BW BOARD
사용 설명서

ARM7 CPU 개발 보드

(SAMSUNG S3C44B0X CPU BOARD)

V1.01

2003년 11월 13일

CPU BOARD 사양

CPU	SAMSUNG S3C44B0X 16/32bit RISC architecture ARM7TDMI(up to 66Mhz) CPU 내부에 다양한 주변기능 내장 - External memory controller.(SDRAM사용가능) - LCD controller(up to 256 color DSTN, LCD전용DMA내장) - 2-ch general DMAs / 2-ch peripheral DMAs - 2-ch UART, 1-ch IIC, 1-ch IIS, 5-ch PWM timers & 1-ch internal timer - Watch Dog Timer - 71 general purpose I/O ports / 8-ch external interrupt source - 8-ch 10-bit ADC. - RTC with calendar function. - On-chip clock generator with PLL. - Power control: Normal, Slow, Idle, and Stop mode
Memory	RAM – SDRAM 32Mbyte(256Mbit), 16bit data bus FLASH – 1Mbyte(8Mbit) 16bit data bus(4Mbyte까지 확장가능)* EEPROM – 2Kbyte(16kbit) Serial ROM(IIC사용)
I/O	- CPU pin out to header connector - 8 switch input/8 LED out - Buzzer
기타	- RS232C 1 port - RTC Battery 내장 크기 114mmX114mm
CPU Board OPTION**	CPLD KIT - XILINX 95144XL 장착 - IDE HARD DISK READ/WRITE가능

* 기존의 Flash(29LV800)를 떼어낸후 2Mbyte or 4Mbyte Chip으로 부착가능

** CPLD KIT옵션 추가시 Board내에 CPLD를 장착하여 출고. 이 옵션이 선택지 않으면 CPLD자리에는 빈공간으로 출고

기타옵션

1. MONO LCD KIT

- 320X240 MONO LCD Module과 LCD Interface 보드로 구성
- 16 Gray Scale
- BACK LIGHT는 달려있지 않음
- 영문폰트(8X16),한글폰트(16X16)관련 소프트웨어 제공

2. COLOR LCD KIT(현재 개발중)

- 640X480 STN COLOR Module과 LCD Interface보드로 구성
- 256 Color
- BACK LIGHT포함
- 영문폰트(8X16),한글폰트(16X16)관련 소프트웨어 제공

제공내용물

1. 기본형

   CPU BOARD

   ROM BOARD

   RS232C Cross Cable

   전원어탭터

   JTAG DEBUGGER

   25Pin 연장케이블

   SOFTWARE CD

2. CPLD KIT

   기본형에 다음물품 포함

   CPLD(95144XL)와 CPLD관련 부품이 부착된 CPU BOARD

   JTAG CABLE

   SW&LED Board

3. MONO LCD KIT

   기본형에 다음물품 포함

   320X240 Mono LCD Module

   LCD Interface 보드

   LCD 연결케이블

4. COLOR LCD KIT

   기본형에 다음물품 포함

   640X480 COLOR LCD Module

   LCD Interface 보드

   LCD 연결케이블

BW BOARD의 소개

BW BOARD(S3C44B0X)는 ARM7TDMI CPU를 기본으로 제작된 연습용 보드로 ARM CPU를 사용법을 처음 배우기 위한 용도에서부터 특정 응용분야에 적용이 가능한 용도까지의 모든 분야에 사용 가능한 CPU보드입니다.

삼성 S3C44B0x CPU는 ARM7TDMI Core를 내장한 CPU로 PDA에 적용을 목적으로 만들어진 다양한 기능을 내장한 CPU입니다. 현재는 PDA시장에 ARM9과 Xscale계열의 CPU가 주류를 이루고 있어 S3C44B0X를 가지고 PDA를 만들지는 않지만, 그래도 다양한 기능을 내장한 이유로 PDA가, 아닌 Embedded System분야에서 많이 사용되는 CPU입니다.

이 BWBOARD는 OS를 사용하지 않는 단순한 Firmware Level의 환경에서 32Bit RISC CHIP의 뛰어난 능력과 속도를 사용자가 직접 컨트롤 할 수 있게 제작된 보드로 이 보드를 사용하는데 필요한 자료는 모두 공개할 예정입니다.

이 보드와 관련된 소프트웨어 소스는 모두 제공된 CD에 포함되어 있으며 지속적으로 홈페이지를 통해서 업그레이드할 예정입니다.

이 보드를 사용하는데 편리한 개발환경을 구축하기 위하여 시리얼포트를 통한 프로그램 Image 다운로드/Flash Write 프로그램 및 실무에 바로 적용가능 한 기본 소스가 제공되며 이와 함께 저렴한 JTAG Debugger도 함께 제공됩니다. 그리고 만일의 경우 Flash메모리의 내용이 지워졌을 경우를 대비하여 Flash메모리를 초기화시키는 롬 보드를 제공합니다.

또한 BWBOARD에는 선택사양으로 CPLD를 장착하여 CPLD를 배우거나, CPLD를 이용한 응용분야에 곧바로 적용이 가능한 보드이며, CPLD KIT는 'Xilinx CPLD ISP CABLE'이 포함되어 제공됩니다. 그리고 CPLD의 I/O핀의 동작을 간단히 확인할 수 있는 LED&SW보드도 제공합니다. CPLD KIT를 이용할 경우 ARM7TDMI에서 IDE HDD(ATA3이상)를 읽거나 쓸 수 있습니다.(LBA단위로 READ/WRITE가 가능하며 28bit LBA를 지원합니다. 현재 File System은 구현되어 있지 않습니다.)

이 이외에 더 자세한 사항은 본 사용설명서상의 각각의 부분을 참고하거나 홈페이지, Email등을 참고하십시요.

BW BOARD를 사용하기 위한 필요사항

필수사항

1 Windows 98/ME/NT/2000/XP 환경의 PC

2 Xilinx WebPACK 소프트웨어 – CPKD_KIT를 사용자

- Xilinx사의 홈페이지에서 사용자 등록을 하면 무료로 다운로드 받을수 있습니다.

권장사항

1.컴파일러 – SDT/ADS or IAR EWARM 컴파일러

- 보드와 함께 제공되는 CD에는 IAR EWARM 3.40컴파일러용 소스와 ADS 1.2컴파일러용 소스가 함께 들어있습니다.

ADS 1.2 컴파일러는 ARM사의 홈페이지에 사용자 등록 후 CD를 신청하면 30일간 사용할 수 있는 데모버전 CD를 우편으로 보내줍니다.

IAR EWARM 3.40 컴파일러는 IAR사의 홈페이지에 사용자 등록 후 다운로드 할 수 있습니다. 그리고 국내의 ㈜마이크로비전 홈페이지에서도 다운로드 할수 있습니다.

ARM 홈페이지 : http://www.arm.com/

IAR 홈페이지 : http://www.iar.com/

(주)마이크로비전 홈페이지 : http://www.microvision.co.kr/

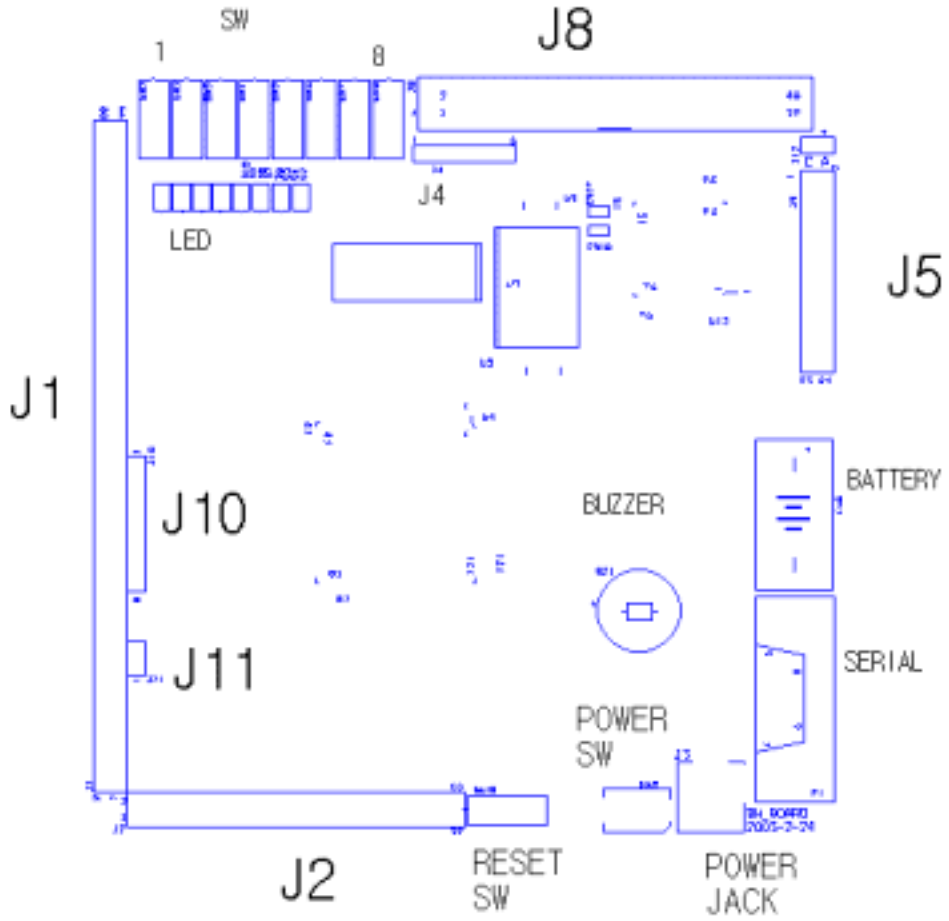
이 사용설명서에서는 IAR EWARM 3.40컴파일러를 기준으로 설명합니다.

2.JTAG Debugger

– Multi ICE, JEENI, MAGIC, Trace32등의 고속의 JTAG 디버깅장비가 있으면 ARM계열의 하드웨어/소프트웨어 개발에 도움이 많이 됩니다.

보드와 함께 제공되는 JTAG DEBUGGER은 IAR EWARM 컴파일러에 들어있는 C-SPY 디버그 소프트웨어에서 사용할 수 있는 wiggler라는 JTAG DEBUGGER와 호환되는 DEBUGGER로 소프트웨어 다운로드속도가 느린 점 이외에는 고속의 JTAG 디버깅장비와 성능은 크게 차이 없습니다. 제한사항이라면 단지 C-SPY에서만 사용할 수 있을 뿐입니다.

BOARD LAY OUT



각각의 부품 설명

J1, J2 – CPU의 Pin을 외부로 확장하기 위한 커넥터

J10, J11 – CPU JTAG PIN Connector

J4 – CPLD JTAG PIN Connector

J5 – CPLD LED&SW Board Connector

J8 – IDE HDD Connector

J4,J5,J8은 CPLD Kit에만 존재하는 핀입니다.

각 핀의 세부사항에 대해서는 CD에 들어있는 회로도 를 참고 하십시요.

각 커넥터에 외부부품을 부착할 경우에는 방향을 특별히 유의해서 연결 하십시요.

잘못 연결되거나 역방향으로 연결될 경우 BW_BOARD 및 외부부품이 파손될 수 있습니다. 이 경우 수리비는 실비로 청구합니다.

함께 제공되는 부속물의 사용법

1 ROM Board

FLASH에 들어있는 모니터 프로그램이 손상되었을 때 ROM보드를 삽입하여 전원을 켜면 모니터프로그램이 FLASH에 WRITE됩니다.

사용방법

- 전원 OFF (부속물의 탈부착은 반드시 전원이 OFF된 상태에서 하십시요)
- 롬 보드를 꼽기 전에 J1의 79,80번 핀의 점퍼를 제거
- 롬 보드의 79,80pin이 CPU보드의 79,80pin과 일치하도록 롬 보드를 삽입
  (역방향 및 다른 번호로 삽입되지 않도록 주의)
- 아래 사진 참조
- PC와 시리얼 크로스 케이블 연결후 시리얼 통신프로그램 실행
  통신 파라미터 115200bps, 8bit,NoParity, No Flow Control
- 전원 ON
- 시리얼통신프로그램 상의 진행상황 관찰
- 전원 OFF
- 롬 보드를 제거한 후에는 반드시 점퍼로 79,80PIN을 연결 하십시요.

롬보드의 사용법

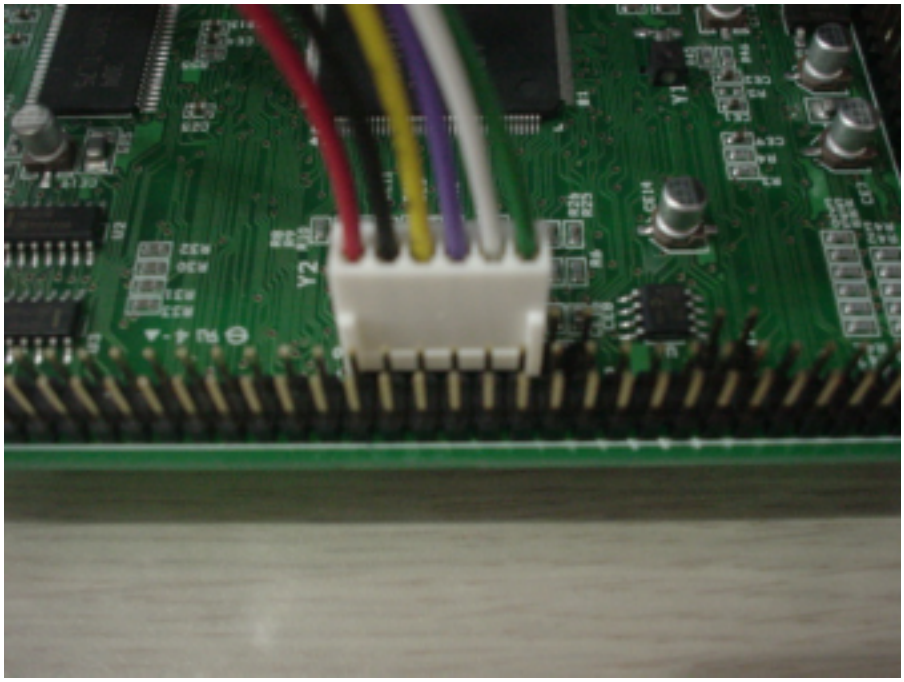
2 JTAG Debugger

IAR EWARM의 C-SPY에서 C/ASM 소스라인 디버깅을 할 때 사용합니다. 이 케이블은 Macraigor사의 wiggler와 호환됩니다.(100%호환되지는 않고, 오로지 BWBOARD에서 C-SPY의 사용 목적으로 제작되었습니다.)

- JTAG Debugger를 함께 제공되는 25Pin 케이블을 통해 PC에 연결한다.

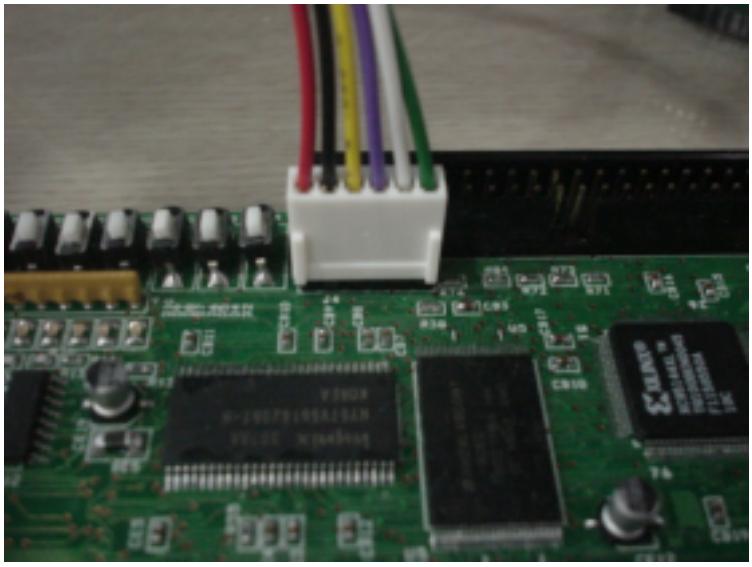
(이때 PC나 BWBOARD의 전원이 꺼져있을 필요는 없습니다. PC나 BWBOARD가 켜진 상태

로 케이블을 연결하거나 끊어도 됩니다.)
- 6핀 커넥터의 빨간선을 J10의 1번핀과 일치하도록 6핀 커넥터를 삽입.
(JTAG Debugger은 6핀이고, J10은 8핀입니다. J10의 7,8핀은 사용하지 않습니다.
6핀 커넥터가 역방향이나, 밀려서 삽입되지 않도록 주의 하십시요)
- IAR EWARM의 C-SPY를 실행 – 자세한 내용은 '**JTAG DEBUGGER 사용법**'란 참조
- IAR EWARM의 화면에서 C-SPY가 **실행 중에 케이블을 연결하거나 끊지 마십시요**.
하드웨어적인 위험은 없지만, C-SPY가 오동작하거나 IAR EWARM이 응답 없음 상태가 되는 원인이 됩니다.



JTAG Debugger의 연결사진

3 Xilinx JTAG Cable(CPLD KIT에 포함)
CPLD에 Configuration Data를 Write할 때 사용합니다. 이 케이블은 xilinx cable이나 Insight IJC2 cable과 100% 호환됩니다. 따라서 Xscale등의 JFLASH에도 사용할 수 있습니다. BWBOARD에서는 xilinx webpack의 IMPACT라는 소프트웨어를 이용하여 CPLD(95144xl)에 데이터를 Write합니다. (이때 PC나 BWBOARD의 전원이 꺼져있을 필요는 없습니다. PC나 BWBOARD가 켜진 상태로 케이블을 연결하거나 끊어도 됩니다.)
사용방법
- Xilinx JTAG Cable를 함께 제공되는 25Pin 케이블을 통해 PC에 연결한다.
(이때 PC가 꺼져있을 필요는 없습니다. PC가 켜진상태로 작업해도 됩니다.)
- 6핀커넥터의 빨간색선을 J4의 1번핀과 일치하도록 6핀커넥터를 삽입.
(6핀커넥터가 밀리거나 역방향으로 삽입되지 않도록 주의하십시요)
- IMPACT를 실행



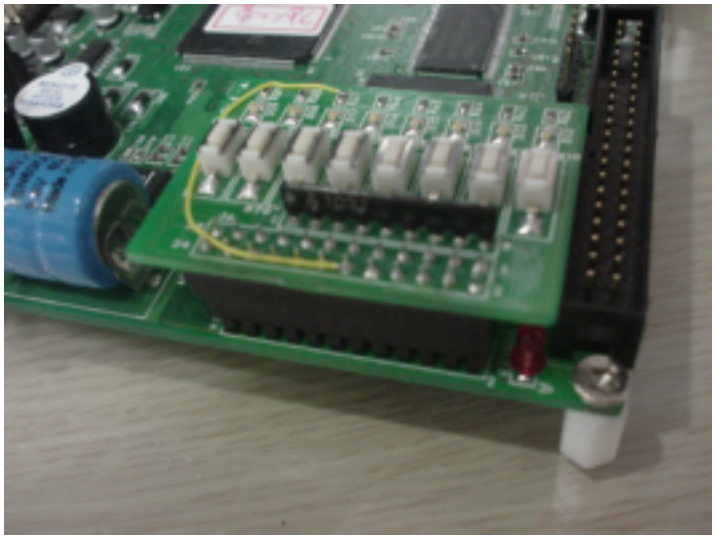
Xilinx JTAG Cable 연결 사진

4. LED&SW Board(CPLD KIT에 포함)

CPLD의 Input/OUTPUT line의 테스트 목적으로 사용합니다.

J5의 1,2번핀이 LES&SW보드의 1,2번핀과 일치하도록 LED보드를 삽입

(역방향으로 삽입되지 않도록 주의 하십시요.)



5. IDE HARD DISK CABLE(이 케이블은 BWBOARD패키지에 포함되어 있지 않습니다)

J8에 40Pin IDE CABLE이 역방향으로 삽입되지 않도록 주의해서 연결합니다.

하드디스크 연결시에는 하드와 CPU보드의 전원소스는 달라도 상관없습니다.

HDD의 40Pin Cable에는 GND라인만 존재하고, VCC라인이 없기 때문에 HDD나 CPUBOARD의 전원이 서로 간섭을 일으키지 않습니다. 단지 하드디스크에 BWBOARD보다 먼저 혹은 동시에 전원이 공급되면 됩니다. BWBOARD는 부팅 초기에 하드디스크를 인식하기 때문이 이때 하드디스크에 전원이 공급되지 않으면 BWBOARD는 하드디스크를 인식하지 못합니다.

CABLE이 역방향으로 삽입되어 오랜 시간 보드에 전원이 투입되어 있으면 CPLD와 BW_BOARD가 파손됩니다. BW보드의 전원을 투입할 때 부저나 전원LED, Board의 LED등이 특별한 반응을 보이지 않으면 전원을 즉시 OFF시키십시요.

보드의 기본적인 사용방법

PC에 시리얼 통신용 프로그램을 실행하고, 통신포트번호를 시리얼 cable이 연결된 포트번호를 지정하고, 통신포트의 속도를 115200bps, 8bit, no parity, '흐름제어 없음'을 선택하십시요. 일반 통신용 프로그램은 윈도우 자체에 내장되어 있는 하이퍼터미널이나 이야기, 새롬데이타맨 등을 사용할 수 있습니다.

PC에서 BW_BOARD로 소프트웨어를 다운로드하기 위해서는 windows NT,2000,XP계열은 반드시 dnw라는 삼성전자에서 제공하는 터미널 프로그램을 사용해야 합니다. dnw는 CD안에 들어있습니다. Windows 98,me계열 사용자는 일반 통신용 프로그램을 사용하며 wkcom2라는 dos command로 소프트웨어를 BW_BOARD로 다운로드 할 수 있습니다.

통신용 프로그램을 실행한 상태에서 BW-BOARD에 시리얼 Cable과 전원을 연결하고 전원스위치를 켜면 통신용 프로그램 창에 부팅되는 메시지가 나타납니다. 부팅메시지가 나타나지 않는다면 케이블이나 전원, 통신프로그램의 설정을 점검하십시요.

BOARD의 FLASH에는 크게 2가지 프로그램이 저장됩니다.

이 2가지는 MONITOR프로그램과 MAIN프로그램으로, 모니터프로그램은 부팅과 FLASH에 DATA를 write하는 것을 관할하고, MAIN프로그램은 실제 사용자가 실행하려는 프로그램입니다.

전원이 켜지면 MONITOR 프로그램이 BOOT절차를 실행하며 보드자체의 기본 설정값들이 필요한 CPU 레지스터 등에 데이터를 쓰고, 모니터모드로 켜질 것 인지 Main 프로그램 실행모드로 켜질 것 인지를 결정합니다.

FLASH에 MAIN프로그램이 존재하면 FLASH는 BOOT후 바로 MAIN프로그램을 실행합니다.

FLASH에 MAIN프로그램이 존재하지 않으면 BOOT후 MONITOR모드로 동작합니다.

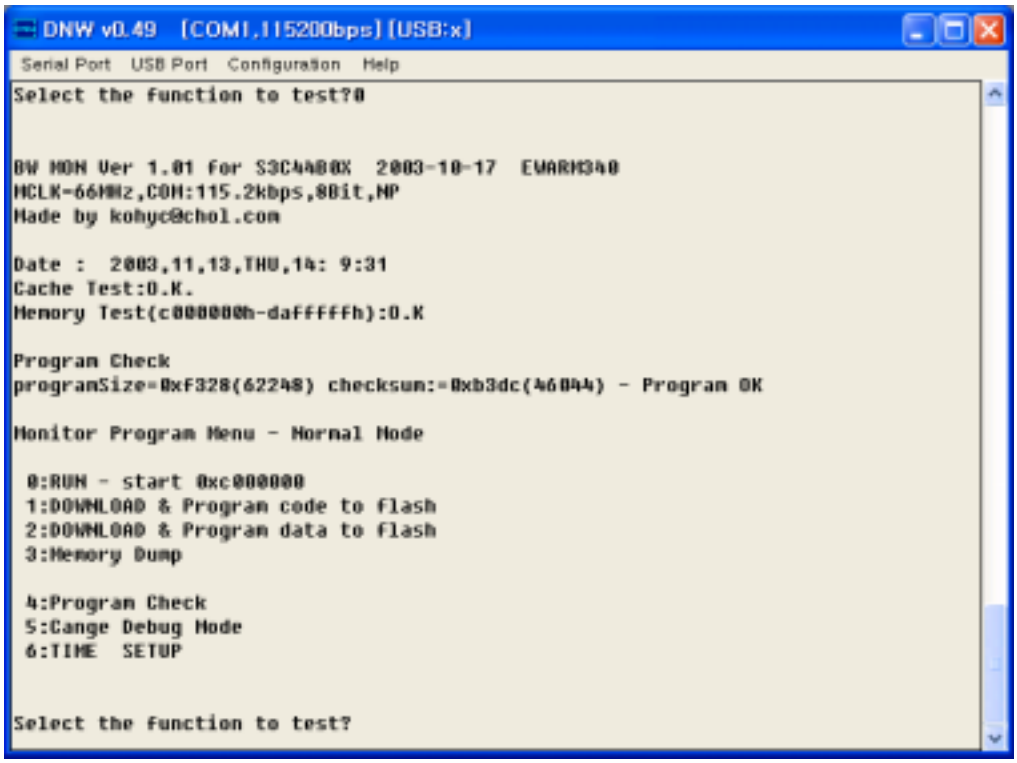
FLASH에 MAIN프로그램이 있을 때 **Monitor모드로 진입하기 위해서는 SW1을 누른 상태에서 전원 스위치를 켜면 모니터 모드로 동작**하여, 통신 화면 창에 모니터모드의 내용을 볼 수 있습니다.

FLASH의 모니터 프로그램영역은 사용상의 부주의 혹은 모니터 프로그램 작성(Boot코드, 모니터 코드의 C언어 수정/컴파일)상의 실수로 지워지거나 부팅될 수 없는 코드가 Write될 수 있습니다. 이 경우에 보드는 부팅되지 않습니다. 이럴 경우 보드와 함께 제공된 롬 보드를 꼽고 부팅을 하면 FLASH에 모니터 프로그램이 다시 WRITE됩니다. WRITE의 진행사항은 통신프로그램 창에서 확인할 수 있습니다.

롬보드를 통한 모니터 프로그램 Write가 끝나면 전원을 끄고, 롬 보드를 제거한 후 반드시 점퍼로 79,80핀을 연결하십시요. 점퍼가 꼽히지 않으면 부팅되지 않고, 회로가 오동작을 일으켜 부팅되는 경우도 있지만, 보드는 정상적으로 작동하지 않습니다.

롬보드로 FLASH를 다시 Write하면, Flash에 들어있던 Main프로그램은 지워집니다.

모니터 프로그램의 사용법



보드가 부팅되어 모니터 모드로 진입하면 간단한 자체테스트를 거쳐 위와 같은 메뉴가 나타납니다.

메인 프로그램이 있을 때 0번을 선택하면 Main프로그램이 실행됩니다.

이 화면에서 1번이나 2번을 선택하면 flash에 원하는 데이터를 write할수 있습니다. 1번을 선택하면 flash에 main 실행프로그램을 write할 수 있고, 2번을 선택하면 binary image를 write할 수 있습니다.

Main프로그램 Image는 flash의 0x10000번지에 4byte의 Main 프로그램크기가 존재하고, 그 뒤에 Main프로그램 Image가 있습니다. 그리고 끝에 2byte가 추가되어 check sum이 있습니다.

1번이나 2번을 선택하면 통신 창에서 프로그램다운로드가 시작되고 보드는 PC의 데이터 전송을 기다립니다. NT/2000/XP에서는 dnw의 **메뉴**의 **Serial Port-Transmit**를 선택한 후 전송하려는 binary image를 선택하면 전송이 시작됩니다. 전송이 끝나면 해당 image가 flash에 write되는 상황을 통신 창에서 확인할 수 있습니다.

1번을 선택해서 다운로드 하면 전송이 끝난 즉시 0x10000번지에 '4byte길이+전송된 bin image+2byte check sum'이 write됩니다.

2번을 선택해서 다운로드 하면 전송이 끝난 후에 전송된 image가 write될 flash의 블록 Address를 입력하는 내용이 나타납니다. 이때 해당 Address를 지정하면 전송된 image는 해당 Address에 write됩니다. Write block address에 0을 입력하면 별도의 password를 묻습니다. 이 password는 '1000'입니다.
0번지에 모니터프로그램 image가 아닌 다른 binary image가 write된다면 보드는 더 이상 켜지지 않을 것 입니다. 이럴 경우에는 롬 보드를 이용하여 모니터 image를 다시 write 하십시요.
0번지(모니터프로그램영역)나 0x10000번지(Main프로그램영역)가 아닌 다른 영역은 사용자가 필요로 하는 데이터-폰트,bitmap,음성등을 다운로드하여 사용하면 됩니다.

3번을 선택하면 메모리 ADDRESS를 입력할 수 있습니다. 여기에 원하는 주소를 입력하면 해당번지의 데이터를 볼 수 있습니다.

4번은 0x10000번지에 Main프로그램이 존재하는지 여부를 check합니다. 프로그램길이만큼의 image date를 더한 값이 마지막의 checksum과 같은지를 확인하여 프로그램 존재여부를 알아냅니다. 하지만 Main프로그램이 아닌 다른 bin image가 1번 'code to flash'를 통해서 다운로드 되었다면 해당 image가 실행 가능한 image인지 아닌지는 알 수 없습니다.
실행 가능한 image인지는 실제로 실행해 봐야 알 수 있습니다.

실행 불가능한 Main 프로그램 image를 1번을 통해서 다운 로드 했을 경우 0번이나 RESET, 전원 투입 등을 통하여 Boot을 시도하면 Boot후에 모니터 모드를 거쳐 main프로그램이 실행되는 과정에서 main프로그램은 실행되지 못하고 Error Message를 통신 창에 표시합니다. 이럴 경우에는 **sw1을 누른상태에서** 부팅하면 모니터모드가 정상적으로 실행되고, 모니터 모드에서 Main image를 다시 다운로드 하면 됩니다.

5번은 Debug Mode/Normal Mode 전환 선택항목입니다. IAR EWARM C-SPY에서 Main프로그램의 C-Source Line Debugging을 하려면 Debug Mode로 설정되어 있어야 합니다.

6번은 CPU의 내장시계(RTC)의 시간을 설정하는 항목입니다. 화면에 표시된 형식으로 입력하면 내장시가의 날짜와 시간이 수정됩니다. 요일은 모니터 프로그램이 자동으로 계산해서 CPU의 레지스터를 Write합니다.

위의 모니터 프로그램 실행화면에서는 Program Check정보와 Normal/Debug모드를 확인할 수 있습니다. 이 두가지 정보를 항상 유의하여 관찰하여 현재 보드의 부팅 상태를 판단하십시요.

기본 소프트웨어의 구성

BWBOARD와 함께 소프트웨어는 크게 3부분으로 구성되어 있습니다.

BW-ROM용 소프트웨어

모니터용 소프트웨어

MAIN용 소프트웨어

이들은 모두 CD의 Source 디렉토리에 있습니다. 이들 각각의 구성과 역할은 다음과 같습니다.

모니터용 소프트웨어

모니터용 소프트웨어는 2개의 소프트웨어(Binary Image)로 구성됩니다.

Bw_boot와 Bw_mon 이 2개의 소스를 통해 각각 생성된 2개의 binary파일이 하나의 image로 합쳐져서 flash의 0번지에 위치합니다.

Bw_boot의 역할은 CPU가 RESET직후 0번지에서 프로그램을 시작할 때, 실행되는 코드로 CPU Boot와 관련된 기본 설정을 한 후, bw_mon image를 램 영역으로 복사한 후, bw_mon을 실행시킵니다. 부팅 중에 실행되는 기본설정에는 WATCH DOG Disable, Clock 설정(20->66Mhz), Memory Control Set, 스택 설정, IRQ Table 설정 등이 있습니다.

Bw_mon을 램으로 복사하는 이유는 모니터프로그램이 Flash영역에서 실행되면서 flash자체에 write할수 없기 때문입니다. 모니터 프로그램이 RAM영역에서 실행되면 Flash는 아무 제한 없이 write가 가능합니다.

Bw_mon은 bw_boot로부터 call되어 실행되면서 모니터모드로 진입할 것인지, Main모드로 진입할 것인지를 결정하여 그에 맞는 실행을 합니다.

모니터모드로 진입할 경우 자체적인 실행을 시작하고, Main모드로 진입할 경우 flash의 0x10000번지에 있는 Main 프로그램 Image를 RAM의 가장 앞부분(0x0c00.0000)에 복사한 후 Main프로그램을 실행시킵니다.

이 두 image는 flash에 아래의 위치는 위치하여 실행됩니다.

0x0 ~ 0x7ff : bw_boot.bin (size 2Kbyte)

0x800~0x87ff : bw_mon.bin(size 32Kbyte)

위 두개의 image를 하나로 합치는 방법은 bw_mon프로젝트의 exe 디렉토리에 있는 batch 파일(*.bat, *.dbg화일 참조)을 통해서 합쳐집니다.

Bw_boot.bin과 bw_mon.bin이 하나의 파일 bwflash.bin으로 합쳐져서 모니터프로그램의 2번 'data to flash'모드나 ROM보드를 통해서 flash의 0번지에 write되면 보드는 정상적으로 부팅되고, 모니터프로그램이 실행될 수 있습니다.

BW-ROM용 소프트웨어

BW-ROM용 소프트웨어는 모니터용 소프트웨어인 bwflash.bin을 flash의 0번지에 write해주는 소프트웨어입니다.
롬 보드가 없을 경우 flash의 0번지(CPU의 BANK 0 ADDRESS)가 CPU의 0번지와 일치하지만, 롬 보드를 꽂을 경우 롬의 0번지가 CPU의 0번지(BANK0)가 되고, flash의 0번지가 CPU의 0x0200.0000(BANK1)에 해당됩니다.
Bw-ROM은 0번지에서 실행되며 롬의 뒷부분의 있는 bwflash.bin을 데이터 image로 처리하여 flash의 0번지(CPU의 0x0200.0000)번지에 write합니다.

Bw_ROM을 컴파일 하여 얻은 bwrom.bin과 bwflash.bin을 하나의 파일로 합쳐서 롬 라이터를 통해 롬에 write하고 해당 롬을 롬 보드를 통해 BWBOARD에 부착하여 부팅시키면 bwflash.bin이 flash에 write됩니다. 롬 보드를 통해 Flash로 write하는 진행사항은 시리얼포트를 통하여 볼 수 있습니다.

이 두 image는 rom의 아래의 위치는 위치합니다.
0x0 ~ 0x77ff : bw_rom.bin (size 30Kbyte)
0x7800~0xffff : bwflash.bin (size 34Kbyte)
0x7800~0x8000 : bw_boot.bin (2Kbyte)
0x8000~0xffff : bw_mon.bin (32Kbyte)

MAIN용 소프트웨어

Main 소프트웨어는 사용자가 실행하려는 작업에 해당하는 소프트웨어입니다. 이 소프트웨어 image는 flash의 0x10004번지에 위치하고, bw_mon에 의해 0x0c00.0000으로 image size만큼(0x10000번지의 정보) 복사된 뒤 실행됩니다.
Main 소프트웨어의 bin image file name은 BW_Main.BIN 입니다.
이 image파일은 bw_main의 디렉토리 밑에 있습니다.

위 3가지에 해당하는 프로그램 소스는 보드와 함께 제공된 CD에 포함되어 있습니다.
각각에 대한 더 자세한 내용은 해당 소스를 참고 하십시요.

컴파일러와 소스에 대한 간략한 설명

BW보드와 함께 제공되는 CD에는 ADS1.2와 EWARM3.40에서 사용할 수 있는 소스가 함께 들어있습니다. ADS나 EWARM은 사용자가 구해서 PC에 설치해야 합니다.

ADS나 EWARM은 C소스의 경우 100%호환이 됩니다.(컴파일러에 종속적인 구문이나 환경변수등을 사용하지 않으면 100%호환이 됩니다. 컴파일러에 종속적인 내용을 C소스에 사용한다면 당연히 호환되지 않습니다. BW_BOARD와 함께 제공된 C소스는 컴파일러에 종속적인 구문을 사용하지 않아 100%호환이 됩니다.)

하지만 ASM소스의 경우 구문에 전혀 다르기 때문에 전혀 호환되지 않습니다. 따라서 BW_BOARD에 사용되는 ASM소스는 _ads.s나 _ewarm.s라는 형식으로 해당 컴파일러에 맞는 소스가 제공됩니다. 예를 들어 'bw_mon_init_ads.s'와 'bw_mon_init.ewarm.s'는 동일한 역할을 하는 소스로 각각의 컴파일러에 맞게 구현된 소스입니다.

ADS 1.2

ADS 1.2용 프로젝트 파일의 확장자는 *.mcp입니다.

CD상의 소스 압축화일을 하드디스크의 적당한 곳에 풀어놓은뒤 ADS1.2를 실행한 후 *.mcp파일을 open하면 해당 프로젝트가 열립니다.

ADS1.2에서 컴파일한 실행 결과는 {project_file}_Data라는 폴더에 *.bin이라는 형태로 실행 화일이 생성됩니다.

ADS용 프로젝트 파일은 bw_boot.mcp, bw_mon.mcp, bw_rom.mcp, bw_main.mcp 이렇게 4가지가 제공됩니다.

위 각각의 프로젝트를 컴파일하여 생성된 binary파일들을 적절하게 BWboard에 Write하면 해당 프로그램이 실행됩니다.

Bw_main.bin은 모니터 프로그램의 1번메뉴를 통해 다운로드 후 flash영역 0x10000번지에 write되면 보드가 부팅되어 Main프로그램 실행시에 실행됩니다.

Bw_boot.bin과 Bw_mon.bin 두개의 파일을 bwflash.bin화일 하나로 합쳐서 모니터프로그램 2번 메뉴를 통해 다운로드 후 flash영역의 0x0번지에 write되면 보드가 부팅될 때 이 image들이 실행됩니다. 두개의 파일을 하나로 합치는 법은 bw_boot.bin을 bw_mon.bin이 있는 폴더로 복사한후에 bw_mon.bin이 있는 폴더의 mk.bat을 실행시키면 하나의 파일 bw_flash.bin이 생성됩니다.

Bw_rom.bin과 bwflash.bin을 하나로 합쳐 ROM.BIN을 생성하고 이 ROM.BIN을 롬라이터를 통해서 27C512롬에 Write한후 이 롬을 롬보드에 꽂은후, 롬보드를 보드에 꽂아서 보드의 전원을 켜면 bwflash.bin이 flash의 0번지에 write됩니다.

Bw_rom.bin과 bwflash.bin을 하나로 합치는 방법은 bwflash.bin파일을 bw_rom.bin이 있는 폴더로 복사한후 해당 폴더의 mk.bat을 실행하면 하나의 파일 ROM.BIN으로 합쳐집니다.

IAR EWARM 3.40

EWARM3.40의 워크스페이스 파일의 확장자는 *.eww입니다.
CD상의 소스 압축화일을 하드디스크의 적당한 곳에 풀어놓은뒤 IAR EWARM 3.40 를 실행한 후 bwboard.eww파일을 open하면 해당 프로젝트가 열립니다.
Bwboard.eww에는 4개의 프로젝트-bw_boot,bw_mon.bw_rom,bw_main이 포함되어 있습니다. 메인창에서 원하는 프로젝트 탭을 선택한 후 컴파일을 하면 실행화일을 얻을수 있습니다. 생성되는 실행화일포멧은 xlink옵션에서 지정할수 있습니다. EWARM은 다양한 포멧으로 실행화일을 생성하는데, BW_BOARD에서 사용하는 bin image을 얻기 위해서는 'Intel-extended' 포멧을 지정하여 *.a79라는 헥사화일을 얻은후에 이 헥사파일을 HEXBIN을 이용하여 bin파일로 변환합니다. 이렇게 해서 얻은 각각의 Image파일을 위의 ADS1.2컴파일러 설명에서 설명한 것처럼 각각의 상황에 맞게 보드에 write하여 실행하면 됩니다.

EWARM을 사용하면 BWBOARD와 함께 제공된 JTAG DEBUGGER를 사용할 수 있습니다. 이 JTAG DEBUGGER은 EWARM에 내장된 C-SPY라는 디버거 소프트웨어에서 사용가능한 하드웨어 디버거입니다. C-SPY와 JTAG DEBUGGER를 사용한다면 ARM CPU에서 코드가 실행되는 것을 실시간으로 모니터하여 실행과 관련된 C/ASM 소스, 메모리, 레지스터, 전역/지역변수등의 내용을 알수 있습니다. 더 자세한 내용은 **'JTAG DEBUGGER 사용법'** 을 참고하십시요.

소스에 대하여

bw_boot, bw_mon, bw_rom의 소스는 별도의 수정이 필요없는 소스입니다. CPU의 부트 절차나 flash write기능등이 필요한 사용자가 스스로 직접 분석하여 수정해서 사용하시면 됩니다. 이 세가지 소스에 대한 자세한 분석과 설명은 ㈜마이크로비전에서 개최하는 교육에서 들을수 있습니다.

사용자가 주로 수정할 부분은 bw_main입니다.

Bw_main에 들어 있는 주요 기능은 기본형 보드에는 부저/KEY/LED/RTC/Serial EEPROM등과 관련된 기초 테스트기능이 들어 있습니다.

여기에 CPLD_KIT 보드에는 CPLD와 관련된 기능- IDE HDD Read/Write, CPLD용 SW&LED보드 테스트 기능 등이 들어 있습니다.

LCD_KIT를 사용한다면 Mono LCD나 Color LCD KIT에 한글/영문 폰트 테스트와 간단한 그래픽 library가 들어 있습니다.(Color LCD KIT는 현재 개발중입니다.)

IIS_KIT는 현재 개발중인 기능으로 필립스 UDA1341 Audio Codec을 이용한 음성 녹음/재생기능이 시험적으로 들어있습니다.

IAR EWARM의 사용법

여기서는 IAR EWARM 3.40과 bw_board(main program)을 기준으로 새로운 프로젝트를 생성하고 컴파일하여 실행 image를 얻는 방법에 대해서 설명합니다.

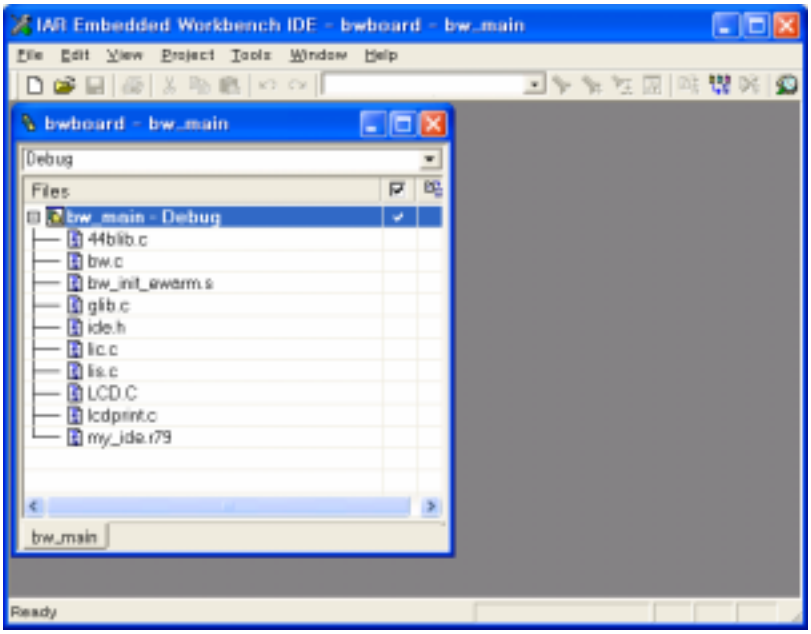
새로운 프로젝트 생성법

EWARM을 설치한 후 IAR Embedded Workbench IDE를 실행합니다.

- 메뉴에서 File-New-Workspace를 선택한 후 확인클릭
- New Workspace 창에서 적당한 폴더(빈폴더)를 지정하여 파일이름을 'bwboard' (workspace name)로 지정한 후 create버튼 클릭
- 메뉴-Project- Create New Project를 선택하여 나오는 창에서 새로운 폴더를 만들어서(마우스 오른쪽 버튼) 폴더 이름을 bw_main으로 지정한후 bw_main폴더로 이동하여 파일이름을 'bw_main' (project name)로 지정한 후 create버튼 클릭
- 메뉴의 File-save all를 이용하여 워크스페이스/프로젝트 저장.
- 제공된 CD에 들어있는 압축소스를 하드디스크에서 풀어낸 후에 bw_main폴더 아래에 있는 src폴더와 inc폴더를 윈도우 탐색기등을 이용하여 방금 생성한 bw_main폴더로 복사
- EWARM IDE에서 메뉴의 project-Add files를 선택한 후 SRC폴더를 선택하여 SRC 폴더에 있는 bw_init_ewarm.s, *.c, my_ide.r79파일을 ADD.(iic_int.c는 ADD하지 마십시요.)
- CPLD kit를 사용하여 IDE HDD를 엑세스 하려면 my_ide.r79 파일도 Add 하십시요.

CPLD kit가 없거나 IDE HDD를 사용하지 않으려면 my_ide.r79를 추가하지 않아도 됩니다.

그러면 다음과 같은 화면이 됩니다.



옵션지정

- 메뉴의 Project-Options를 선택하면 이 프로젝트의 옵션을 지정할 수 있습니다.

(위의 화면에서 bw_main-Debug라인에서 오른쪽마우스 버튼을 눌러도 옵션을 지정할 수 있씁니다.)

- Options의 General category

target에서 Process Variant를 ARM7TDMI를 지정.

Generate interwork code는 선택하지 않습니다.

Code model-Large, Endian-little, Process mode-ARM, Stack Align-8byte를 지정

ICCARM이나 AARM에서는 특별한 options 지정이 필요 없습니다.

각소스 화일별로 list파일을 보려면 list파일 생성을 지정 하십시요.

XLINK category

- Output tab에서 Format을 other에 intel-extended를 지정
- List tab에서 Generate linker listing을 선택하여 map파일을 출력하도록 설정
- Include tab에서 Ignore CSTARTUP in Library 선택
  XCL file name에서 Overide Default를 선택
  그리고 보드 원본소스(CD에서 가져온 소스)의 bw_main폴더에 있는 'bw_main_lnk.xcl' 파일을 이 프로젝트의 폴더로 복사한후에 '…'아이콘을 눌러서 이 파일을 지정토록함.
  (원래는 '…'아이콘 위에 있는 벌레아이콘을 눌러서 GUI환경에서 xcl파일의 내용을 지정해서 xcl파일을 새로이 생성해야 하지만 EWARM3.40에서는 어떤이유에서인지 이 아이콘을 누르면 에러메시지를 생성하며 GUI환경의 xcl화일 생성창을 볼수 없습니다. 따라서 어쩔수 없이 xcl파일을 일반 에디터로 편집하여 위와 같이 지정합니다. xcl화일 편집에 대한 자세한 내용은 '***xcl파일에 대하여***'를 참고하십시요. **그리고 XCL파일의 구성과 설정방법이 ARM CPU 컴파일러 사용법의 거의 대부분을 차지한다는 것을 먼저 알고계십시요.**)

C-SPY category

Setup tab에서 Driver를 Macraigor로 지정, Runto 를 선택안함

Macraigor tab에서 OCD Interface device를 wiggler로 지정

OK버튼을 눌러서 옵션설정을 저장

메뉴의 File-SaveAll을 선택하여 지금까지의 내용 저장.

컴파일

메뉴의 Project-Make선택

Message창에서 진행사항이 표시되며 컴파일 종료후 Error/Warning메시지가 표시됨.
Error가 없으면 Debug폴더의 Exe폴더에 'bw_board.a79' 파일 생성

Bw_board.a79파일은 인텔 헥사 포멧의 ASCII파일이다.
이 파일을 hex2bin 유틸리티를 이용하여 bin파일로 변환한다.
명령프롬프트에서 다음과 같이하여 bin파일로 변환한다.

C: \ ~ ~ ~ \ debug \ exe>hexbin bw_board.a79 bw_board.bin i

이렇게 하면 최종적은 bw_board.bin이라는 binary image를 얻는다.
이 image파일을 모니터 메뉴의 1번 code to flash를 이용하여 flash에 write한다.

보드를 RESET시키면 bw_board가 실행되는 내용이 통신 창에 나타난다.

참고사항)
Project의 Inc폴더에 있는 option.h의 내용에 따라서 보드의 옵션상태를 조절하여 컴파일을 할 수 있습니다.
CPLD옵션이나 LCD옵션 키트,음성키트를 사용하려면 해당 옵션이 있는 줄의 코멘트('//')를 제거하여 해당 define이 유효하게 만들면 됩니다.

```
//#define CPLD_KIT   // if you have CPLD_KIT, uncomment this line
//#define LCD_KIT   // if you have LCD_KIT, uncomment this line
//#define IIS_KIT   // if you have IIS_KIT, uncomment this line
```

Mono LCD를 사용할 경우는 LCD_MONO가 유효하도록 다음 라인들 중에서 코멘트('//')를 제거하면 됩니다.

```
#ifdef LCD_KIT
    //#define LCD_MONO
#endif
```

LCD_MONO가 유효하지 않으면(define 되어 있지 않으면) COLOR LCD모드로 컴파일됩니다.

IDE_HDD를 ACCESS하려면 IDE_HDD가 유효하도록 다음라인들중에서 코멘트('//')를 제거

하면됩니다.

```
#ifdef CPLD_KIT
    //#define IDE_HDD
#endif
```

LCD Kit(Mono or Color)를 가지고 있지 않아도 LCD_KIT를 define하여 컴파일하여 보드에서 실행하여도 실행에는 지장이 없습니다. 다만 binary size만 커질 뿐입니다.
CPLD Kit를 가지고 있지 않은 상태에서 CPLD_KIT를 define하여 컴파일하여 보드에서 실행하면 실행상에는 소프트웨어나 하드웨어적인 문제가 없지만, 동작상에 지장이 있습니다. 자세한 내용은 직접 실행해 보면 확인할 수 있습니다.

JTAG DEBUGGER 사용법

IAR EWARM 컴파일러에 포함되어 있는 C-SPY 라는 디버거와 BWBOARD에 포함되어 있는 JTAG DEBUGGER를 이용하면 CPU의 프로그램 소스를 실행중에 직접 디버깅할 수 있습니다. J

TAG DEBUGGER는 Macraigor사에서 판매하는 wiggler라는 JTAG Cable과 호환되는 케이블로 C-SPY 옵션에서 wiggler로 지정을 하면 디버깅 환경을 구축할수 있습니다.

여기서는 이에 대한 절차를 설명합니다.

(참고1) IAR EWARM 컴파일러의 데모버전에는 컴파일러와 C-SPY가 함께 제공됩니다. 하지만 실제 판매되는 버전은 Full버전과 LE버전으로 나누어서 판매되고, LE버전에는 C-SPY는 들어있지 않고 가격이 쌉니다. 디버깅 소프트웨어가 필요하다면 (C-SPY를 이용하려면) 컴파일러를 구입할 때 Full버전을 구입해야 합니다.

(참고2) BWBOARD와 함께 제공되는 JTAG DEBUGGER는 Wiggler에서 nTRST line이 빠진 제품입니다. BWBOARD의 사용중에는 nTRST단자가 특별히 필요하지 않기 때문에 nTRST line를 제거했습니다.

- JTAG 디버깅을 하기 위한 준비사항

JTAG DEBUGGER는 프린터 포트를 사용하여 JTAG LINE(TDI/TMS/TCK/TDO)를 컨트롤하기 때문에 윈도우 환경의 소프트웨어(C-SPY)가 프린터 포트를 직접 제어하게 됩니다.

하지만 Win XP/2000/NT 계열의 OS는 응용프로그램이 하드웨어(프린터포트)를 직접 제어하는 것을 허용하지 않기 때문에 별도의 프린터 포트 제어 디바이스 드라이버를 설치해야 합니다. 이 디바이스 드라이버는 wiggler를 만든 Macraigor사에서 제공하는 ocp_dbgr.exe라는 low_level 디버거를 설치하면 PC에 자동으로 설치됩니다. 이 설치 화일은 Macraigor사의 홈페이지(http://www.ocdemon.net/)의 Free software메뉴에서 OCD Commander windows 버전을 구하거나 IAR EWARM CD내에서 구할수 있습니다.

Windows 98/Me계열은 응용프로그램이 프린터포트를 직접 제어할 수 있으므로 별도로 ocp_dbgr을 설치할 필요는 없습니다. 하지만 low_level 디버깅 환경을 사용해 보려면 설치해도 됩니다.

JTAG DEBUGGER의 실제 사용

BWBOARD에서 JTAG DEBUGGER를 사용하는 방법은 크게 다음과 같은 두 가지로 사용 할 수 있습니다.

첫번째 Symbol을 이용하는 방법

Symbol을 이용하는 방법은 MAIN IMAGE를 DNW, 시리얼 케이블, 모니터 프로그램을 이용하여 FLASH에 저장한 후 실행시켜 디버깅하는 방법입니다.

이 방법은 JTAG DEBUGGER을 이용하여 다운로드 하지 않기 때문에 다운로드시간을 절약하는 장점이 있지만, 실행이미지(.bin화일)를 얻기 위한 컴파일/HEXBIN변환 절차와 Debug Image(.d79 Symbol 정보)를 얻기 위한 컴파일 절차 두 번의 컴파일 과정이 필요하고 C-SPY의 디버깅 환경이 C-SPY를 종료할 때 저장되지 않는다는 단점이 있습니다.(C-SPY환경이 저장되지 않는 이유는 SDRAM에 존재하는 실행 IMAGE가 JTAG DEBUGGER를 통해서 Write된 Image가 아닌, 모니터프로그램이 Flash에서 복사해온 Image라서 디버깅정보가 다르기 때문입니다.)

두번째 SDRAM에 직접 다운로드 하는 방법.

이 방법은 JTAG DEBUGGER를 이용하여 실행이미지를 SDRAM에 직접 Write하는 방식입니다. 이 방법을 사용하기 위해서는 모니터 프로그램에서 Flash메모리의 Main영역(0x10004번지이후)을 SDRAM의 앞부분으로 복사하는 부분을 삭제해야 합니다. 디버깅을 시작할 때 JTAG CABLE을 통해서 실행 Image를 SDRAM영역에 Write한 후 C-SPY의 메뉴에서 CPU를 RESET시키고 GO 를 시키면 0x0번지의 boot프로그램과 모니터 프로그램을 거쳐 메인 프로그램이 실행되는데, 이 과정에서 모니터 프로그램이 Flash에 있는 Main Image를 SDRAM영역에 복사하는 것을 방지하기 위하여 모니터 프로그램 소스상에서 Main영역 복사 부분을 필요에 따라 생략해야 합니다. 이런 복사부분을 생략하는 모드가 모니터 프로그램상의 5번메뉴를 통해서 선택하는 **JTAG_DEBUG MODE**입니다.

모니터 프로그램이 JTAG_DEBUG MODE이면 flash의 Main부분을 SDRAM으로 복사하는 부분은 생략됩니다. 따라서 C-SPY로 디버깅할 때 C-SPY가 JTAG DEBUGGER을 통해서 SDRAM에 Write한 Image가 실행됩니다. 만일 C-SPY를 사용하지 않으면서 모니터 프로그램의 모드가 JTAG_DEBUG MODE이면 CPU는 모니터를 실행한후 메인을 실행하기 위해 SDRAM영역으로 진입할 때 SDRAM영역에 아무 데이터도 존재하지 않기 때문에 Main프로그램이 정상적으로 실행되지 않습니다. 반대로 모니터프로그램모드가 JTAG_DEBUG MODE가 아닌 NORMAL MODE이면서 C-SPY를 실행한다면, C-SPY가 프로그램 Image를 SDRAM에 Write한후에 C-SPY메뉴에서 RESET을 걸고, GO를 실행하면, 모니터프로그램이 Flash의 Main Image를 SDRAM에 복사하여 실항하게 됩니다. 따라서 C-SPY가 다운로드한 Image는 지워지게 되어 C-SPY의 정상적인 실행이 불가능합니다.

이 두가지 방법에 대한 구체적인 절차는 다음과 같습니다.

1. Symbol을 이용하는 방법

1. BW_Main(Main Image)의 내용을 컴파일하여 bw_board.a79(hex파일)을 생성하고, hex2bin을 통해 bw_board.bin을 생성한 후 dnw를 이용하여 code영역에 download 한다.

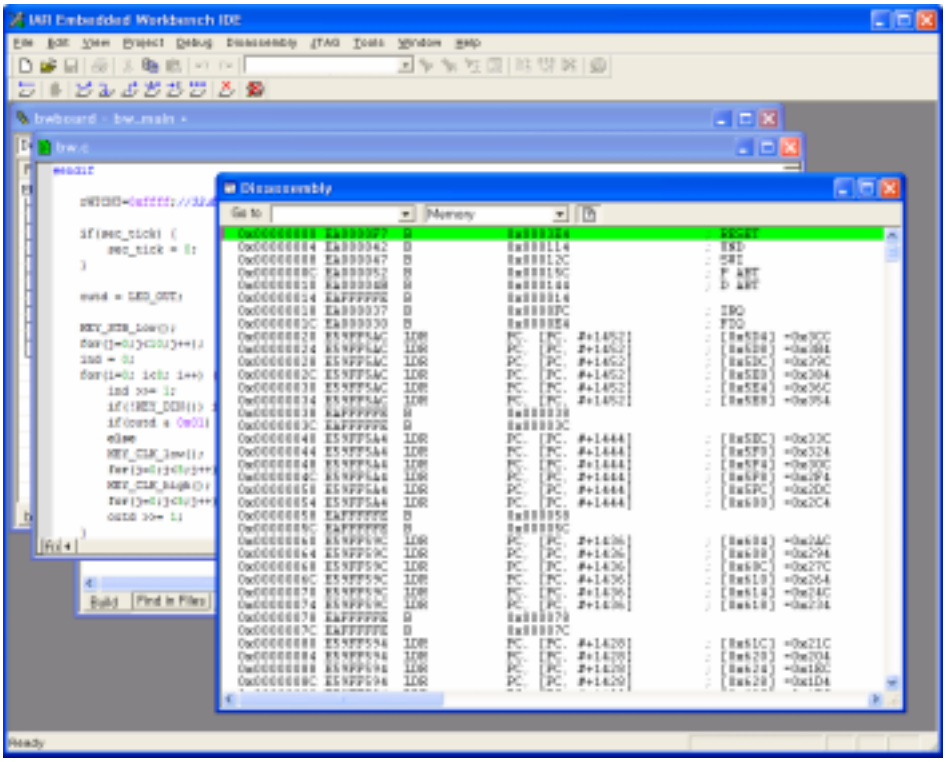
2. EWARM의 메뉴의 project-options-xlink를 선택하여 출력 포멧을 debug info로 지정한후 컴파일하여 bw_board.d79를 생성한다.(bw_board.bin 과 bw_board.d79는 같은 소스로 컴파일한 이미지여야 합니다. 두 화일의 소스가 다르면 정상적인 디버깅은 불가능합니다)

3. EWARM의 메뉴의 project-options-C-SPY옵션에서 Macraigor tab에서 Suppress Download를 선택한다.

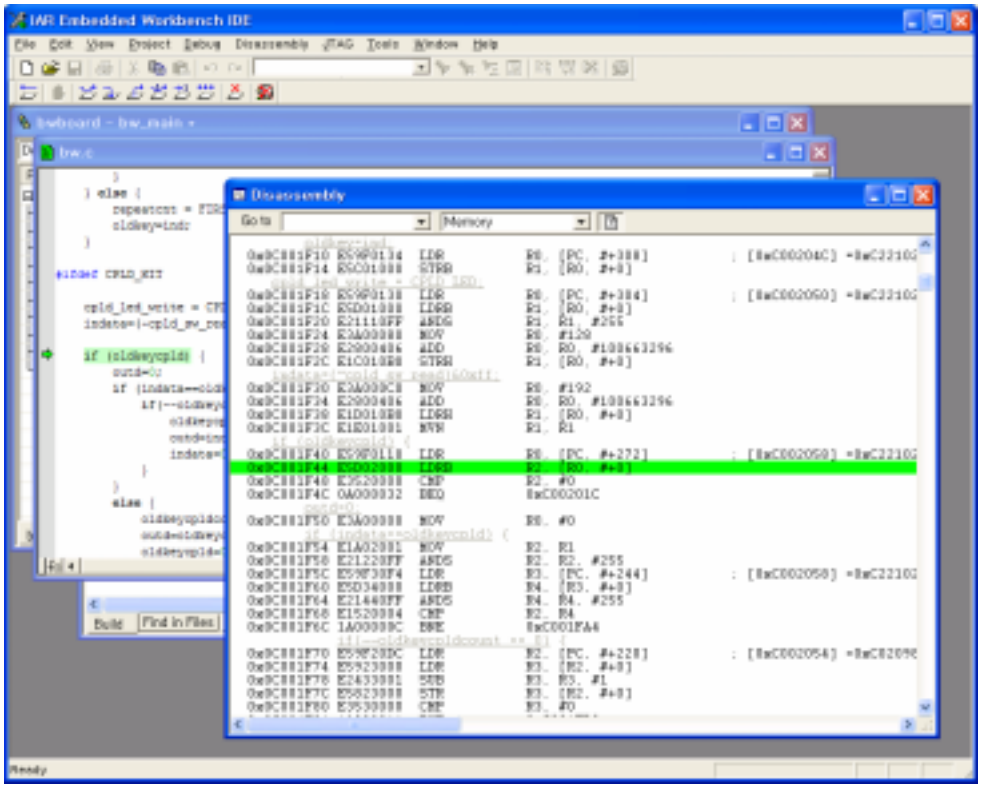
(이 선택은 C-SPY 시작시에 실행 image를 SDRAM으로 다운로드 하지 않고, 단지 CPU의 Register내용과 메모리내용만을 가지고 디버깅을 하는 것을 지정하는 선택이다. CPU가 실행되는 동안 CPU의 Program Counter Register를 참조하여 현재의 실행 번지를 구하고, 이 실행번지를 C-SPY내부에서 Debug Table의 Symbol영역 정보를 이용하여 현재 실행되는 시점의 C-Source Line을 파악하게 해준다.)

4. BWBOARD와 JTAG Debugger를 연결하고, JTAG Debugger를 PC의 패러럴 포트에 연결하고 BW_BOARD의 전원을 켠다. BWBOARD와 시리얼포트를 연결하고, 통신용 프로그램도 실행시킨다. 보드의 메인 이미지는 자동으로 실행된다.

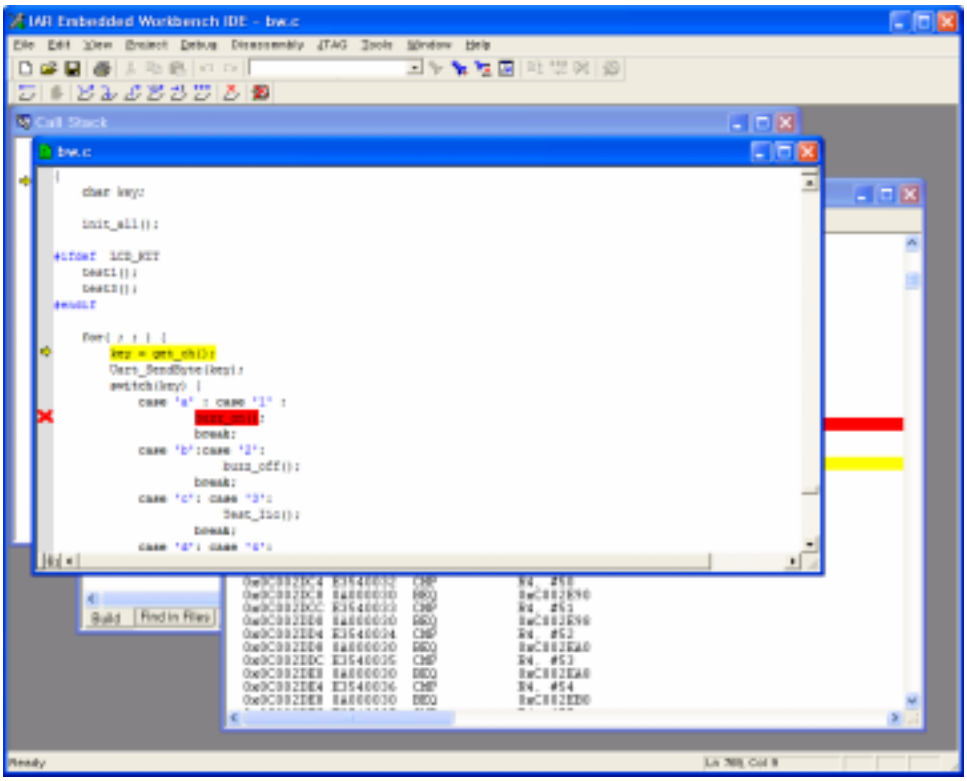
5. EWARM의 메뉴의 project-debug를 선택하면 EWARM은 C-SPY 디버그 모드로 진입하며 Macraigor Driver의 download상태가 잠시 나타났다 사라지며 C-SPY화면이 아래과 같이 나타난다.



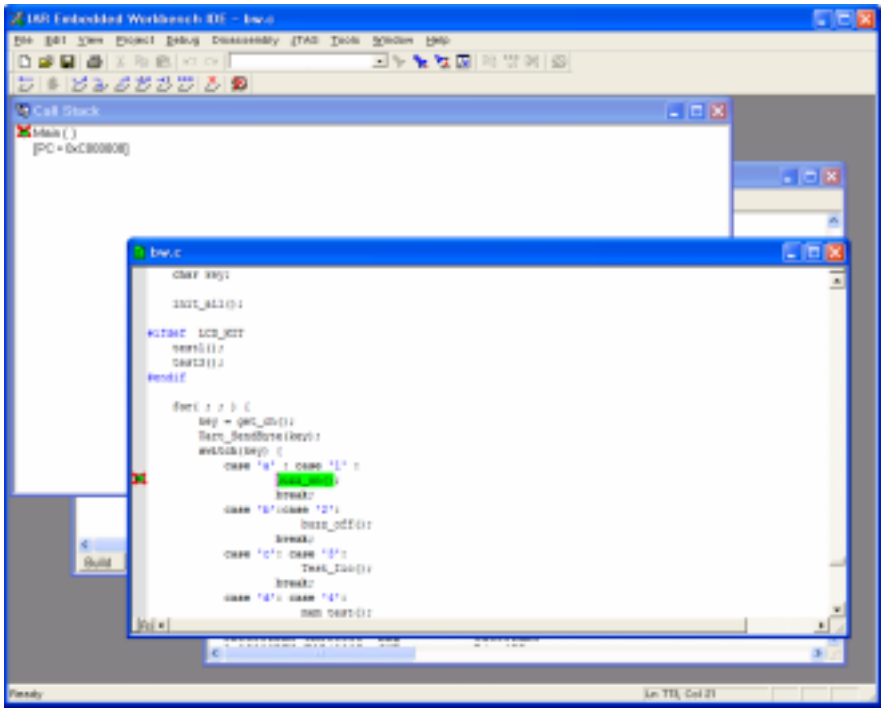
6. 화면 좌측 상단의 왼쪽화살표 아이콘을 눌러 CPU를 리셋 시킨다.
(RESET시키면 Disassembly창은 0x0번지로 이동하며 0x0라인이 초록색이 된다.
초록색이 현재의 실행지점을 나타냅니다.)
7. 화면 좌측 상단의 왼쪽화살표세개 아이콘을 눌러 CPU를 GO시킨다.
시리얼 통신창을 통해 프로그램이 정상적으로 초기화 과정을 실행하는지 확인하고, LED가 주기적으로 점멸하는지 확인하여 프로그램이 정상으로 실행되는지 확인한다.
8. 화면 좌측 상단의 빨간색 손바닥 ICON을 클릭한다. 그러면 CPU는 정지하고 현재 중단점의 주소와 데이터가 dsassembly창에 표시되며 바로 뒤에 창에 C소스 창과 C소스상에서의 중단점이 표시된다.(그림상에서는 44blib.c의 Uart_SendByte함수내부에서 멈추었음. 수동으로 정지시킬 때의 정지지점은 수시로 변함)



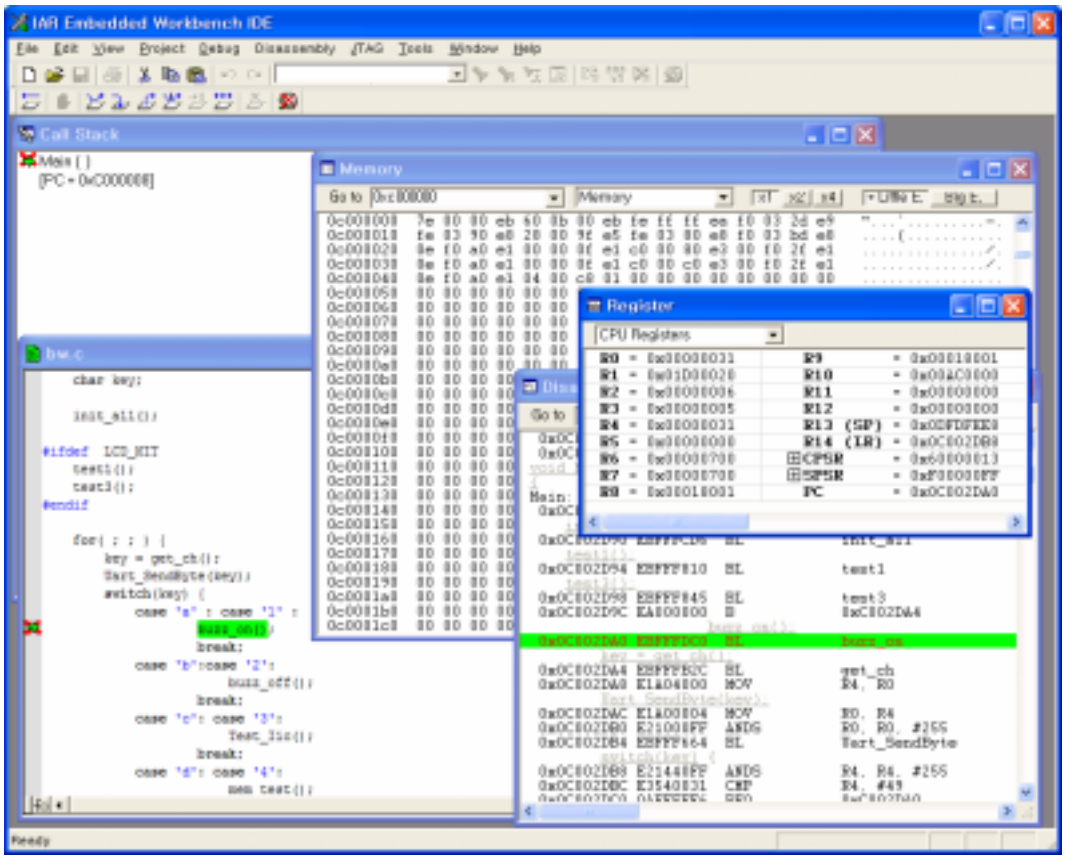
8. Call Stack창(Menu-View-Call Stack)을 선택하여 Main()함수를 더블클릭한다. 그러면 Disassembly창이 앞에 나오고 bw.c창이 뒤에 위에 위치한다.
9. bw.c창을 선택하여 마우스커서를 buzz_on();라인에 위치시키고, 마우스 오른쪽 버튼을 눌러서 Break를 Enable시킨다. Break가 Enable되면 해당 라인이 빨간색으로 바뀐다.



10. 화면 좌측 상단의 왼쪽화살표세개 아이콘을 눌러 CPU를 GO시킨다.
CPU는 다시 계속 실행을 진행하며 LED는 계속 점멸한다.
11. 통신창에서 PC키보드의 숫자키 '1'을 누른다. IAR EWAR창의 buzz_no()의 좌측에 초록색 화살표가 나오면서 CPU는 정지한다.



12. 정지상태에서 원하는 정보-메뉴의 View를 선택하여 를 파악한다. (그림은 메모리와 레지스터를 본 그림)



13. CPU를 중단시키고 필요한 정보를 모두 파악했으면 다시 GO(왼쪽화살표3개) 시키고, (부저가 시끄러우니) 통신창에서 PC키보드의 숫자키 '2'을 누른다. – 부저가 꺼진다.
위와 같은 방법으로 특정 시점에서 CPU를 정지시키고, 정지한 상태에서 각종 정보를 파악한 후 다시 실행시키거나 STEP by STEP으로 소스를 디버깅 할수 있다.

보통의 경우(JTAG CABLE로 SDRAM에 직접 이미지를 다운로드한경우) C-SPY를 실행전에 소스상에서 지정한 브레이크 포인트가 C-SPY실행후에도 계속 유효한 상태로 남아 CPU의 실행이 해당지점에서 도달하면 자동으로 멈추게되지만, CPU내의 다른 프로그램이 SDRAM에 복사한 이미지의 경우에는 브레이크설정이 사라지게 되어 C소스상에 브레이크가 표시되어 있어도 그 위치에서 정지하지 않습니다. 따라서 Symbolic Debug의 경우에는 프로그램의 최초 시작 후 반드시 한번은 수동으로 멈춰서 원하는 위치에 Break를 설정하고 진행하여야 합니다.

2. SDRAM에 직접 다운로드 하는 방법.

이 방법을 사용하기 위해서는 우선 모니터프로그램의 모드를 변경해야 합니다. 모니터 모드에서 5번을 선택하여 모드가 'JTAG_DEBUG MODE'인지 확인한후에 아래 사항을 실행하십시요.

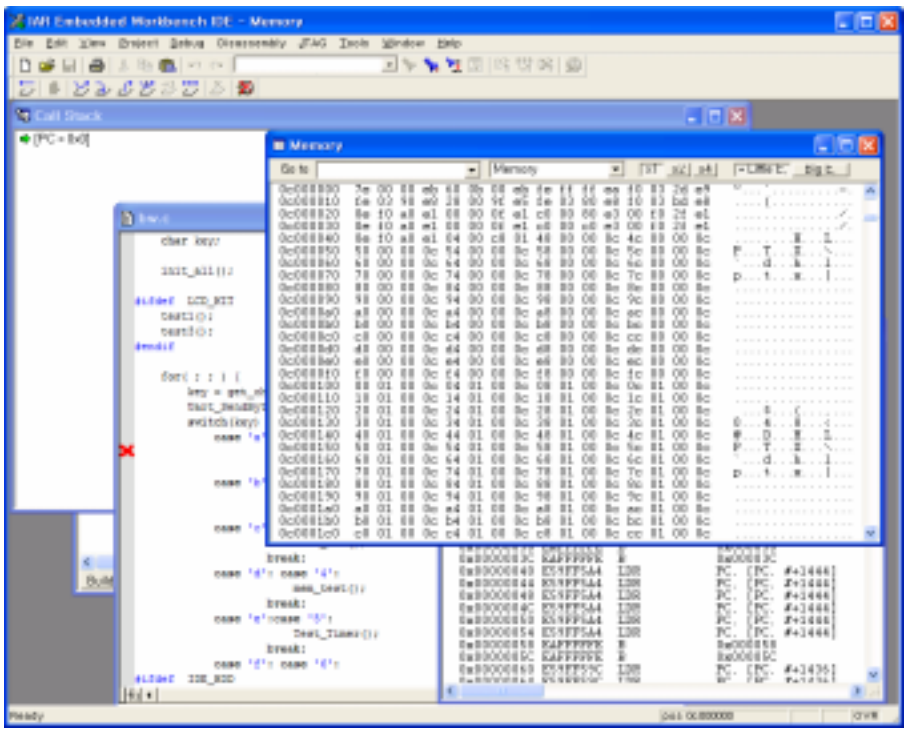
C-SPY를 통한 **디버깅이 끝나면 반드시** 모니터모드(다시 모니터 프로그램의 5번메뉴를 사용하여)를 **Normal모드로 바꾸십시요**.

모니터프로그램에 JTAG_DEBUG MODE라는 메시지가 있는지 확인합니다.

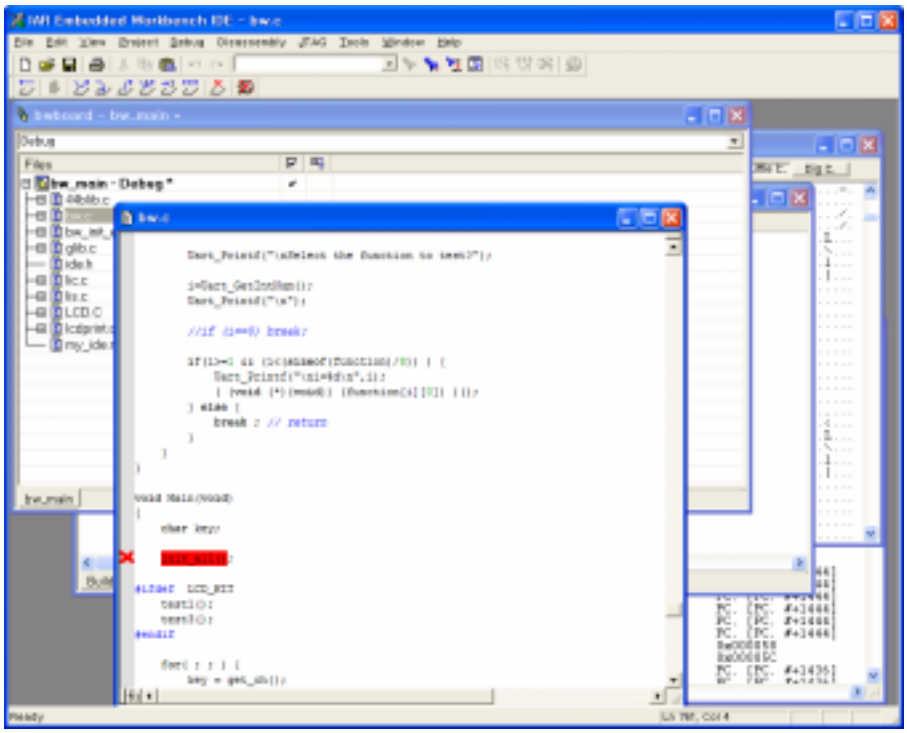
이 이후부터는 전원을 켜면 모니터 모드로 진입이 가능하지만, SDRAM의 영역에 프로그램이 복사되지 않으므로 Main프로그램은 실행되지 않습니다.

이 상태에서 다음 절차로 디버깅을 시작하면 됩니다.

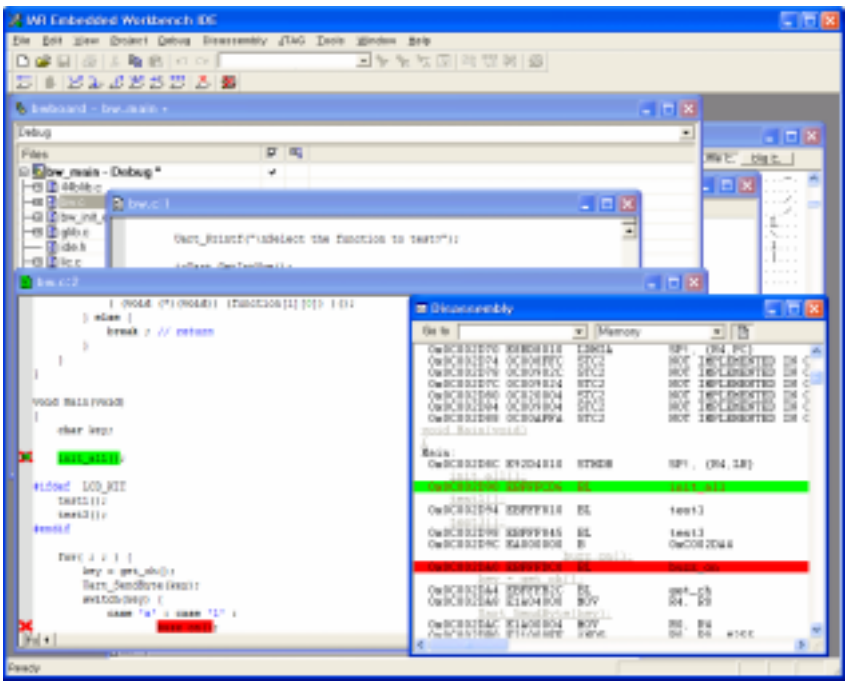
1. bw_main 프로젝트에서 메뉴의 project-options-xlink를 선택하여 출력 포멧을 debug info로 지정한 후 컴파일하여 bw_board.d79를 생성한다.
2. EWARM의 메뉴의 project-options-C-SPY옵션에서 Macraigor tab에서 Suppress Download를 선택하지 않는다.
3. BWBOARD와 JTAG Debugger를 연결하고, JTAG Debugger를 PC의 패러럴 포트에 연결하고 BW_BOARD의 전원을 켠다. BWBOARD와 시리얼포트를 연결하고, 통신용 프로그램도 실행시킨다. 이때 보드의 SDRAM에는 메인 프로그램이 없기 때문에 알 수 없는 상태가 됩니다.
4. EWARM의 메뉴의 project-debug를 선택하면 EWARM은 C-SPY 디버그 모드로 진입하며 Macraigor Driver의 download상태창이 나타나며 JTAG CABLE을 통해서 이미지 데이터가 SDRAM영역에 Write된다.(다운로드 속도는 약 8~10kbyte/sec정도입니다)
5. Download가 끝나면 C-SPY가 실행되며 각종 창이 화면에 나타납니다.



6. 이 상태에서 bw.c소스(화면에 보이지 않는다면 메뉴widow bwboard-bw_main창을 열어서 bw.c를 더블클릭하면 화면에 나타남)의 Main()함수의 진입지점에 break를 설정합니다.(init_all()이 있는 라인에 마우스커서를 놓고 오른쪽 버튼을 눌러 break를 표시하는 빨간 X표시가 표시되도록 합니다.)



7. 화면 좌측 상단에 있는 왼쪽화살표 아이콘을 눌러 CPU를 RESET시킵니다.
8. 화면 좌측 상단에 있는 오른쪽화살표세개 아이콘을 눌러 CPU를 GO시킵니다.
CPU는 0번지부터 순차적으로 프로그램을 실행하여 Main()함수의 init_all()지점에서 동작을 멈춥니다.



9. 동작이 정지한 상태에서 사용자는 CPU에 대한 디버깅 정보를 볼수 있습니다.
10. 다시 CPU를 GO시키면(오른쪽화살표세개 아이콘 클릭) 프로그램은 정상적으로 동작합니다. (프로그램의 정상동작확인은 LED의 점멸로 확인)
11. 이 이후의 상황은 Symbol로 디버깅하는법의 8번항목 이후의 절차를 참고하십시요.

(주의사항)
1. 디버그 모드에서 디버깅이 끝나면 모니터프로그램 모드를 Normal Mode로 반드시 교체하십시요.
2. 디버그하기 전에 컴파일된 Image(*.a79,*d79,*.bin)의 날짜와 시간을 탐색기등을 통하여 반드시 확인하십시요. 실제로 디버깅하는 Image가 최종적으로 소스를 수정한 파일인지 확인하십시요. 실제로 수정된 소스와 FLASH/SDRAM에 다운로드된 Image와 디버깅 Symbol이 들어 있는 Image가 동일한 내용인지 반드시 확인해야 합니다. 최종 소스와 컴파일된 Image, 실제 SDRAM에 다운로드한 image, 디버깅용 Symbol image가 달라서 디버깅에 혼란을 가져오는 경우가 상당히 많습니다.

xcl파일에 대하여

EWARM컴파일러에서 xcl파일은 생성되는 binary image의 실행번지와 밀접한 관련이 있습니다. CPU의 특정 번지(특정메모리영역)에 특정 binary image를 위치시켜 실행하기 위해서는 xcl파일을 통해서 ASM/C언어상의 함수/변수위치를 해당 위치에 맞게 할당해 주어야 합니다.

시스템 엔지니어는 하드웨어를 설계할 때 CPU의 어느 번지에 어떤 메모리(램/롬)을 붙일 것인가를 설계하고, 해당 번지를 어떤 용도로 사용할 것인지 결정하게 됩니다. 이때 CPU의 주소공간에 마음대로 램이나 롬을 붙일수 있는 것이 아니라, Reset Vector영역에는 롬을 붙여야 하고, 그 외 다른 주소 영역에 램을 붙입니다. 롬과 램 용량의 결정은 어떤 소프트웨어를 실행할 것인가에 달려있습니다. 이렇게 해서 롬과 램의 주소와 용량이 결정되면 그에 맞게 프로그램을 작성하여 해당 영역(롬/램 영역)에 프로그램을 위치시킨 후 프로그램을 실행합니다.

보통의 경우 프로그램을 작성할 때는(ASM/C소스를 작성할 때) 이런 롬/램의 영역이나 용량은 고려의 대상이 아니라 단순히 소스만 작성하게 됩니다.(물론 데이터의 양은 고려의 대상입니다.) 이렇게 작성된 명령어/데이터들의 집합체인 이미지를 하드웨어 설계시 의도했던 번지에 배치하는 작업이 필요한데, 이 작업은 link작업시에 이루어지며, link 프로그램은 xcl파일을 참조하여 명령어와 데이터를 적절한 주소 영역에 배치합니다.

프로그램의 코드는 보통 롬에 위치하고, 램에 위치하기도 합니다. 프로그램의 데이터는 램에 위치합니다. 프로그래머는 xcl파일을 통해서 코드의 위치(주소)와 데이터의 위치(주소)를 링커에게 알려주고, 링커는 이에 맞게 binary image를 배치합니다.

IAR EWARM 컴파일러는 Segment라는 단위로 코드와 데이터를 배치합니다. 따라서 프로그래머는 Segment단위로 주소영역을 지정합니다.

이들 Segment중에 코드와 관련된 세그먼트는 INTVEC, ICODE, NEARFUNC_A등이 있으며 데이터와 관련된 세그먼트는 HUGE_ID,HUGE_C,HUGE_I,HUGE_Z,HEAP,CSTACK등이 있습니다.(더 자세한 사항은 IAR EWARM 관련 자료참조)

이들 세그먼트중에서 가장 중요한 점은 코드의 시작위치와 데이터의 시작위치입니다.

코드의 시작위치가 반드시 롬/플래쉬 메모리의 시작위치일 필요는 없습니다.

코드의 시작위치는 롬/플래쉬의 중간이나 램의 중간부분 어디에 위치해도 상관없습니다.

이 코드(실행 image)를 실행하기 위해서는 단지 CPU의 PC register(Program Counter)를 이 코드의 시작위치로 세팅하기만 하면 됩니다.

데이터의 위치 역시 램의 시작위치일 필요는 없습니다. READ와 WRITE겸용 데이터(전역번수/Stack)는 반드시 램에 위치해야 하지만, 이 역시 램의 시작위치일 필요는 없습니다.

READ전용 데이터의 위치는 램에 위치해도 되고, 롬에 위치해도 됩니다.

보통의 경우 롬에 코드와 READ전용데이터가 위치하고, 램에 READ/WRITE데이터가 위치하지만, 램에 코드와 READ전용데이터, READ/WRITE데이터가 위치해도 상관없습니다.

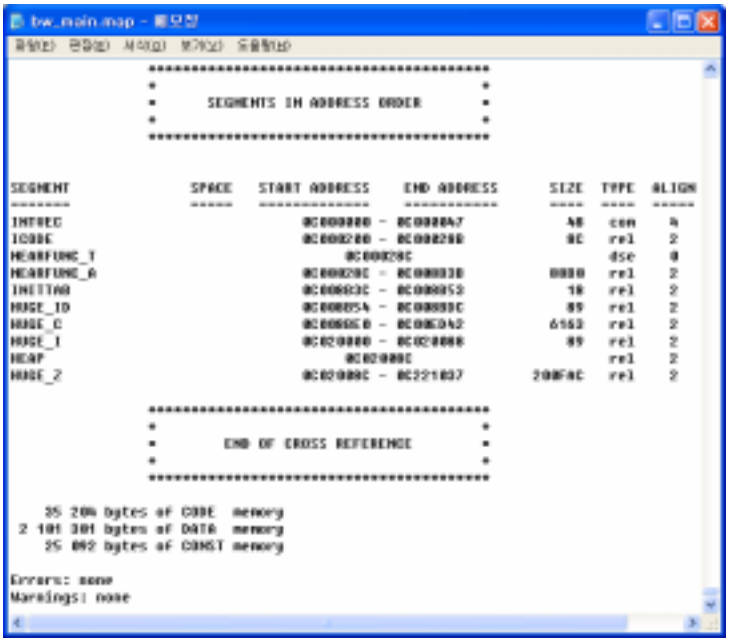
IAR EWARM 컴파일러는 xcl파일을 통해서 이들 Segment의 실제 위치를 지정하고,

이렇게 해서 컴파일 된 binary image는 xcl파일을 통해서 지정된 위치에 실제로 Load되어야 올바르게 실행됩니다.

BWBOARD와 함께 제공된 소스의 각각의 프로젝트에는 각각에 해당되는 *.xcl파일이 있습니다. 이 파일을 잘 분석하면 각각의 image의 위치를 알수 있습니다.

그리고 최종적으로 컴파일된 binary iamge의 Segment위치는 MAP파일을 통해서 알수 있습니다. xcl파일은 단지 범위을 지정하기만 할 뿐이고, 최종적인 결과의 위치는 MAP파일을 통해서 알수 있습니다.

결과적으로 xcl파일이 소스와 binary image 실행번지를 연결해주는 파일입니다.



bw_main.bin의 map 파일

각각의 프로그램에 대한 메모리 요약정보

1 BW_BOOT – bw_boot.bin
Xlink 지정(실제 실행위치)
CODE : 0x0
DATA : 0x0df0.0000
저장위치 –
CODE : 0x0000.0000 ~ 0x0000.07ff - Flash Memory AREA (BANK0)
역할 : Boot후 CPU의 기본적인 사항을 설정한후 0x800~0x87ff번지에 있는 BW_MON을 0x0db0.0000번지로 복사한후 PC(Program Counter)를 0x0db0.0000번지로 옮김

2 BW_MON – bw_mon.bin
Xlink 지정(실제 실행위치)
CODE : 0x0db0.0000
DATA : 0x0db1.0000
저장위치
CODE : 0x0000.0800 ~ 0x0000.87ff
- Flash Memory(BANK0)
- BW_BOOT가 0x0db0.0000번지로 복사후 실행
역할 : 모니터 프로그램.
올바른 Main프로그램이 존재하고, Main프로그램이 실행될 조건이라면 flash 메모리의 0x10004번지 이후에 있는 Main 프로그램 image을 0x0c00.0000 번지로 복사한후 PC(Program Counter)를 0x0c00.0000번지로 옮김

3 BW_MAIN – bw_main.bin
Xlink 지정(실제 실행위치)
CODE : 0x0c00.0000
DATA : 0x0c02.0000
저장위치
CODE : 0x0001.0004 ~
- Flash Memory(BANK0)
- BW_BOOT가 0x0c00.0000번지로 복사 후 실행
역할 : Main 프로그램. 사용자가 필요에 따라 수정해서 사용하는 프로그램

4 BW_ROM – ROM.BIN – CODE와 RO DATA 의 저장위치와 실행위치가 같음.
CODE : 0x0000.0000~0x0000.77ff
DATA(RO): 0x0000.0000~0x0000.7800 – FLASH에 Write될 모니터 프로그램 Image Data
DATA (RW) :0x0df0.0000 ~

OPTION KIT에 대한 소개

1. CPLD KIT

- Xilinx CPLD XC96144XLTQ100 사용
- CPU Interface : Address 8bit/Data 16bit/기타 Control Line
- 외부 I/O Interface : IDE HDD커넥터(28I/O), 24핀커넥터(19 I/O pin)
  IDE HDD를 사용하지 않을경우 28I/O를 다른 목적으로 사용할수 있음.
- I/O 테스트용으로 LED/SW보드 제공
- IDE HDD규격 : ATA-3 PIO 모드 지원

2. MONO LCD KIT

- 320X240 STN Mono GRAPHIC Module
- 4bit Single Scan 방식
- LOGIC 전압 : 5V
- LCD DRIVE 전압 : -22V (MAX748 DC/DC Converter)
- **BACK LIGHT는 없음**
- 16 Gray Scale Level, 4bit/pixel
  440bx CPU가 지원하는 Frame Rate Control를 이용하여 Gray Scale 구현
- Frame Per Second : 약 72.3Hz
- Pixell Clock : 1.41Mhz
- Pixell Memory : 3,686,400byte(320*8X240*12*4bit) – 96개(8X12)의 가상화면이 있음.
- Pixell Memory 위치 0x0dc0.0000~

3. COLOR LCD KIT

- 640X480 STN COLOR GRAPHIC Module
- 8bit Single Scan 방식
- LOGIC 전압 : 3V~5V
- LCD DRIVE 전압 : 0~2.5V (DC/DC Converter 내장 Module)
- CCFL BACK LIGHT
- 256 Color(Red 3bit, Green 3bit, Blue 2bit)

440bx CPU가 지원하는 Frame Rate Control를 이용하여 Color 구현

- Frame Per Second : 약 72.5Hz
- Pixell Clock : 8.25Mhz
- Pixell Memory : 3,686,400byte(640*3X480*4*8bit) – 12개(3X4)의 가상화면이 있음.
- Pixell Memory 위치 0x0dc0.0000~

OPTION 관련 제공 소프트웨어

1. CPLD 구동 소프트웨어
- I/O 테스트 루틴 제공(LED/SW보드 테스트)
- IDE HDD 관련부분 Object로 제공
- CPLD Bit Stream Image 제공

2. LCD구동 소프트웨어 – Mono,Color겸용
- 8X16 영문폰트와 16X16 한글폰트가 제공
- 간단한 Graphic Library 제공

맺음말

이 사용설명서 상에 언급된 내용이나, BWBOARD와 관련된 내용, ARM CPU와 관련된 내용 중에 궁금한 사항이 있으면 홈페이지

http://cgi.chol.com/~kohyc/armcpld/index.cgi/

나 Email kohyc@chol.com 으로 문의하십시요.

최선을 다해서 기술지원을 하겠습니다.