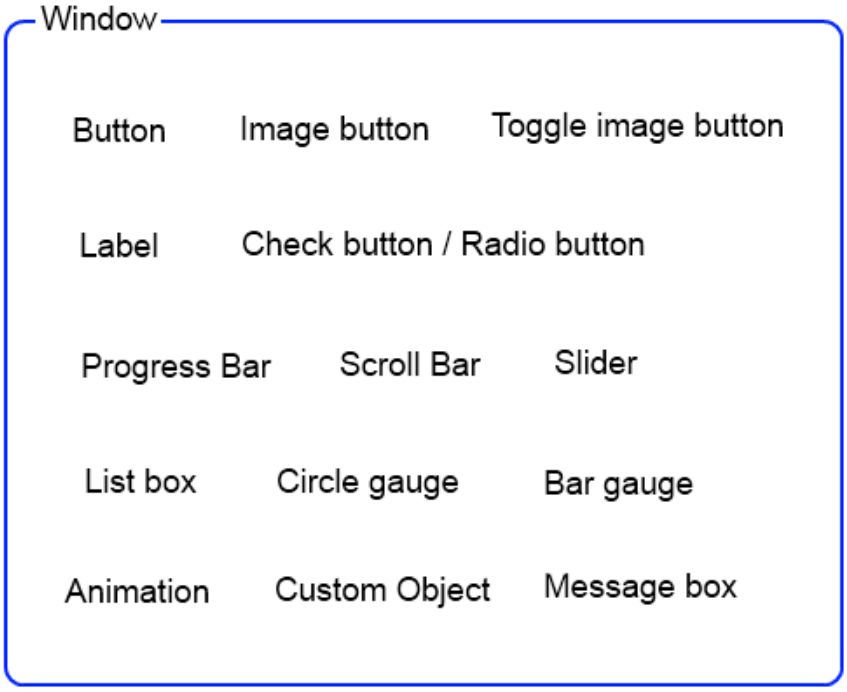
13. EGL Library

1) 소개

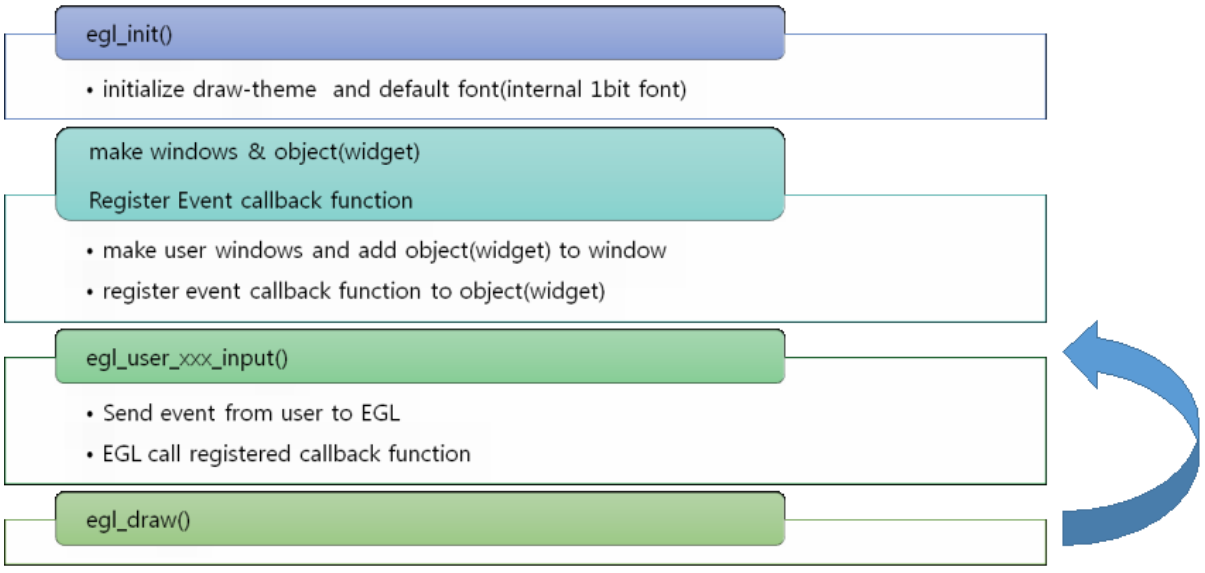
EGL(Embedded Graphic Library)은 graphic application 개발시간을 단축하고 다양한 object를 편리하게 사용할 수 있도록 제작 되었다. EGL은 다양한 object를 간편하게 사용할 수 있도록 함수를 제공하며, event에 대한 처리와 object 관리를 library내에서 하도록 되어 있다. Event에 대한 추가 동작은 각 object별 사용자 callback 함수를 두어, 개발자가 원하는 동작의 callback 함수를 작성하여 사용할 수 있다.

2) 구성

EGL은 다음의 object 들로 구성되어 있다.



3) Program 기본 구조



```
< Source Code >

#include "adstar.h"

static void btn_callback(EGL_HANDLE  h, int  event) // button callback function.
{
    /* 사용자 call back 함수 */
}

extern BOOL process_touch(BOOL*  touchdown, EGL_POINT*  pPoint);

int main()
{
...
  EGL_POINT  touch_pt;
  BOOL  touchdown = FALSE;
  EGL_HANDLE  hWin;
  EGL_HANDLE  btn;
  egl_init();
  hWin = egl_create_window("Main Window");         // create window
  btn = egl_create_button(100,100,200,50,"Button 1");  // create button obj
  egl_btn_callback(btn, btn_callback);
  egl_window_add_object(hWin, btn);                  // button obj add.
  egl_window_show(hWin);                            // window show

  while(1)
  {
       if( process_touch( &touchdown, &touch_pt ) )      // touch 입력.
       egl_user_touch_input( touchdown, &touch_pt ); // touch 처리, msg 처리.

     egl_draw();
   }
...
}
```

Window

Function	Description
egl_create_window	Window를 생성한다.
egl_window_show	해당 window를 화면에 출력할지를 결정한다.
egl_window_isshow	해당 window가 현재 보여지고 있는지를 확인한다.
egl_window_invalidate	현재 window안의 object를 다시 draw한다.
egl_window_invalidate_rect	현재 window의 지정한 영역의 object를 다시 draw한다.
egl_window_redraw_rect	현재 window의 지정한 영역을 다시 draw한다.
egl_window_set_bg	Window의 배경 이미지를 설정한다.
egl_window_get_bg	Window의 배경 이미지를 가져온다.
egl_window_add_object	Window에 object를 추가한다.
egl_window_set_callback	Window 이벤트가 발생했을 때 호출될 callback함수를 등록한다.
egl_window_get_active	현재 보이는 window의 handle값을 반환한다.
egl_user_touch_input	Msg_handler로 touch message를 전달한다.
egl_draw	현재 windows의 object를 draw한다.
egl_visible_object	Object의 visible 상태를 설정한다.
egl_window_delete_object	해당 window에서 object를 제거한다.
egl_release_window	Window를 제거한다.

▶**egl_create_window** function

```
EGL_HANDLE  egl_create_window(
      const char*  title
);
```

Overview

Window를 생성한다. Window의 크기는 사용하는 해상도와 같다.

Parameter

const char* title	Window의 title.

Return Value

생성된 Window의 handle.

Example

```
EGL_HANDLE  hWin;

hWin = egl_create_window("Main Window");
```

▶**egl_window_show** function

```
BOOL  egl_window_show(
    EGL_HANDLE  hWin,
    BOOL  bShow
);
```

Overview

해당 window를 화면에 출력할지를 결정한다.

Parameter

EGL_HANDLE hWin	출력을 결정 할 window handle.
BOOL bShow	출력 여부 결정.

Return Value

TRUE or FALSE

Example

```
EGL_HANDLE  hWin;
hWin = egl_create_window("Main Window");
...
egl_window_show(hWin, TRUE);
```

▶**egl_window_isshow** function

```
BOOL  egl_window_isshow(
    EGL_HANDLE  hWin
);
```

Overview

해당 window가 현재 보여지고 있는지를 확인한다.

Parameter

EGL_HANDLE hWin	출력을 상태를 확인 할 window handle.

Return Value

TRUE or FALSE

Example

```
EGL_HANDLE  hWin;
hWin = egl_create_window("Main Window");
…
if(egl_window_isshow(hWin))
{
…
}
```

▶**egl_window_invalidate** function

```
void  egl_window_invalidate( void );
```

Overview

현재 window안의 object를 다시 draw한다.

Parameter

없음.

Return Value

없음.

Example

```
EGL_HANDLE  hWin;
hWin = egl_create_window("Main Window");
...
egl_window_invalidate( );
```

▶**egl_window_invalidate_rect** function

```
void  egl_window_invalidate_rect(
    EGL_RECT*  pRect
);
```

Overview

현재 window의 지정한 영역의 object를 다시 draw한다.

Parameter

EGL_RECT* pRect 지정 영역 value.

```
typedef  Struct  _tag_RECT{
    int  x;
    int  y;
    int  w;
    int  h;
}EGL_RECT
```

Return Value

없음.

Example

```
EGL_HANDLE  hWin;
EGL_RECT pRect;
hWin = egl_create_window("Main Window");
…
pRect.x = 100;
pRect.y = 200;
pRect.w = 150;
pRect.h = 150;
egl_window_invalidate_rect(&pRect );
```

▶**egl_window_redraw_rect** function

```
Void  egl_window_redraw_rect(
     EGL_RECT*  pRect
);
```

Overview

현재 window의 지정한 영역을 다시 draw한다.

Parameter

EGL_RECT* pRect 지정 영역 value.

```
typedef  Struct  _tag_RECT{
    int  x;
    int  y;
    int  w;
    int  h;
}EGL_RECT
```

Return Value

없음.

Example

```
EGL_HANDLE hWin;
EGL_RECT pRect;
hWin = egl_create_window("Main Window");
...
pRect.x = 100;
pRect.y = 200;
pRect.w = 150;
pRect.h = 150;
egl_window_redraw_rect(&pRect);
```

▶**egl_window_set_bg** function

```
BOOL  egl_window_set_bg(
    EGL_HANDLE  hWin;
    SURFACE*  Img;
);
```

Overview

Window의 배경 이미지를 설정한다.

Parameter

EGL_HANDLE hWin	배경 이미지를 설정할 Window handle.
SURFACE* Img	배경 이미지 SURFACE

Return Value

TRUE or FALSE

Example

```
EGL_HANDLE hWin;
SURFACE* surface = loadbmp("windowbg.bmp");
hWin = egl_create_window("Main Window");
egl_window_set_bg(hWin, surface);
```

▶**egl_window_get_bg** function

```
SURFACE*  egl_window_get_bg(
    EGL_HANDLE  hWin;
);
```

Overview

Window의 배경 이미지를 가져온다.

Parameter

EGL_HANDLE hWin	배경 이미지를 가져 올 Window handle.

Return Value

해당 window의 image.

Example

```
EGL_HANDLE hWin;
SURFACE* getimage;
hWin = egl_create_window("Main Window");
...
getimage = egl_window_get_bg(hWin);
```

▶**egl_window_add_object** function

```
BOOL  egl_window_add_object(
    EGL_HANDLE  hWin,
    EGL_HANDLE  hObj
);
```

Overview

Window에 object를 추가한다.

Parameter

EGL_HANDLE hWin	Object를 추가 할 window handle.
EGL_HANDLE hObj	Window에 추가 할 object handle.

Return Value

TRUE or FALSE

Example

```
EGL_HANDLE hWin;
EGL_HANDLE btn;
hWin = egl_create_window("Main Window");
btn = egl_create_button(100, 100, 100, 50, "Button");
egl_window_add_object(hWin, btn);
```

▶**egl_window_set_callback** function

```
void  egl_window_set_callback(
    EGL_HANDLE  hWin,
    EVENT_CALLBACK  cb
);
```

Overview

Window 이벤트가 발생했을 때 호출 되는 callback함수를 등록한다.

Parameter

EGL_HANDLE hWin	Callback 함수가 등록되는 Window handle.
EVENT_CALLBACK cb	Event 발생 시 호출되는 callback 함수.

Return Value

없음.

Example

```
Void window_callback(EGL_HANDLE h, int event)
{
…
}
int main()
{
   …
   EGL_HANDLE hWin;
   hWin = egl_create_window("Main Window");
   egl_window_callback(hWin, window_callback);
   …
}
```

▶**egl_window_get_active** function

```
EGL_HANDLE  egl_window_get_active( void );
```

Overview

현재 출력되고 있는 window의 handle값을 반환한다.

Parameter

없음.

Return Value

현재 출력되고 있는 window handle.

Example

```
EGL_HANDLE hWin;
EGL_HANDLE active_win;
hWin = egl_create_window("Main Window");
...
active_win = egl_window_get_active( );
```

▶**egl_user_touch_input** function

```
void  egl_user_touch_input(
    BOOL  bPressed,
    EGL_POINT*  pt
);
```

Overview

Msg_handler로 touch message를 전달한다.

Parameter

BOOL bPressed	Touch가 눌린 상태.
EGL_POINT* pt	Touch 된 좌표.

```
typedef struct _tagPOINT
{
    int x;
    int y;
} EGL_POINT;
```

Return Value

없음.

Example

```
EGL_POINT touch_pt;
BOOL touchdown = FALSE;
...
if(process_touch(&touchdown, &touch_pt))
   egl_user_touch_input(touchdown, &touch_pt);
```

▶**egl_draw** function

```
void  egl_draw( void );
```

Overview

현재 windows의 object를 draw한다.

Parameter

없음.

Return Value

없음.

Example

```
if(process_touch(&touchdown, &touch_pt))
    egl_user_touch_input(touchdown, &touch_pt);
egl_draw();
```

▶**egl_visible_object** function

```
void  egl_visible_object (
   EGL_HANDLE  h,
   bool b
);
```

Overview

Object의 visible 상태를 설정한다.

Parameter

EGL_HANDLE h	Visible 상태를 변경 할 object handle.
bool b	Visible 상태, TRUE : visible, FALSE : invisible

Return Value

없음.

Example

```
egl_visible_object( button, FALSE );
//egl_visible_object(button, TRUE);
…
egl_draw( );
```

▶**egl_window_delete_object** function

```
bool  egl_window_delete_object (
   EGL_HANDLE  hWin,
   EGL_HANDLE  hObj
);
```

Overview

해당 window의 object를 제거한다. Window object list에서만 제거된다. Object를 메모리 상에서 완전히 제거하기 위해서는 release해주어야 한다.

Parameter

EGL_HANDLE hWin	Object가 포함 된 window handle.
EGL_HANDLE hObj	제거 할 object handle.

Return Value

TRUE or FALSE.

Example

```
egl_window_delete_object(hWin, button);
...
egl_draw( );
```

▶**egl_release_window** function

```
bool  egl_release_window (
   EGL_HANDLE  hObj
);
```

Overview

해당 window를 제거한다.

Parameter

EGL_HANDLE hObj 제거 할 window handle.

Return Value

TRUE or FALSE.

Example

```
if(hWin != NULL)
{
    egl_release_window(hWin);
    hWin = NULL;
}
```

▶ Window Example.

Example

```
#include "adstar.h"

static void btn1_callback(EGL_HANDLE h, int event)
{
     if( event == BTN_CLICKED )
     {
         debugprintf("btn1 clicked₩r₩n");
         egl_window_show( hWin2, TRUE );
       }
}

static void btn2_callback(EGL_HANDLE h, int event)
{
       if( event == BTN_CLICKED )
     {
         debugprintf("btn2 clicked₩r₩n");
         egl_window_show( hWin1, TRUE );
       }
}

extern BOOL process_touch(BOOL* touchdown, EGL_POINT* pPoint);

int main()
{
    ...
   EGL_POINT touch_pt;
   BOOL touchdown = FALSE;
   EGL_HANDLE hWin1;
   EGL_HANDLE hWin2;
   EGL_HANDLE btn1;
    EGL_HANDLE btn2;
    SURFACE* bg = loadbmp("bg.bmp");
    egl_init();
   hWin1 = egl_create_window("First Window");
   egl_windows_set_bg(hWin1, bg);
    hWin2 = egl_create_window("Second Window");
   btn1 = egl_create_button(100, 100, 200, 50, "Second Window");
```

```
    egl_btn_callback(btn1, btn1_callback);
  btn2 = egl_create_button(100, 200, 200, 50, "First Window");
    egl_btn_callback(btn2, btn2_callback);
  egl_window_add_object(hWin1, btn1);
  egl_window_add_object(hWin2, btn2);
  egl_window_show(hWin1);
   egl_draw();

  while(1)
  {
       if( process_touch(&touchdown, &touch_pt ) )
       egl_user_touch_input(touchdown, &touch_pt );

     egl_draw();
  }
}
```

Button

Function	Description
egl_create_button	Button을 생성한다.
egl_button_callback	Button 이벤트가 발생했을 때 호출될 callback 함수를 등록한다.
egl_release_button	Button을 제거한다.

Define

BUTTON_EVENT

```
typedef enum
{
   BTN_CLICKED = 0,
   BTN_PRESSED,
   BTN_MAX,
}BUTTON_EVENT
```

▶**egl_create_button** function

```
EGL_HANDLE   egl_create_button(
      int   x,
      int   y,
      int   w,
      int   h,
      const char*   text
);
```

Overview

Button을 생성한다.

Parameter

int x	생성될 Button의 x좌표.
int y	생성될 Button의 y좌표.
int w	생성될 Button의 가로 크기.
int h	생성될 Button의 세로 크기.
const char* text	Button에 출력될 text.

Return Value

생성된 Button의 handle.

Example

```
EGL_HANDLE btn[2];

btn[0] = egl_create_button(100, 100, 100, 50, "Button1");

btn[1] = egl_create_button(300, 100, 100, 50, "Button2");
```

▶**egl_button_callback** function

```
BOOL  egl_button_callback(
      EGL_HANDLE  hObj,
      EVENT_CALLBACK  cb
);
```

Overview

Button 이벤트가 발생했을 때 호출될 callback 함수를 등록한다.

Parameter

EGL_HANDLE hObj	Callback 함수가 등록되는 Button handle.
EVENT_CALLBACK cb	Event가 발생 시 호출되는 callback 함수.

Return Value

TRUE or FALSE

Example

```
void btn_callback(EGL_HANDLE h, int event)
{
If(event == BTN_CLICKED)
      {
            …
    }
}
int main()
{
   …
   EGL_HANDLE btn;
   btn = egl_create_button(100, 100, 100, 50, “Button1”);
   egl_button_callback(btn,btn_callback);
   …
}
```

▶**egl_release_button** function

```
BOOL  egl_release_button(
      EGL_HANDLE  hObj,
);
```

Overview

Button을 제거한다. 해당 object가 window에 add 되어 있을 경우, egl_window_delete_object 함수로 window에서 object를 제거한 후 release 해주어야 한다.

Parameter

EGL_HANDLE hObj	제거 할 Button handle.

Return Value

TRUE or FALSE

Example

```
if(button != NULL)
{
    egl_window_delete_object(hWin, button);
    egl_release_button(button);
    button = NULL;
}
```

▶ Button Example.

Example

```
void btn_callback(EGL_HANDLE h, int event)
{
      if(event == BTN_CLICKED)
      {
          debugprintf("buttton clicked.");
     }
}

extern BOOL process_touch(BOOL* touchdown, EGL_POINT* pPoint);

int main()
{
   …
    EGL_POINT touch_pt;
  BOOL touchdown = FALSE;
  EGL_HANDLE hWin;
  EGL_HANDLE btn;
  egl_init();
    btn = egl_create_button(100, 100, 100, 50, "Button Ex");
    egl_button_callback(btn,btn_callback);
  egl_window_add_object(hWin, btn);
  egl_window_show(hWin);
   egl_draw();

while(1)
{
      if( process_touch( &touchdown, &touch_pt ) )
     egl_user_touch_input( touchdown, &touch_pt );

    egl_draw();
 }
}
```

Image button

< img >

< pressed_img >

Function	Description
egl_create_image_button	Image button을 생성한다.
egl_image_button_callback	Image button 이벤트가 발생했을 때 호출될 callback 함수를 등록한다
egl_release_image_button	Image button을 제거한다.

▶**egl_create_image_button** function

```
EGL_HANDLE  egl_create_image_button(
    SURFACE*  img,
    SURFACE*  pressed_img,
    int  x,
    int  y,
    int  w,
    int  h
);
```

Overview

Image button을 생성한다.

Parameter

SURFACE* img	Image button의 표시 될 image.
SURFACE* pressed_img	눌렸을 때 표시 될 image.
int x	Image button의 x좌표.
int y	Image button의 y좌표.
int w	Image button의 가로 크기.
int h	Image button의 세로 크기.

Return Value

생성 된 Image button handle.

Example

```
EGL_HADLE imagebtn;
SURFACE* img = loadbmp("btnimg.bmp");;
SURFACE* pressed_img = loadbmp("pressedimg.bmp");
imagebtn = egl_create_image_button(img, pressed_img, 100, 100, 150, 50);
```

▶**egl_image_button_callback** function

```
BOOL  egl_image_button_callback(
    EGL_HANDLE  hObj,
     EVENT_CALLBACK  cb
);
```

Overview

Image button 이벤트가 발생했을 때 호출될 callback 함수를 등록한다

Parameter

EGL_HADLE hObj	Callback 함수가 등록되는 Image button handle.
EVENT_CALLBACK cb	Event가 발생 시 호출되는 callback 함수.

Return Value

TRUE or FALSE

Example

```
void imgbtn_callback(EGL_HANDLE h, int event)
{
      …
}
int main()
{
   …
   EGL_HADLE imagebtn;
   SURFACE* img = loadbmp(“btnimg.bmp”);;
   SURFACE* pressed_img = loadbmp(“pressedimg.bmp”);
   imagebtn = egl_create_image_button(img, pressed_img, 100, 100, 150, 50);
   egl_image_button_callback(imgbtn, imgbtn_callback);
   …
}
```

▶**egl_release_image_button** function

```
BOOL  egl_release_image_button(
    EGL_HANDLE  hObj
);
```

Overview

Image button을 제거한다. 해당 object가 window에 add 되어 있는 경우, egl_window_delete_object 함수로 window에서 object를 제거한 후 release해주어야 한다.

Parameter

EGL_HADLE hObj	제거 할 Image button handle.

Return Value

TRUE or FALSE

Example

```
if(imgbtn != NULL)
{
    egl_window_delete_object(hWin, imgbtn);
    egl_release_image_button(imgbtn);
    imgbtn = NULL;
}
```

▶ Image Button Example.

Example

```
void imgbtn_callback(EGL_HANDLE h, int event)
{
      if(event == BTN_CLICKED)
      {
          debugprintf("image buttton clicked.");
     }
}

extern BOOL process_touch(BOOL* touchdown, EGL_POINT* pPoint);

int main()
{
   ...
   EGL_POINT touch_pt;
  BOOL  touchdown = FALSE;
  EGL_HANDLE hWin;
  EGL_HANDLE imgbtn;
   SURFACE* img = loadbmp("btnimg.bmp");;
   SURFACE* pressed_img = loadbmp("pressedimg.bmp");
   egl_init();
   imgbtn = egl_create_image_button(img, pressed_img, 100, 100, 150, 50);
   egl_button_callback(imgbtn, imgbtn_callback);
  egl_window_add_object(hWin, imgbtn);
  egl_window_show(hWin);
   egl_draw();

while(1)
{
      if( process_touch( &touchdown, &touch_pt ) )
      egl_user_touch_input( touchdown, &touch_pt );

   egl_draw();
 }
}
```

Toggle image button

< surf_on >

< surf_off >

Function	Description
egl_create_toggle_image	Toggle image button을 생성한다.
egl_toggle_image_callback	Toggle image button 이벤트가 발생했을 때 호출될 callback 함수를 등록한다.
egl_toggle_image_set_on	Toggle image button의 상태를 설정한다.
egl_release_toggle_image	Toggle image를 제거한다.

Define

TOGGLE_IMAGE_EVENT

```
typedef enum
{
    TOGGLE_IMAGE_ON = 0,
    TOGGLE_IMAGE_OFF,
} TOGGLE_IMAGE_EVENT;
```

▶**egl_create_toggle_image** function

```
EGL_HANDLE  egl_create_toggle_image(
    SURFACE*  surf_off,
    SURFACE*  surf_on,
    int  x,
    int  y,
    int  w,
    int  h
);
```

Overview

Toggle image button을 생성한다.

Parameter

SURFACE* surf_off	Toggle imgae 중 off시에 출력 할 image. (default상태 : 0ff)
SURFACE* surf_on	Toggle image 중 on시에 출력 할 image.
int x	Toggle image의 x 좌표.
int y	Toggle image의 y 좌표.
int w	Toggle image의 가로 크기.
int h	Toggle image의 세로 크기.

Return Value

생성 된 toggle image handle.

Example

```
EGL_HADLE toggleimg;
SURFACE* surf_off = loadpng("off.png");;
SURFACE* surf_on = loadpng("on.png");
toggleimg = egl_create_toggle_image(surf_off, surf_on, 100, 100, 150, 50);
```

▶**egl_toggle_image_callback** function

```
BOOL  egl_toggle_image_callback(
    EGL_HANDLE  hObj,
    EVENT_CALLBACK  cb
);
```

Overview

Toggle image button 이벤트가 발생했을 때 호출될 callback 함수를 등록한다

Parameter

EGL_HANDLE hObj	Callback 함수가 등록되는 toggle image handle.
EVENT_CALLBACK cb	Event가 발생 시 호출되는 callback 함수.

Return Value

TRUE or FALSE

Example

```
void toggleimg_callback(EGL_HANDLE h, int event)
{
   if(event == TOGGLE_IMAGE_ON)
            …
    else
            …
}
int main()
{
   EGL_HADLE toggleimage;
   …
   toggleimage = egl_create_toggle_image(surf_off, surf_on, 100, 100, 150, 50);
   egl_toggle_image_callback(toggleimage, toggleimg_callback);
   …
}
```

▶**egl_toggle_image_set_on** function

```
BOOL  egl_toggle_image_set_on(
    EGL_HANDLE  hObj,
    BOOL  b
);
```

Overview

Toggle image button의 상태를 설정한다.

Parameter

EGL_HANDLE hObj	상태를 설정 할 toggle image handle.
BOOL b	설정 값. TRUE => On FALSE => Off

Return Value

TRUE or FALSE

Example

```
EGL_HADLE toggleimg;
SURFACE* surf_off = loadpng("off.png");;
SURFACE* surf_on = loadpng("on.png");
toggleimg = egl_create_toggle_image(surf_off, surf_on, 100, 100, 150, 50);
egl_toggle_image_set_on(toggleimg, TRUE);
```

▶**egl_release_toggle_image** function

```
BOOL  egl_release_toggle_image(
    EGL_HANDLE  hObj
);
```

Overview

Toggle image button을 제거한다. 해당 object가 window에 add 되어 있는 경우,
egl_window_delete_object 함수로 window에서 object를 제거한 후 release 해주어야 한다.

Parameter

EGL_HANDLE hObj	제거 할 toggle image button handle.

Return Value

TRUE or FALSE

Example

```
if(toggle != NULL)
{
    egl_window_delete_object(hWin, toggle);
    egl_release_toggle_image(toggle);
    toggle = NULL;
}
```

▶ Toggle Image Button Example.

Example

```
void toggleimg_callback(EGL_HANDLE h, int event)
{
    if(event == TOGGLE_IMAGE_ON)
            debugprintf(" toggle image on ₩r₩n");
    else
          debugprintf(" toggle image off ₩r₩n");
}

extern BOOL process_touch(BOOL* touchdown, EGL_POINT* pPoint);

int main()
{
    ...
    EGL_POINT touch_pt;
  BOOL  touchdown = FALSE;
  EGL_HANDLE  hWin;
   EGL_HANDLE  toggleimg;
    SURFACE* surf_off = loadpng("off.png");;
    SURFACE* surf_on = loadpng("on.png");
    egl_init();
    toggleimg = egl_create_toggle_image(surf_off, surf_on, 100, 100, 150, 50);
    egl_toggle_image_callback(toggleimg, toggleimg_callback);
    egl_toggle_image_set_on(toggleimg, TRUE);
   egl_window_add_object(hWin, toggleimg);
   egl_window_show(hWin);
    egl_draw();

while(1)
{
      if( process_touch( &touchdown, &touch_pt ) )
      egl_user_touch_input( touchdown, &touch_pt );

    egl_draw();
  }
}
```

Label

Function	Description
egl_create_label	Label object를 생성한다.
egl_label_callback	Lavel 이벤트가 발생했을 때 호출 될 callback 함수를 등록한다.
egl_label_set_text	Label text를 설정한다.
egl_label_set_redraw_bg	Label 배경을 draw할지를 결정한다.
egl_label_set_color	Label text color를 설정한다.
egl_release_label	Label을 제거한다.

▶**egl_create_label** function

```
EGL_HANDLE   egl_create_label(
    int  x,
    int  y,
    int  w,
    int  h,
    const char* text
);
```

Overview

Label object를 생성한다.

Parameter

int x	label의 x 좌표.
int y	label의 y 좌표.
int w	label의 가로 크기.
int h	label의 세로 크기.
const char* text	label의 text.

Return Value

생성 된 label handle.

Example

```
EGL_HADLE label;
label = egl_create_label( 100, 100, 150, 50, "label test");
```

▶**egl_label_callback** function

```
BOOL  egl_label_callback(
    EGL_HANDLE  hObj,
    EVENT_CALLBACK  cb
);
```

Overview

Label 이벤트가 발생했을 때 호출 될 callback 함수를 등록한다.

Parameter

EGL_HANDLE hObj	Callback 함수가 등록되는 label handle.
EVENT_CALLBACK cb	Event가 발생 시 호출되는 callback 함수

Return Value

TRUE or FALSE

Example

```
void label_callback(EGL_HANDLE  h, int  event)
{
    If(event == LABEL_CLICKED)
    {
       ....
    }
}

int main()
{
    EGL_HANDLE  label;
    ....
    label = egl_create_label(100,100,150,50,"label");
    egl_label_callback(label, label_callback);
    ...
}
```

▶**egl_label_set_text** function

```
BOOL  egl_label_set_text(
    EGL_HANDLE  h,
char*  text
);
```

Overview

Label의 text를 변경한다.

Parameter

EGL_HANDLE h	Text를 변경 할 label handle.
char text	변경 할 text.

Return Value

TRUE or FALSE

Example

```
EGL_HADLE label;
label = egl_create_label( 100, 100, 150, 50, "label test" );
egl_label_set_text( label, "change text");
```

▶**egl_label_set_redraw_bg** function

```
void  egl_label_set_redraw_bg(
    EGL_HANDLE  h,
    BOOL  b
);
```

Overview

Label 배경을 draw할지를 결정한다. label을 단독 object로 사용할 경우 TRUE로 설정해야 한다.

Parameter

EGL_HANDLE hObj	상태를 설정 할 label handle.
BOOL b	설정 값. TRUE or FALSE

Return Value

없음.

Example

```
EGL_HADLE label;
label = egl_create_label( 100, 100, 150, 50, "label test" );
egl_label_set_redraw_bg(label, TRUE);
```

▶**egl_label_set_color** function

```
void  egl_label_set_color(
    EGL_HANDLE  h,
    EGL_COLOR  clr
);
```

Overview

Label text color를 설정한다.

Parameter

EGL_HANDLE h	Color를 변경 할 label handle.
EGL_COLOR clr	Color value. MAKE_COLORREF(r, g, b) 매크로 함수를 사용하면 color 지정을 쉽게 할 수 있다.

Return Value

없음.

Example

```
EGL_HADLE label;
label = egl_create_label( 100, 100, 150, 50, "label test" );
egl_label_set_color( label, MAKE_COLORREF( 0, 255, 0 ));
```

▶**egl_release_label** function

```
BOOL  egl_label_set_color(
    EGL_HANDLE  hObj
);
```

Overview

Label을 제거한다. 해당 object가 window에 add 되어 있는 경우, egl_window_delete_object 함수로 window에서 object를 제거한 후 release해주어야 한다.

Parameter

EGL_HANDLE hObj	제거 할 label handle.

Return Value

TRUE or FALSE

Example

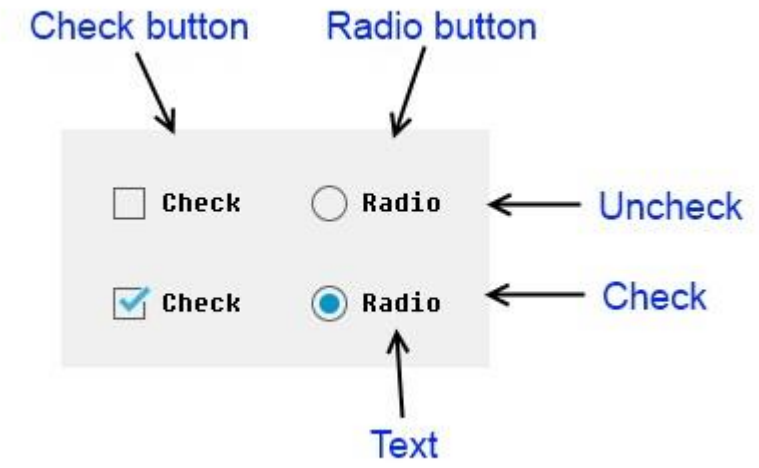
```
if(label != NULL)
{
    egl_window_delete_object(hWin, label);
    egl_release_label(label);
    label = NULL;
}
```

▶ Label Example.

Example

```
extern BOOL process_touch(BOOL* touchdown, EGL_POINT* pPoint);

int main()
{
    ...
    EGL_POINT touch_pt;
  BOOL touchdown = FALSE;
  EGL_HANDLE hWin;
   EGL_HANDLE label;
  egl_init();
    label = egl_create_label(50, 50, 100, 30, "label");
    egl_label_set_redraw_bg(label, TRUE);
    egl_label_set_color(label, MAKE_COLORREF(0, 0, 255);
    egl_window_add_object(hWin, label);
  egl_window_show(hWin);
   egl_draw();

while(1)
{
      if( process_touch( &touchdown, &touch_pt ) )
      egl_user_touch_input( touchdown, &touch_pt );

    egl_draw();
  }
}
```

Check button / Radio button



Function	Description
egl_create_checkbutton	Check / Radio button을 생성한다.
egl_checkbutton_callback	Check / Radio button 이벤트가 발생했을 때 호출될 callback 함수를 등록한다.
egl_checkbutton_set_check	Check / Radio button을 check / uncheck 상태를 설정한다.
egl_checkbutton_get_check	Check / Radio button의 현재 check 상태 값을 가져온다.
egl_checkbutton_set_style	Check / Radio button의 style을 변경한다.
egl_release_checkbutton	Check / Radio button을 제거한다.

Define

CHECK_EVENT

```
typedef enum
{
    CHECK_CHECKED = 0,
    CHECK_UNCHECKED,,
} CHECK_EVENT;
```

▶**egl_create_checkbutton** function

```
EGL_HANDLE   egl_create_checkbuttonk(
      int   x,
      int   y,
      int   w,
      int   h,
      const char*   text,
      CHECK_STYLE   style
);
```

Overview

Check / Radio button을 생성한다.

Parameter

int x	생성될 Check / Radio button의 x좌표.
int y	생성될 Check / Radio button의 y좌표.
int w	생성될 Check / Radio button의 가로 크기.
int h	생성될 Check / Radio button의 세로 크기.
const char* text	Check / Radio button에 출력될 text.
CHECK_STLE style	Check / Radio button의 style. Check button ->CHECK_STYLE_CHECKBUTTON. Radio button ->CHECK_STYLE_RADIOBUTTON.

Return Value

생성된 Check / Radio button의 handle.

Example

```
EGL_HANDLE check;
EGL_HANDLE radio;
check = egl_create_checkbutton(100,100, 100, 32, "Check", CHECK_STYLE_CHECKBUTTON);
radio = egl_create_checkbutton(250,  100, 100, 32, "Radio", CHECK_STYLE_RADIOBUTTON);
```

▶**egl_checkbutton_callback** function

```
BOOL  egl_button_callback(
      EGL_HANDLE  hObj,
      EVENT_CALLBACK  cb
);
```

Overview

Check / Radio button 이벤트가 발생했을 때 호출될 callback 함수를 등록한다.

Parameter

EGL_HANDLE hObj	Callback 함수가 등록되는 Check / Radio button handle.
EVENT_CALLBACK cb	Event 발생 시 호출되는 callback 함수.

Return Value

TRUE or FALSE

Example

```
Void check_callback(EGL_HANDLE h, int event)
{
if(event == CHECK_CHECKED)
…
     else if(event == CHECK_UNCHECKED)
…
}
int main()
{
   …
   EGL_HANDLE check;
   check = egl_create_checkbutton(100, 100, 100, 32, "Check", STYLE_CHECKBUTTON);
   egl_checkbutton_callback(check, check_callback);
   …
}
```

▶**egl_checkbutton_set_check** function

```
void egl_checkbutton_set_check(
    EGL_HANDLE  hObj,
    BOOL  b
);
```

Overview

Check / Radio button을 check / uncheck 상태를 설정한다.

Parameter

EGL_HANDLE hObj	Check 상태를 설정할 Check / Radio button handle.
BOOL b	TRUE = Check. FALSE = Uncheck.

Return Value

없음.

Example

```
EGL_HANDLE check;

EGL_HANDLE radio;

check = egl_create_checkbutton(100, 100, 100, 32, "Check", STYLE_CHECKBUTTON);

radio = egl_create_checkbutton(200, 100, 100, 32, "Radio", STYLE_RADIOBUTTON);

egl_checkbutton_set_check(check, TRUE);

egl_checkbutton_set_check(radio, FASLE);
```

▶**egl_checkbutton_get_check** function

```
BOOL egl_checkbutton_get_check(
    EGL_HANDLE  hObj,
);
```

Overview

Check / Radio button의 현재 check 상태 값을 가져온다.

Parameter

EGL_HANDLE hObj	Check 상태를 가지고 올 Check / Radio button handle.

Return Value

Check 상태를 반환. TRUE = check.
FALSE = uncheck.

Example

```
EGL_HANDLE check;

EGL_HANDLE radio;

BOOL check_status = 0;

BOOL radio_status = 0;

check = egl_create_checkbutton(100, 100, 100, 32, "Check", STYLE_CHECKBUTTON);

radio = egl_create_checkbutton(200, 100, 100, 32, "Radio", STYLE_RADIOBUTTON);

check_status = egl_checkbutton_get_check(check);

radio_status = egl_checkbutton_get_check(radio);
```

▶**egl_checkbutton_set_style** function

```
void egl_checkbutton_set_style(
    EGL_HANDLE  hObj,
     CHECK_STYLE  style
);
```

Overview

Check / Radio button의 style을 변경한다.

Parameter

EGL_HANDLE hObj	Style을 변경할 Check / Radio button handle.
CHECK_STYLE style	Check / Radio button의 style. Check button ->CHECK_STYLE_CHECKBUTTON. Radio button ->CHECK_STYLE_RADIOBUTTON.

Return Value

없음

Example

```
EGL_HANDLE check;

EGL_HANDLE radio;

check = egl_create_checkbutton(100,  100, 100, 32, "Check", CHECK_STYLE_CHECKBUTTON);

radio = egl_create_checkbutton(200, 100, 100, 32, "Radio", CHECK_STYLE_RADIOBUTTON);

egl_checkbutton_set_style(check, CHECK_STYLE_RADIOBUTTON);

egl_checkbutton_set_style(radio, CHECK_STYLE_CHECKBUTTON);
```

▶**egl_release_checkbutton** function

```
BOOL egl_release_checkbutton (
    EGL_HANDLE  hObj
);
```

Overview

Check / Radio button을 제거한다. 해당 object가 window에 add 되어 있는 경우,
egl_window_delete_object 함수로 window에서 object를 제거한 후 release 해주어야 한다.

Parameter

EGL_HANDLE hObj	제거 할 Check / Radio button의 handle.

Return Value

TRUE or FALSE

Example

```
if(check != NULL)
{
    egl_window_delete_object(hWin, check);
    egl_release_checkbutton(check);
    check = NULL;
}
```

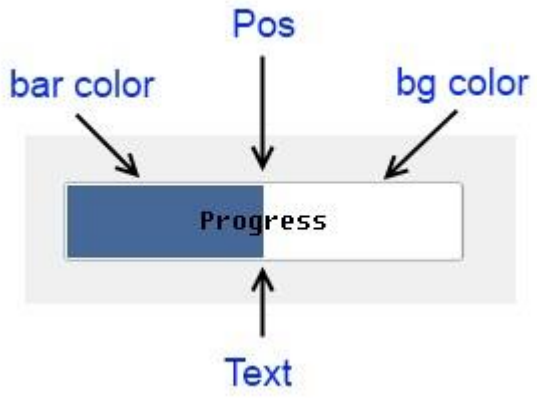
▶ Check / Radio button Example.

Example

```
 EGL_HANDLE check;
 EGL_HANDLE radio1;
 EGL_HANDLE radio2;

void check_callback(EGL_HANDLE h, int event)
{
    if(event == CHECK_CHECKED)
    {
        if(egl_checkbutton_get_check())
        {
            debugprintf("status : checked₩r₩n");
        }
        else
        {
            debugprintf("status : unchecked₩r₩n");
           }
     }
}

void radio1_callback(EGL_HANDLE h, int event)
{
    if(event == CHECK_CHECKED)
    {
        debugprintf("radio1 select₩r₩n");
        egl_checkbutton_set_check(radio2, FALSE);
     }
}

void radio2_callback(EGL_HANDLE h, int event)
{
    if(event == CHECK_CHECKED)
    {
        debugprintf("radio2 select₩r₩n");
        egl_checkbutton_set_check(radio1, FALSE);
     }
```

```
}

extern BOOL process_touch(BOOL* touchdown, EGL_POINT* pPoint);

int main()
{
   ...
   EGL_POINT touch_pt;
  BOOL touchdown = FALSE;
  EGL_HANDLE hWin;

   egl_init();
    check = egl_create_checkbutton(100, 100, 100, 32, "Check", CHECK_STYLE_CHECKBUTTON);
   egl_checkbutton_callback(check, check_callback);
    radio1 = egl_create_checkbutton(250, 100, 100, 32, "Radio1", CHECK_STYLE_RADIOBUTTON);
   egl_checkbutton_callback(radio1, radio1_callback);
    radio2 = egl_create_checkbutton(250, 160, 100, 32, "Radio2", CHECK_STYLE_RADIOBUTTON);
   egl_checkbutton_callback(radio2, radio2_callback);
    egl_window_add_object(hWin, check);
    egl_window_add_object(hWin, radio1);
    egl_window_add_object(hWin, radio2);
   egl_window_show(hWin);
    egl_draw();

while(1)
{
      if( process_touch( &touchdown, &touch_pt ) )
      egl_user_touch_input( touchdown, &touch_pt );

    egl_draw();
 }
}
```

Progress Bar



Function	Description
egl_create_progressbar	Progress bar를 생성한다.
egl_progressbar_set_barcolor	Progress bar의 bar color를 설정한다.
egl_progressbar_set_bgcolor	Progress bar의 back ground color를 설정한다.
egl_progressbar_set_textcolor	Progress bar의 text color를 설정한다.
egl_progressbar_set_text	Progress bar의 text를 설정한다.
egl_progressbar_set_pos	Progress bar의 bar position을 설정한다.
egl_progressbar_get_pos	Progress bar의 현재 bar position value를 가져온다.
egl_release_progressbar	Progress bar를 제거한다.

▶**egl_create_progressbar** function

```
EGL_HANDLE  egl_create_progressbar(
    int  x,
    int  y,
    int  w,
    int  h,
    const  char* text,
   BOOL  style
   BOOL  bVertical
);
```

Overview

Progress bar를 생성한다.

Parameter

int x	생성될 Progress bar의 x좌표.
int y	생성될 Progress bar의 y좌표
int w	생성될 Progress bar의 가로 크기
int h	생성될 Progress bar의 세로 크기
const char* text	Progress bar에 출력 될 text.
BOOL style	Progress bar의 style. Nomal = STYLE_PGBAR_NOMAL. Divide bar = STYLE_PGVAR_DIV.
BOOL bVertical	Progress의 출력 형태. TRUE = Vertical. FALSE = Horizontal.

Return Value

생성 된 Progress bar의 handle.

Example

```
EGL_HANDLE  pgbar;

pgbar = egl_create_progressbar(100, 100, 200, 40, "Progress", STYLE_PGBAR_NOMAL,FALSE);
```

▶**egl_progressbar_set_barcolor** function

```
void egl_progressbar_set_barcolor(
    EGL_HANDLE  hObj,
    EGL_COLOR  clr
 );
```

Overview

Progress bar의 bar color를 설정한다.

Parameter

EGL_HANDLE hObj	Bar color를 설정할 Progress bar handle.
EGL_COLOR clr	Color value. MAKE_COLORREF(r,g,b) 매크로 함수를 사용하면 color지정을 쉽게 할 수 있다.

Return Value

없음.

Example

```
EGL_HANDLE pgbar;

pgbar = egl_create_progressbar(100,100,200,40,"Progress", STYLE_PGBAR_NOMAL, FALSE);

egl_progressbar_set_barcolor(pgbar, MAKE_COLORREF(0, 0, 0xff));
```

▶**egl_progressbar_set_bgcolor** function

```
void egl_progressbar_set_bgcolor(
    EGL_HANDLE  hObj,
    EGL_COLOR  clr
);
```

Overview

Progress bar의 back ground color를 설정한다.

Parameter

EGL_HANDLE hObj	Back ground color를 설정 할 Progress bar handle.
EGL_COLOR clr	Color value. MAKE_COLORREF(r,g,b) 매크로 함수를 사용하면 color 지정을 쉽게 할 수 있다.

Return Value

없음.

Example

```
EGL_HANDLE pgbar;

pgbar = egl_create_progressbar(100,100,200,40,"Progress", STYLE_PGBAR_NOMAL, FALSE);

egl_progressbar_set_bgcolor(pgbar, MAKE_COLORREF(0, 0xff, 0));
```

▶**egl_progressbar_set_textcolor** function

```
void egl_progressbar_set_textcolor(
    EGL_HANDLE  hObj,
    EGL_COLOR  clr
 );
```

Overview

Progress bar의 text color를 설정한다.

Parameter

EGL_HANDLE hObj	Text color를 설정 할 Progress bar handle.
EGL_COLOR clr	Color value. MAKE_COLORREF(r,g,b) 매크로 함수를 사용하면 color 지정을 쉽게 할 수 있다.

Return Value

없음.

Example

```
EGL_HANDLE pgbar;

pgbar = egl_create_progressbar(100,100,200,40,"Progress", STYLE_PGBAR_NOMAL, FALSE);

egl_progressbar_set_textcolor(pgbar, MAKE_COLORREF(0xff, 0, 0xff));
```

▶**egl_progressbar_set_text** function

```
void egl_progressbar_set_text(
    EGL_HANDLE   hObj,
    const char*  text
);
```

Overview

Progress bar의 text를 설정한다.

Parameter

EGL_HANDLE hObj	Text를 설정할 Progress bar handle.
const char* text	설정 할 Text.

Return Value

없음.

Example

```
EGL_HANDLE pgbar;

pgbar = egl_create_progressbar(100,100,200,40,"Progress", STYLE_PGBAR_NOMAL, FALSE);

egl_progressbar_set_text(pgbar, "text1");
```

▶**egl_progressbar_set_pos** function

```
void egl_progressbar_set_pos(
    EGL_HANDLE  hObj,
    int  value
);
```

Overview

Progress bar의 bar position을 설정한다.

Parameter

EGL_HANDLE hObj	Bar position을 변경 할 Progress bar handle.
int value	Bar position 값 (0 ~ 100%)

Return Value

없음.

Example

```
EGL_HANDLE pgbar;

pgbar = egl_create_progressbar(100,100,200,40,"Progress", STYLE_PGBAR_NOMAL, FALSE);

egl_progressbar_set_pos(pgbar, 50);
```

▶**egl_progressbar_get_pos** function

```
int egl_progressbar_get_pos(
    EGL_HANDLE  hObj
);
```

Overview

Progress bar의 현재 bar position value를 가져온다.

Parameter

EGL_HANDLE hObj	Bar position value를 가져 올 Progress bar handle.

Return Value

Bar position value. (0 ~ 100%)

Example

```
EGL_HANDLE pgbar;

int pgbar_pos;

pgbar = egl_create_progressbar(100,100,200,40,"Progress", STYLE_PGBAR_NOMAL, FALSE);

pgbar_pos = egl_progressbar_get_pos(pgbar);
```

▶**egl_release_progressbar** function

```
BOOL egl_release_progressbar(
    EGL_HANDLE  hObj
);
```

Overview

Progress bar를 제거한다. 해당 object가 window에 add되어 있는 경우, egl_window_delete_object 함수로 window에서 object를 제거한 후 release 해주어야 한다.

Parameter

EGL_HANDLE hObj	제거 할 Progress bar handle.

Return Value

TRUE or FALSE

Example

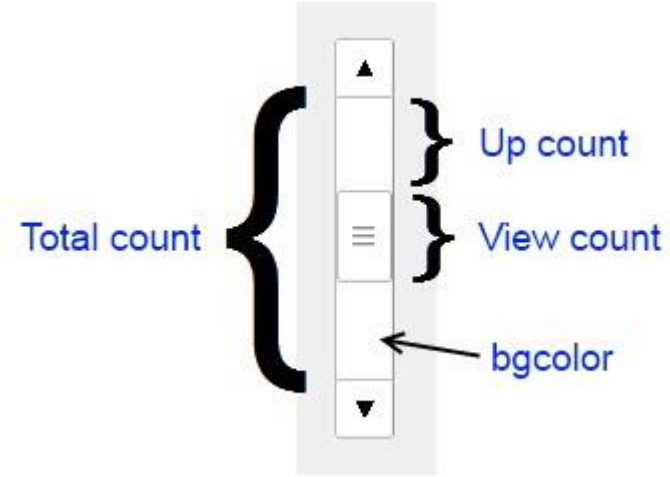
```
if(pgbar != NULL)
{
    egl_window_delete_object(hWin, pgbar);
    egl_release_progressbar(pgbar);
    pgbar = NULL;
}
```

▶ Progress Bar Example.

Example

```
EGL_HANDLE pgbar;
int value = 0;

void pgbar_callback(EGL_HANDLE h, int event)
{
        value++;
        if(value >100)
            value = 0;
      egl_progressbar_set_pos(pgbar, value);
}

extern BOOL process_touch(BOOL* touchdown, EGL_POINT* pPoint);

int main()
{
   ...
   EGL_POINT  touch_pt;
  BOOL  touchdown = FALSE;
  EGL_HANDLE  hWin;
   egl_init();
    pgbar = egl_create_progressbar(100, 100, 200, 40, "progress", STYLE_PGBAR_NOMAL,FALSE);
    egl_progress_callback(pgbar, pgbar_callback);
    egl_window_add_object(hWin, pgbar);
    egl_window_show(hWin);
    egl_draw();

while(1)
{
      if( process_touch( &touchdown, &touch_pt ) )
      egl_user_touch_input( touchdown, &touch_pt );

    egl_draw();
 }
}
```

Scroll Bar



Function	Description
egl_create_scrollbar	Scroll bar를 생성한다.
egl_scrollbar_callback	Scroll bar 이벤트가 발생했을 때 호출될 callback 함수를 등록한다.
egl_scroll_set_position	Scroll bar의 thumb position을 설정한다.
egl_scroll_get_position	Scroll bar의 thumb position을 가져온다.
egl_scroll_set_totalcount	Scroll bar의 total count를 설정한다.
egl_scroll_get_totalcount	Scroll bar의 total count를 가져온다.
egl_scroll_set_view_count	Scroll bar의 view count를 설정한다.
egl_scroll_get_view_count	Scroll bar의 view count를 가져온다.
egl_scroll_set_upcount	Scroll bar의 up count를 설정한다.
egl_scroll_get_upcount	Scroll bar의 up count를 가져온다.
egl_scroll_set_bgcolor	Scroll bar의 back ground color를 설정한다.
egl_scroll_set_size	Scrollbar의 크기를 설정한다.

Function	Description
egl_release_scrollbar	Scrollbar를 제거한다.
Define	
SCROLLBAR_EVENT	typedef enum { SCBAR_CLICKED = 0, } SCROLLBAR_EVENT;

▶**egl_create_scrollbar** function

```
EGL_HANDLE   egl_create_scrollbar(
    int  x,
    int  y,
    int  w,
    int  h,
    int  totalcount,
    int  viewcount,
    BOOL bVertical
);
```

Overview

Scroll bar를 생성한다.

Parameter

int x	생성 될 Scroll bar의 x좌표.
int y	생성 될 Scroll bar의 y좌표.
int w	생성 될 Scroll bar의 가로 크기.
int h	생성 될 Scroll bar의 세로 크기.
int totalcount	Scroll bar가 관리할 항목의 전체 크기.
int viewcount	Scroll bar의 thumb가 표시할 항목의 크기
BOOL bVertical	Scroll bar의 출력 형태. TRUE = Vertical. FALSE = Horizontal.

Return Value

생성 된 Scroll bar handle.

Example

```
EGL_HANDLE scroll;

scroll = egl_create_scrollbar(100, 100, 30, 200, 30, 10, TRUE);
```

▶**egl_scrollbar_callback** function

```
BOOL egl_scrollbar_callback(
    EGL_HANDLE  hObj,
    EVENT_CALLBACK  cb
);
```

Overview

Scroll bar 이벤트가 발생했을 때 호출될 callback 함수를 등록한다.

Parameter

ELG_HANDLE hObj	Callback 함수를 등록 할 Scroll bar handle.
EVENT_CALLBACK cb	호출 될 callback 함수.

Return Value

TRUE or FALSE

Example

```
void scroll_callback(EGL_HANDLE h, int event)
{

}
int main()
{
   ...
   EGL_HANDLE scroll;
   scroll = egl_create_scrollbar(100,100, 30, 200, 30, 10, TRUE);
   egl_scroll_callback(check, scroll_callback);
   ...
}
```

▶**egl_scroll_set_position** function

```
void  egl_scroll_set_position(
      EGL_HANDLE  hObj,
    int  totalcount,
    int  viewcount,
    int  upcount
);
```

Overview

Scroll bar의 thumb position을 설정한다.

Parameter

EGL_HANDLE hObj	Thumb position을 설정 할 Scroll bar handle.
int totalcount	Scroll bar가 관리 할 항목의 전체 크기.
int viewcount	Scroll bar의 thumb가 표시할 항목의 크기.
int upcount	Scroll bar의 thumb 이전 항목의 크기.

Return Value

없음.

Example

```
EGL_HANDLE scroll;

scroll = egl_create_scrollbar(100,100, 30, 200, 30, 10, TRUE);

egl_scroll_set_position(scroll, 25, 5, 10);
```

▶**egl_scroll_get_position** function

```
void  egl_scroll_get_position(
    EGL_HANDLE  hObj,
    int*  totalcount,
    int*  viewcount,
    int*  upcount
);
```

Overview

Scroll bar의 thumb position을 가져온다.

Parameter

EGL_HANDLE hObj	Thumb position을 가져 올Scroll bar handle.
int* totalcount	Total count를 저장 할 포인터.
int* viewcount	View count를 저장 할 포인터.
int* upcount	Up count를 저장 할 포인터.

Return Value

없음.

Example

```
EGL_HANDLE scroll;
int totalcount = 0;
int viewcount = 0;
int upcount = 0;
scroll = egl_create_scrollbar(100,100, 30, 200, 30, 10, TRUE);
egl_scroll_get_position(scroll, &totalcount, &viewcount, &upcount);
```

▶**egl_scroll_set_totalcount** function

```
void  egl_scroll_set_totalcount(
    EGL_HANDLE  hObj,
    int  totalcount
);
```

Overview

Scroll bar의 total count를 설정한다.

Parameter

EGL_HANDLE hObj	Total count를 설정 할 Scroll bar handle.
int totalcount	Scroll bar가 관리 할 항목의 전체 크기.

Return Value

없음.

Example

```
EGL_HANDLE scroll;

scroll = egl_create_scrollbar(100,100, 30, 200, 30, 10, TRUE);

egl_scroll_set_totalcount(scroll, 45);
```

▶**egl_scroll_get_totalcount** function

```
int  egl_scroll_get_totalcount(
    EGL_HANDLE  hObj
);
```

Overview

Scroll bar의 total count를 가져온다.

Parameter

EGL_HANDLE hObj	Total count를 가져 올 Scroll bar handle.

Return Value

Total count value.

Example

```
EGL_HANDLE scroll;
int totalcount = 0;
scroll = egl_create_scrollbar(100,100, 30, 200, 30, 10, TRUE);
totalcount = egl_scroll_get_totalcount(scroll);
```

▶**egl_scroll_set_viewcount** function

```
void  egl_scroll_set_viewcount(
    EGL_HANDLE  hObj,
    int  viewcount
);
```

Overview

Scroll bar의 view count를 설정한다.

Parameter

EGL_HANDLE hObj	View count를 설정 할 Scroll bar handle.
int viewcount	Scroll bar의 thumb가 표시할 항목의 크기.

Return Value

없음.

Example

```
EGL_HANDLE scroll;

scroll = egl_create_scrollbar(100,100, 30, 200, 30, 10, TRUE);

egl_scroll_set_viewcount(scroll, 5);
```

▶ **egl_scroll_get_viewcount** function

```
int  egl_scroll_get_viewcount(
    EGL_HANDLE  hObj
);
```

Overview

Scroll bar의 view count를 가져온다.

Parameter

EGL_HANDLE hObj	View count를 가져 올 Scroll bar handle.

Return Value

View count value.

Example

```
EGL_HANDLE scroll;
int viewcount = 0;
scroll = egl_create_scrollbar(100,100, 30, 200, 30, 10, TRUE);
viewcount = egl_scroll_get_viewcount(scroll);
```

▶**egl_scroll_set_upcount** function

```
void  egl_scroll_set_upcount(
    EGL_HANDLE  hObj,
    int  upcount
);
```

Overview

Scroll bar의 up count를 설정한다.

Parameter

EGL_HANDLE hObj	Up count를 설정 할 Scroll bar handle.
int upcount	Scroll bar의 thumb 이전 항목의 크기.

Return Value

없음.

Example

```
EGL_HANDLE scroll;

scroll = egl_create_scrollbar(100,100, 30, 200, 30, 10, TRUE);

egl_scroll_set_upcount(scroll, 10);
```

▶**egl_scroll_get_upcount** function

```
int  egl_scroll_get_upcount(
    EGL_HANDLE  hObj
);
```

Overview

Scroll bar의 up count를 가져온다.

Parameter

EGL_HANDLE hObj	Up count를 가져 올 Scroll bar handle.

Return Value

Up count value.

Example

```
EGL_HANDLE scroll;
int upcount = 0;
scroll = egl_create_scrollbar(100,100, 30, 200, 30, 10, TRUE);
upcount = egl_scroll_get_upcount(scroll);
```

▶**egl_scroll_set_bgcolorbar** function

```
void  egl_scroll_set_bgcolor(
    EGL_HANDLE  hObj,
    unsigned char  r,
    unsigned char  g,
    unsigned char  b
)
```

Overview

Scroll bar의 back ground color를 설정한다.

Parameter

EGL_HANDLE hObj	Back ground color를 설정 할 Scroll bar handle.
unsigned char r	RGB 중 Red 값.
unsigned char g	RGB 중 Green 값.
unsigned char b	RGB 중 Blue 값.

Return Value

없음.

Example

```
EGL_HANDLE scroll;

scroll = egl_create_scrollbar(100,100, 30, 200, 30, 10, TRUE);

egl_scroll_set_bgcolor(scroll, 0, 0, 0xff);
```

▶**egl_scroll_set_size** function

```
void  egl_scroll_set_size(
    EGL_HANDLE  hObj,
    int  w,
    int  h
)
```

Overview

Scroll bar의 크기를 설정한다.

Parameter

EGL_HANDLE hObj	크기를 설정 할 Scroll bar handle.
int w	가로 크기.
int h	세로 크기.

Return Value

없음.

Example

```
EGL_HANDLE scroll;
scroll = egl_create_scrollbar(100,100, 30, 200, 30, 10, TRUE);
...
egl_scroll_set_size(scroll, 40, 200);
```

▶**egl_release_scrollbar** function

```
BOOL  egl_release_scrollbar(
    EGL_HANDLE  hObj
)
```

Overview

Scroll bar를 제거한다. 해당 object가 window에 add 되어 있는 경우, egl_window_delete_object 함수로 window에서 object를 제거한 후 release 해주어야 한다.

Parameter

EGL_HANDLE hObj	제거 할 scroll bar handle.

Return Value

TRUE or FALSE.

Example

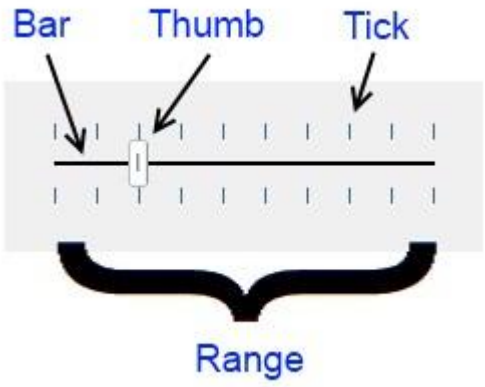
```
if( scroll != NULL)
{
    egl_window_delete_object(hWin, scroll);
    egl_release_scrollbar(scroll);
    scroll = NULL;
}
```

▶ Scroll Bar Example.

Example

```
EGL_HANDLE scroll;
void scroll_callback(EGL_HANDLE h, int event)
{
      int totalcount = 0;
      int viewcount = 0;
      int upcount = 0;

     if(event == SCBAR_CLICKED)
     {
          egl_scroll_get_position(scroll, &totalcount, &viewcount, &upcount);
          debugprintf("%d, %d, %d ₩r₩n", totalcount, viewcount, upcount);
       }
}
extern BOOL process_touch(BOOL* touchdown, EGL_POINT* pPoint);
int main()
{
   …
   EGL_POINT  touch_pt;
  BOOL  touchdown = FALSE;
  EGL_HANDLE  hWin;
   egl_init();
    scroll = egl_create_scrollbar(100, 100, 30, 200, 30, 10, TRUE);
    egl_scroll_set_totalcount(scroll, 40);
   egl_scroll_callback(scroll, scroll_callback);
   egl_window_add_object(hWin, scroll);
    egl_window_show(hWin);
   egl_draw();
while(1)
{
      if( process_touch( &touchdown, &touch_pt ) )
      egl_user_touch_input( touchdown, &touch_pt );

    egl_draw();
  }
}
```

Slider



Function	Description
egl_create_slider	Slider를 생성한다.
egl_slider_callback	Slider 이벤트가 발생했을 때 호출될 callback 함수를 등록한다.
egl_slider_set_pos	Slider의 thumb position을 설정한다.
egl_slider_get_pos	Slider의 thumb position 값을 가져온다.
egl_slider_set_range	Slider의 range를 설정한다.
egl_slider_get_range	Slider의 range를 가져온다.
egl_slider_stepit	Slider를 전후로 한 step이동한다.
egl_slider_set_tick_frequency	Slider의 tick 빈도 수를 설정한다.
egl_slider_set_tick_style	Slider의 tick style을 설정한다.
egl_slider_set_thumb_size	Slider의 thumb size를 설정한다.
egl_slider_get_thumb_size	Slider의 thumb size를 가져온다.
egl_slider_set_barcolor	Slider의 bar color를 설정한다.
egl_slider_set_tickcolor	Slider의 tick color를 설정한다.

Function	Description
egl_slider_set_transparent	Slider의 배경을 투명하게 할지를 결정한다.
egl_release_slider	Slider를 제거한다.
define	
SLIDER_EVENT	typedef enum { SLR_CLICKED = 0, } SLIDER_EVENT;

▶**egl_create_slider** function

```
EGL_HANDLE   egl_create_slider(
    int   x,
    int   y,
    int   w,
    int   h,
    int   range,
    TICKSTYLE   style,
    BOOL   bVertical
);
```

Overview

Slider를 생성한다.

Parameter

int x	생성 될 Slider의 x좌표.
int y	생성 될 Slider의 y좌표.
int w	생성 될 Slider의 가로 크기.
int h	생성 될 Slider의 세로 크기.
int range	생성 될 Slider의 Range.
TICKSTYLE style	생성 될 Slider의 Tick style TICK_NONE : 눈금 표시 없음. TICK_TOPLEFT : 눈금 표시가 위쪽 또는 좌측에 표시. TICK_BOTTOMRIGHT : 눈금 표시가 아래쪽 또는 우측에 표시. TICK_BOTH : 눈금 표시가 양쪽에 표시.
BOOL bVertical	Slider의 출력 형태. TRUE = Vertical. FALSE = Horizontal.

Return Value

생성 된 Slider handle.

Example

```
EGL_HANDLE slider;
slider = egl_create_slider(100, 100, 200, 40, 10, TICK_BOTH, FALSE);
```

▶**egl_slider_callback** function

```
BOOL  egl_slider_callback(
    EGL_HANDLE  hObj,
    EVENT_CALLBACK  cb
);
```

Overview

Slider 이벤트가 발생했을 때 호출될 callback 함수를 등록한다.

Parameter

EGL_HANDLE hObj	Callback 함수를 등록 할 Slider handle.
EVENT_CALLBACK cb	호출 될 callback 함수.

Return Value

TRUE or FALSE

Example

```
Void slider_callback(EGL_HANDLE h, int event)
{

}
int main()
{
   …
   EGL_HANDLE slider;
   slider = egl_create_slider(100,100, 200, 40, 10, TICK_BOTH, FALSE);
   egl_slider_callback(check, slider_callback);
   …
}
```

▶**egl_slider_set_pos** function

```
void  egl_slider_set_pos(
    EGL_HANDLE  hObj,
    int  nPos
);
```

Overview

Slider의 thumb position을 설정한다.

Parameter

EGL_HANDLE hObj	Position을 설정 할 Slider handle.
int nPos	Position 위치. (nPos는 정수로 0 ~ range)

Return Value

없음.

Example

```
EGL_HANDLE slider;
slider = egl_create_slider(100,100, 200, 40, 10, TICK_BOTH, FALSE);
egl_slider_set_pos(slider, 2);
```

▶**egl_slider_get_pos** function

```
int  egl_slider_get_pos(
    EGL_HANDLE  hObj
);
```

Overview

Slider의 thumb position 값을 가져온다.

Parameter

EGL_HANDLE hObj Position을 가져 올 Slider handle.

Return Value

Position value.

Example

```
EGL_HANDLE slider;
int slider_pos = 0;
slider = egl_create_slider(100,100, 200, 40, 10, TICK_BOTH, FALSE);
slider_pos = egl_slider_get_pos(slider);
```

▶**egl_slider_set_range** function

```
void  egl_slider_set_range(
    EGL_HANDLE  hObj,
    int  nMinPos,
    int  nMaxPos
);
```

Overview

Slider의 range를 설정한다.

Parameter

EGL_HANDLE hObj	Range를 설정 할 Slider handle.
int nMinPos	Min position 값. (range = Max position – Min position).
int nMaxPos	Max position 값. (range = Max position – Min position).

Return Value

없음.

Example

```
EGL_HANDLE slider;
slider = egl_create_slider(100,100, 200, 40, 10, TICK_BOTH, FALSE);
egl_slider_set_range(slider,  0, 15);
```

▶**egl_slider_get_range** function

```
void egl_slider_get_range(
    EGL_HANDLE  hObj,
     int*  lpMinPos,
     int*  lpMaxPos
);
```

Overview

Slider의 range를 가져온다.

Parameter

EGL_HANDLE hObj	Range 값을 가져 올 Slider handle.
int* lpMinPos	Min position 값을 저장 할 포인터.
int* lpMaxPos	Max position 값을 저장 할 포인터.

Return Value

없음.

Example

```
EGL_HANDLE slider;
int max_pos = 0;
int min_pos = 0;
slider = egl_create_slider(100, 100, 200, 40, 10, TICK_BOTH, FALSE);
egl_slider_get_range(slider, &min_pos, &max_pos);
```

▶**egl_slider_stepit** function

```
void  egl_slider_stepit(
    EGL_HANDLE  hObj,
    BOOL  inc
);
```

Overview

Slider를 전후로 한 step이동한다.

Parameter

EGL_HANDLE hObj	Step 이동 할 Slider handle.
BOOL inc	Step의 증가/감소를 결정. TRUE = 증가. FALSE = 감소.

Return Value

없음.

Example

```
EGL_HANDLE slider;
slider = egl_create_slider(100, 100, 200, 40, 10, TICK_BOTH, FALSE);
egl_slider_stepit(slider, TRUE);
//egl_slider_stepit(slider, FALSE);
```

▶**egl_slider_set_tick_frequency** function

```
void  egl_slider_set_tick_frequency(
    EGL_HANDLE  hObj,
    int freq
);
```

Overview

Slider의 tick 빈도 수를 설정한다.

Parameter

EGL_HANDLE hObj	Tick 빈도 수를 설정 할 Slider handle.
int freq	Tick 출력 빈도 수.

Return Value

없음.

Example

```
EGL_HANDLE slider;
slider = egl_create_slider(100, 100, 200, 40, 10, TICK_BOTH, FALSE);
egl_slider_set_tick_frequency(slider, 2);
```

▶**egl_slider_set_tick_style** function

```
void  egl_slider_set_tick_frequency(
    EGL_HANDLE  hObj,
    TICKSTYLE  style
);
```

Overview

Slider의 tick style을 설정한다.

Parameter

EGL_HADNE hObj	Tick style을 설정 할 Slider handle.
TICKSTYLE style	TICK style. TICK_NONE : 눈금 표시 없음. TICK_TOPLEFT : 눈금 표시가 위쪽 또는 좌측에 표시. TICK_BOTTOMRIGHT : 눈금 표시가 아래쪽 또는 우측에 표시. TICK_BOTH : 눈금 표시가 양쪽에 표시.

Return Value

없음.

Example

```
EGL_HANDLE slider;
slider = egl_create_slider(100, 100, 200, 40, 10, TICK_BOTH, FALSE);
egl_slider_set_tick_style(slider, TICK_BOTTOMRIGHT);
```

▶**egl_slider_set_thumb_size** function

```
void  egl_slider_set_thumb_size(
    EGL_HANDLE  hObj,
    int  width,
    int  height
);
```

Overview

Slider의 thumb size를 설정한다.

Parameter

EGL_HANDLE hObj	Thumb size를 설정 할 Slider handle.
int width	Thumb의 가로 크기.
int height	Thumb의 세로 크기.

Return Value

없음.

Example

```
EGL_HANDLE slider;
slider = egl_create_slider(100, 100, 200, 40, 10, TICK_BOTH, FALSE);
egl_slider_set_thumb_size(slider, 10, 30);
```

▶**egl_slider_get_thumb_size** function

```
void  egl_slider_get_thumb_size(
    EGL_HANDLE  hObj,
    int*  w,
    int*  h
)
```

Overview

Slider의 thumb size를 가져온다.

Parameter

EGL_HANDLE hObj	Thumb size를 가져 올 Slider handle.
int* w	Thumb 가로 크기를 저장할 포인터.
int* h	Thumb 세로 크기를 저장할 포인터.

Return Value

없음.

Example

```
EGL_HANDLE slider;
int w = 0;
int h = 0;
slider = egl_create_slider(100, 100, 200, 40, 10, TICK_BOTH, FALSE);
egl_slider_get_thumb_size(slider, &w, &h);
```

▶**egl_slider_set_barcolor** function

```
void  egl_slider_set_barcolor(
    EGL_HANDLE  hObj,
    EGL_COLOR  clr
);
```

Overview

Slider의 bar color를 설정한다.

Parameter

EGL_HANDLE hObj	Bar color를 설정 할 Slider handle.
EGL_COLOR clr	Color Value. MAKE_COLORREF(r,g,b) 매크로 함수를 사용하면 color 지정을 쉽게 할 수 있다.

Return Value

없음.

Example

```
EGL_HANDLE slider;
slider = egl_create_slider(100, 100, 200, 40, 10, TICK_BOTH, FALSE);
egl_slider_set_barcolor(slider,MAKE_COLORREF( 0, 0xff, 0));
```

▶**egl_slider_set_tickcolor** function

```
void  egl_slider_set_tickcolor(
    EGL_HANDLE  hObj,
    EGL_COLOR  clr
);
```

Overview

Slider의 tick color를 설정한다.

Parameter

EGL_HANDLE hObj	Tick color를 설정 할 Slider handle.
EGL_COLOR clr	Color value. MAKE_COLORREF(r,g,b) 매크로 함수를 사용하면 color 지정을 쉽게 할 수 있다.

Return Value

없음.

Example

```
EGL_HANDLE slider;
slider = egl_create_slider(100, 100, 200, 40, 10, TICK_BOTH, FALSE);
egl_slider_set_tickcolor(slider, MAKE_COLORREF(0xff, 0, 0));
```

▶**egl_slider_set_transparent** function

```
void  egl_slider_set_transparent(
    EGL_HANDLE  hObj,
    BOOL  bflag
);
```

Overview

Slider의 배경을 투명하게 할지를 결정한다.

Parameter

EGL_HANDLE hObj	배경을 투명하게 설정 할 Slider handle.
BOOL bflag	TRUE => 배경 투명. FALSE => 배경 불투명.

Return Value

없음.

Example

```
EGL_HANDLE slider;
slider = egl_create_slider(100, 100, 200, 40, 10, TICK_BOTH, FALSE);
egl_slider_set_transparent(slider, TRUE);
```

▶**egl_release_slider** function

```
BOOL  egl_release_slider(
    EGL_HANDLE  hObj
);
```

Overview

Slider를 제거한다. 해당 object가 window에 add 되어 있는 경우, egl_window_delete_object 함수로 window에서 object를 제거한 후 release 해주어야 한다.

Parameter

EGL_HANDLE hObj	제거 할 slider handle.

Return Value

TRUE or FALSE.

Example

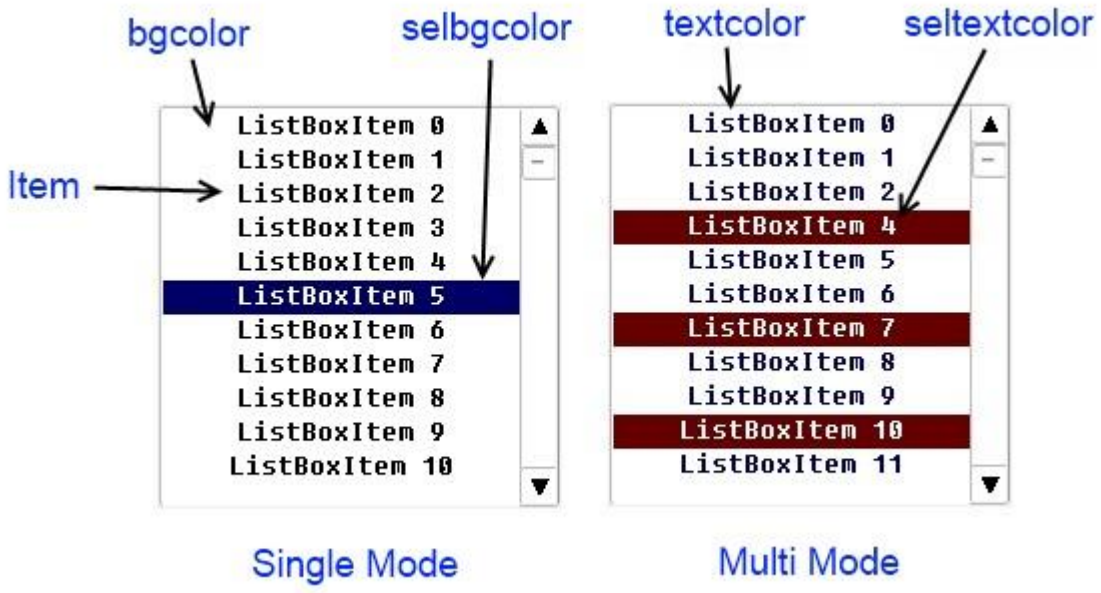
```
if(slider != NULL)
{
    egl_window_delete_object(hWin, slider);
    egl_release_slider(slider);
    slider = NULL;
}
```

▶ Slider Example.

Example

```
EGL_HANDLE slider;
void slider_callback(EGL_HANDLE h, int event)
{
      if(event == SLR_CLICKED)
      {
          debugprintf("position = %d₩r₩n",egl_slider_get_pos());
       }
}
extern BOOL process_touch(BOOL* touchdown, EGL_POINT* pPoint);
int main()
{
   …
   EGL_POINT touch_pt;
 BOOL touchdown = FALSE;
 EGL_HANDLE hWin;
  egl_init();
   slider = egl_create_slider(100, 100, 200, 40, 10, TICK_BOTH, FALSE);
   egl_sliderl_callback(slider, slider_callback);
  egl_window_add_object(hWin, slider);
   egl_window_show(hWin);
  egl_draw();
while(1)
{
      if( process_touch( &touchdown, &touch_pt ) )
      egl_user_touch_input( touchdown, &touch_pt );

   egl_draw();
 }
}
```

List Box



Function	Description
egl_create_listbox	List box를 생성한다.
egl_listbox_callback	List box이벤트가 발생했을 때 호출될 callback 함수를 등록한다.
egl_listbox_additem	List box에 Item을 추가한다.
egl_listbox_delitem	List box의 마지막 Item을 제거한다.
egl_listbox_delitem_text	List box에서 text에 해당되는 Item을 제거한다.
egl_listbox_delitem_num	List box에서 num번째 Item을 제거한다.
egl_listbox_alldelitem	List box의 모든 Item을 제거한다.
egl_listbox_get_all_itemlist	List box의 모든 Item들의 text목록을 반환한다.
egl_listbox_get_sel_item	List box가 Single Mode일 때 선택된 Item의 text를 반환한다.
egl_listbox_get_multiple_sel_itemlist	List box가 Multiple Mode일 때 선택된 Item들의 text목록을 반환한다.
egl_listbox_set_bgcolor	List box의 배경색을 지정한다.

Function	Description
egl_listbox_set_selbgcolor	List box에서 선택된 Item의 배경색을 지정한다.
egl_listbox_set_textcolor	List box의 글자 색을 지정한다.
egl_listbox_set_seltextcolor	List box에서 선택된 item의 글자 색을 지정한다.
egl_listbox_set_textalign	List box의 글자 align을 설정한다.
egl_listbox_set_scrollwidth	List box의 scrollbar의 가로 크기를 설정한다.
egl_listbox_change_item_text	List box에서 text에 해당하는 item 명을 변경한다.
egl_listbox_change_item_num	List box에서 num 번째 item 명을 변경한다.
egl_release_listbox	List box를 제거한다.

Define

LIST_EVENT

```
typedef enum
{
    LIST_CHANGED = 0,
} LIST_EVENT;
```

▶**egl_create_listbox** function

```
EGL_HANDLE  egl_create_listbox(
            int  x,
            int  y,
            int  w,
            int  h,
            bool bMultiple
);
```

Overview

List box를 생성한다.

Parameter

int x	생성될 List box의 x좌표.
int y	생성될 List box의 y좌표.
int w	생성될 List box의 가로 크기.
int h	생성될 List box의 세로 크기.
Bool bMultiple	생성될 List box에서 복수 item 선택 여부를 설정한다.

Return Value

생성된 List box의 pointer

Example

```
EGL_HANDLE listbox[2];

listbox [0] = egl_create_button (100, 100, 200, 200, TRUE);

listbox [1] = egl_create_button (400, 100, 200, 200, FALSE);
```

▶**egl_listbox_callback** function

```
BOOL  egl_listbox_callback (
      EGL_HANDLE  hObj,
      EVENT_CALLBACK  cb
);
```

Overview

List box 이벤트가 발생했을 때 호출될 callback 함수를 등록한다.

Parameter

EGL_HANDLE hObj	이벤트가 발생되는 List box의 handle
EVENT_CALLBACK cb	이벤트가 발생했을 때 호출될 callback 함수

Return Value

TRUE or FALSE.

Example

```
static void listbox_callback(EGL_HANDLE h, int event)
{
if(event == LIST_CHANGED)
    debugprintf("LIST_CHANGED");
}

void main(void)
{
   EGL_HANDLE listbox;

   egl_listbox_callback(listbox, listbox_callback);
}
```

▶**egl_listbox_additem** function

```
void  egl_listbox_additem (
      EGL_HANDLE  hObj,
      const char* text
);
```

Overview

List box에 Item을 추가한다.

Parameter

EGL_HANDLE hObj	Item을 추가할 List box의 handle
const char* text	추가할 Item의 text

Return Value

없음.

Example

```
EGL_HANDLE listbox;

egl_listbox_additem(listbox, "ListboxItem 1");
```

▶**egl_listbox_delitem** function

```
void  egl_listbox_delitem (
      EGL_HANDLE  hObj
);
```

Overview

List box의 마지막 Item을 제거한다.

Parameter

EGL_HANDLE hObj	Item을 제거할 List box의 handle

Return Value

없음.

Example

```
EGL_HANDLE listbox;

egl_listbox_delitem(listbox);
```

▶**egl_listbox_delitem_text** function

```
void  egl_listbox_delitem_text (
      EGL_HANDLE  hObj,
      const char* text
);
```

Overview

List box에서 text에 해당되는 Item을 제거한다.

Parameter

EGL_HANDLE hObj	Item을 제거할 List box의 handle
const char* text	제거할 item의 text.

Return Value

없음.

Example

```
EGL_HANDLE listbox;

egl_listbox_delitem_text(listbox, "item1");
```

▶**egl_listbox_delitem_num** function

```
BOOL  egl_listbox_delitem_num (
      EGL_HANDLE  hObj,
      Int  num
);
```

Overview

List box에서 num번째 Item을 제거한다.

Parameter

EGL_HANDLE hObj	Item을 제거할 List box의 handle
int num	제거할 item number.

Return Value

TRUE or FALSE

Example

```
EGL_HANDLE listbox;

egl_listbox_delitem_num(listbox, 3);
```

▶**egl_listbox_alldelitem** function

```
void  egl_listbox_alldelitem (
      EGL_HANDLE  hObj,
);
```

Overview

List box의 모든 Item을 제거한다.

Parameter

EGL_HANDLE hObj	Item을 제거할 List box의 handle

Return Value

없음

Example

```
EGL_HANDLE listbox;

egl_listbox_alldelitem(listbox);
```

▶egl_listbox_ get_all_itemlist function

```
const char**  egl_listbox_get_all_itemlist (
      EGL_HANDLE  hObj,
      int* itemcnt
);
```

Overview

List box의 모든 Item들의 text목록을 반환한다.

Parameter

EGL_HANDLE hObj	Item의 text 목록을 반환 받을 List box의 handle
int* itemcnt	List box의 Item개수를 반환 받을 변수의 pointer

Return Value

모든 Item의 text list pointer

※ 주의 : 반드시 return 받은 list pointer의 메모리를 해제하여야 한다.

Example

```
EGL_HANDLE listbox;
int ItemCnt;
const char** ItemTextList;

ItemTextList = egl_listbox_all_itemlist(listbox, &ItemCnt);

free(ItemTextList);
```

▶**egl_listbox_ get_sel_item** function

```
const char*  egl_listbox_get_sel_item (
      EGL_HANDLE  hObj,
      int* index
);
```

Overview

List box가 Single Mode 일 때 선택된 Item의 Text를 반환한다

Parameter

EGL_HANDLE hObj	선택된 Item의 text를 반환 받을 List box의 handle
Int* index	선택된 Item의 순번을 반환 받을 변수의 pointer

Return Value

선택된 Item의 text pointer

Example

```
EGL_HANDLE listbox;
int selectItemIndex;
const char* selectItemText;

selectItemText = egl_listbox_get_sel_item(listbox, &selectItem);
```

▶egl_listbox_ get_ multiple_sel_itemlist function

```
const char**  egl_listbox_get_multiple_sel_itemlist (
      EGL_HANDLE  hObj,
      int* selcnt
);
```

Overview

List box가 Multiple Mode일 때 선택된 Item의 text들의 목록을 반환한다.

Parameter

EGL_HANDLE hObj	선택된 Item들의 text를 반환 받을 List box의 handle
int* selcnt	선택된 Item의 개수를 반환 받을 변수의 pointer

Return Value

선택된 Item들의 text list pointer

※ 주의 : 반드시 return 받은 list pointer의 메모리를 해제하여야 한다.

Example

```
EGL_HANDLE listbox;
int selectItemCnt;
const char** selectItemTextList;

selectItemTextList = egl_listbox_get_sel_itemlist (listbox, & selectItemCnt);

free(selectItemTextList);
```

▶**egl_listbox_ set_bgcolor** function

```
void egl_listbox_ set_bgcolor (
      EGL_HANDLE   hObj,
      EGL_COLOR   clr
);
```

Overview

List box의 배경색을 지정한다.

Parameter

EGL_HANDLE hObj	배경색을 지정할 List box의 handle
EGL_COLOR clr	Color value. MAKE_COLORREF(r,g,b) 매크로 함수를 사용하면 color 지정을 쉽게 할 수 있다.

Return Value

없음

Example

```
EGL_HANDLE listbox;

egl_listbox_set_bgcolor(listbox, MAKE_COLORREF(0x0, 0x0, 0xFF));  // Blue
```

▶**egl_listbox_ set_selbgcolor** function

```
void egl_listbox_ set_selbgcolor (
      EGL_HANDLE   hObj,
      EGL_COLOR   clr
);
```

Overview

List box에서 선택된 Item의 배경색을 지정한다.

Parameter

EGL_HANDLE hObj	선택된 Item의 배경색을 지정할 List box의 handle
EGL_COLOR clr	Color value. MAKE_COLORREF(r, g, b) 매크로 함수를 사용하면 color 지정을 쉽게 할 수 있다.

Return Value

없음

Example

```
EGL_HANDLE listbox;

egl_listbox_set_selbgcolor(listbox, MAKE_COLORREF(0xFF, 0x0, 0x0));  // Red
```

▶**egl_listbox_ set_textcolor** function

```
void egl_listbox_ set_textcolor (
      EGL_HANDLE   hObj,
      ELG_COLOR   clr
);
```

Overview

List box의 글자 색을 지정한다.

Parameter

EGL_HANDLE hObj	글자 색을 지정할 List box의 handle
EGL_COLOR clr	Color value. MAKE_COLORREF(r, g, b) 매크로 함수를 사용하면 color 지정을 쉽게 할 수 있다.

Return Value

없음

Example

```
EGL_HANDLE listbox;

egl_listbox_set_textcolor(listbox, MAKE_COLORREF(0x0, 0x0, 0x0));  // Black
```

▶**egl_listbox_ set_seltextcolor** function

```
void egl_listbox_ set_seltextcolor (
      EGL_HANDLE   hObj,
      EGL_COLOR   clr
);
```

Overview

List box에서 선택된 Item의 글자 색을 지정한다.

Parameter

EGL_HANDLE hObj	선택된 Item의 글자 색을 지정할 List box의 handle
EGL_COLOR clr	Color value. MAKE_COLORREF(r, g, b) 매크로 함수를 사용하면 color 지정을 쉽게 할 수 있다.

Return Value

없음

Example

```
EGL_HANDLE listbox;

egl_listbox_set_seltextcolor(listbox, MAKE_COLORREF(0xFF, 0xFF, 0xFF));  // White
```

▶**egl_listbox_ set_textalign** function

```
void egl_listbox_ set_seltextcolor (
      EGL_HANDLE   hObj,
      Int   align
);
```

Overview

List box의 item align을 설정한다.

Parameter

EGL_HANDLE hObj	Align을 설정 할 List box의 handle.
int alingn	Align value. EGL_ALIGN_LEFT / EGL_ALIGN_RIGHT / EGL_ALIGN_CENTER EGL_ALIGN_TOP / EGL_ALIGN_BOTTOM

Return Value

없음

Example

```
EGL_HANDLE listbox;

egl_listbox_set_textalign(listbox, EGL_ALIGN_LEFT );
```

▶**egl_listbox_ set_scrollwidth** function

```
void egl_listbox_ set_scrollwidth (
      EGL_HANDLE   hObj,
      int   width
);
```

Overview

List box의 scrollbar의 가로 크기를 설정한다.

Parameter

EGL_HANDLE hObj	Scrollbar의 가로크기를 설정 할 List box의 handle
int width	Scrollbar의 가로 크기.

Return Value

없음

Example

```
EGL_HANDLE listbox;

egl_listbox_set_scrollwidth(listbox, 40);
```

▶**egl_listbox_change_item_text** function

```
BOOL  egl_listbox_change_item_text (
      EGL_HANDLE  hObj,
      const char*  text,
      const char*  changetext
);
```

Overview

List box에서 text에 해당하는 item 명을 변경하다.

Parameter

EGL_HANDLE hObj	Item명을 변경 할 List box의 handle.
const char* text	Item text.
const char* changetext	변경 할 Item text.

Return Value

TRUE or FALSE

Example

```
EGL_HANDLE listbox;

egl_listbox_change_item_text( listbox, "test1", "test2");
```

▶**egl_listbox_change_item_num** function

```
BOOL  egl_listbox_change_item_num (
      EGL_HANDLE  hObj,
      Int  num,
      const char*  changetext
);
```

Overview

List box에서 num번째 item 명을 변경하다.

Parameter

EGL_HANDLE hObj	Item명을 변경 할 List box의 handle.
int num	Item number.
const char* changetext	변경 할 Item text.

Return Value

TRUE or FALSE

Example

```
EGL_HANDLE listbox;

egl_listbox_change_item_text( listbox,  3, "test2");
```

▶**egl_release_listbox** function

```
BOOL egl_listbox_ set_scrollwidth (
      EGL_HANDLE   hObj
);
```

Overview

List box를 제거한다. 해당 object가 window에 add되어 있는 경우, egl_window_delete_object 함수로 window에서 object를 release해주는 것이 좋다.

Parameter

EGL_HANDLE hObj	제거 할 list box handle.

Return Value

TRUE or FALSE

Example

```
if(listbox != NULL)
{
    egl_window_delete_object(hWin, listbox);
    egl_releaase_listbox( listbox);
    listbox = NULL;
}
```

▶ List box Example.

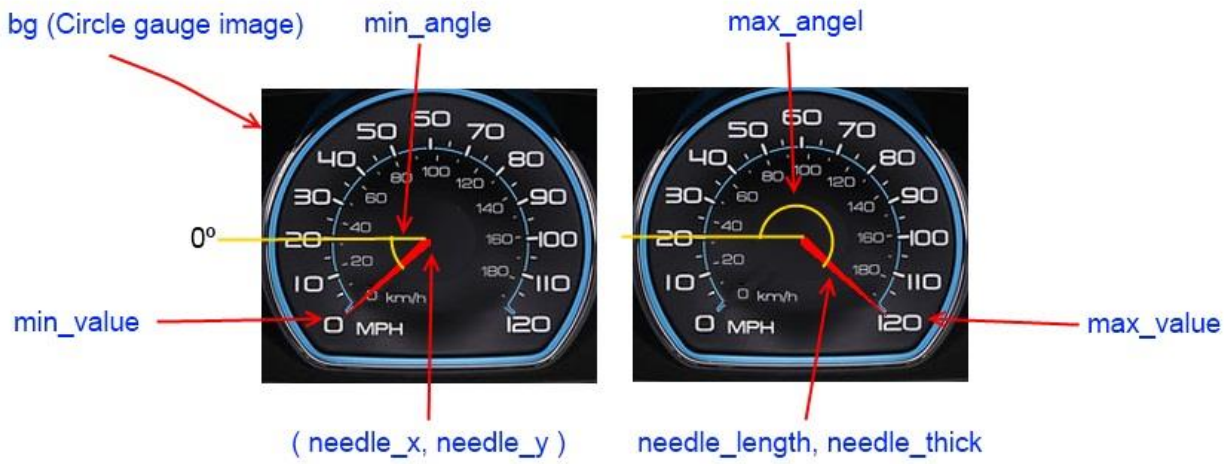
Example

```
#include "adStar.h"
extern BOOL process_touch(BOOL* touchdown,EGL_POINT* pPoint);

static void listbox_cb1(EGL_HANDLE h, int event)
{
	int itemcnt, index;
	const char **itemlist;
	const char *selitem;
	int i;

	if(event == LIST_CHANGED) {
		debugprintf("LIST_CHANGED₩r₩n");
		// Get all item list
		itemlist = egl_listbox_get_all_itemlist(h, &itemcnt);
		debugprintf("All Item %d₩r₩n", itemcnt);
		for(i = 0; i < itemcnt; i++)
			debugprintf("Item : [%s]₩r₩n", itemlist[i]);
		// Get selected item list
		selitem = egl_listbox_get_sel_item(h, &index);
		debugprintf("Selected Item [%d] : [%s]₩r₩n", index, selitem);
	}
	free(itemlist);
}

static void listbox_cb2(EGL_HANDLE h, int event)
{
	int selcnt;
	const char **selitemlist;
	int i;

	if(event == LIST_CHANGED) {
		debugprintf("LIST_CHANGED₩r₩n");
		selitemlist = egl_listbox_get_multiple_sel_itemlist(h, &selcnt);
		debugprintf("Selected Items %d ₩r₩n", selcnt);
		for(i = 0; i < selcnt; i++)
			debugprintf("Selected Item : [%s]₩r₩n", selitemlist[i]);
	}
```

```
	free(selitemlist);
}
void user_main()
{
	float addpos=1.0f;
	EGL_POINT touch_pt;
	BOOL touchdown=FALSE;
	EGL_HANDLE hWin;
	EGL_HANDLE listbox[2];

	egl_init();

	hWin = egl_create_window("Main Window");

	listbox[0] = egl_create_listbox(100, 100, 200, 200, FALSE);
	listbox[1] = egl_create_listbox(400, 100, 200, 200, TRUE);

	egl_listbox_callback(listbox[0], listbox_cb1);
	egl_listbox_callback(listbox[1], listbox_cb2);

	{
		int i;
		char str_text[16];
		for(i=0;i<100;i++)
		{
			sprintf(str_text,"ListBoxItem %d",i);
			egl_listbox_additem(listbox[0],str_text);
			egl_listbox_additem(listbox[1],str_text);
		}
	}
	// Delete last item
	egl_listbox_delitem(listbox[0]);
	// Delete "ListBoxItem 3" item
	egl_listbox_delitem_text(listbox[1], "ListBoxItem 3");

	// Set background color
	egl_listbox_set_bgcolor(listbox[1], MAKE_COLORREF(0xff, 0xff, 0xff));
	// Set selected background color
	egl_listbox_set_selbgcolor(listbox[1], MAKE_COLORREF(0x66, 0x0, 0x0));
	// Set text color
	egl_listbox_set_textcolor(listbox[1], MAKE_COLORREF(0, 0, 0x66));
```

```
        // Set selected text color
        egl_listbox_set_seltextcolor(listbox[1], MAKE_COLORREF(0xff, 0xff, 0xff));
        egl_window_add_object(hWin, listbox[0]);
        egl_window_add_object(hWin, listbox[1]);

       egl_window_show(hWin, TRUE);
        egl_draw();

        while(1)
        {
                BOOL bEvent = TRUE;

                if(process_touch(&touchdown, &touch_pt))
                        egl_user_touch_input(touchdown, &touch_pt);
                else
                        bEvent = FALSE;
                egl_draw();
        }
}
```

Circle Gauge



Function	Description
egl_create_circle_gauge	Circle gauge를 생성한다.
egl_circle_gauge_set_value	Circle gauge 값을 설정한다.
egl_circle_gauge_get_value	Circle gauge 값을 가져온다.
egl_release_circle_gauge	Circle gauge를 제거한다.

▶**egl_create_circle_gauge** function

```
EGL_HANDLE  egl_create_circle_gauge(
    SURFACE*  bg,
      int  x,
      int  y,
    EGL_CIRCLE_GAUGE_INFO*  pInfo
);
```

Overview

Circle gauge를 생성한다.

Parameter

SURFACE* bg	Circle gauge image.
int x	Circle gauge x좌표.
int y	Circle gauge y좌표.
EGL_CIRCLE_GAUGE_INFO* pInfo	Circle gauge 정보.

```
typedef struct  _tagCIRCLE_GAUGE_INFO
{
    int  needle_x;          // gauge needle x좌표.
    int  needle_y;          // gauge needle y좌표.
    int  needle_length;     // gauge needle 길이.
    int  needle_thick;      // gauge needle 두께.
    int  min_value;         // gauge의 최소 값.
    int  max_value;         // gauge의 최대 값.
    int  min_angle;         // gauge의 최소 각.
    int  max_angle;         // gauge의 최대 각.
}EGL_CIRCLE_GAUGE_INFO;
```

Return Value

생성 된 circle gauge handle

Example

```
SURFACE* gauge_bg;
EGL_HANDLE c_gauge;
EGL_CIRCLE_GAUGE_INFO CGInfo;
CGInfo.needle_x = 113;
CGInfo.needle_y = 101;
CGInfo.min_angle = -42;
CGInfo.max_angle = 222;
CGInfo.min_valie = 0;
CGInfo.max_value = 120;
CGInfo.needle_length = 80;
CGInfo.needle_thick = 3;
gauge_bg = loadbmp("gauge.bmp");
c_gauge = egl_create_circle_gauge(gauge_bg, 286, 37, &CGInfo);
```

▶**egl_circle_gauge_set_value** function

```
BOOL  egl_circle_gauge_set_value(
    EGL_HANDLE  h,
    int  value
);
```

Overview

Circle gauge 값을 설정한다.

Parameter

EGL_HANDLE h	Gauge 값을 설정 할 circle gauge handle.
int vaule	설정 gauge 값.

Return Value

TRUE or FALSE

Example

```
...
c_gauge = egl_create_circle_gauge(gauge_bg, 286,37,&CGInfo);
egl_circle_gauge_set_value(c_gauge, 80);
...
```

▶**egl_circle_gauge_get_value** function

```
int  egl_circle_gauge_get_value(
    EGL_HANDLE  h
);
```

Overview

Circle gauge 값을 가져온다.

Parameter

EGL_HANDLE h	Gauge 값을 가져 올 circle gauge handle.

Return Value

Gauge 값.

Example

```
...
c_gauge = egl_create_circle_gauge(gauge_bg, 286,37,&CGInfo);
int gauge_value = egl_circle_gauge_get_value(c_gauge);
...
```

▶**egl_release_circle_gauge** function

```
BOOL  egl_release_circle_gauge (
    EGL_HANDLE  hObj
);
```

Overview

Circle gauge를 제거한다. 해당 object가 window에 add 되어 있는 경우, egl_window_delete_object 함수로 window에서 object를 제거한 후 release 해주어야 한다.

Parameter

EGL_HANDLE hObj	제거 할 circle gauge handle.

Return Value

TRUE or FALSE

Example

```
if(circle != NULL)
{
    egl_window_delete_object(hWin, circle);
    egl_release_circle(circle);
    circle = NULL;
}
```

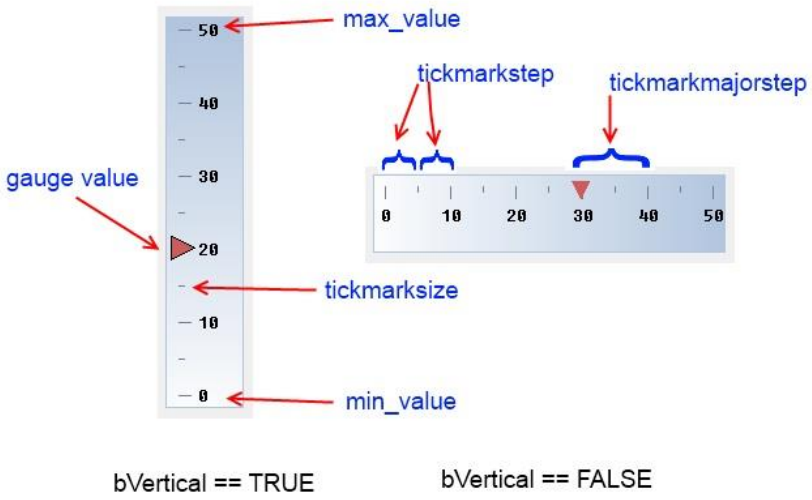
▶ Circle Gauge Example.

Example

```
extern BOOL process_touch(BOOL* touchdown, EGL_POINT* pPoint);
int main()
{
   …
   EGL_POINT touch_pt;
 BOOL touchdown = FALSE;
 EGL_HANDLE hWin;
   SURFACE* gauge_bg;
   EGL_HANDLE c_gauge;
   EGL_CIRCLE_GAUGE_INFO CGInfo;
  egl_init();
   CGInfo.needle_x = 113;
   CGInfo.needle_y = 101;
   CGInfo.min_angle = -42;
   CGInfo.max_angle = 222;
   CGInfo.min_valie = 0;
   CGInfo.max_value = 120;
   CGInfo.needle_length = 80;
   CGInfo.needle_thick = 3;
   gauge_bg = loadbmp("gauge.bmp");
   c_gauge = egl_create_circle_gauge(gauge_bg, 286, 37, &CGInfo);
   egl_window_add_object(hWin, c_gauge);
   egl_window_show(hWin);
  egl_draw();
  int gauge_value = 0;
  int gauge_flag = 0;
while(1)
{
      if( process_touch( &touchdown, &touch_pt ) )
     egl_user_touch_input( touchdown, &touch_pt );

   if(gauge_flag)
   {
      gauge_value++;
       if(gauge_value > 120)
```

```
            {
              gauge_value = 120;
              gauage_flag = 0;
            }
          }
          else
          {
             gauge_value--;
             if(gauge_value < 0)
             {
                gauge_value = 0;
                gauge_flag = 1;
              }
           }
           egl_circle_gauge_set_valut(c_gauge, gauge_value);
           egl_draw();
      }
}
```

Bar Gauge



bVertical == TRUE bVertical == FALSE

Function	Description
egl_create_bar_gauge	bar gauge를 생성한다.
egl_bar_gauge_set_value	bar gauge 값을 설정한다.
egl_bar_gauge_get_value	bar gauge 값을 가져온다.
egl_release_bar_gauge	Bar gauge를 제거한다.

▶**egl_create_bar_gauge** function

```
EGL_HANDLE  egl_create_bar_gauge(
    int  x,
    int  y,
    int  w,
    int  h,
    EGL_BAR_GAUGE_INFO*  pInfo
);
```

Overview

bar gauge를 생성한다.

Parameter

int x	Bar gauge x좌표.
int y	Bar gauge y좌표.
int w	Bar gauge 가로 크기.
int h	Bar gauge 세로 크기.
EGL_BAR_GAUGE_INFO* pInfo	Bar gauge 정보

```
typedef struct  _tagBAR_GAUGE_INFO
{
int min_value;            // gauge 최소 값.
int max_value;            // gauge 최대 값.
int tickmarksize;         // tick size. major tick = tick size * 2
int tickmarkstep;         // tick step size.
int tichmarkmajorstep;    // major tick step size.
   BOOL bVertical;        // vertical mode select.
   SURFACE* bg;           // back ground image.
} EGL_BAR_GAUGE_INFO;
```

Return Value

생성 된 bar gauge handle.

Example

```
EGL_HANDLE b_gauge;
EGL_BAR_GAUGE_INFO BGInfo;
BGInfo.min_value = 0;
BGInfo.max_value = 50;
BGInfo.bVertical = TRUE;
BGInfo.tickmarksize = 5;
BGInfo.tickmarkstep = 5;
BGInfo.tickmarkmajorstep = 10;
BGInfo.bg = NULL;
b_gauge  = egl_create_bar_gauge(10, 40, 60, 300, &BGInfo);
```

▶**egl_bar_gauge_set_value** function

```
BOOL  egl_bar_gauge_set_value(
    EGL_HANDLE  h,
    int  value
);
```

Overview

bar gauge 값을 설정한다.

Parameter

EGL_HANDLE h	값을 설정 할 bar gauge handle.
int value	설정 gauge 값.

Return Value

TRUE or FALSE

Example

```
...
b_gauge = egl_create_bar_gauge(10, 40, 60, 300, &BGInfo);
egl_bar_gauge_set_value(b_gauge, 20);
...
```

▶**egl_bar_gauge_get_value** function

```
int  egl_bar_gauge_get_value(
    EGL_HANDLE  h
);
```

Overview

bar gauge 값을 가져온다.

Parameter

EGL_HANDLE h	Gauge 값을 가져 올 bar gauge handle.

Return Value

Gauge 값

Example

```
...
b_gauge = egl_create_bar_gauge(10, 40, 60, 300, &BGInfo);
int gauge = egl_bar_gauge_get_value(b_gauge);
...
```

▶**egl_release_bar_gauge** function

```
BOOL  egl_release_bar_gauge(
    EGL_HANDLE  hObj
);
```

Overview

Bar gauge를 제거한다. 해당 object가 window에 add되어 있는 경우, egl_window_delete_object 함수로 window에서 object를 제거한 후 release 해주어야 한다.

Parameter

EGL_HANDLE hObj	제거 할 bar gauge handle.

Return Value

TRUE or FALSe

Example

```
if(bar_gauge != NULL)
{
    egl_window_delete_object(hWin, bar_gauge);
    egl_release_bar_gauge(bar_gauge);
    bar_gauge = NULL;
}
```

▶ Bar Gauge Example.

Example

```
extern BOOL process_touch(BOOL* touchdown, EGL_POINT* pPoint);
int main()
{
   ...
   EGL_POINT touch_pt;
 BOOL touchdown = FALSE;
 EGL_HANDLE hWin;
   EGL_HANDLE b_gauge;
   EGL_BAR_GAUGE_INFO BGInfo;
   egl_init();
   BGInfo.min_value = 0;
   BGInfo.max_value = 50;
   BGInfo.bVertical = TRUE;
   BGInfo.tickmarksize = 5;
   BGInfo.tickmarkstep = 5;
   BGInfo.tickmarkmajorstep = 10;
   BGInfo.bg = NULL;
   b_gauge  = egl_create_bar_gauge(10,40,60,300,&BGInfo);
   egl_window_add_object(hWin, b_gauge);
   egl_window_show(hWin);
  egl_draw();
  int gauge_value = 0;
  int gauge_flag = 0;
while(1)
{
      if( process_touch( &touchdown, &touch_pt ) )
     egl_user_touch_input( touchdown, &touch_pt );

   if(gauage_flag)
   {
      gauge_value++;
       if(gauge_value > 50)
       {
         gauge_value = 50;
         gauge_flag = 0;
```

```
            }
        }
        else
        {
            gauge_value--;
            if(gauge_value < 0)
            {
                gauge_value = 0;
                gauge_flag = 1;
             }
         }
         egl_bar_gauge_set_valut(b_gauge, gauge_value);
         egl_draw();
    }
}
```

Picture

Function	Description
egl_create_picture	Picture object를 생성한다.
egl_picture_callback	Picture 이벤트가 발생했을 때 호출될 callback 함수를 등록한다.
egl_picture_set	Picture object의 image를 변경한다.
egl_release_picture	Picture를 제거한다.

▶**egl_create_picture** function

```
EGL_HANDLE  egl_create_picture(
    SURFACE* surf,
    int  x,
    int  y,
    int  w,
    int  h
);
```

Overview

Picture object를 생성한다.

Parameter

SURFACE* surf	Picture object의 image.
int x	Picture object x좌표.
int y	Picture object y좌표.
int w	Picture object 가로 크기.
int h	Picture object 세로 크기.

Return Value

생성 된 picture object handle.

Example

```
SURFACE* surf = loadbmp("image.bmp");
EGL_HANDLE  picture;
picture  = egl_create_picture(surf , 10, 40, 128, 128 );
```

▶**egl_picture_callback** function

```
BOOL  egl_picture_callback(
    EGL_HANDLE  hObj,
    EVENT_CALLBACK  cb
);
```

Overview

Picture 이벤트가 발생했을 때 호출될 callback 함수를 등록한다.

Parameter

EGL_HANDLE hObj	이벤트가 발생되는 Picture의 handle.
EVENT_CALLBACK cb	이벤트가 발생했을 때 호출 될 callback 함수.

Return Value

TRUE or FALSE

Example

```
static void picture_callback(EGL_HANDLE h, int event)
{
    if(event == PICTURE_CLICKED)
        debugprintf("Picture clicked₩r₩n");
}

void main(void)
{
   EGL_HANDLE picture;
   ...
   egl_picture_callback(picture, picture_callback);
   ...
}
```

▶**egl_picture_set** function

```
SURFACE*  egl_picture_set(
    EGL_HANDLE  hObj,
    SURFACE*  surf
);
```

Overview

Picture의 image를 변경한다.

Parameter

EGL_HANDLE hObj	Image를 변경할 Picture handle.
SURFACE* surf	변경할 image.

Return Value

이전 설정되어 있던 surface.

Example

```
…
SURFACE* new_image = loadbmp("new_image.bmp");
SURFACE* old_image;
old_image = egl_picture_set(picture, new_image);
…
```

▶**egl_release_picture** function

```
BOOL  egl_release_picture(
    EGL_HANDLE  hObj
);
```

Overview

Picture를 제거한다. 해당 object가 window에 add되어 있는 경우, egl_window_delete_object 함수로 window에서 object를 제거한 후 release 해주어야 한다.

Parameter

EGL_HANDLE hObj	제거 할 picture handle.

Return Value

TRUE or FALSE

Example

```
If(picture != NULL)
{
    egl_window_delete_object( hWin, picture);
    egl_release_picture(picture);
    Picture = NULL;
}
```

▶ Picture Example.

Example

```
extern BOOL process_touch(BOOL* touchdown, EGL_POINT* pPoint);
int main()
{
      …
      EGL_POINT touch_pt;
      BOOL touchdown = FALSE;
     EGL_HANDLE hWin;
      EGL_HANDLE picture;
      egl_init();
      hWin = egl_create_window("window1");

      SURFACE*  surf = loadbmp("test.bmp");
      picture  = egl_create_picture(surf, 10, 40, 128, 128);
      egl_window_add_object(hWin, b_gauge);
      egl_window_show(hWin);
     egl_draw();

    while(1)
    {
        if( process_touch( &touchdown, &touch_pt ) )
              egl_user_touch_input( touchdown, &touch_pt );

        egl_draw();
    }
}
```

Animation

Function	Description
egl_create_animation	Animation object를 생성한다.
egl_release_animation	Animation을 제거한다.

▶**egl_create_animation** function

```
EGL_HANDLE  egl_create_animation(
    int  x,
    int  y,
    int  w,
    int  h,
    SURFACE**  surflist,
    int  surfcnt,
    int  delaycnt
);
```

Overview

Animation object를 생성한다. Animation object는 연속 이미지를 번갈아 가며 draw한다.

Parameter

int x	Animation object x좌표.
int y	Animation object y좌표.
int w	Animation object 가로 크기.
int h	Animation object 세로 크기.
SURFACE** surflist	Animation image list.
int surfcnt	Animation image count.
int delaycnt	Animation image 전환 delay.

Return Value

생성 된 animation object handle.

Example

```
EGL_HANDLE btn_ani;
SURFACE* surf_ani[10];
char fname[12];
int i;
for(i=0;i<10;i++)
{
   sprint(fname, "Frame%d.bmp",i);
   surf_ani[i] = loadbmp(fname);
}
btn_ani = egl_create_animation(350, 150, 128, 128, surf_ani, 10, 0);
```

▶**egl_release_animation** function

```
BOOL  egl_release_animation(
      EGL_HANDLE  hObj
);
```

Overview

Animation을 제거한다. 해당 object가 window에 add 되어 있는 경우, egl_window_delete_object 함수로 window에서 object를 제거한 후 release 해주어야 한다.

Parameter

EGL_HANDLE hObj	제거 할 animation handle.

Return Value

TRUE or FALSE

Example

```
if(animation != NULL)
{
    egl_window_delete_object(hWin, animation);
    egl_release_animation(animation);
    animation = NULL;
}
```

▶ Animation Example.

Example

```
extern BOOL process_touch(BOOL* touchdown, EGL_POINT* pPoint);
int main()
{
    ...
    EGL_POINT touch_pt;
  BOOL touchdown = FALSE;
  EGL_HANDLE hWin;
    EGL_HANDLE btn_ani;
    SURFACE* surf_ani[10];
    char fname[12];
    egl_init();
    int i;
    for(i=0;i<10;i++)
    {
        sprint(fname, "Frame%d.bmp",i);
        surf_ani[i] = loadbmp(fname);
    }
    btn_ani = egl_create_animation(350, 150, 128, 128, surf_ani, 10, 0);
    egl_window_add_object(hWin, btn_ani);
    egl_window_show(hWin);
   egl_draw();
  while(1)
{
      if( process_touch( &touchdown, &touch_pt ) )
     egl_user_touch_input( touchdown, &touch_pt );

      egl_draw();
  }
}
```

Custom Object

Function	Description
egl_create_custom_object	Custom object를 생성한다.

Define

EGL_MSG_ID

```
typedef  enum  enumMSG
{
    EGL_MSG_DRAW = 0,
    EGL_MSG_DELETE,
    EGL_MSG_FOCUS,
    EGL_MSG_UNFOCUS,
    EGL_MSG_KEY_UP,
    EGL_MSG_KEY_DOWN,
    EGL_MSG_TOUCHED,
    EGL_MSG_UNTOUCHED,
    EGL_MSG_MOVE,
    EGL_MSG_TIMETICK
} EGL_MSG_ID;
```

EGL_MSG

```
typedef  struct  _tagMessage
{
   EGL_MSGID  msgID;
   EGL_HANDLE  hObj;
   EGL_HANDLE  hWin;
   Union
   {
      EGL_POINT  point;
      U32  key;
   } param;
} EGL_MSG;
```

▶**egl_create_custom_object** function

```
EGL_HANDLE   egl_create_custom_object(
    int  x,
    int  y,
    int  w,
    int  h,
    void* (*msg_handler)(EGL_MSG*  pMsg)
);
```

Overview

Custom object를 생성한다.

Parameter

int x	Custom object x좌표.
int y	Custom object y좌표.
int w	Custom object 가로 크기.
int h	Custom object 세로 크기.
void* (*msg_handler)(EGL_MSG* pMsg)	Custom object msg 처리 함수.

Return Value

생성 된 custom object handle.

Example

```
Static void* custom_obj_msghandler(EGL_MSG* pMsg)
{
     Switch(pMsg->msgID)
    {
        Case EGL_MSG_DRAW:
```

```
            /* draw */
            Break;
        }
        Return NULL;
}
 …
 Int main()
 {
 …
 EGL_HANDLE custom_obj;
 custom_obj = egl_create_custom(350, 240, 100, 30, custom_obj_msghandler);
 …
 }
```

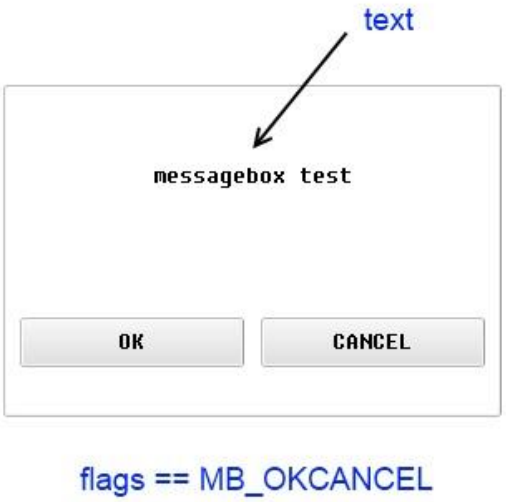
▶ Custom Example.

Example

```
static char speed_str[16];
static EGL_HANDLE hSpeed;

static void speed_draw(EGL_OBJECT_PTR pObj)
{
    draw_text_in_box(pObj->pFont, &pObj->rect, speed_str, EGL_ALIGN_CENTER);
}

static void* speed_msghandler(EGL_MSG* pMsg)
{
    switch(pMsg->msgID)
    {
        case EGL_MSG_DRAW:
            speed_draw(EGL_HANDLE_TO_OBJECT(pMsg->hObj));
            break;
     }
}

extern BOOL process_touch(BOOL* touchdown, EGL_POINT* pPoint);

int main()
{
   ...
   EGL_POINT touch_pt;
 BOOL  touchdown = FALSE;
 EGL_HANDLE hWin;
   EGL_HANDLE hSpeed;
   EGL_FONT* myfont;
   egl_init();
   myfont = egl_create_default_font();
   hSpeed = egl_create_custom_object(350, 240, 100, 30, speed_msghandler);
   egl_window_add_object(hWin, hSpeed);
   egl_set_font(hSpeed, myfont);
   strcpy(speed_str, "    0 Mile/h");
   egl_window_show(hWin);
  egl_draw();
```

```
    while(1)
   {
        if( process_touch( &touchdown, &touch_pt ) )
        egl_user_touch_input( touchdown, &touch_pt );

        egl_draw();
    }
}
```

Messagebox



Function	Description
egl_show_messagebox	Messagebox를 출력한다.

▶**egl_show_messagebox** function

```
int  egl_show_messagebox(
    const char* text,
    int  flags,
    BOOL  (*user_input_function)(EGL_MSG* pMsg)
);
```

Overview

Messagebox를 출력한다.

Parameter

const char* text	Messagebox에 출력할 문자.
int flags	Messagebox type. MB_OK MB_OKCANCEL MB_YESNO MB_YESNOCANCEL
BOOL (*user_input_function)(EGL_MSG* pMsg)	사용자 입력 함수. Messagebox 안의 어떤 버튼이 눌렸는지 확인을 위한 입력 함수.

Return Value

Message box 안의 눌린 버튼의 type 값을 반환.

```
enum
 {
   IDOK = 0,
   IDCANCEL,
   IDABORT,
   IDRETRY,
   IDIGNORE,
   IDYES,
   IDNO,
   IDCONTINUE,
 }
```

Example

```
BOOL my_touchinput(EGL_MSG* pMsg)
{
    BOOL touchdown;
    EGL_POINT touch_pt;
    if(process_touch(&touchdown, &touch_pt))
    {
        if(touchdown)
           pMsg -> msgID = EGL_MSG_TOUCHED;
        else
           pMsg -> msgID = EGL_MSG_UNTOUCHED;
        pMsg -> param.point.x = touch_pt.x;
        pMsg -> param.point.y = touch_pt_y;
        return TRUE;
    }
    return FALSE;
}

int main()
{
…
…
int type = egl_show_messagebox("messagebox test", MB_OKCANCEL, my_touchinput);
…
 }
```

▶ MessageBox Example.

Example

```
void btn_callback(EGL_HANDLE h, int event)
{
      if(event == BTN_CLICKED)
      {
          debugprintf("buttton clicked.");
    }
}

BOOL my_touchinput(EGL_MSG* pMsg)
{
    BOOL  touchdown;
    EGL_POINT  touch_pt;
    if(process_touch(&touchdown, &touch_pt))
    {
        if(touchdown)
           pMsg -> msgID = EGL_MSG_TOUCHED;
        else
           pMsg -> msgID = EGL_MSG_UNTOUCHED;
        pMsg -> param.point.x = touch_pt.x;
        pMsg -> param.point.y = touch_pt_y;
        return TRUE;
    }
    return FALSE;
}

extern BOOL process_touch(BOOL* touchdown, EGL_POINT* pPoint);

int main()
{
   …
   EGL_POINT touch_pt;
 BOOL touchdown = FALSE;
 EGL_HANDLE hWin;
 EGL_HANDLE btn;
 egl_init();
   btn = egl_create_button(100, 100, 100, 50, "Button Ex");
   egl_button_callback(btn, btn_callback);
```

```
    egl_window_add_object(hWin, btn);
    egl_window_show(hWin);
     egl_draw();

    int type = egl_show_messagebox("messagebox test", MB_OKCANCEL, my_touchinput);

  while(1)
  {
       if( process_touch( &touchdown, &touch_pt ) )
      egl_user_touch_input( touchdown, &touch_pt );

     egl_draw();
   }
}
```

EGL Font

Function	Description
egl_get_font	Object가 사용하는 font 정보를 반환한다.
egl_set_font	Object가 사용 할 font를 설정한다.
egl_font_set_bkmode	Font의 배경색 사용 여부를 결정한다.
egl_font_get_bk_color	Font의 배경색 정보를 가져온다.
egl_font_set_bk_color	Font의 배경색을 설정한다.
egl_font_get_color	Font의 색 정보를 가져온다.
egl_font_set_color	Font의 색을 설정한다.
create_bitfont	Font사용을 위해 Bit방식의 font를 생성한다.
release_bitfont	Create한 bit방식의 font를 release한다.
create_bmpfont	Font사용을 위해 이미지 font를 생성한다.
bmfont_release	Create한 이미지 font를 release한다.
draw_text	문자를 출력한다.
draw_text_pivot	문자를 좌/우 90도 회전하여 출력한다.
draw_text_len	문자를 지정한 길이만큼 출력한다.
draw_text_in_box	지정한 영역안에 문자를 출력한다.
text_width	문자열의 길이를 확인한다.

▶**egl_get_font** function

```
EGL_FONT*  egl_get_font(
    EGL_HANDLE  h
);
```

Overview

Object가 사용하는 font 정보를 반환한다.

Parameter

EGL_HANDLE h	Font 정보를 가져 올 object handle.

Return Value

Object가 사용하고 있는 font 정보. (EGL_FONT Struct)

Example

```
EGL_HANDLE object;
…
…
EGL_FONT* cur_font = egl_get_font(object);
```

▶**egl_set_font** function

```
EGL_FONT*  egl_set_font(
    EGL_HANDLE  h,
    EGL_FONT*  font
);
```

Overview

Object가 사용 할 font를 설정한다.

Parameter

EGL_HANDLE h	Font를 설정할 object handle.
EGL_FONT* font	설정 할 font.

Return Value

이전에 설정되어 있던 font 정보.

Example

```
EGL_HANDLE object;
EGL_FONT* font;
...
font = create_bitfont();
egl_set_font(object , font);
```

▶**egl_font_set_bkmode** function

```
void  egl_font_set_bkmode(
    EGL_FONT*  font,
    int  mode
);
```

Overview

Font의 배경색 사용 여부를 결정한다.

Parameter

EGL_FONT* font	설정을 변경 할 font.
int mode	배경 색 사용 여부. TRUE or FALSE.

Return Value

없음.

Example

```
 EGL_FONT* font;
 …
 font = create_bitfont();
 egl_set_font(object , font);
egl_font_set_bkmode(font, TRUE);
```

▶**egl_font_get_bk_color** function

```
EGL_COLOR  egl_font_get_bk_color(
    EGL_FONT*  font
);
```

Overview

Font의 배경색 정보를 가져온다.

Parameter

EGL_FONT* font	배경색 정보를 가져 올 font..

Return Value

현재 설정 되어 있는 font의 배경색.

Example

```
EGL_FONT* font;
EGL_COLOR font_bk_color;
...
font = create_bitfont();
font_bk_color = egl_font_get_bk_color(font);
```

▶**egl_font_set_bk_color** function

```
EGL_COLOR  egl_font_set_bk_color(
    EGL_FONT*  font,
    EGL_COLOR  clr
);
```

Overview

Font의 배경색을 설정한다.

Parameter

EGL_FONT* font	배경색을 설정 할 font.
EGL_COLOR clr	배경색. (MAKE_COLORREF( ) 매크로를 사용하여 설정)

Return Value

이전에 설정 되어 있던 font의 배경색.

Example

```
EGL_FONT* font;
EGL_COLOR font_bk_color;
...
font = create_bitfont();
font_bk_color = egl_font_set_bk_color(font, MAKE_COLORREF(0, 255, 0));
```

▶**egl_font_get_color** function

```
EGL_COLOR  egl_font_get_color(
    EGL_FONT*  font
);
```

Overview

Font의 색 정보를 가져온다.

Parameter

EGL_FONT* font	색 정보를 가져 올 font.

Return Value

현재 설정 되어 있는 font 색.

Example

```
EGL_FONT* font;
EGL_COLOR font_color;
...
font = create_bitfont();
font_color = egl_font_get_color(font);
```

▶**egl_font_set_color** function

```
EGL_COLOR  egl_font_set_color(
    EGL_FONT*  font,
    EGL_COLOR  clr
);
```

Overview

Font의 색을 설정한다.

Parameter

EGL_FONT* font	색을 설정 할 font.
EGL_COLOR clr	Font 색. (MAKE_COLORREF( ) 매크로를 사용하여 설정)

Return Value

이전에 설정 되어 있던 font 색.

Example

```
EGL_FONT* font;
EGL_COLOR font_color;
...
font = create_bitfont();
font_color = egl_font_set_color(font, MAKE_COLORREF(0, 0, 255));
```

▶**create_bitfont** function

```
EGL_FONT*   create_bitfont( void );
```

Overview

Font사용을 위해 Bit방식의 font를 생성한다.

Parameter

없음.

Return Value

생성된 bit font 구조체 포인터.

Example

```
EGL_FONT* bit_font;
EGL_HANDLE object;
...
bit_font = create_bitfont();
egl_set_font(object, bit_font );
...
draw_text(bit_font, 100, 100, "font test");
```

▶**release_bitfont** function

```
void  release_bitfont(
      EGL_FONT*  pFont
 );
```

Overview

Create한 bit방식의 font를 release한다.

Parameter

EGL_FONT* pFont Release 대상 bit font.

Return Value

없음.

Example

```
EGL_FONT* bit_font;
…
bit_font = create_bitfont();
…
release_bitfont(bit_font);
```

▶**create_bmpfont** function

```
EGL_FONT*  create_bmpfont(
      const char *fname
);
```

Overview

Font사용을 위해 이미지 font를 생성한다.

Parameter

const char *fname	이미지 font file name.

Return Value

생성된 이미지 font 구조체 포인터.

Example

```
EGL_FONT* bm_font;
EGL_HANDLE object;
...
bm_font = create_bmpfont("font/batang_24.fnt");
egl_set_font(object, bm_font);
...
draw_text(bm_font, 100, 100, "image font test");
```

▶**bmfont_release** function

```
void  bmfont_release(
      EGL_FONT*  pFont
);
```

Overview

Create한 이미지 font를 release한다.

Parameter

EGL_FONT* pFont Release 대상 이미지 font.

Return Value

없음.

Example

```
EGL_FONT* bm_font;
…
bm_font = create_bmpfont("font/batang_24.fnt");
…
bmfont_release(bm_font);
```

▶**draw_text** function

```
int  draw_text(
     EGL_FONT*  pFont,
     int  x,
     int  y,
     const char*  text
);
```

Overview

문자를 출력한다.

Parameter

EGL_FONT* pFont	문자 출력에 사용할 font.
int x	출력할 x좌표.
int y	출력할 y좌표.
const char* text	출력할 문자열.

Return Value

출력한 문자열 갯수.

Example

```
EGL_FONT* bit_font;
…
bit_font = create_bitfont();
…
draw_text(bit_font, 100, 100, "font test");
```

▶**draw_text_pivot** function

```
int  draw_text_pivot(
      EGL_FONT*  pFont,
      int  x,
      int  y,
      const char*  text,
      int  pivot
);
```

Overview

문자를 좌/우 90도 회전하여 출력한다.

Parameter

EGL_FONT* pFont	문자 출력에 사용할 font.
int x	출력할 x좌표.
int y	출력할 y좌표.
const char* text	출력할 문자열.
int pivot	좌/우 90도 회전 여부. PIVOT_RIGHT / PIVOT_LEFT

Return Value

출력한 문자열 갯수.

Example

```
EGL_FONT* bit_font;
…
bit_font = create_bitfont();
…
draw_text_pivot(bit_font, 100, 100, “font test”,PIVOT_RIGHT);
```

▶**draw_text_len** function

```
void  draw_text_len(
      EGL_FONT*  pFont,
      int  x,
      int  y,
      const char*  text,
      int  len
);
```

Overview

문자를 지정한 길이만큼 출력한다.

Parameter

EGL_FONT* pFont	문자 출력에 사용할 font.
int x	출력할 x좌표.
int y	출력할 y좌표.
const char* text	출력할 문자열.
int len	출력할 문자 개수

Return Value

없음.

Example

```
EGL_FONT* bit_font;
…
bit_font = create_bitfont();
…
draw_text_len(bit_font, 100, 100, "font test",4);
```

▶draw_text_in_box function

```
void  draw_text_in_box(
      EGL_FONT*  pFont,
      EGL_RECT*  pRect,
      const char*  text,
      int align
);
```

Overview

지정한 영역 안에 문자를 출력한다.

Parameter

EGL_FONT* pFont	문자 출력에 사용할 font.
EGL_RECT* pRect	출력할 영역.

```
typedef struct _tag_RECT{
    int  x;
    int  y;
    int  w;
    int  h;
} EGL_RECT;
```

const char* text	출력할 문자열.
int align	영역 안에서의 정렬.

```
EGL_ALIGN_LEFT,
EGL_ALIGN_RIGHT,
EGL_ALIGN_TOP,
EGL_ALIGN_BOTTOM,
EGL_ALIGN_CENTER,
EGL_ALIGN_MULTILINE
```

Return Value

없음.

Example

```
EGL_FONT* bit_font;
EGL_RECT rect;
…
bit_font = egl_create_default_font();
rect.x = 100;
rect.y = 100;
rect.w = 200;
rect.h = 40;
draw_text_in_box(bit_font, &rect, "font test", EGL_ALIGN_CENTER);
```

▶**text_width** function

```
int  text_width(
     EGL_FONT*  font,
     const char*  str
);
```

Overview

문자열의 길이(pixel 단위)를 확인한다.

Parameter

EGL_FONT* font	문자열에 사용 할 font. (font마다 한글자의 width가 다르기 때문에 지정해 주어야 된다.)
const char* str	Width값을 확인 할 문자열.

Return Value

문자열 width.

Example

```
EGL_FONT* bit_font;
int font_width;
…
bit_font = egl_create_default_font();
…
font_width = text_width(bit_font, “font test”);
```

EGL Primitives

Function	Description
draw_line	line을 그린다.
draw_hline	가로 line을 그린다.
draw_vline	세로 line을 그린다.
draw_thickline	Thickline line을 그린다.
draw_rect	사각형을 그린다.
draw_rectfill	안이 채워진 사각형을 그린다.
draw_rectfill_gradient	gradient효과가 들어간 사각형을 그린다.
draw_rectfill_h_gradient	가로로 gradient효과가 들어간 사각형을 그린다.
draw_rectfill_v_gradient	세로로 gradient효과가 들어간 사각형을 그린다.
draw_roundrect	모서리가 둥근 사각형을 그린다.
draw_roundrectfill	모서리가 둥근 안이 채워진 사각형을 그린다.
draw_arc	호, 반원, 원을 그린다.
draw_pie	파이 모양의 도형을 그린다.
draw_piefill	안이 채워진 파이 모양의 도형을 그린다.
draw_ellipse	타원을 그린다.
draw_ellipsefill	안이 채워진 타원을 그린다.
draw_circle	원을 그린다.
draw_circlefill	안이 채워진 원을 그린다.

Function	Description
draw_bezier	베이어 곡선을 그린다.
draw_polyline	지정한 좌표를 잇는 선을 그린다.
draw_polygon	지정한 좌표를 잇는 다각형을 그린다.
draw_polygonfill	지정한 좌표를 잇는 안이 채워진 다각형을 그린다.

▶**draw_line** function

```
void  draw_line(
    int  x,
    int  y,
    int  x2,
    int  y2,
    EGL_COLOR  c
);
```

Overview

(x, y)에서 (x2,y2)까지 line을 그린다.

Parameter

int x	Line의 시작 x좌표.
int y	Line의 시작 y좌표.
int x2	Line의 끝 x좌표.
int y2	Line의 끝 y좌표.
EGL_COLOR c	Line의 color. (MAKE_COLORREF(r,g,b) 매크로를 사용하여 설정)

Return Value

없음.

Example

```
draw_line(100, 100, 200, 200, MAKE_COLORREF(0, 0, 0);
draw_line(100, 100, 100, 200, MAKE_COLORREF(255, 0, 0);
```

▶**draw_hline** function

```
void draw_hline(
    int  x,
    int  y,
    int  x2,
    EGL_COLOR  c
);
```

Overview

(x, y)에서 (x2, y)까지 가로 line을 그린다.

Parameter

int x	Line의 시작 x좌표.
int y	Line의 시작과 끝 y좌표.
int x2	Line의 끝 x좌표.
EGL_COLOR c	Line의 color. (MAKE_COLORREF(r,g,b) 매크로를 사용하여 설정)

Return Value

없음.

Example

```
draw_hline(100, 100, 200, MAKE_COLORREF(0, 0, 0));
draw_hline(100, 200, 200, MAKE_COLORREF(0, 0, 255));
```

▶**draw_vline** function

```
void draw_vline(
    int  x,
    int  y,
    int  y2,
    EGL_COLOR  c
);
```

Overview

(x, y)에서 (x, y2)까지 세로 line을 그린다.

Parameter

int x	Line의 시작과 끝 x좌표.
int y	Line의 시작 y좌표.
int y2	Line의 끝 y좌표.
EGL_COLOR c	Line의 color. (MAKE_COLORREF(r,g,b) 매크로를 사용하여 설정)

Return Value

없음.

Example

```
draw_vline(100, 100, 200, MAKE_COLORREF(0, 0, 0));
draw_vline(200, 100, 200, MAKE_COLORREF(0, 255, 0));
```

▶**draw_thickline** function

```
void draw_thickline(
    int  x1,
    int  y1,
    int  x2,
    int  y2,
    U8  width,
    EGL_COLOR  color
);
```

Overview

(x1, y1)에서 (x2,y2)까지 설정한 두께의 line을 그린다.

Parameter

int x1	Thick line의 시작 x좌표.
int y1	Thick line의 시작 y좌표.
int x2	Thick line의 끝 x좌표.
int y2	Thick line의 끝 y좌표.
U8 width	Thick line의 두께.
EGL_COLOR color	Thick line의 color. (MAKE_COLORREF(r,g,b) 매크로를 사용하여 설정)

Return Value

없음.

Example

```
draw_thickline(100, 130, 200, 130, 2, MAKE_COLORREF(0, 0, 0));
draw_thickline(100, 200, 100, 300, 3, MAKE_COLORREF(255, 0, 0));
```

▶**draw_rect** function

```
void draw_rect(
    int  x,
    int  y,
    int  w,
    int  h,
    EGL_COLOR  c
);
```

Overview

(x, y)부터 가로크기가 w, 세로크기가 h인 사각형을 그린다.

Parameter

int x	사각형의 시작 x좌표.
int y	사각형의 시작 y좌표.
int w	사각형의 가로 크기.
int h	사각형의 세로 크기.
EGL_COLOR c	사각형 color. (MAKE_COLORREF(r,g,b) 매크로를 사용하여 설정)

Return Value

없음.

Example

```
draw_rect(100, 100, 200, 200, MAKE_COLORREF(0, 0, 0));
draw_rect(150, 150, 100, 100, MAKE_COLORREF(0, 0, 255));
```

▶**draw_rectfill** function

```
void draw_rectfill(
    int  x,
    int  y,
    int  w,
    int  h,
    EGL_COLOR  c
);
```

Overview

(x, y)부터 가로크기가 w, 세로크기가 h인 안이 채워진 사각형을 그린다.

Parameter

int x	사각형의 시작 x좌표.
int y	사각형의 시작 y좌표.
int w	사각형의 가로 크기.
int h	사각형의 세로 크기.
EGL_COLOR c	사각형 color. (MAKE_COLORREF(r,g,b) 매크로를 사용하여 설정)

Return Value

없음.

Example

```
draw_rectfill(100, 100, 50, 50, MAKE_COLORREF(0, 255, 0));
draw_rectfill(200, 200, 50, 50, MAKE_COLORREF(0, 0, 255));
```

▶**draw_rectfill_gradient** function

```
void draw_rectfill_gradient(
    int  x,
    int  y,
    int  w,
    int  h,
    EGL_COLOR startcolor,
    EGL_COLOR endcolor,
    BOOL  bVertical
);
```

Overview

(x, y)부터 가로크기가 w, 세로크기가 h인 gradient 사각형을 그린다.

Parameter

int x	사각형의 시작 x좌표.
int y	사각형의 시작 y좌표.
int w	사각형의 가로 크기.
int h	사각형의 세로 크기.
EGL_COLOR startcolor	Gradient 시작 color. (MAKE_COLORREF(r,g,b) 매크로를 사용하여 설정)
EGL_COLOR endcolor	Gradient 끝 color. (MAKE_COLORREF(r,g,b) 매크로를 사용하여 설정)
BOOL bVertical	Gradient 방향. TRUE == vertical, FALSE == horizontal.

Return Value

없음.

Example

```
draw_rectfill_gradient(100,100,200,50,MAKE_COLORREF(0,255,0),MAKE_COLORREF(255,255,255),FALSE);
draw_rectfill_gradient(100,200,50,200,MAKE_COLORREF(255,255,255),MAKE_COLORREF(0,0,255),TRUE);
```

▶**draw_rectfill_h_gradient** function

```
void draw_rectfill_h_gradient(
    int  x,
    int  y,
    int  w,
    int  h,
    EGL_COLOR  startcolor,
    EGL_COLOR  endcolor
);
```

Overview

(x, y)부터 가로크기가 w, 세로크기가 h인 가로방향 gradient 사각형을 그린다.

Parameter

int x	사각형의 시작 x좌표.
int y	사각형의 시작 y좌표.
int w	사각형의 가로 크기.
int h	사각형의 세로 크기.
EGL_COLOR startcolor	gradient 시작 color. (MAKE_COLORREF(r,g,b) 매크로를 사용하여 설정)
EGL_COLOR endcolor	gradient 끝 color. (MAKE_COLORREF(r,g,b) 매크로를 사용하여 설정)

Return Value

없음.

Example

```
draw_rectfill_h_gradient(50,50,200,50,MAKE_COLORREF(255,255,255),MAKE_COLORREF(255,0,0));
```

▶**draw_rectfill_v_gradient** function

```
void draw_rectfill_v_gradient(
    int  x,
    int  y,
    int  w,
    int  h,
    EGL_COLOR  startcolor,
    EGL_COLOR  endcolor
);
```

Overview

(x, y)부터 가로크기가 w, 세로크기가 h인 세로방향 gradient 사각형을 그린다.

Parameter

int x	사각형의 시작 x좌표.
int y	사각형의 시작 y좌표.
int w	사각형의 가로 크기.
int h	사각형의 세로 크기.
EGL_COLOR startcolor	gradient 시작 color. (MAKE_COLORREF(r,g,b) 매크로를 사용하여 설정)
EGL_COLOR endcolor	gradient 끝 color. (MAKE_COLORREF(r,g,b) 매크로를 사용하여 설정)

Return Value

없음.

Example

```
draw_rectfill_v_gradient(50,50,200,50,MAKE_COLORREF(255,255,255),MAKE_COLORREF(255,0,0));
```

▶**draw_roundrect** function

```
void draw_roundrect(
    int  x0,
    int  y0,
    int  w,
    int  h,
    int  corner,
    EGL_COLOR  c
);
```

Overview

(x0, y0)부터 가로크기가 w, 세로크기가 h인 둥근 사각형을 그린다.

Parameter

int x0	둥근 사각형의 시작 x좌표.
int y0	둥근 사각형의 시작 y좌표.
int w	둥근 사각형의 가로 크기.
int h	둥근 사각형의 세로 크기.
int corner	모서리 둥근 정도.
EGL_COLOR c	둥근 사각형 color. (MAKE_COLORREF(r,g,b) 매크로를 사용하여 설정)

Return Value

없음.

Example

```
draw_roundrect(100, 100, 100, 100, 10, MAKE_COLORREF(0, 0, 0);
draw_roundrect(400, 200, 200, 200, 20, MAKE_COLORREF(0, 255, 0);
```

▶**draw_roundrectfill** function

```
void draw_roundrectfill(
    int  x0,
    int  y0,
    int  w,
    int  h
    int  corner,
    EGL_COLOR  c
);
```

Overview

(x0, y0)부터 가로크기가 w, 세로크기가 h인 안이 채워진 둥근 사각형을 그린다.

Parameter

int x0	둥근 사각형의 시작 x좌표.
int y0	둥근 사각형의 시작 y좌표.
int w	둥근 사각형의 가로 크기.
int h	둥근 사각형의 세로 크기.
int corner	모서리 둥근 정도.
EGL_COLOR c	둥근 사각형 color. (MAKE_COLORREF(r,g,b) 매크로를 사용하여 설정)

Return Value

없음.

Example

```
draw_roundrectfill(100, 100, 100, 100, 10, MAKE_COLORREF(255, 0, 0));
draw_roundrectfill(300, 200, 200, 200, 15, MAKE_COLOEREF(0, 255, 0));
```

▶**draw_arc** function

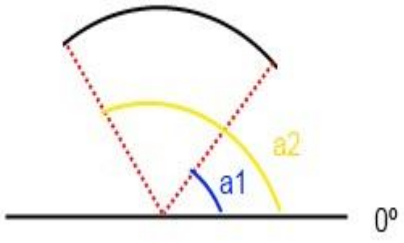
```
void draw_arc(
    int  x,
    int  y,
    int  rx,
    int  ry,
    int  a1,
    int  a2,
    EGL_COLOR  c
);
```

Overview

(x, y)를 원점으로 하고, 가로 반지름이 rx, 세로 반지름이 ry인 곡면을 그린다.
a1을 시작 각으로 하여 각 a2까지 곡면을 그린다.

Parameter

int x	원점의 x좌표.
int y	원점의 y좌표.
int rx	곡면의 가로 반지름.
int ry	곡면의 세로 반지름.
int a1	곡면의 시작 각도.
int a2	곡면의 끝 각도.
EGL_COLOR c	곡면의 color. (MAKE_COLORREF(r,g,b) 매크로를 사용하여 설정)



Return Value

없음.

Example

```
draw_arc(100, 100, 30, 30, 0, 90, MAKE_COLORREF(255, 0, 0));
draw_arc(200, 200, 40, 30, 90, 180, MAKE_COLORREF(0, 0, 255));
```

▶**draw_pie** function

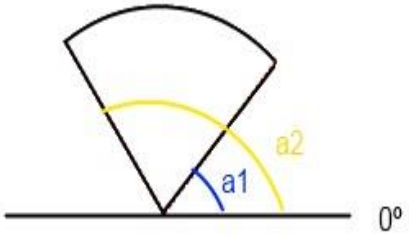
```
draw_pie(
    int  x,
    int  y,
    int  rx,
    int  ry,
    int  a1,
    int  a2,
    EGL_COLOR  c
);
```

Overview

(x, y)를 원점으로 하고, 가로 반지름이 rx, 세로 반지름이 ry인 파이모양을 그린다.
a1을 시작 각으로 하여 각 a2까지 그린다.

Parameter

int x	원점의 x좌표.
int y	원점의 y좌표.
int rx	파이의 가로 반지름.
int ry	파이의 세로 반지름.
int a1	파이의 시작 각도.
int a2	파이의 끝 각도.
EGL_COLOR c	파이의 color. (MAKE_COLORREF(r,g,b) 매크로를 사용하여 설정)



Return Value

없음.

Example

```
draw_pie(100, 100, 50, 50, 45, 90, MAKE_COLORREF(0, 0, 255));
draw_pie(200, 100, 60, 60, 90, 180, MAKE_COLORREF(255, 0, 0));
```

▶**draw_piefill** function

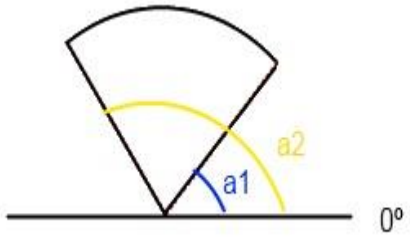
```
void  draw_piefill(
    int  x,
    int  y,
    int  rx,
    int  ry,
    int  a1,
    int  a2,
    EGL_COLOR  c
);
```

Overview

(x, y)를 원점으로 하고, 가로 반지름이 rx, 세로 반지름이 ry인 채워진 파이모양을 그린다.
a1을 시작 각으로 하여 각 a2까지 그린다.

Parameter

int x	원점의 x좌표.
int y	원점의 y좌표.
int rx	파이의 가로 반지름.
int ry	파이의 세로 반지름.
int a1	파이의 시작 각도.
int a2	파이의 끝 각도.
EGL_COLOR c	파이의 color. (MAKE_COLORREF(r,g,b) 매크로를 사용하여 설정)



Return Value

없음.

Example

```
draw_pie(100, 100, 50, 50, 45, 90, MAKE_COLORREF(0, 0, 255));
draw_pie(200, 100, 60, 60, 90, 180, MAKE_COLORREF(255, 0, 0));
```

▶**draw_ellipse** function

```
void draw_ellipse(
    int  x,
    int  y,
    int  rx,
    int  ry,
    EGL_COLOR  c
);
```

Overview

(x, y)를 원점으로 하고, 가로 반지름이 rx, 세로 반지름이 ry인 타원을 그린다.

Parameter

int x	원점의 x좌표.
int y	원점의 y좌표.
int rx,	가로 반지름.
int ry	세로 반지름.
EGL_COLOR c	타원의 color. (MAKE_COLORREF(r,g,b) 매크로를 사용하여 설정)

Return Value

없음.

Example

```
draw_ellipse(100, 100, 50, 100, MAKE_COLORREF(0, 0, 255));
draw_ellipse(300, 200, 100, 50, MAKE_COLORREF(255, 0, 0));
```

▶**draw_ellipsefill** function

```
void draw_ellipsefill(
    int  x,
    int  y,
    int  rx,
    int  ry,
    EGL_COLOR  c
);
```

Overview

(x, y)를 원점으로 하고, 가로 반지름이 rx, 세로 반지름이 ry인 채워진 타원을 그린다.

Parameter

int x	원점의 x좌표.
int y	원점의 y좌표.
int rx,	가로 반지름.
int ry	세로 반지름.
EGL_COLOR c	타원의 color. (MAKE_COLORREF(r,g,b) 매크로를 사용하여 설정)

Return Value

없음.

Example

```
draw_ellipsefill(100, 100, 50, 100, MAKE_COLORREF(0, 0, 255));
draw_ellipsefill(300, 200, 100, 50, MAKE_COLORREF(255, 0, 255));
```

▶**draw_circle** function

```
void draw_circle(
    int  x,
    int  y,
    int  r,
    EGL_COLOR  c
);
```

Overview

원점이 (x, y)고 반지름이 r인 원을 그린다.

Parameter

int x	원점의 x좌표.
int y	원점의 y좌표.
int r	반지름.
EGL_COLOR c	원의 color. (MAKE_COLORREF(r,g,b) 매크로를 사용하여 설정)

Return Value

없음.

Example

```
draw_circle(100, 100, 50, MAKE_COLORREF(255, 0, 255));
draw_circle(300, 200, 100, MAKE_COLORREF(0, 0, 255));
```

▶**draw_circlefill** function

```
void draw_circlefill(
    int  x,
    int  y,
    int  r,
    EGL_COLOR  c
);
```

Overview

원점이 (x, y)고 반지름이 r인 채워진 원을 그린다.

Parameter

int x	원점의 x좌표.
int y	원점의 y좌표.
int r	반지름.
EGL_COLOR c	원의 color. (MAKE_COLORREF(r,g,b) 매크로를 사용하여 설정)

Return Value

없음.

Example

```
draw_circle(100, 100, 70, MAKE_COLORREF(0, 0, 255));
draw_circle(300, 200, 50, MAKE_COLORREF(0, 255, 0));
```

▶**draw_bezier** function

```
void draw_bezier(
    EGL_POINT*  pts,
    int  n,
    int  s,
    EGL_COLOR  c
);
```

Overview

Bezier curve를 그린다.

Parameter

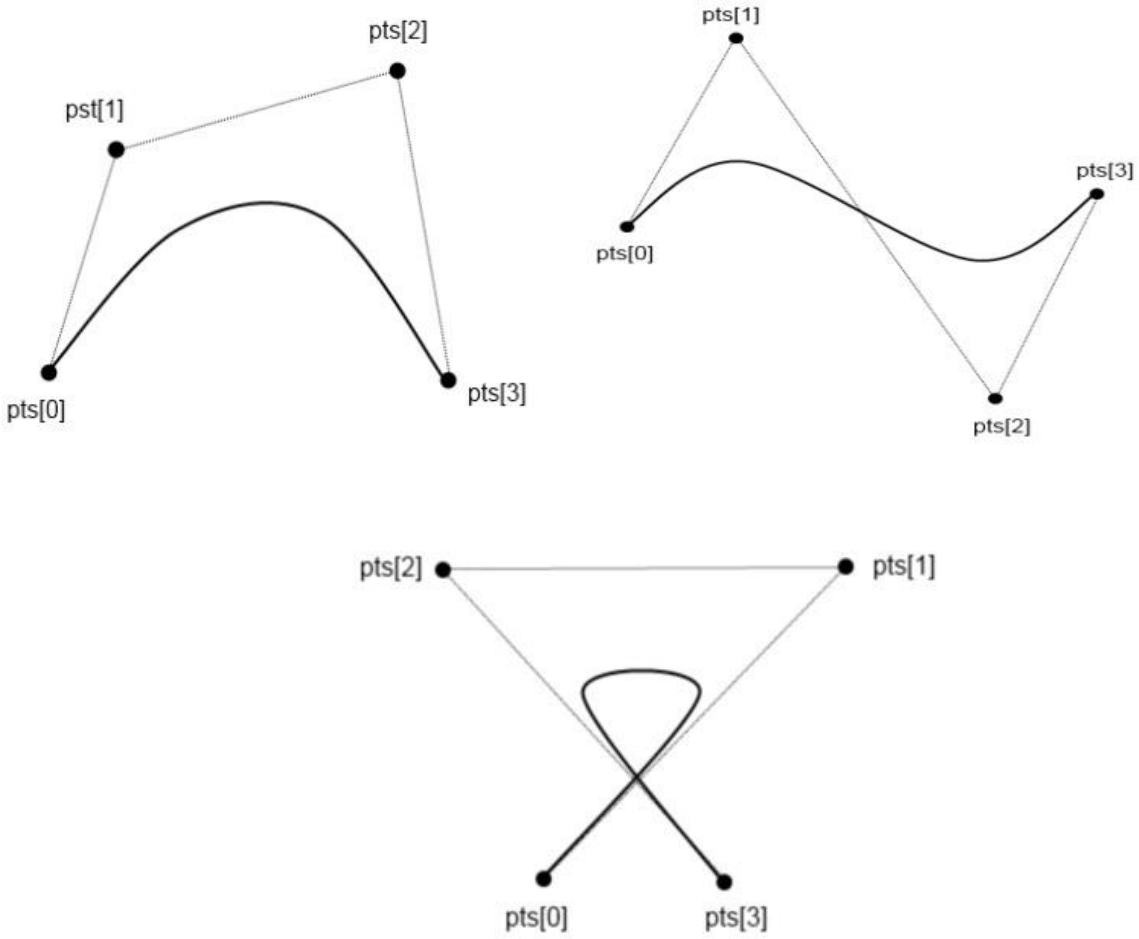
EGL_POINT* pts	좌표 배열.
int n	좌표 개수.
int s	point 개수. 좌표와 좌표사이에 몇 개의 point를 draw할지 결정.
EGL_COLOR c	curve의 color.

Return Value

없음.

Example

```
EGL_POINT* pts[4];
pts[0].x = 100; pts[0].y = 200;
pts[1].x = 200; pts[1].y = 50;
pts[2].x = 300; pts[2].y = 50;
pts[3].x = 400; pts[3].y = 200;

draw_bezier(pts, 4, 20, MAKE_COLORREF(255, 0, 255));
```



< Bezier curve image >

▶**draw_polyline** function

```
void draw_polyline(
    EGL_POINT*  p,
    int  n,
    EGL_COLOR  c
);
```

Overview

지정한 좌표를 잇는 선을 그린다.

Parameter

EGL_POINT* p	좌표 배열.
int n	좌표 개수.
EGL_COLOR c	선의 color. (MAKE_COLORREF(r,g,b) 매크로를 사용하여 설정)

Return Value

없음.

Example

```
EGL_POINT p[3];
p[0].x = 100; p[0].y = 100;
p[1].x = 130; p[1].y = 100;
p[2].x = 150; p[2].y = 150;
draw_polyline(p, 3, MAKE_COLORREF(0, 0, 0));
```

▶**draw_polygon** function

```
void draw_polygon(
    EGL_POINT* ptable,
    int  cnt,
    EGL_COLOR  c
);
```

Overview

지정한 좌표를 잇는 다각형을 그린다.

Parameter

EGL_POINT* p	좌표 배열.
int n	좌표 개수.
EGL_COLOR c	다각형의 color. (MAKE_COLORREF(r,g,b) 매크로를 사용하여 설정)

Return Value

없음.

Example

```
EGL_POINT ptable[5];
ptable[0].x = 100; ptable[0].y = 100;
ptable[1].x = 150; ptable[1].y = 50;
ptable[2].x = 200; ptable[2].y = 100;
ptable[3].x = 135; ptable[3].y = 150;
ptable[4].x = 115; ptable[4].y = 150;
draw_polygon(ptable, 5, MAKE_COLORREF(0, 0, 255));
```

▶**draw_polygonfill** function

```
void draw_polygonfill(
    EGL_POINT*  ptable,
    int  cnt,
    EGL_COLOR  c
);
```

Overview

지정한 좌표를 잇는 채워진 다각형을 그린다.

Parameter

EGL_POINT* p	좌표 배열.
int n	좌표 개수.
EGL_COLOR c	다각형의 color. (MAKE_COLORREF(r,g,b) 매크로를 사용하여 설정)

Return Value

없음.

Example

```
EGL_POINT ptable[3];
ptable[0].x = 100; ptable[0].y = 100;
ptable[1].x = 150; ptable[1].y = 150;
ptable[2].x = 100; ptable[2].y = 150;
draw_polygon(ptable, 3, MAKE_COLORREF(0, 255, 0));
```

EGL etc.

Function	Description
egl_init	egl 초기화 함수.
egl_show_object	Object를 draw 여부를 결정한다.
egl_object_set_redraw	Object redraw할지를 결정한다.

Define	
EGL_MSG_ID	typedef enum enumMSG { EGL_MSG_DRAW = 0, EGL_MSG_DELETE, EGL_MSG_FOCUS, EGL_MSG_UNFOCUS, EGL_MSG_KEY_UP, EGL_MSG_KEY_DOWN, EGL_MSG_TOUCHED, EGL_MSG_UNTOUCHED, EGL_MSG_MOVE, EGL_MSG_TIMETICK } EGL_MSG_ID;
EGL_MSG	typedef struct _tagMessage { EGL_MSGID msgID; EGL_HANDLE hObj; EGL_HANDLE hWin; Union { EGL_POINT point; U32 key; } param; } EGL_MSG;

▶**egl_init** function

```
BOOL  egl_init( void );
```

Overview

egl 초기화를 수행한다. embedded graphic library를 사용하기 위해서 반드시 호출해주어야 한다.

Parameter

없음.

Return Value

TRUE or FALSE

▶**egl_show_object** function

```
void  egl_show_object(
      EGL_HANDLE  h,
      BOOL  bShow
);
```

Overview

Object를 draw 여부를 결정한다. bShow 값을 FALSE설정 하면 egl_draw( ) 호출 시 해당 object는 draw되지 않는다.

Parameter

EGL_HANDLE h	Draw 여부를 결정할 object handle.
BOOL bShow	Draw 여부. TRUE or FALSE.

Return Value

없음.

▶**egl_object_set_redraw** function

```
void  egl_object_set_redraw(
      EGL_HANDLE  handle
);
```

Overview

Object redraw할지를 결정한다. 참고로 library에서 object의 변화가 있을 경우 이 함수를 호출하지 않아도 redraw가 자동으로 설정된다.

Parameter

EGL_HANDLE handle	Object를 redraw 설정 할 object handle.

Return Value

없음.

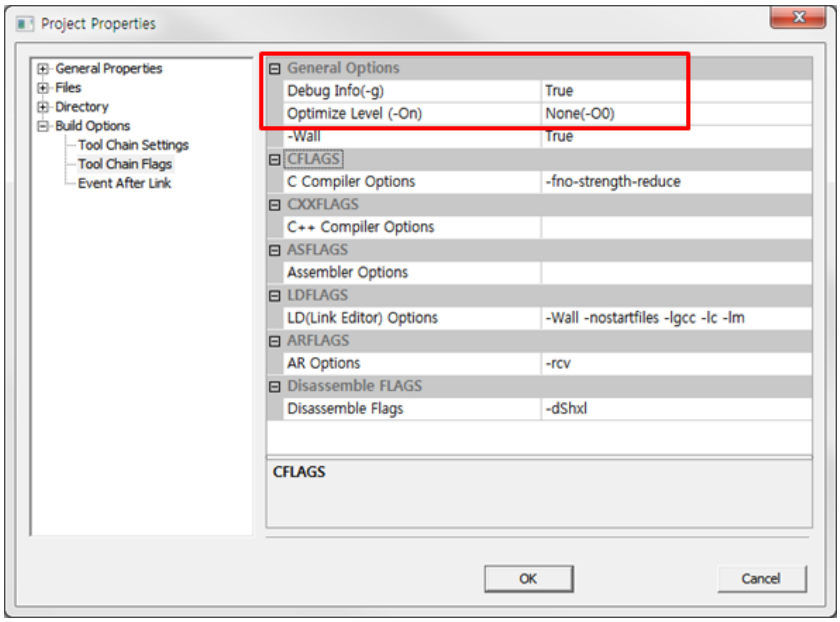
14. Debugging

이번 장에서는 debugging 에 대해서 설명한다.

Debugging 은 E-Con 과 EConMan 프로그램을 필요로 한다. E-Con 을 사용하기 위해서는 device driver 가 설치 되어야 하는데, 설치를 하지 않았다면, "1. Software 개발환경"의 device driver 설치하는 부분을 참고하기 바란다.

14.1 Debugging 준비

E-Con 과 EConMan 이 준비 되었다면 debugging 을 할 project 를 열어 Project → Properties 를 실행한다.



실행하면 위와 같은 창이 뜨게 되고, 빨간 상자 안의 내용, Debug Info(-g)를 TRUE 로 설정하고, Optimize Level(-On)을 None(-O0)로 설정을 한 후 OK 를 누른다. 그런 다음 Build → Rebuild Project 를 실행하여 Rebuild 를 진행한다.

Debugging 을 위해서는 Debug Info 를 TRUE 로 하여 Build 에 의해 생성되는 elf 파일에 Debug 정보를 포함시켜야 하고, Optimize Level 을 None 으로 하여 최적화 없이 compile 해야 debugging 시 보다 정확하게 source 를 debugging 할 수 있다.

14.2 Debugging

Debugging 은 다음의 두 가지 상황에 따라 시작하는 방법이 다르다.

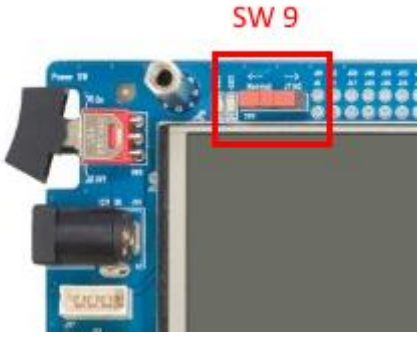
1. Bootloader 없이 0 번지(Serial Flash)에서 Program 이 실행되는 경우.
2. Bootloader 를 사용하여 Ram 에서 Program 이 실행되는 경우.

위의 두 경우는 linker script 를 가지고도 구분할 수 있다. Linker Script 가 amazon2.ld 파일로 되어 있으면 0 번지에서 Program 이 실행되는 것이고, amazon2_ram.ld 파일로 되어 있으면 Ram 상에서 Program 이 실행되는 경우이다. Serial Flash booting 을 사용하는 경우에는 1 번과 2 번 방법 모두 사용가능하고, NAND booting 을 사용할 경우에는 2 번 방법으로 debugging 을 진행해야 한다.

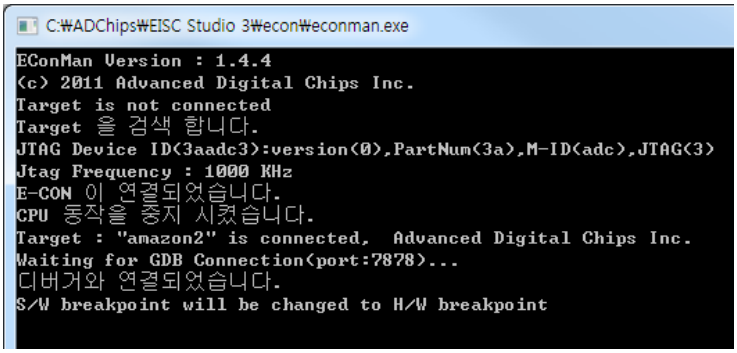
▶ 0번지(Serial Flash)에서 Program이 실행되는 경우

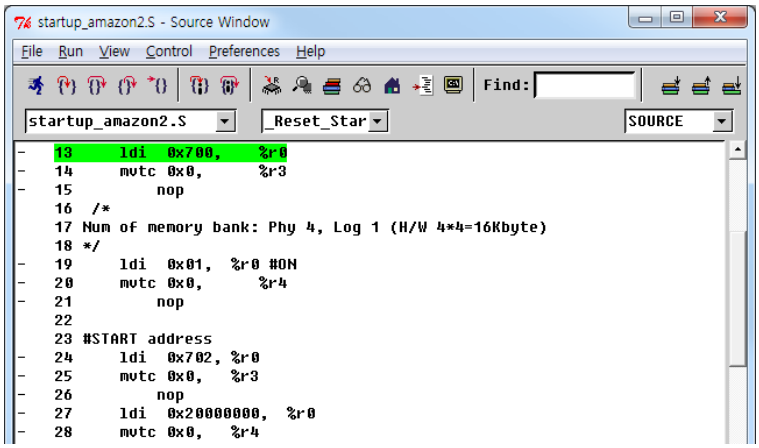
Program 이 0 번지(Serial Flash)에서 실행되는 경우 다음과 같이 debugging 을 진행한다.

1) Build 하여 생성 된 bin 파일을 0 번지에 download 한다.
2) SW9 을 설정하여 Debug mode (JTAG mode)로 놓은 상태에서 E-Con 을 연결하고 전원을 켠다. (SW9 을 JTAG 쪽으로 놓으면 Debug mode 상태이다.)



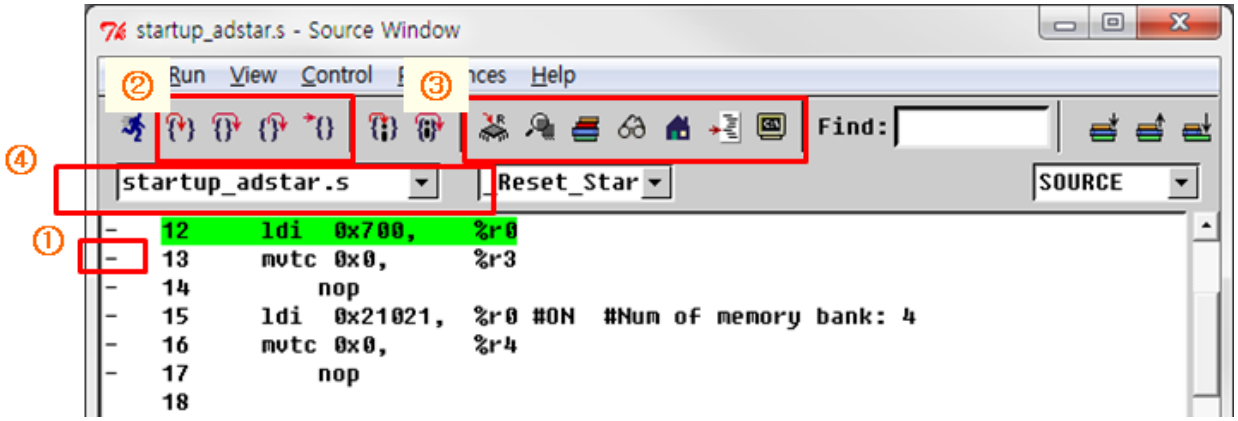
3) Debugging 할 project 가 열려있는 상태에서 Debug → Start Debugger (F5)를 실행한다. 그러면 다음과 같이 두개의 창이 뜨면서 debugging 을 시작할 수 있다.





위와 같은 창이 뜨고, 녹색 바가 표시되면 정상 연결된 것으로 debugging 을 시작할 수 있다. 녹색 바가 아닐 경우 연결이 제대로 안된 것으로 다시 시도해야 한다.

4) Debugger 창에 대해 설명하면 다음과 같다.



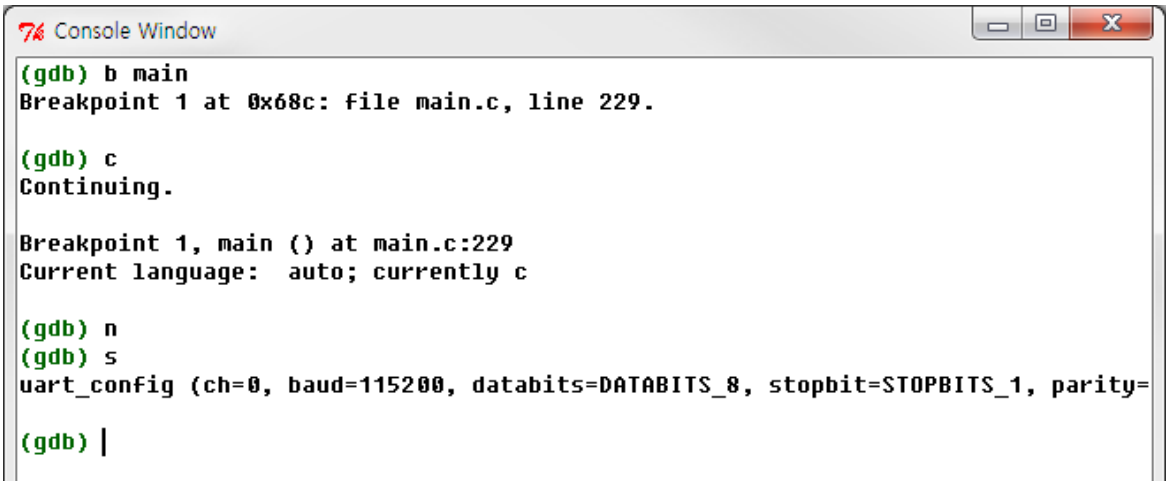
① : 마우스로 '-' 클릭 시 break point 설정/해제

② : 앞에서부터 'step', 'next', 'finish', 'continue' 동작

③ : 앞에서부터 'register', 'memory', 'stack', 'watch expressions', 'local variable' 확인창 출력. 제일 마지막 'c: '는 'console'창을 띄운다. Console 창에서는 GDB 명령을 사용하여 debugging 을 수행할 수 있다.

④ : Source code 이동.

5) console 창에서의 debugging.

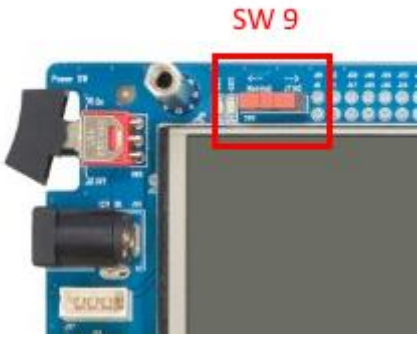


b : break point 설정.
c : continue 명령. 다음 break point 까지 진행.
n : next 명령. 하나의 코드 실행. (함수일 경우에도 그대로 진행)
s : step 명령. 하나의 코드 실행. (함수인 경우 함수 내부로 들어가 진행)
q : 종료.

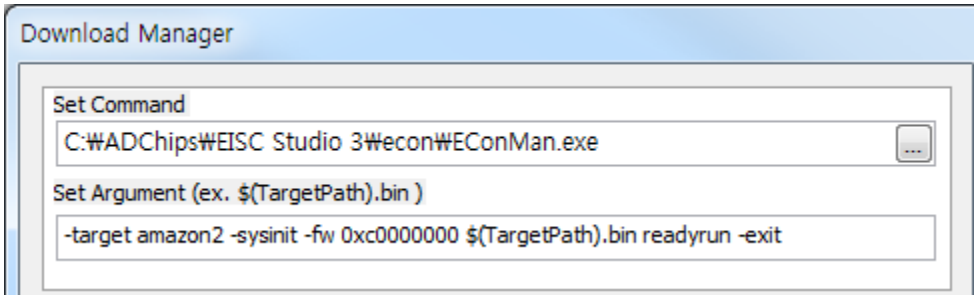
▶ Ram에서 Program이 실행되는 경우

Bootloader 를 사용하는 경우 Program 이 Ram 에서 동작하게 되는데, 이 경우에는 이전과 debugging 을 시작하는 방법이 조금 다르다.

1) Bootloader 가 download 되어 있는 상태에서, SW9 을 설정하여 Debug mode (JTAG mode)로 놓은 상태에서 E-Con 을 연결하고 전원을 켠다.
(SW9 을 JTAG 쪽으로 놓으면 Debug mode 상태이다.)



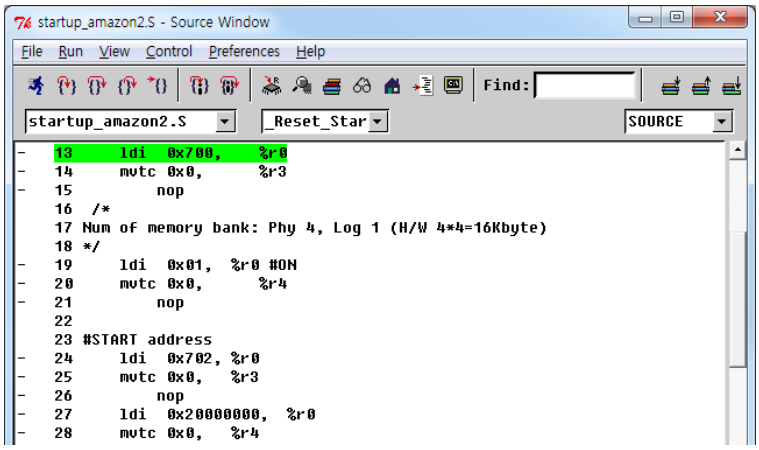
2) Debugging 할 project 를 열어서 build 를 한 후 download option 을 실행하여 다음과 같이 argument 를 적어준다.



-target amazon2 –sytsinit –fw 0xc0000000 $(TargetPath).bin readyrun –exit

(-fw 는 파일을 원하는 ram 영역에 write 하는 명령이고, 끝에 readyrun 을 쓰게 되면, ram 영역에 write 된 프로그램이 처음 실행되는 위치에서 대기하게 된다. 이렇게 실행 위치에 대기시켜 놓은 뒤에 Start Debugger 를 실행하여 debugging 을 수행할 수 있다.)

3) 위 처럼 Download option 을 작성한 후 Download to Target 을 실행하여 프로그램을 ram 영역에 download 하고, Debug → Start Debugger (F5)를 실행한다. 그러면 다음과 같은 창이 뜨면서 debugging 을 시작할 수 있다.



4) 이제부터는 이전 0 번지에서 동작하는 program 을 debugging 할 때와 같은 방법으로 진행하면 된다.