Setup - Installation of Vitis on Ubuntu

Vitis has the most documentation on linux and Ubuntu is one of the recommended platforms. This setup tutorial shows you how to:

Install Vitis — Contains Vitis IDE, Vitis HLS, Vivado and Vivado HLS, Install OpenCL Client Driver — OpenCL drivers, Install the Xilinx Runtime — XRT for OpenCL, Install the appropriate platform (we install U200 by Xilinx)

Based off of (and heavily borrows from) https://www.hackster.io/news/microzed-chronicles-vitis-acceleration-creating-a-vm-setup-363946fb4ede

Download Vitis from here: https://www.xilinx.com/products/design-tools/vitis.html

Download the following OpenCL drivers:

sudo apt install ocl-icd-libopencl1 sudo apt install opencl-headers

sudo apt install ocl-icd-opencl-dev

Install xilinx runtime (dpkg file) from here:

https://www.xilinx.com/bin/public/openDownload?filename=xrt_201920.2.3.1301_18.04-xrt.deb

Install the target device (dpkg file) from here:

https://www.xilinx.com/products/boards-and-kits/alveo/u200.html#xrt1804_20191_xdma

The dpkg files need to be installed with sudo apt install <dpkg_file_name>.dpkg

To check if everything is installed correctly

git clone https://github.com/Xilinx/Vitis-Tutorials.git
cd Vitis-Tutorials/docs/my-first-program/reference-files
g++ -I$XILINX_XRT/include/ -I$XILINX_VIVADO/include/ -Wall -O0 -g -std=c++11 ./src/host.cpp -o 'host' -L$XILINX_XRT/lib/ -lOpenCL -lpthread -lrt -lstdc++

This should run with no error.

Bug Fixes

On Ubuntu 18.04, if youget the following error: **../src/host.cpp:47:10: fatal error:** CL/cl2.hpp: No such file or directory

- Then unzip https://www.xilinx.com/member/forms/download/design-license-xef.html?filename=rdf0476-zcu111-rf-dc-eval-tool-2019-2.zip

And run

```
cp $(download_location)/rdf0476-zcu111-rf-dc-eval-tool-2019-2/pl/MTSDesign_8x8/pre-built/rfsoc_trd/zcu111_rfsoc_trd_wrapper.xsa $(vitis_install_location)/2019.2/data/embeddedsw/lib/fixed_hwplatforms
```

- If Vitis is asking for XILINX_VITIS to be set, then run the following:

  source opt/xilinx/xrt/setup.sh

How to Run Software Emulation and Hardware Emulation

In vitis_emulation/, run the following for Software Emulation:

g++ -I$XILINX_XRT/include/ -I$XILINX_VIVADO/include/ -Wall -O0 -g -std=c++11 host.cpp dense2.cpp dense1.cpp conv7.cpp conv6.cpp conv5.cpp conv4.cpp conv3.cpp conv2.cpp conv1.cpp bnd2_a_b.cpp bnd1_a_b.cpp bn7_a_b.cpp bn6_a_b.cpp bn5_a_b.cpp bn4_a_b.cpp bn3_a_b.cpp bn2_a_b.cpp bn1_a_b.cpp -o 'host' -L$XILINX_XRT/lib/ -lOpenCL -lpthread -lrt -lstdc++

v++ -t sw_emu --config design.cfg -c -k compute_network -o'compute_network.xilinx_u200_xdma_201830_2.xo' 'tw_vgg10.cpp' 'tw_vgg10.h' 'pred_output.h' 'dense2.h' 'dense2.cpp' 'dense1.h' 'dense1.cpp' 'conv7.h' 'conv7.cpp' 'conv6.h' 'conv6.cpp' 'conv5.h' 'conv5.cpp' 'conv4.h' 'conv4.cpp' 'conv3.h' 'conv3.cpp' 'conv2.h'

'conv2.cpp' 'conv1.h' 'conv1.cpp' 'bnd2_a_b.h' 'bnd2_a_b.cpp' 'bnd1_a_b.h' 'bnd1_a_b.cpp' 'bn7_a_b.h' 'bn7_a_b.cpp' 'bn6_a_b.h' 'bn6_a_b.cpp' 'bn5_a_b.h' 'bn5_a_b.cpp' 'bn4_a_b.h' 'bn4_a_b.cpp' 'bn3_a_b.h' 'bn3_a_b.cpp' 'bn2_a_b.h' 'bn2_a_b.cpp' 'bn1_a_b.h' 'bn1_a_b.cpp' 'vgg_dense_3.h'

v++ -t sw_emu --config design.cfg -l -o'compute_network.xilinx_u200_xdma_201830_2.xclbin' compute_network.xilinx_u200_xdma_201830_2.xo

emconfigutil --platform xilinx_u200_xdma_201830_2
export XCL_EMULATION_MODE=sw_emu
./host compute_network.xilinx_u200_xdma_201830_2.xo

In vitis_emulation/, run the following for Hardware Emulation (Warning: 2hour+ run time):

v++ -t hw_emu --config design.cfg --save-temps -c -k compute_network -o'compute_network.xilinx_u200_xdma_201830_2.xo' 'tw_vgg10.cpp' 'tw_vgg10.h' 'pred_output.h' 'dense2.h' 'dense2.cpp' 'dense1.h' 'dense1.cpp' 'conv7.h' 'conv7.cpp' 'conv6.h' 'conv6.cpp' 'conv5.h' 'conv5.cpp' 'conv4.h' 'conv4.cpp' 'conv3.h' 'conv3.cpp' 'conv2.h' 'conv2.cpp' 'conv1.h' 'conv1.cpp' 'bnd2_a_b.h' 'bnd2_a_b.cpp' 'bnd1_a_b.h' 'bnd1_a_b.cpp' 'bn7_a_b.h' 'bn7_a_b.cpp' 'bn6_a_b.h' 'bn6_a_b.cpp' 'bn5_a_b.h' 'bn5_a_b.cpp' 'bn4_a_b.h' 'bn4_a_b.cpp' 'bn3_a_b.h' 'bn3_a_b.cpp' 'bn2_a_b.h' 'bn2_a_b.cpp' 'bn1_a_b.h' 'bn1_a_b.cpp' 'vgg_dense_3.h'

v++ -t hw_emu --config design.cfg -l -o'compute_network.xilinx_u200_xdma_201830_2.xclbin' compute_network.xilinx_u200_xdma_201830_2.xo

emconfigutil --platform xilinx_u200_xdma_201830_2

export XCL_EMULATION_MODE=hw_emu

./host compute_network.xilinx_u200_xdma_201830_2.xclbin

If we had the hardware, we could simply run:

unset XCL_EMULATION_MODE

./host compute_network.xilinx_u200_xdma_201830_2.xclbin

This file (xclbin) can be run on chip, in an SD card.

Summary (so far)

To get to this point we had to create and run vitis host files (host.cpp/host.h) and a compute_network kernel (tw_cpp.c) in vitis_emulation.

Then we ran in vitis on the **u200_xdma_201830_2** target device.

Once officially supported by Xilinx, this will easily be ported to the ZCU111 platform simply by downloading the appropriate platform and changing the target device.