RISC-V Cross-Compilation Toolchain Setup

Pre-requisites:

- Download `WSL` or `Virtual Machine`.
- Install any version of `UBUNTU` on `WSL` or `VM`. (preferably the latest)
- Download `Terminal` from Microsoft Store. (optional)

Step-by-Step guide:

ν Open the terminal and start following the given steps.

- **Set your environment variables or Paths:**
  - **Open the `.bashrc` file:**
    - `nano ~/.bashrc`
  - **Paste the following commands at the end of the `.bashrc` file.**
  - `export RISCV_PATH="$HOME/proj/binaries/riscv-toolchain"`
  - `export RISCV_TOOLCHAIN=$RISCV_PATH`
  - `export SPIKE_PATH=$RISCV_TOOLCHAIN/bin`
  - `export SPIKE_PATH_32=$RISCV_TOOLCHAIN/bin/spike32`
  - `export SPIKE_PATH_64=$RISCV_TOOLCHAIN/bin/spike64`
  - `export SPIKE_PATH_DEFAULT=$RISCV_TOOLCHAIN/bin/default`
  - `export PK_PATH=$RISCV_TOOLCHAIN/bin`
  - `export PK_PATH_32=$RISCV_TOOLCHAIN/bin/pk32`
  - `export PK_PATH_64=$RISCV_TOOLCHAIN/bin/pk64`
  - `export PATH=$RISCV_TOOLCHAIN/rv64/bin:$PATH`
  - `export PATH=$RISCV_TOOLCHAIN/rv32/bin:$PATH`
  - `export PATH=$RISCV_TOOLCHAIN/multi/bin:$PATH`
  - `export PATH=$RISCV_TOOLCHAIN/riscv32-unknown-elf/bin:$PATH`
  - `export PATH=$RISCV_TOOLCHAIN/riscv64-unknown-elf/bin:$PATH`
  - `export PATH=$SPIKE_PATH_32/bin:$PATH`
  - `export PATH=$SPIKE_PATH_64/bin:$PATH`
  - `export PATH=$SPIKE_PATH_DEFAULT/bin:$PATH`
  - `export PATH=$PK_PATH_32/riscv32-unknown-elf/bin:$PATH`
  - `export PATH=$PK_PATH_64/bin:$PATH`
    `export pk=$RISCV_TOOLCHAIN/bin/pk32/riscv32-unknown-elf/bin/pk`
  - **Press `Ctrl+S` to save. Then press `Ctrl+X` to exit.**
  - **To apply the changes we made in the `.bashrc` file, run the following command on the terminal:**
  - `source ~/.bashrc`

- **Make directories:**

- `mkdir -p ~/proj/tools; cd ~/proj; mkdir -p binaries/riscv-toolchain; cd tools;`

- **The following command will clone the official `RISC-V Toolchain` GitHub repository and it will make the directory `riscv-gnu-toolchain` inside the `tools` directory:**
  `git clone --recursive  https://github.com/riscv/riscv-gnu-toolchain`

NOTE: You can remove `--recursive` if you have a stable internet connection throughout the installation process. The recursive approach downloads all submodules of the `Git` and vice versa.

- **For `Ubuntu` dependencies, run the following commands:**
- `sudo apt-get -y install autoconf automake autotools-dev curl python3 libmpc-dev libmpfr-dev libgmp-dev gawk build-essential bison flex texinfo gperf libtool patchutils bc zlib1g-dev libexpat-dev python3-pip`
- **Inside the `riscv-gnu-toolchain` directory, make a directory `build` and open it:**
- `cd riscv-gnu-toolchain; mkdir build; cd build`

- **Configure the toolchain:**

  - **For pure `64-bit` toolchain:**
    ```
    ../configure --prefix=$RISCV_PATH/rv64  --enable-multilib --with-arch=rv64imafdqcv_zifencei_zicsr
    make -j3
    ```
  - **For pure `32-bit` toolchain:**
    ```
    ../configure --prefix=$RISCV_PATH/rv32  --enable-multilib --with-arch=rv32imafdqcv_zifencei_zicsr
    make -j3
    ```
  - **For `multi-bit support` toolchain:**
    ```
    ../configure --prefix=$RISCV_PATH/multi  --enable-multilib
    make -j3
    ```
    NOTE: `C` extension of RISC-V architecture in `--with-arch=rv32/64 imafdqcv_zifencei_zicsr` is for compressed instructions support, you can remove it if you don't want compressed instructions. In the command `make -j3`, `3` represents the utilization of the number of `CPU cores` while installing, you can increase this accordingly.

- **Installation of `spike`:**
  - **Move to the `tools` directory and clone the link below which will make a directory `spike-riscv` inside the `tools` directory:**
    `git clone --recursive https://github.com/riscv-software-src/riscv-isa-sim.git spike-riscv`

  - **This step will build the necessary dependencies:**

    `sudo apt-get install device-tree-compiler`
  - **Inside the `spike-riscv` directory:**
    `mkdir build;cd build`
    - **Configure `32-bit spike`: (for `32-bit` toolchain)**
      ```
      ../configure --prefix=$SPIKE_PATH_32 --with-isa=RV32IMAFDQCV
      make -j3
      ```

```
sudo make install
```

- **Configure `64-bit spike`: (for `64-bit` toolchain)**

```
../configure --prefix=$SPIKE_PATH_64
make -j3
sudo make install
```

NOTE: Also here `C` extension should be removed if you have done this earlier while configuring the toolchain.

- **Installation of `Pk`:**
  - **Clone inside the `tools` directory which will make a directory `riscv-pk` inside the `tools` directory:**
  - `git clone --recursive https://github.com/riscv-software-src/riscv-pk.git`
  - **Inside the `riscv-pk` directory:**

```
mkdir build;cd build
```

- **Configure 32-bit pk: (for 32-bit toolchain & 32-bit spike)**

```
../configure --prefix=$PK_PATH_32 --host=riscv32-unknown-elf --
with-arch=rv32imafdqcv_zifencei_zicsr --with-abi=ilp32d
make -j3
sudo make install
```

- **Configure 64-bit pk: (for 64-bit toolchain & 64-bit spike)**

```
../configure --prefix=$PK_PATH_64 --host=riscv64-unknown-elf --
with-arch=rv64imafdqcv_zifencei_zicsr
make -j3
sudo make install
```

NOTE: Also here `C` extension should be removed if you have done this earlier while configuring the toolchain and the `spike`.

Toolchain test with an example:

- **Download `VS Code` and link it with `WSL`.**
- **On the terminal, create a separate directory in the `proj` directory for test code files. (good practice)**
- **Write a short `C` code in VS code and save it in the test code folder you made in the previous step.**

- **On the terminal, move to the code directory and run the C code using the following command:**

- `riscv32-unknown-elf-gcc -o (output filename) (C filename).c`

- **You will find the output file in the same directory.**

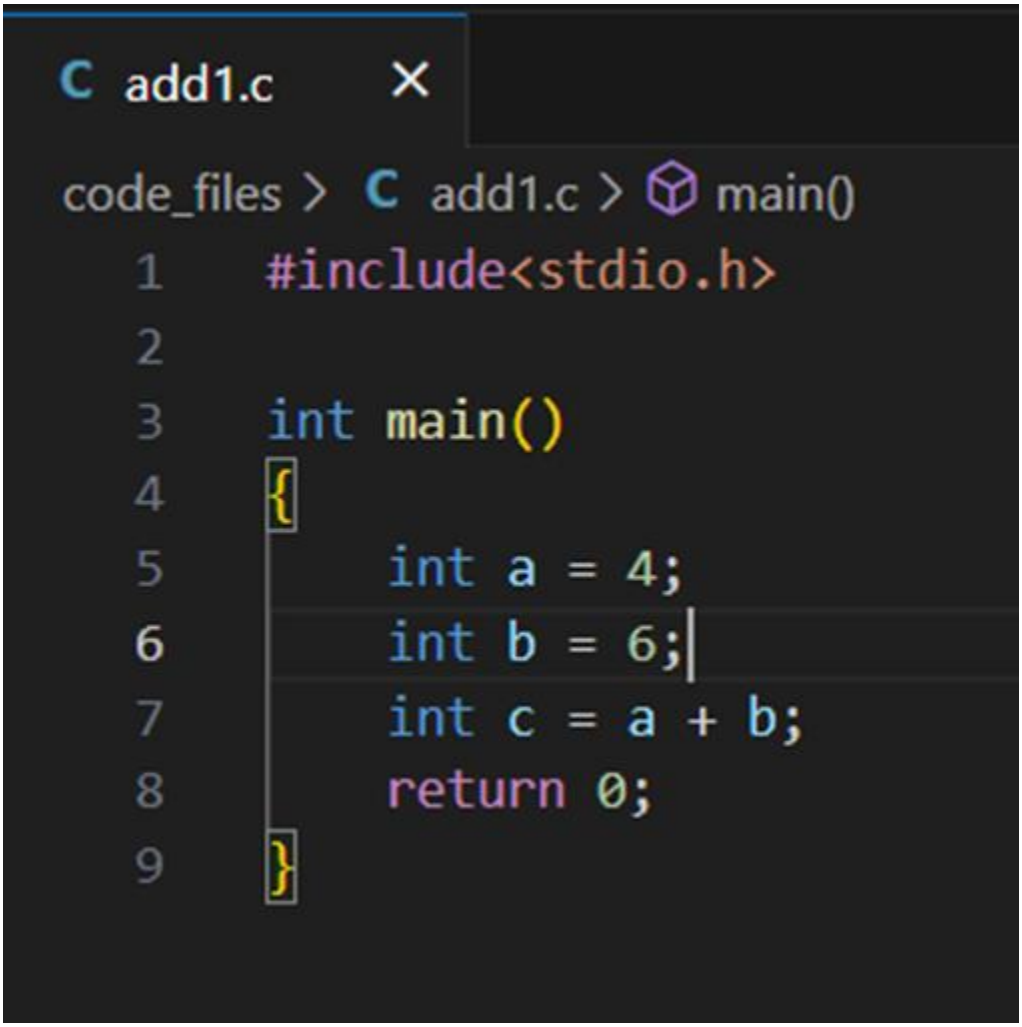
- **Now, run the output file using `spike` simulator and `pk` using the following command:**
    - **To run the assembly program normally:**

```
spike $pk (output filename)
```

    - **To run the assembly program line-by-line:**

```
spike -d $pk (output filename)
```

Screenshots of Example:



```
[redacted]:~/proj$ cd code_files/
[redacted]:~/proj/code_files$ ls
add.c  add1  add1.c  add1.dis  add2  add2.dis
[redacted]:~/proj/code_files$ riscv32-unknown-elf-gcc -o add2 add1.c
[redacted]:~/proj/code_files$ spike -d $pk add2
(spike)
core   0: 0x00001000 (0x00000297) auipc   t0, 0x0
(spike)
core   0: 0x00001004 (0x02028593) addi    a1, t0, 32
(spike)
core   0: 0x00001008 (0xf1402573) csrr    a0, mhartid
(spike)
core   0: 0x0000100c (0x0182a283) lw      t0, 24(t0)
(spike)
core   0: 0x00001010 (0x00028067) jr      t0
(spike)
core   0: 0x80000000 (0x1f80006f) j       pc + 0x1f8
(spike)
core   0: 0x800001f8 (0x00000093) li      ra, 0
(spike) reg 0 ra
0x00000000
(spike)
```