Ubuntu to Ubuntu Cross C++ Development

Pieter P

The steps are almost identical to the Ubuntu to Raspberry Pi OS Cross C++ Development guide, with the main differences being that you use a different distribution and release when calling `mk-sbuild`, and that you'll use the 64-bit toolchain.

Brief instructions for Ubuntu 21.10 (Impish Indri), 64-bit:

```
1   # Install the necessary tools:
2   sudo apt install ubuntu-dev-tools
3   # Then create a schroot environment for the arm64 environment, using Ubuntu
4   # Impish (remember the name you used, you'll need it later):
5   mk-sbuild --arch=arm64 --skip-proposed --skip-updates --skip-security --name=rpi3-impish impish
6   # If this is the first time you run mk-sbuild, you'll have to follow some
7   # instructions. Since we won't be publishing any software, you can simply accept
8   # the default configuration. Afterwards, reboot or log out and back in again, as
9   # instructed. Alternatively, use
10  su - $USER # to flush group membership.
11  # Then run the command again, this time it will actually create the schroot:
12  mk-sbuild --arch=arm64 --skip-proposed --skip-updates --skip-security --name=rpi3-impish impish
13  # Install some dependencies in the schroot. Use the sbuild-apt wrapper around
14  # the apt-get tool, and give it the name of the schroot you created earlier,
15  # with the architecture as suffix:
16  sudo sbuild-apt rpi3-impish-arm64 apt-get install libboost-all-dev
17  # Download and extract the cross-compilation toolchain:
18  wget -qO- https://github.com/tttapa/docker-arm-cross-toolchain/releases/latest/download/x-tools-aarch64-rpi3-linux-gnu.tar.bz2 | tar xJ -C ~/opt
19  # Add the toolchain to your path:
20  export PATH="$HOME/opt/x-tools/aarch64-rpi3-linux-gnu/bin:$PATH"
21  # Download the repository with the example CMake C++ project:
22  git clone https://github.com/tttapa/RPi-Cross-Cpp-Development.git
23  # Enter it:
24  cd RPi-Cross-Cpp-Development
25  # Replace the schroot name in the toolchain file by the name used earlier:
26  sed -i 's/schroot-name-arm64/rpi3-impish-arm64/' cmake/aarch64-rpi3-linux-gnu.cmake
27  # Configure the project for cross-compilation:
28  cmake -S. -Bbuild -DCMAKE_TOOLCHAIN_FILE="$PWD/cmake/aarch64-rpi3-linux-gnu.cmake"
29  # Build the project:
30  cmake --build build -j
31  # Install the project into the staging area:
32  cmake --install build
33  # Install the dependencies on the Raspberry Pi:
34  ssh RPi3 sudo apt-get install -y libboost-program-options1.74.0
35  # Copy the "Hello World" example program to the Raspberry Pi:
36  ssh RPi3 mkdir -p '~/.local/bin'
37  scp ~/RPi-dev/staging-aarch64-rpi3/bin/hello RPi3:~/.local/bin
38  # Run the example program:
39  ssh RPi3 bash --login -c hello
40  # Or simply log in using
41  ssh RPi3
42  # and then run the program there:
43  hello
```

```
1   # Copy the toolchain's gdbserver to the Pi:
2   scp ~/opt/x-tools/aarch64-rpi3-linux-gnu/aarch64-rpi3-linux-gnu/debug-root/usr/bin/gdbserver RPi3:~
3   # Install it:
4   ssh RPi3 sudo mv gdbserver /usr/local/bin
5   # Test it:
6   ssh RPi3 gdbserver --version
7   # Start GDB with the hello program on your computer:
8   aarch64-rpi3-linux-gnu-gdb ~/RPi-dev/staging-aarch64-rpi3/bin/hello
9   # Inside of GDB, set the sysroot:
10  set sysroot /var/lib/schroot/chroots/rpi3-impish-arm64
11  # Select and start the target:
12  target remote | ssh RPi3 gdbserver - '~/.local/bin/hello' --name Pieter
13  # Run program:
14  continue
15  # Exit GDB:
16  quit
```

```
$ aarch64-rpi3-linux-gnu-gdb ~/RPi-dev/staging-aarch64-rpi3/bin/hello
GNU gdb (crosstool-NG UNKNOWN) 10.2
Copyright (C) 2021 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "--host=x86_64-build_pc-linux-gnu --target=aarch64-rpi3-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ~/RPi-dev/staging-aarch64-rpi3/bin/hello...
(gdb) set sysroot /var/lib/schroot/chroots/rpi3-impish-arm64
(gdb) target remote | ssh RPi3 gdbserver - '~/.local/bin/hello' --name Pieter
Remote debugging using | ssh RPi3 gdbserver - '~/.local/bin/hello' --name Pieter
stdin/stdout redirected
Process /home/ubuntu/.local/bin/hello created; pid = 5815
Remote debugging using stdio
Reading symbols from /var/lib/schroot/chroots/rpi3-impish-arm64/lib/ld-linux-aarch64.so.1...
(No debugging symbols found in /var/lib/schroot/chroots/rpi3-impish-arm64/lib/ld-linux-aarch64.so.1)
0x0000fffff7fc8d40 in ?? ()
    from /var/lib/schroot/chroots/rpi3-impish-arm64/lib/ld-linux-aarch64.so.1
(gdb) c
Continuing.
Hello, Pieter!
[Inferior 1 (process 5815) exited normally]
(gdb) q
```