# CodeViser ~ Target board 연결

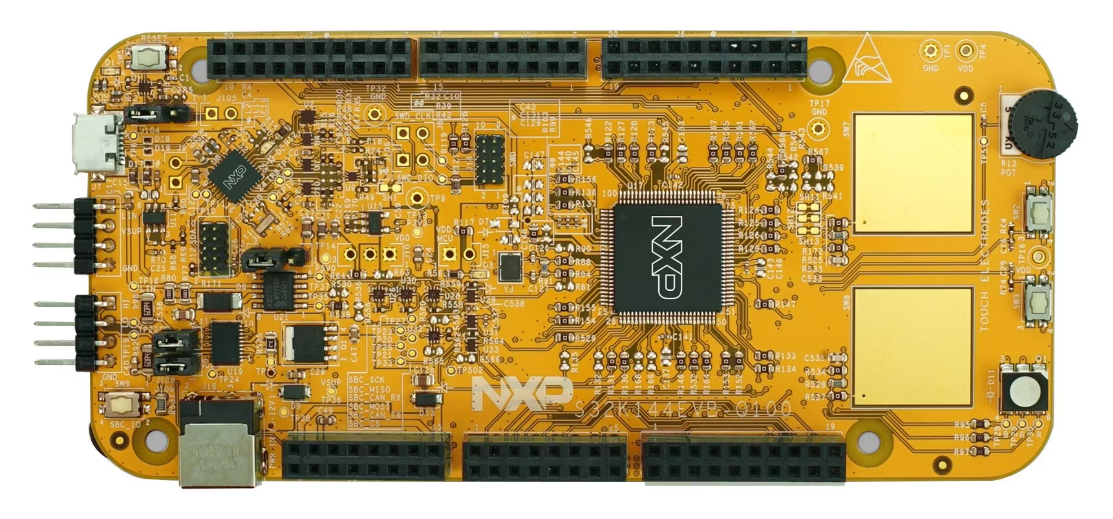
CVD JTAG 보드와 Target 연결

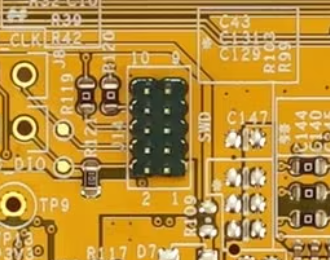
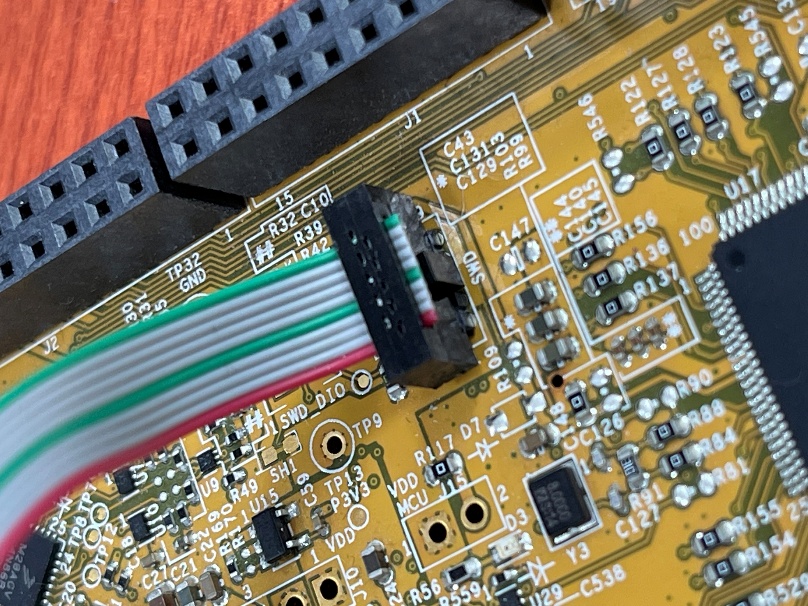
S32K144는 10pin SWD port를 사용합니다.

CodeViser에 동봉해드린 커넥터를 통해 JTAG 보드를 연결해 주셔야 합니다.



이미지와 같이 보드 SWD port는 보드 중간에 위치해 있으며 1번 pin을 확인하시어 JTAG 커넥터와 연결해주세요.





이제 codeviser 의 USB cable 을 host pc 와 연결해 주시면 됩니다.

CVD 프로그램을 실행 해 주세요

CVD 를 새로 설치하신 경우 Help-About CVD 에 들어가셔서 CVD 와 MDSFirmware 를 확인해 주세요

CVD 를 설치하실 때 updata mdsfirmware 옵션을 클릭하시면 기본적으로 mdsfirware 도 update 진행 하지만 codeviser 가 연결되지 않았거나 cvd 에서 simulation mode 로 선택한 경우에는 firmware update 가 진행이 되지 않습니다.

그러므로 버전을 확인하시고 firmware update 되지 않은 경우 수동으로 업데이트 해주시기 바랍니다.

Program – Firmware 에서 cvd 설치경로의 MDSfirmware directory 에서 최신 firmware 를 선택해서 업데이트 해주시면 됩니다.

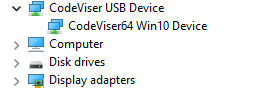
현재 하단에 status bar 가 sysoff 로 되어 있는데 이는 cvd 가 codeviser 를 인식한 상태이고 target 전원을 감지하지 못한 상태임을 의미합니다.

이제 target에 power on 을 하면 sysdown으로 변하게 됩니다.

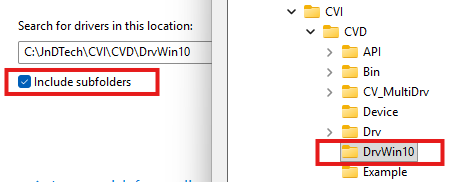
혹시 연결 했을 때 idle 상태로 뜬다면 이는 cvd가 codeviser 를 인식하지 못한 상태입니다. Usb 나 jtag 연결은 다시한번 확인해 주시고 드라이버가 제대로 설치되었는지 확인해 주세요

하기와 같이 codeviser usb device 로 잡혀야 제대로 설치가 된 상태입니다.

기타 장비로 잡혔을 경우에는 업데이트 해주세요.



드라이버는 cvd 설치경로 DvrWin10 해주시면 됩니다.



이제 기본적인 환경 설정은 끝났습니다.

S32k144 보드를 flash 하고 디베깅 데모를 보여드리겠습니다.

Flash 를 하기 위해서는 세가지 파일이 필요합니다.

1. Flash 할 소스파일
2. Flash controller 에 flash 정보를 넘겨줄 flash loader
3. Flash loader 를 실행 할 cvd 스크립트 파일

그중에서 flash loader 와 script 는 jnd tech 에서 제공을 해드립니다.

Flash를 하실 때에는 cvd에서 스크립트를 실행만 해주시면 됩니다.

스크립트를 우선 설명 드리겠습니다.

저희 제공해드린 flash loader 와 flash 하실 data 의 경로를 지정하고

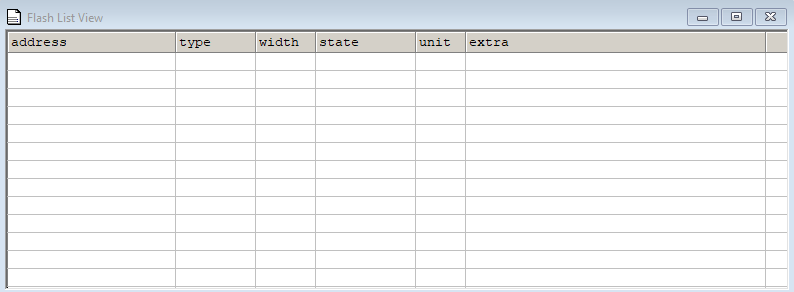
CVD 설정을 해줍니다.

기본적인 cpu type, debug interface type 를 선택하고 그 외의 옵션을 수요에 따라서 설정을 해주시면 됩니다.

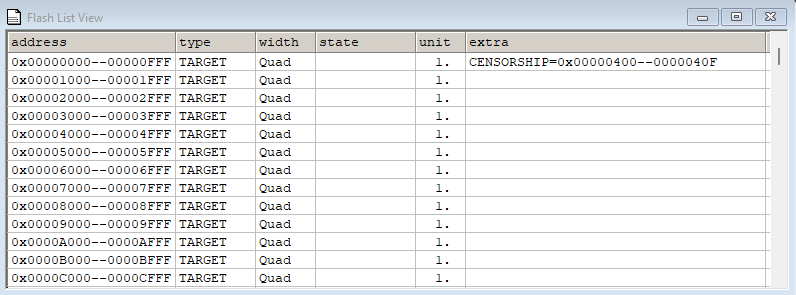
그리고 flash 전에 bootrom과 watchdog을 disable 해주고

선택된 cpu 에 따라 flash sector을 생성해 줍니다.

지금 상태에서 cflash.list가 비어있습니다.

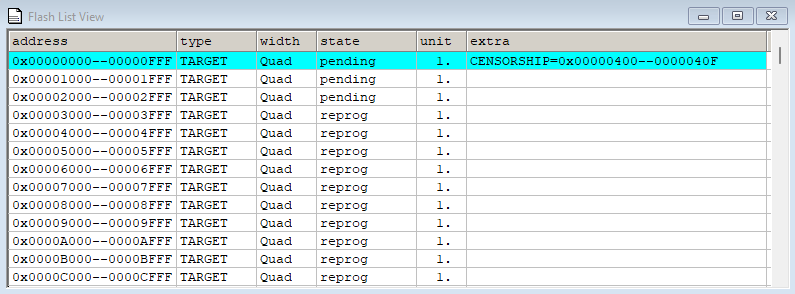


Flash declaration sub routine 을 완성 하고 다시 cflash.list 를 해보면 flash sector 가 생성된 걸 확인할 수 있습니다.



Cflash.reprogram.all 를 하시면 cvd 가 생성된 flash들을 확인하고 state는 reprogram 으로 변합니다.

Data.load.auto 를 하면 flash 할 source 를 확인해서 state 를 다시 pending 으로 바꿔줍니다.



마자막으로 cflash.reprogram.off 를 하면 flash programming 이 실행 및 완료됩니다.

이제 flash 한 이미지를 이용해서 디버깅을 해보도록 하겠습니다.

디버깅 환경 설정도 스크립트로 하는데요,

cd.do 로 debugging script를 실행 하면 debugging 환경 설정이 끝나게 됩니다.

만약 해당 script로 flash 만 하고 싶다 하실 경우에는 해당 라인을 주석처리 해주시면 됩니다.

저는 step을 해서 해당 스크립트를 불러오도록 하겠습니다.

디버그 스크립트도 아까와 마찬가지로 환경 설정을 해주고요

Elf 이미지를 올리고 debug window 설정을 해줍니다.

저희는 이미 아까 flash 를 하면서 이미지를 올렸기 때문에 load elf 를 주석으로 처리 했는데요

CVD는 프로그램이 꺼지면 cvd 에 로딩된 elf 도 날라납니다.

그래서 target 에는 code 가 올라가 있는데 cvd 에만 elf 를 올려서 소스코드를 보고자 할 경우, / nocode 옵션으로 elf 를 로딩해 주시면 됩니다.

아래는 자주 사용하는 debug view 를 미리 실행시키는 command 들입니다.

저는 debuglist, breakpoint, symbol browse, variable, watch 창을 열어두었습니다.

Debuglist 는 코드를 볼 수 있는데요 debuglist 라는 커멘드로 현제 pc가 가리키는 code를 불러 올 수 있습니다.

Show next statement 아이콘을 클릭해서 현재 pc 값을 기준으로 debuglist 를 불러옵니다.

지금 pc 가 reset handler 에 위치해 있는데 밑에 command 창에 go main 을 입력해서 프로그램의 main 까지 실행해주 실수 있습니다.

Debug list view 에서 mode 를 클릭 해서 assembly 를 확인하실 수 있습니다.

함수를 더블 클릭 하면 해당 함수 코드를 확인 하실 수 있습니다.

Address line 을 클릭 해서 breakpoint 를 추가 할 수 있습니다.

Breakpoint view 를 보시면 클릭한 address 에 breakpoint 가 설정이 되었는데요

보통 address line 을 클릭해서 breakpoint 를 설정할 경우 software breakpoint 가 찍힙니다. 그러나 cvd 가 target 의 flash 정보를 갖고 있는 경우, software breakpoint 가 flash 에서 실행 되지 못하는걸 알기 때문에 자동으로 hw breakpoint 가 찍히게 됩니다.

우클릭을 해서 강제로 software breakpoint 를 찍어보면 이렇게 error 가 발생합니다.

Breakpoint view 의 set에 symbol을 검색해서 breakpoint 를 설정 할 수 있습니다.

이 breakpoint 설정 화면서 bp 설정 시 flash 영역은 반드시 수동으로 Hard 를 선택해 주셔야 합니다.

자주 사용하는 breakpoint 인 경우 save load 기능을 이용하 실 수 있고 아니면 script 에 breakpoint 를 미리 추가해 놓을 수 있습니다.

Break.set main /onchip

Main 함수에 hardware breakpoint 를 설정할 수 있습니다.

이번에는 Watch view 로 넘어가겠습니다.

Watch view 에도 역시 미리 변수를 등록해 놓을 수 있습니다 .아까 스크립트로 미리 등록을 해 놓은 변수들입니다.

Symbol을 클릭해서 변수를 추가로 등록 할 수 있습니다.

혹시 location이이나 타입정보가 보이지 않을 경우에는 우클릭 – format 을 선택해서 수요되는 옵션을 체크해주시면 됩니다.

Watchlist view 는 확인하고자 하는 글로벌 variable을 보통 등록해서 쓰고 local 변수는 local variable view 로 확인을 하시면 됩니다.

호출 경로를 추적할 경우에는 frame.view command 혹은 callstack 아이콘을 클릭 하시면 됩니다.

함수별 컨텍스트를 확인 하시고 source 박스를 클릭 하면 해당 함수를 호출한 소스코드 라인을 직접 표시해줍니다.

특정 심볼이나, global 변수나, 함수, source 를 검색할 경우 browse view를 활용하시면 됩니다.

변수 일 경우 더블클릭 해서 바로 watch list 에 추가를 해도 되고 다른 옵션들은 더블클릭해서 소스 디버그 리스트를 불러옵니다.

우클릭을 해서 더 많은 기능을 확인해 보세요.

이번에는 register 값을 확인하는 방법을 설명드리겠습니다.

Register 아이콘을 클릭 하셔서 general register 값을 확인 하실 수 있고 시스템 레지스터 등은 register-cp register 를 통해 확인하실 수 있습니다.

하지만 register 의 경우 peripheral 도 그렇고 특정 address 에 mapping 된 register 값인 경우 memory dump를 해서 확인하시는 것을 추천 드립니다.

Debug DHCSR register 인 경우 0xE000EDF0 에 위치해 있습니다.

메모리 덤프 하는 방법을 설명드리겠습니다

register 값을 확인하는 방법을 설명드리겠습니다.

Memory 아이콘을 클릭 해서 메모리를 dump 창을 실행 해 주시고

Address 를 입력하시면 됩니다.

혹은 dump 창에 d 0xe000edf0을 입력하셔도 됩니다.

기본 디버깅 가이드를 마치겠습니다.



