

RISC-V eXpress Install Manual

한규승, 이석호, 장형욱, 김상필, 이재진

한국전자통신연구원

May 17, 2024

Contents

1	개요	3
2	주의사항	3
3	준비물	3
4	알아두어야 할 사항	5
5	설치 1단계	6
5.1	목표	6
5.2	Git Program	6
5.3	Python 3	6
5.3.1	Linux	6
5.3.2	Windows	7
5.4	Make	7
6	설치 2단계	8
6.1	목표	8
6.2	Git Repository	8
6.3	RVX를 위한 python 3 설정	8
6.4	RVX를 위한 pip 3 설치 및 설정	9
6.5	Install	9
7	설치 3단계	10
7.1	Olimex Driver 설치 (Windows Only)	10
7.2	Telnet 설치 (Windows Only)	10
7.3	Java (GUI를 사용할 경우)	10
8	설치 4단계 (라이센스 프로그램)	10
9	동기화	11
10	계정 초기화 및 동기화	11
11	설정파일	11

12 Install 이후	12
12.1 Install 확인	12
12.2 User Manual	12
12.3 자동 Update	12
12.4 수동 Update	12

1 개요

- RISC-V eXpress (이하 RVX) 는 RISC-V 환경을 활용하여 MCU 급의 저전력 SoC를 개발하기 위한 HW/SW development kit 이다.
 - 논문제목: Developing TEI-Aware Ultralow-Power SoC Platforms for IoT End Nodes
 - 게재지: IEEE Internet of Things Journal, vol. 8, no. 6, 2021
 - <https://ieeexplore.ieee.org/document/9208740>
- 본 메뉴얼은 rvx mini git을 기반으로 사용자의 컴퓨터에 RVX를 설치하기 위한 메뉴얼이다.

2 주의사항

- RVX를 이용하여 만든 모든 결과물은,
 - 특정 수업 또는 연구과제 등 사전에 지정된 목적과 범위를 벗어나서 활용할 수 없다.
 - 지정된 사용자 또는 사용기관 이외에 제3자에게 배포할 수 없다.
 - 상업적 목적으로 사용하려면 기술이전이 필요하다.
- ETRI에서 제공하는 RVX server을 사용할 경우,
 - RVX mini를 통한 제공된 기능만을 이용해야한다.
 - 이를 의도적으로 위반하여 사고가 발생하면 사용자가 비용을 포함한 모든 책임을 진다.

3 준비물

- 라이선스 툴
 - RTL 시뮬레이터 = Modelsim/Questa or Xcelium/NCSim/Incisive
 - FPGA 툴 = Xilinx Vivado 2021.2 이상
- 성능이 좋은 컴퓨터
 - 라이선스 툴들을 돌리기 위해서는 고성능이 필요하다.
 - 실제로 필요한 성능은 각 라이선스 툴의 요구사항을 참조한다.
 - 참고로, RVX 자체 엔진은 고성능을 요구하지 않는다.
- 아래 해당하는 OS 중 라이선스 툴이 안정적으로 돌아가는 버전 설치 권장
 - CentOS 7, 8
 - Redhat 8, 9
 - Ubuntu 18, 19, 20, 22, 24
 - Windows 10 (Home, Enterprise)
 - Windows 11 Home
- 컴퓨터의 관리자 권한이 있는 계정
 - Linux는 sudo 권한, Windows는 관리자 권한이 있어야한다.
- RVX git 주소

- 교수 또는 프로젝트 관리자 등에게 받도록 한다.
- RVX server의 IP 주소, 계정, 비밀번호
 - 교수 또는 프로젝트 관리자 등에게 받도록 한다.
 - 사용할 컴퓨터 및 컴퓨터 계정마다 하나씩 필요하다.

4 알아두어야 할 사항

- RVX git repository가 설치된 위치를 `${RVX_MINI_HOME}`라고 부른다.
- 메뉴얼에서 알려주는 명령 중에서 `#`로 시작되는 부분은 상황에 맞게 바꿔서 입력해야한다.
- Linux에서 명령어창은 bash shell을 사용한다.
- Linux에서 특정 디렉토리를 PATH 변수에 추가하는 방법
 - ① `.bashrc` 파일에 `"export PATH=$PATH:$(특정디렉토리)"`를 추가한다.
- Windows에서 명령어창은 [Windows Power Shell]을 사용한다.
- Windows에서 [Windows Power Shell]를 실행하는 방법
 - ① 검색에서 power를 검색한다.
 - ② [Windows Power Shell]를 클릭한다.
- Windows에서 특정 디렉토리 위치에서 명령어창을 실행하는 방법
 - ① Windows 탐색기에서 원하는 디렉토리로 이동한다.
 - ② 오른쪽 파일뷰에서 쉬프트키를 누른 상태로 마우스 오른쪽 클릭을 한다.
 - ③ [여기서 PowerShell 창 열기]를 클릭한다.
- Windows에서 환경변수를 설정하는 방법
 - ① [시스템 속성]을 연다.
 - ex) 검색창에서 SystemPropertiesAdvanced.exe를 검색해서 실행한다.
 - ex) 또는, `${RVX_MINI_HOME}\install\open_system_properties.bat`를 실행한다.
 - ex) 또는, 명령어창에서 SystemPropertiesAdvanced.exe를 실행한다.
 - ② [고급]탭에서 [환경변수] 클릭
 - ③ 시스템 변수가 아니라 사용자 변수에 추가한다.
 - ④ 추가하려는 변수가 이미 존재하면 해당 변수를 클릭한 후 [편집]을 한다.
 - ⑤ 추가하려는 변수가 존재하지 않으면 [새로 만들기]를 한다.
 - ⑥ 환경변수를 수정하고 나서는 기존의 명령어창을 꺾다가 다시 켜야한다.

5 설치 1단계

5.1 목표

Git, make, python3, Java 환경을 설치한다.

널리 사용되는 프로그램들로, [목표]만 만족한다면 다른 방식으로 설치해도 무관하다.

5.2 Git Program

- [목표] git 1.7.10 이상을 설치하고 확인한다.
- [확인] `git --version`
- [방법 CentOS]
 - ① `sudo yum install git`
 - ② 버전 확인을 해서 맞지 않으면 각자 알아서 설치한다.
- or
- [방법 Ubuntu]
 - ① `sudo apt-get install git`
 - ② 버전 확인을 해서 맞지 않으면 각자 알아서 설치한다.
- or
- [방법 Windows] <https://git-scm.com/download/win>
- [참고] Section 6.2에 따라 repository를 받으면
최신 매뉴얼이 `${RVX_MINI_HOME}/rvx_install`에 있다.

5.3 Python 3

- [목표] python 3.8에서 3.10 사이를 설치하고 확인한다.
- [확인] `python3 --version` or `python --version`
- [참고] 사용자 환경에 따라 명령어가 `python3` 또는 `python`

5.3.1 Linux

- [방법 CentOS]
 - ① `sudo yum install python3`
 - ② 설치가 안 되거나 버전 확인을 해서 맞지 않으면 [방법 Linux] 또는 각자 알아서 설치한다.
- or
- [방법 Ubuntu]
 - ① `sudo apt-get install python3`
 - ② 설치가 안 되거나 버전 확인을 해서 맞지 않으면 [방법 Linux] 또는 각자 알아서 설치한다.
- or

– [방법 Linux]

- ① `sudo yum install gcc zlib zlib-devel openssl openssl-devel`
- ② `wget https://www.python.org/ftp/python/3.8.16/Python-3.8.16.tar.xz`
- ③ `tar xvf Python-3.8.16.tar.xz`
- ④ `cd Python-3.8.16; ./configure; make; sudo make install`

5.3.2 Windows

– [방법 Windows]

- ① Microsoft Store에서 python3를 검색하여 3.8 버전을 설치한다.
⇒ 가능하다면 이 방법을 더 추천한다.
⇒ 3.7에서 3.9 사이이면 다 괜찮다.

or

– [방법 Windows]

- ① <https://www.python.org>에서 3.9.12 버전을 다운받아 설치한다.
- ② 3.7에서 3.9 사이이면 다 괜찮지만 Windows installer 가 있는 버전으로 해야한다.
- ③ 설치할 때 반드시 [Add Python 3.x to PATH]를 활성화한다.
- ④ 만약 실수로 하지 않았다면, Path 환경변수에 두 디렉토리를 직접 추가한다.
ex) C:\Users\kshan\AppData\Local\Programs\Python\Python3x
ex) C:\Users\kshan\AppData\Local\Programs\Python\Python3x\Scripts

5.4 Make

– [목표] Makefile이 동작하는 환경을 만든다.

– [방법 Linux] 기본적으로 지원하기 때문에 별도로 설치할 필요가 없다.

– [방법 Windows]

- ① <http://gnuwin32.sourceforge.net/packages/make.htm> 에서 Complete package를 받아 설치한다.
- ② Path 환경변수에 설치된 디렉토리를 추가한다.
ex) C:\Program Files (x86)\GnuWin32\bin

6 설치 2단계

6.1 목표

RVX git repository를 설치하고 RVX를 위한 python3 환경을 설정한다.
python3가 올바르게 설치되고 설정되었는지 확인하고
여러 버전의 python3를 가지고 있을 경우 충돌을 해결한다.

6.2 Git Repository

- [방법 Linux & Windows]

- ① 설치하고 싶은 디렉토리의 상위 디렉토리로 이동한다.
- ② `git clone --recursive #(git 주소)`
 - ⇒ 생성된 디렉토리를 `${RVX_MINI_HOME}`로 부른다.
 - ⇒ 실제 변수 설정은 Section 6.5에서 이루어진다.

or

- ② `git clone #(git 주소)`
 - ⇒ 생성된 디렉토리를 `${RVX_MINI_HOME}`로 부른다.
 - ⇒ 실제 변수 설정은 Section 6.5에서 이루어진다.
- ③ 생성된 디렉토리로 이동한다.
- ④ `git submodule init`
- ⑤ `git submodule update`

6.3 RVX를 위한 python 3 설정

- [목표] RVX에서 사용할 python3 명령어를 설정하고 확인한다.

- [확인] `make check_python`

- [방법 Linux]

- ① `${RVX_MINI_HOME}`로 이동한다.
- ② `make config_python`
 - ⇒ `rvx_python_config.mh` 이 생성됨을 확인
- ③ `rvx_python_config.mh`을 텍스트 편집기로 열어 확인하고 필요한 경우 수정한다.
 - ⇒ 경로가 하나도 없다면 python이 설치되지 않았거나 PATH 설정에 문제가 있는 것이다.
 - ⇒ 경로가 2개 이상이라면 사용할 1개만 남기고 지운다.

- [방법 Windows]

- ① `${RVX_MINI_HOME}`로 이동한다.
- ② `make config_python`
 - ⇒ `rvx_python_config.mh` 이 생성됨을 확인 (수정하지 않음)
 - ⇒ `python3.bat` 이 생성됨을 확인
- ③ `python3.bat`을 텍스트 편집기로 열어 확인하고 필요한 경우 수정한다.
 - ⇒ 경로가 하나도 없다면 python이 설치되지 않았거나 PATH 설정에 문제가 있는 것이다.

- ⇒ 경로가 2개 이상이라면 사용할 1개만 남기고 지운다.
 - * 맨마지막에 있는 "%*" 는 지우면 안 된다.
- ⇒ 디렉토리 이름에 빈칸이 있다면 아래와 같이 따옴표 처리 한다.
 - ex) C:\Users\"Kyuseung Han"\... %*

6.4 RVX를 위한 pip 3 설치 및 설정

- [목표] RVX에서 사용할 python package를 설치하고 확인한다.
- [확인] make check_pip
- [방법 Linux & Windows]
 - ① make pip
- * 문제가 생겼거나 손상이 생긴 경우
 - ① make fix_pip
 - ② make pip

6.5 Install

- [방법 Linux]
 - ① \${RVX_MINI_HOME}로 이동한다.
 - ② make install
 - ⇒ ./rvx_setup.sh 파일 생성 확인
 - ③ .bashrc에 "source #(RVX_MINI_HOME)/rvx_setup.sh"를 추가한다.
 - ⇒ # 부분은 반드시 절대 주소로 바꿔서 추가한다.
 - ④ .bashrc을 적용하기 위해 명령창을 꺾다 다시 켜다.
- [방법 Windows]
 - ① \${RVX_MINI_HOME}로 이동한다.
 - ② make install
 - ③ PATH 환경변수에 대해
 - ⇒ 성공적으로 업데이트 되었다는 INFO가 출력된 경우는 다음으로 넘어간다.
 - ⇒ 직접 추가하라고 WARNING이 출력된 경우는
 - "\${RVX_MINI_HOME}\windows_binary"를 직접 추가한다.
 - ④ RVX_MINI_HOME 환경변수가 새로 추가되었으니 명령창을 꺾다 다시 켜다.

7 설치 3단계

7.1 Olimex Driver 설치 (Windows Only)

– [방법 Windows]

- ① Olimex ARM-USB-TINY-H 케이블을 컴퓨터에 연결한다.
- ② zadig 프로그램을 실행한다.
⇒ `${RVX_MINI_HOME}\windows_binary` 에 있는 `zadig-2.5.exe`를 실행한다.
or
⇒ <https://zadig.akeo.ie>에서 다운받아 실행한다.
- ③ [Olimex OpenOCD JTAG ARM-USB-TINY-H (Interface 0)] 를 선택하고 [Install Driver]를 누른다.
- ④ [Olimex OpenOCD JTAG ARM-USB-TINY-H (Interface 1)] 를 선택하고 [Install Driver]를 누른다.

7.2 Telnet 설치 (Windows Only)

– [방법 Windows]

- ① [제어판] 실행
ex) control 검색
- ② [프로그램] 클릭
- ③ [Windows 기능 켜기/끄기] 클릭
- ④ [텔넷 클라이언트] 체크 활성화
- ⑤ 확인

7.3 Java (GUI를 사용할 경우)

- [목표] OpenJDK를 설치하고 확인한다.
- [확인] `java -version`
- [방법 Linux] `yum` 또는 `apt` 으로 각자 검색해서 설치한다. 버전은 크게 상관 없다.
- [방법 Windows]
 - ① <https://openjdk.java.net/>에서 다운받는다. 버전은 크게 상관 없다.
 - ② 압축을 풀고 생성되는 디렉토리를 설치하고자 하는 위치로 이동시킨다.
 - ③ 설치한 디렉토리의 하위 디렉토리인 `bin`을 `PATH` 변수에 추가한다.

8 설치 4단계 (라이선스 프로그램)

- ① Xilinx Vivado 2021.2이상의 버전을 설치하고 설치경로를 `PATH` 변수에 추가한다.
⇒ Windows에서는 자동으로 추가될 수도 있다.
- ② ModelSim 또는 NCsim 을 설치하고 설치경로를 `PATH` 변수에 추가한다.
⇒ Windows에서는 자동으로 추가될 수도 있다.

9 동기화

- RVX 서버에서 구동환경을 가져오는 작업이다.
 - ⇒ Git을 처음 받았을 때는 반드시 한번 실행해야한다.
 - ⇒ 이후에는 update 과정에서 자동으로 실행된다.
 - ⇒ 동기화가 끝나면 최신 버전의 메뉴얼들이 `${RVX_MINI_HOME}/manual`에 생긴다.
- 동기화 과정 중에 RVX 서버의 계정정보를 입력하게 된다.
 - ⇒ 계정정보는 미리 알고 있어야한다.
 - ⇒ `${RVX_MINI_HOME}/.rvx_server_config`에 저장된다.
 - ⇒ RVX 계정정보를 변경하고 싶은 경우 해당 파일을 삭제한다.
- [방법 Linux & Windows]
 - ① `${RVX_MINI_HOME}`로 이동한다.
 - ② `make sync`
 - ⇒ SSH 접속 여부는 `no`로 대답한다.

10 계정 초기화 및 동기화

- `sync` 과정에서 문제가 생길 때 실행한다.
- [방법 Linux & Windows]
 - ① `${RVX_MINI_HOME}`로 이동한다.
 - ② `make resync`

11 설정파일

- 설정에 문제가 있을 경우 아래 파일목록 중 관련된 파일 삭제하고 진행한다.
 - ⇒ `${RVX_MINI_HOME}/.rvx_path_config`
 - ⇒ `${RVX_MINI_HOME}/.rvx_sudo_config`
 - ⇒ `${RVX_MINI_HOME}/.rvx_tool_config`
 - ⇒ `${RVX_MINI_HOME}/.rvx_server_config`
- `${RVX_MINI_HOME}/.rvx_key`를 삭제한 경우, 위에 모든 파일을 삭제한다.

12 Install 이후

12.1 Install 확인

- make check
⇒ 도중에 에러가 나면 안 된다 !!!

12.2 User Manual

- \${RVX_MINI_HOME}/manual에 있다.

12.3 자동 Update

- 사용 중에 Update하라는 메시지가 뜨면 진행한다.
- [방법 Linux & Windows]
 - ① 작업한 내용이 있다면 platform 디렉토리를 다른 곳으로 백업한다.
 - ② \${RVX_MINI_HOME}로 이동한다.
- [방법 Linux (Cont.)]
 - ③ ./update.sh
⇒ update.sh 파일이 없으면 make config
 - ④ ./rvx_setup.sh 파일이 새로 생겼으므로 명령어창을 새로 켜서 작업한다.
- [방법 Windows (Cont.)]
 - ③ update.bat
⇒ update.bat 파일이 없으면 make config
 - ④ 환경변수가 업데이트 되었으므로 명령어창을 새로 켜서 작업한다.
 - ⑤ 플랫폼을 clean을 하고 syn부터 다시 작업한다.

12.4 수동 Update

- 관리자의 요청에 의해 진행한다.
- [방법 Linux & Windows]
 - ① \${RVX_MINI_HOME}로 이동한다.
 - ② 작업한 내용이 있다면 platform 디렉토리를 다른 곳으로 백업한다.
 - ③ git checkout .
 - ④ git pull origin master
 - ⑤ git submodule init
 - ⑥ git submodule update
 - ⑦ make reconfig_python
 - ⑧ make pip3
 - ⑨ make config
 - ⑩ 명령어창을 새로 켜서 작업한다.
 - ⑪ make sync
 - ⑫ 플랫폼을 clean을 하고 syn부터 다시 작업한다.