[3.5 Emulator]

소프트웨어 흉내내기가 시뮬레이션이라면 하드웨어적으로 흉내내기는 에뮬레이터라고 한다. Emulator는 JTAG 더 정확히는 ULINK2™를 이용하여 ARM의 내부 메모리에 로딩한 뒤 소스코드에서 집적 디버깅을 할 수 있으므로 I/O 포트, 메모리 값 등의 실제적인 하드웨어에 관련된 정보도 읽어 올수 있어 실시간으로 확인이 가능하다.

물리적으로 에뮬레이터 하기 위해서 USB로 연결되는 [사진:JTAG 장비 ULINK2]처럼 JTAG ULINK가 준비되어 있고 USB 드라이버 또는 MDK-ARM 컴파일러가 설치되어 있어야 한다.

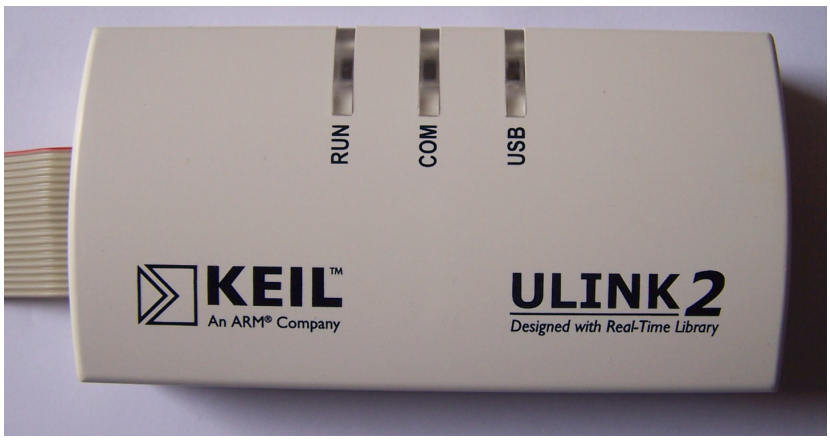


그림 43. JTAG 장비 ULINK2

USB 전용의 ULINK2를 Target Board와 연결하고 환경을 Setting해 보자. 여기서는 이해하기 쉽도록 TI사의 Cortex-M4F TM4C1230로 설계된 "FWB-TM4C-8750"을 이용하여 사진과 함께 설정환경을 설명하고자 했다.

Target Board, ULINK2 그리고 USB Cable을 각각 연결하고 ULINK2를 FWB-TM4C-8750 Target Board의 20핀에서 10핀으로 변경하는 변환 보드를 통해 JTAG에 연결한다. 역삽을 방지하기 위해 가운데 배꼽이 나와 있으므로 순리대로 연결하면 잘못되지 않는다. 여기서 사용된 샘플 프로그램은 다운로드 cafe를 통해 이루어졌다.

Cortex-M4F 평가를 위한 소스코드 :
- http://cafe.naver.com/fws의 "Cortex-M4"

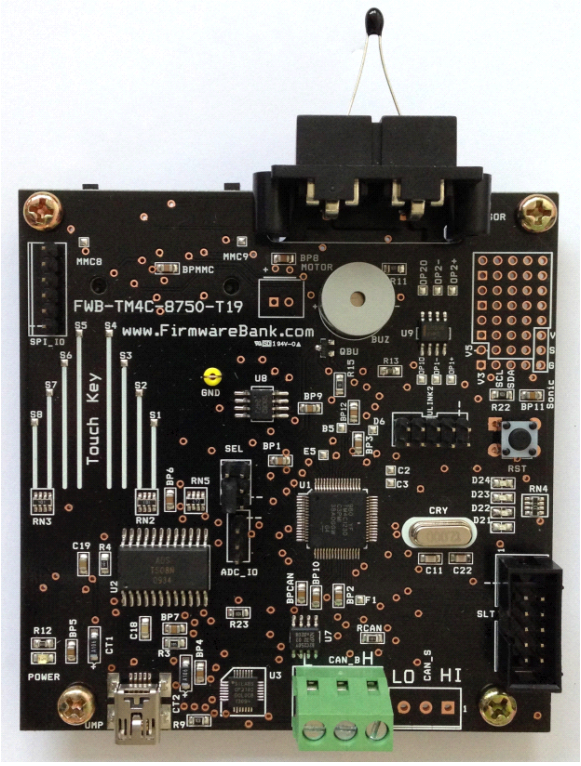


그림 75. FWB-TM4C-8750 타겟보드

'Project/Option for Target'(🔧)에서 나머지는 앞에서 다룬 대로 설정하면 되며 Emulator 하기 위해 'Project/Option for Target/Debug' 항목으로 이동하여 몇 가지 확인하고 익혀 둘 필요가 있다. 디버깅 윈도우의 오른쪽 'Use' 화면처럼 하면 된다.

Debug Tab 설명 :

- 'Use' : 디버그 장비의 디바이스 이름설정, 장비 디바이스를 MDK-ARM에 설치한 종류만큼 디스플레이 된다. 여기서는 ULINK2 또는 ME 선택
- 'Load Application at Startup' : Boot에 해당하는 Startup Code를 실행하고 장비에서 응용 프로그램을 로딩 한다. Startup Code 자체를 디버깅하지 않는다면 필요 없으므로 선택하지 않는다.
- 'Run to main()' : C 소스 프로그램의 main 함수를 호출하여 실행할 준비를 갖춘다.
- 'Restore Debug Session Settings' : Tool 윈도우의 프로그램을 시작할 때 기본 실행 아이템.
- 'Dialog DLL' : 오른쪽에 있는 Parameter와 함께 디버그 장비의 드라이버 이름을 설정하는 것으로 업그레이드되거나 새로운 디바이스나 ARM 칩이 나오게 되면 관련회사에서 공급하게 된다.

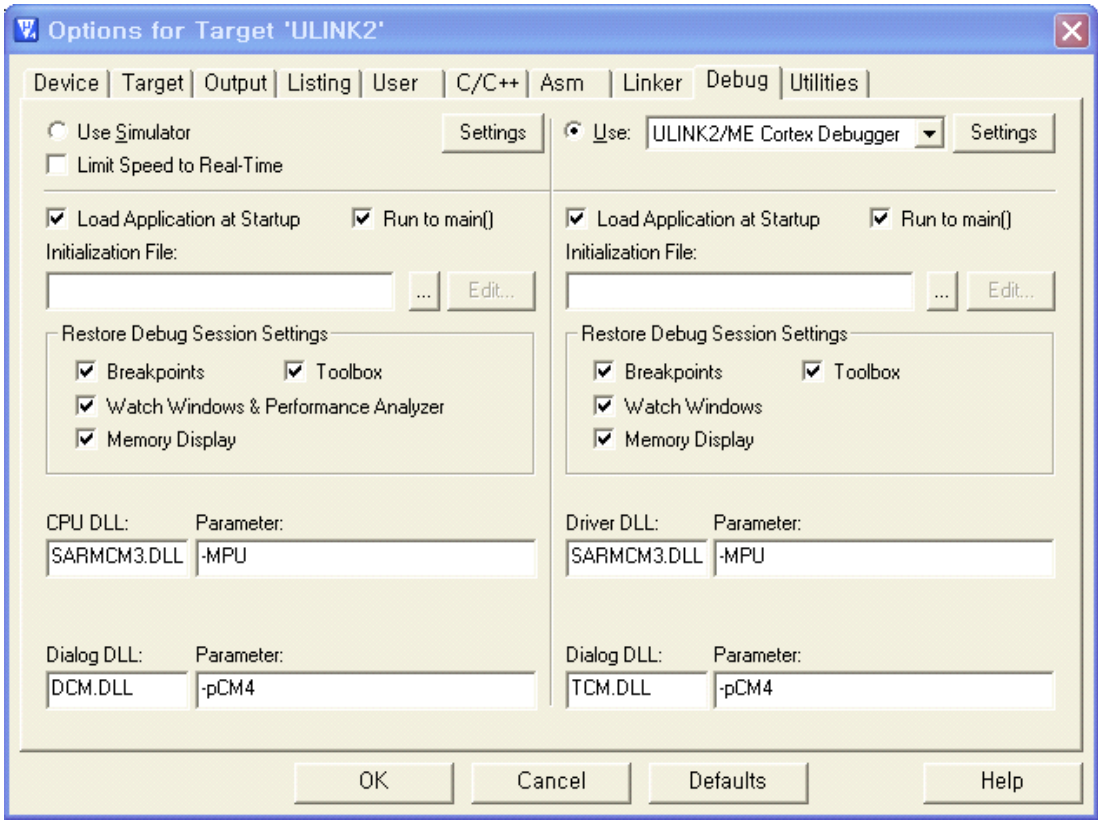


그림 76. Emulator 하기 위한 Project/Option for Target/Debug 설정

'Target Driver Setup'은 'Project/Option for Target/Debug'에서 오른쪽의 'Setting' 항목을 선택하면 팝업된다. 이곳은 ULINK2 장비의 드라이버 설정과 버전을 보여주는 항목으로 JTAG과 Target이 서로 연결 되어 있고 보드에 전원이 공급되었을 경우에 MCU의 ID와 디바이스를 'JTAG Device Chain(JTAG연결)' 또는 'SW Device(SWD연결)' 항목에 디스플레이 해 준다. 이때 복수개가 될 수도 있으며 여러 가지 이유로 연결되지 않았을 경우는 빈 공간으로 되어 있다. 혹시 Target과 JTAG이 물리적으로 인터페이스가 잘 안되어 나타나지 않을 경우는 타겟보드의 Reset 버튼을 이용하자. 주의하고 알아야 할 것은 JTAG ULINK2를 통하여 Target 보드에 전원이 공급되지 않으므로 JTAG은 USB 전원으로 Target은 별도로 전원 공급해 주어야 한다.

Debug/Setting 설명 :

- 'Serial No' : JTAG ULINK2의 장비 번호
- 'ULINK Version' : 장비이름, 예전 ULINK는 지원되지 않는다.
- 'Device Family' : ARM, Cortex-M 등의 종류 표시
- 'Firmware Version' : ULINK2의 Firmware 버전
- 'Max JTAG Clock' : 5㎑~10㎒ 선택가능하며 JTAG Cable이 길수록 최대 속도는 작아지도록 수동 설정한다.
- 'Download Option' : 다운로드 후 원본과 확인 기능, Flash 전 영역을 지운 후 다운로드 또는 선택된 영역만 지운 후 다운로드 Option이 있다.

• 'Misc Options' : Reset에 의해 시작 할 것인가를 선택한다.

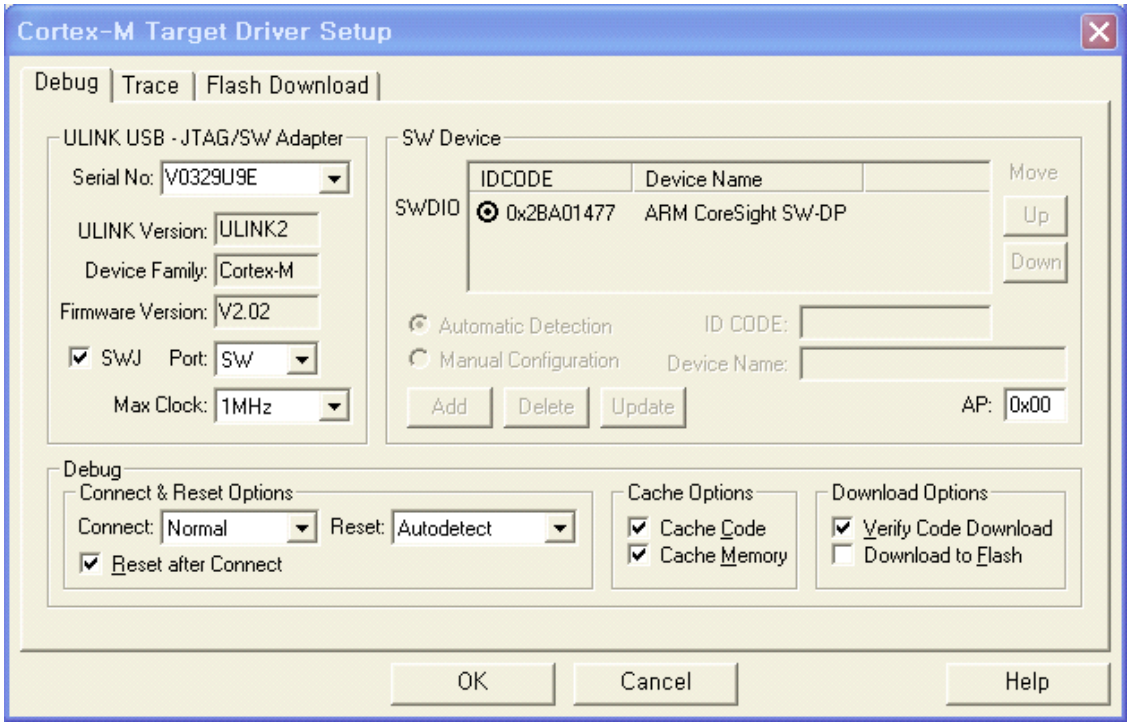


그림 77. Project/Option for Target/Debug/Setting의 JTAG 설정

여기까지 에뮬레이터(Emulator) 설정을 다 마친 것이다. 'OK'와 '확인'으로 선택된 모든 환경을 저장하고 JTAG을 통하여 하드웨어 흉내내기를 해 보자. 혹시 환경이 변경되어 새로이 반영할 것이 있을 수 있으므로 다시 한번 컴파일을 하도록 한다. 가능하다면 'Project/Rebuild all Target'을 눌러 컴파일을 전부 시행 하도록 하자. 순조롭게 타겟보드를 에뮬레이터 하기 위해서는 에러가 발생하지 않아야 하며 경우에 따라서는 에뮬레이터 하기 힘든 치명적인 워닝도 발생하지 않아야 한다.

ULINK2와 Target 장비가 연결된 상태에서 'Debug/Start-Stop Debug Session'실행 또는 디스코프()를 선택하여 에뮬레이션을 시작 할 수 있다. 평가버전일 경우는 컴파일 후 소스코드의 크기에 32K 한계가 있으므로 생성된 코드의 ROM 크기를 예측하여 시작하도록 하자. 32K 코드 이내이므로 '확인'을 누르자. 디스코프는 디버그와 스코프의 합성어로 돋보기 모양 안에 적색으로 'd'라고 적혀 있는 아이콘이다. 디스코프를 실행한 뒤에는 아이콘의 활성화 된 부분이 MDK-ARM의 실행 메뉴와 다른 것을 느낄 것이다.

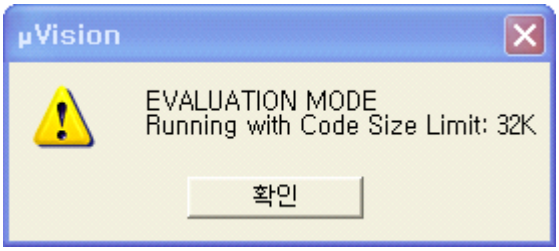


그림 78. Code 제한 팝업창

Debug 환경에 들어오면 Flash Memory에 기계어 코딩을 포팅하는 절차와 같은 여러 가지 기능의 아이콘들이 활성화 된다. 여기서 많은 아이콘들을 단번에 익힐 수 없기 때문에 코딩과 로딩, 디버깅을 하면서 점차 숙달하도록 하는 것이 좋다. 활성화 된 디스코프의 아이콘 중에서 실행 화면상단 왼쪽 처음에 있는 적색 화살표와 같이 있는 버튼부터 알아보자.

- 'RST'() 버튼 : 에뮬레이터일 경우 소프트웨어 Reset과 하드웨어 Reset이 동시 처리된다.
- 'Run'()//F5 : 현재 노란 커서가 있는 위치부터 연속 실행한다. 'Stop Running'으로 정지한다.
- 'Stop Running'() 버튼 : Run되기 전에는 활성화되어 있지 않지만 프로그램이 Run 되어 멈추기 전에는 적색으로 나타나며 X 표시로 되어있고 정지한다.
- 'Step'()//F11 : C의 한 문장 또는 어셈블리 하나의 라인을 실행한다. 화살표가 중괄호 안쪽에 표시되어 있다. 현재의 실행 예정라인은 노란색으로 표시된다.
- 'Step Over'()//F10 : 화살표가 외부에 있고 C/C++언어로 된 소스는 함수를 어셈블리 언어소스는 매크로 단위로 한 블록씩 실행할 수 있다.
- 'Step Out'()//Ctrl+F11 : C 함수의 마지막(Return)까지 실행하며 화살표가 안쪽에서 밖으로 나와 있다.
- 'Run to Course Line'()//Ctrl+F10 : 마우스 커서가 위치 한 곳까지 실행한다.

그림 79. Debug Toolbar1

디버그는 눈에 보이지 않는 부분의 알고리즘 오류를 검증하는 것이므로 꼼꼼하게 대처해야 한다. 사실 개발하는 것 보다 버그퇴치를 올바르게 해야 하는데 적재적소(適材適所)에 잘 사용할 수 있도록 숙련되게 익혀야 하며 보통 개발은 디버그 하는 기간을 개발기간과 같은 정도로 두고 스케줄링을 하는 것이 바람직하다. 그래도 JTAG을 통해서 Emulation 하게 되므로 C소스코드의 소스레벨로 검증과 레지스터, 메모리의 정확한 값을 실시간으로 읽어 디버깅하기 때문에 개발속도와 효율을 극대화 할 수 있다.

디버깅을 더 화려하고 편리하게 하고 싶다면 다음의 아이콘을 눈여겨 볼만하다. 펌웨어기술자들은 계측기가 자신의 설계절반을 해준다는 것에 동감할 것인데. 여기 소개되는 아이콘이 하드에어와 JTAG을 통하여 실시간으로 실험해 볼 수 있는 계측기의 버튼에 해당된다.

- 'Break Points'//Ctrl+B : 프로그램 실행 중에 정지점을 지정하여 이곳에서 일시 멈추게 된다. JTAG을 이용 할 때는 Break Point 개수에 제한을 받는다.
- 'Insert/Remove Breakpoint'()//F9 : 프로그램 실행 중에 정지점을 현재 마우스 포인터가 있는 Line에 설정하거나 동일한 Line에서는 지우게 된다. JTAG을 이용 할 때는 Break Point 개수에 제한을 받으며 토글 동작된다.
- 'Enable/Disable Breakpoint'()//Ctrl+F9 : 설정된 정지점을 활성화 하거나 일시정지 하지만 Break Point 그 자체가 지워지지는 않는다. 설정 때마다 적색/흰색 동그라미로 변경된다.
- 'Disable All Breakpoint'() : 모든 설정된 Break Point의 기능을 상실하게 해 주지만 지우지는 않는다.
- 'Kill All Breakpoint'()//Ctrl+Shift+F9 : 모든 설정된 Break Point를 지운다.

그림 80. Debug Toolbar2

- 'Show Next Statement'() : 다음 실행의 PC(Program Count)를 나타내며 C 소스 코드, Disassembly 상태의 행 번호에 노란색 화살표로 표시한다.
- 'Disassembly Window'() : 소스코드를 어셈블리로 나타내준다.
- 'Watch Windows'() : 스택 상태를 나타내주며 글로벌 변수와 로컬 변수를 불러와 내용의 값을 볼 수 있다.
- 'Serial Windows'() : 가상적 시리얼 포트로 출력되는 결과를 보여준다. 이용하기 위해 시리얼 통신의 초기화를 선행해야 한다. 보통은 printf 함수의 Run Time 결과값을 확인하는 출력 윈도우로 활용된다. 또한 지원되는 Serial Window는 CPU마다 달라 그 개수만큼 지원된다.
- 'Code Coverage'() : 함수의 호출 빈도를 측정하여 코드의 효율을 따질 수 있다.
- 'Memory Window'() : 어드레스 번지를 Hex로 넣으면 MPU에 있는 실제의 해당 번지 메모리 값 또는 어드레스가 있는 읽기 가능한 레지스터의 상태를 볼 수 있게 해 준다.
- 'Performance Analyzer'() : Run 중인 함수의 실행 시간을 측정하여 막대그래프로 한눈에 나타내준다. 함수의 실행 평가에 이용한다.
- 'Logic Analyzer'() : 로직해석기로 사용하기 위해 만들어진 Utility이며 잘 사용하기 위해서는 MDK-ARM의 특별한 키워드를 알고 있어야 한다.
- 'Symbol Window'() : 프로젝트에서 사용된 레지스터를 포함한 Define된 Symbol을 관찰 할 수 있다.
- 'Stack Frame'() : 스택에 현재 이용된 함수의 상태를 볼 수 있다.
- 'Toolbox'() : Update Window 버튼이 기본적으로 생성되며 자주 사용되는 디버그 명령 코드를 *.ini 파일을 통해 자신이 Toolbox를 설계 할 수도 있다.

그림 81. Debug Toolbar3

다음 실행 결과는 행 번호 46번의 if()문 이전까지 실행하게 한 다음 멈춤 결과이다. 물론 JTAG을 이용하기 때문에 ARM 마이크로프로세서 안에 있는 상태값을 불러와 실시간으로 디스플레이 한 것이다.

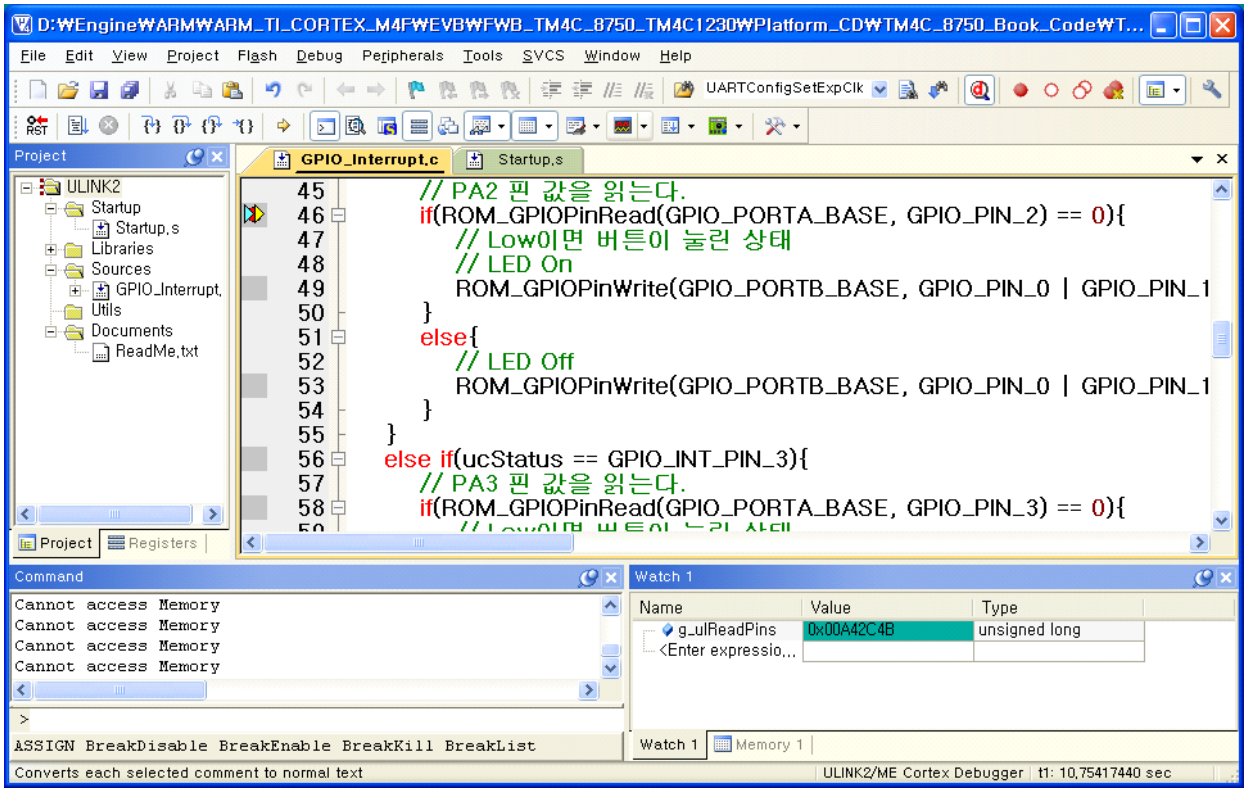


그림 82. ULINK2의 Emulator 실행결과

알나지 아버지의 뒷모습

믿음직한 남편이자 다정한 아버지로 인정받는 선배들과 저녁 식사를 하다가 '로또에 당첨이 된다면 무얼 할까' 하는 쪽으로 이야기가 흘렀다. '직장에 바로 사표 내고, 근사한 집 사고, 멋진 차 한 대 뽑고, 보디가드 고용하고…' 같은 대답이 나올 줄 알았는데 그중 누군가 낮은 소리로 이렇게 말했다.

"당첨금 다 가족에게 줘서 잘살게 한 후 나는 혼자서 멀리 떠날 거야." 놀랍게도 그 자리에 있던 대부분의 남자들이 이 말에 가슴 저릿한 표정으로 고개를 끄덕였다. 가족 보살피느라 자기 인생의 의미 같은 건 잊어버린 지 오래. 가슴이 답답해 끊었던 담배를 다시 피운다는 말이 이어졌다. 이런 이야기를 들으며, 도대체 이 모범적인 아버지들을 멀리 홀로 떠나고 싶게 만드는 이유는 무엇일까 궁금해졌다.

요즘 아버지는 사랑하는 가족이 힘에 부친다. 똑같은 돈을 들고 가도 절반밖에 채우지 못한 장바구니에 울상 짓는 아내는 좀 더 두둑한 월급봉투를 기대한다. 자식에게 기대기 미안한 연로한 부모님은 무슨 일이든 "난 괜찮다" 하신다. 힘들게 일해 입을 것, 먹을 것, 쉴 곳을 마련해 주었더니 사춘기 자녀들은 "우리가 아빠를 정말로 필요로 할 때 어디에 계셨나요?" 하고 입을 삐죽인다. 그럴 때마다 아버지들은 모든 게 다 자신의 잘못 같다. 조금 더 젊었더라면, 조금 더 능력 있었더라면, 조금 더 영리했더라면

좋지 않았을까 후회한다.
중략...
얼마 전 발표에 따르면 1인당 교육비가 1억원을 넘으면서 OECD 국가 중 우리나라가 민간 교육비 지출 1위를 기록했단다. 여기에 덧붙여 노동시간 긴 것으로도 1위라고 한다. 가족을 위해 이전보다 더 오래 더 열심히 일을 해야 하고, 이와 비례해 가족과 함께하는 시간은 점점 줄어든다. 어떤 책에서는 40대 남성들이 우울증이 아닌 '우울하지 않은 척 증(症)'을 앓고 있다고 진단했다.
돈이 없어 아이들을 제대로 먹이지 못하고 공부시키지 못하는 아버지와 남들 다 보낸다는 조기 유학을 보내지 못해 가족들에게 주눅 든 아버지는 스스로 좋은 가장이 되지 못한다는 자괴감에 어깨가 점점 더 처진다. 우울하면서도 우울하지 않은 척, 힘들면서 힘들지 않은 척해야 하는 이 땅의 아버지들이 문득 어디론가 훌쩍 떠나 버리고 싶은 마음이 드는 것은 어쩌면 당연한 일일지도 모른다.
"아빠, 앞으로 어떤 일이 일어날까요? 내가 그 일을 잘 해낼 수 있을까요?" 세상 모든 아이들이 아빠에게 묻고 싶은 궁극의 질문은 이런 것이다. 평범한 아버지들은 그 대답을 말로 하지 않고 대신 자신의 인생을 통해 직접 보여주었다. 정직하게 일하고 가족을 보호하며 무엇이 옳고 그른지 자녀에게 가르쳐 주고 함께 웃고 이야기를 나누었다. 그것만으로 충분히 좋은 아버지 역할을 했다고 자랑스러워할 수 있는 날은 언제일까.
곳곳에서 '기업인을 뛰게 만들자' '기업하기 좋은 나라 만들자'는 이야기가 나온다. 하지만 그보다 먼저 해야 할 것은 이 땅의 아버지들이 지나친 부담을 벗어 버리고 신나게 뛰도록 해 주는 일이다. 퇴근 길 지하철 안, 한숨 내쉬는 아버지들 사이에 서서 '비즈니스 프렌들리'보다 몇 배 더 급한 것이 바로 '패밀리 프렌들리'임을 다시 한 번 생각한다. 아침논단, 김은령(월간 럭셔리 편집장/번역가) 08년4월24일 조선일보

Book

p.458, 30000원
ISBN 9788957174043

- **제목 :** ARM® Cortex™-M4F TI Tiva™ 펌웨어개발
- **내용 :** Cortex-M4F 확실 이용, 쉬운 고속처리/부동소수점 연산, API-ROM, 컴파일러 연관 사용법
- **대상 :** TI 고성능 32Bit RISC MCU Cortex-M4F의 Embedded System 개발자
- **필요학과 :** 컴퓨터/전자공학/ROBOT 관련학과 (RTOS 학습, CAN 통신, 모터/모션제어)
- **실습 Target Board :** FWB-TM4C-8750 또는 FWB-TM4C-7750
- **타겟 특징 :** 최대 80MHz 동작, 32K Flash, 12K SRAM, 2K EEPROM, CMSIS ROM 내장, Floating Point 엔진, 64Bit Timer, 12Ch ADC, CAN 통신, API-ROM, 부트코드 탑재
- **타겟 가격 :** 78,000원(일체 액세서리 포함, 반조립 상태)
- **타겟 구입처 :** www.FirmwareBank, www.Devicemart.co.kr, shop.naver.com/hoit
- **무료 Source Code 및 카페 :** cafe.naver.com/fws의 "Cortex-M4"
- **오픈 강의 :** 2014년 3월 29일, 4월 26일, 6월 10일 => 문의(02-582-2105)
- **서점 진열 위치 :** 컴퓨터IT/임베디드시스템, 컴퓨터공학/마이크로프로세서

p.598, 30000원
ISBN 9788957173657

- **제목 :** ARM® Cortex™-M0 NUC140 펌웨어개발
- **내용 :** Cortex-M0 확실 사용법, 너무 쉽고 방대한 예제, CAN 통신과 USB 익히기
- **대상 :** Nuvoton 32Bit RISC Cortex-M0의 Embedded 개발자
- **필요학과 :** 컴퓨터/전자/제어계측/통신공학 관련학과(CAN 통신, ROBOT, Toy개발)
- **실습 Target Board :** PAT-DAT-D8NN
- **타겟 특징 :** 최대 50MHz 동작, USB2.0, CAN Module, 내부OSC, USB, 부트코드 탑재
- **타겟 가격 :** 50,000원(일체 액세서리 포함, 반조립 상태)
- **타겟 구입처 :** www.FirmwareBank, www.Devicemart.co.kr
- **무료 Source Code 및 카페 :** cafe.naver.com/fws의 "Cortex-M0"
- **오픈 강의 :** 2014년 3월 19일, 6월 26일, 9월 26일, 12월 5일 => 문의 02-582-2105
- **서점 진열 위치 :** 컴퓨터IT/임베디드시스템, 컴퓨터공학/마이크로프로세서

p.488, 27000원
ISBN 9788957173312

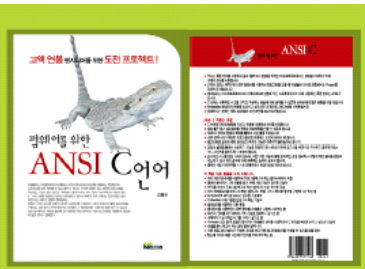
- **제목 :** ARM® Cortex™-M0 Nuvoton NuMicro 펌웨어개발
- **내용 :** Cortex-M0 엄청 쉬운 사용법, GPIO-DMA까지 방대한 예제
- **대상 :** Nuvoton 32Bit RISC Cortex-M0의 Embedded 개발자
- **필요학과 :** 컴퓨터/전자/제어계측/통신공학 관련학과(CAN 통신, 모듈/Toy개발)
- **실습 Target Board :** PAT-DAT-D7NM 또는 PAT-DAT-D8NE
- **타겟 특징 :** 내부 22MHz 동작, 내부OSC, Avago 광센서, 부트코드 탑재
- **타겟 가격 :** 27,000원(일체 액세서리 포함, 반조립 상태)
- **타겟 구입처 :** www.FirmwareBank, www.Devicemart.co.kr, www.icbanq.com
- **무료 Source Code 및 카페 :** cafe.naver.com/fws의 "Cortex-M0"
- **오픈 강의 :** 2014년 3월 19일, 6월 26일, 9월 26일, 12월 5일 => 문의 02-582-2105
- **서점 진열 위치 :** 컴퓨터IT/임베디드시스템, 컴퓨터공학/마이크로프로세서

p.311, 25000원
ISBN 9788957172759

- **제목 :** TI Cortex™-M3 펌웨어개발
- **내용 :** Cortex-M3 빠르게 익히기, 간단한 하드웨어 구성 방법, 컴파일러 연관 사용법
- **대상 :** TI사의 32Bit RISC MCU Cortex-M3의 Embedded System 개발자
- **필요학과 :** 컴퓨터/전자공학/ROBOT 관련학과 (Cortex-M3 기초 학습, RISC 구성)
- **실습 Target Board :** PAT-DAT-D7UM(LM3S811 Module)
- **타겟 특징 :** 최대 50MHz 동작, 표준48핀 개발, 인터럽트, ADC, 기본기능 구성, 부트코드 탑재
- **타겟 가격 :** 35,000원(일체 액세서리 포함, 반조립 상태)
- **타겟 구입처 :** www.FirmwareBank, www.Devicemart.co.kr
- **무료 Source Code 및 카페 :** cafe.naver.com/fws의 "Cortex-M3"
- **서점 진열 위치 :** 컴퓨터IT/임베디드시스템, 컴퓨터공학/마이크로프로세서

p.496, 25000원
ISBN 9788957171752

- **제목 :** RealView MDK ARM 컴파일러
- **내용 :** 32Bit RISC Cortex-M0/M3/M4, ARM 컴파일러 사용법, C/C++로 MCU 제어 방법, 회로설계와 풍부한 예제, 상용 라이브러리, Analog Device, TI, NI, NXP, ST, Cortex-M3의 제품 Quick Start 매뉴얼수록
- **대상 :** ARM7, ARM9, Cortex-M 시리즈 32Bit RISC MCU Embedded System 개발자
- **필요학과 :** 컴퓨터/전자공학/ROBOT 관련학과 (Cortex-M 기초 학습, RISC 프로그램)
- **실습 Target Board :** Cortex-M0/M3/M4, ARM으로 설계된 Embedded Board
- **카페 :** cafe.naver.com/fws의 "RTOS/컴파일러"
- **서점 진열 위치 :** 컴퓨터IT/임베디드시스템, 컴퓨터공학/마이크로프로세서



p.380, 25000원
ISBN 895717124X

- **제목 :** 펌웨어를 위한 ANSI C
- **내용 :** MCU/32Bit RISC Embedded System의 프로그램을 위한 C 언어 학습, 시스템 개발 및 적용 방법수록, API 함수 이용법
- **대상 :** Embedded, 마이크로프로세서, MCU 프로그램, Firmware 엔지니어 기초 기술 필요한 경우, C 프로그램에 의해 제어하고자 하는 기술자
- **필요학과 :** 컴퓨터/Embedded/ROBOT 관련학과 (C 언어 기초 학습, 전자회로와 관련 학습)
- **실습 Target Board :** Cortex-M0/M3/M4, ARM으로 설계된 Embedded Board
- **카페 :** cafe.naver.com/fws의 "단행본"
- **서점 진열 위치 :** 컴퓨터IT/임베디드시스템, 컴퓨터공학/마이크로프로세서, 컴퓨터공학/소프트웨어공학