■ Axonerve/AWS-F1

AWS-F1 ふたたびを参考に.

S3 バケットの用意

- S3 バケットを作成．名前は aws-f1-axonerve-kvs
- フォルダを作成．名前は axonerve-kvs-20190503

F1 インスタンスの用意

- c4.4xlarge で作成．$0.796/ 時間
- ストレージはルートを 100GB 程度あると安心．二つ目のストレージは不要.
- キーペアは既存のキーペアを選択 (以前つくった aws-f1-test-key)

F1 インスタンス起動後の設定

- aws configure を 忘れない
- /home/centos の下で git clone https://github.com/aws/aws-fpga.git $AWS_FPGA_REPO_DIR をする
- /home/centos の下で git clone https://github.com/miyo/axonerve_util.git を する
- AXONERVE_all.vp をアップロードする
  - scp -i pem ファイル AXONERVE_all.vp centos@AWS インスタンスの IP

作業ディレクトリに移動して環境設定

- cd $AWS_FPGA_REPO_DIR
- source sdaccel_setup.sh

SDx プロジェクトを作成

- cd ~/
- sdx -workspace build
- "Create a Xilinx(R) SDx(TM) Application project をクリック
- プロジェクト名を axonerve_kvs としてプロジェクト作成
- Platform は，aws-vu9p-f1... を選択
  - 候補がでないときは，Platform で Add Custom Platform... から，/home/centos/src/project_data/aws-fpga/SDAccel/aws_platform を追加
- Empty Application を選択

RTL カーネルの用意

- Kernel Wizard で雛形を作成
  - メニューの Xilinx から RTL Kernel Wizard を選択
  - General Settings
    - kernel name を axonerve_kvs_rtl，kernel vendor を wasalabo と設定して Next.
    - クロック数を 2，Has Reset を 1 に設定
  - Scalars
    - 数は 1 のまま．Argument name を data_num に変更して，Next.

    - Global Memory
        - そのまま Next
- ファイルの置換と追加
    - 取り除く : axonerve_kvs_rtl_example.sv，axonerve_kvs_rtl_example_vadd.sv, axonerve_kvs_rtl_ooc.xdc と axonerve_kvs_rtl_user.xdc
    - 追加 (1): /home/centos/axonerve_util/kvs/sdaccel/src/hdl の 下の axonerve_kvs_rtl_example.sv，axonerve_kvs_rtl_example_vadd.sv，user_logic.sv
    - 追加 (2): /home/centos/axonerve_util/kvs/hdl/sources の 下の axonerve_kvs_kernel.sv
    - 追加 (3): /home/centos/axonerve_util/kvs/sdaccel/src/xilinx-ip/aws-f1-vu9p の 下の xci ファイル．これはプロジェクトにコピー
    - 追加 (4): アップロードした Axonerve_all.vp
    - 追加 (5): /home/centos/axonerve_util/kvs/sdaccel/src/xdc/vu9p の 下の axonerve_kvs_rtl_ooc.xdc と axonerve_kvs_rtl_user.xdc
        - Soureces ペインの Libraries タブに切り替えると作業しやすい
        - - Design Sources → SystemVerilog → xil_defaultlib の axonerve_kvs_rtl_example.sv と axonerve_kvs_rtl_example_vadd.sv を取り除く
        - - Design Sources で右クリックして，コンテクステメニューから Add Sources を選択．"Add or create design sources" を選択して Next
        - - /home/centos/axonerve_util/kvs/sdaccel/src/hdl の 下の axonerve_kvs_rtl_example.sv, axonerve_kvs_rtl_example_vadd.sv，user_logic.sv を 追加
        - - /home/centos/axonerve_util/kvs/hdl/sources の 下の axonerve_kvs_kernel.sv を 追加
        - - /home/centos/axonerve_util/kvs/sdaccel/src/xilinx-ip/aws-f1-vu9p の 下の xci ファイルを追加 (これはプロジェクトにコピーする - Copy sources into project へのチェックを忘れない)
        - - アップロードした Axonerve_all.vp を 追加
        - - Constraints → constrs_1 の axonerve_kvs_rtl_ooc.xdc と axonerve_kvs_rtl_user.xdc を取り除く
        - - "Add or create constraints" を選択して Next
        - - /home/centos/axonerve_util/kvs/sdaccel/src/xdc/vu9p の 下の axonerve_kvs_rtl_ooc.xdc と axonerve_kvs_rtl_user.xdc を 追加
- Generate RTL Kernel を実行
    - source-only kernel を 選択

SDx でシステム全体のビルド

- ファイルの削除と追加
    - host_example.cpp を削除．axonerve_kvs.cpp，axonerve_kvs.hpp，host.cpp，xcl2.cpp, xcl2.hpp を追加
        - Project Explorer の src → sdx_rtl_kernel → axonerve_kvs_rtl の 下の host_example.cpp (2018.2 以前のバージョンなら main.c だった) は 削除
        - Project Explorer の トップ の axonerve_kvs で右クリック．コンテクストメニューから Import Sources... を選択
        - Browse... で /home/centos/axonerve_util/kvs/sdaccel/src を 選択
        - axonerve_kvs.cpp，axonerve_kvs.hpp，host.cpp，xcl2.cpp，xcl2.hpp を選択して Finish
- ターゲットを System に変更
    - 右ペイン，右上の Active build configuration で System を選択
- コンパイルオプションに --kernel_frequncy "0:60|1:120" を追加.
    - Project Explorer の トップ の axonerve_kvs で右クリック．コンテクストメニューから, C/C++ Build Settings を選択.
    - 左ペインの Settings をクリック
    - Configuration タブ を System にセット (この手順ならセットされているはず)
    - Tool Settings タブを開く
    - SDx XOCC Kernel Compiler → Miscellaneos で --kernel_frequency "0:60|1:120" を追加
    - SDx XOCC Kernel Linker → Miscellaneos で --kernel_frequency "0:60|1:120" を追加
    - Apply and Close で閉じる
- Hardware Functions(ハードウェア側の関数) として axonerve_kvs_rtl を設定

    - 右ペインの Hardware Functions の右にある Add Hardware Function... ボタン (稲妻みたいなアイコン) をクリック
    - axonerve_kvs_rtl を選択して OK
- メニューの Project → Build Project で ビルド
    - ツールバーのハンマーみたいなアイコンをクリックしてもいい

AFI イメージを作る

- cd /home/centos/build/axonerve_kvs/System で移動して binary_container_1.xclbin があるのを確認
- binary_container_1 フォルダが邪魔になるので mv binary_container_1 binary_container_1.o でリネーム
- AFI イメージ作る↓で binary_container_1.awsxclbin が できる

```
$SDACCEL_DIR/tools/create_sdaccel_afi.sh \
  -xclbin=<xclbin file name>.xclbin \
  -s3_bucket=<bucket-name> \
  -s3_dcp_key=<dcp-folder-name> \
  -s3_logs_key=<logs-folder-name>
```

    - 今回の例だと

```
$SDACCEL_DIR/tools/create_sdaccel_afi.sh \
  -xclbin=binary_container_1.xclbin \
  -s3_bucket=aws-f1-axonerve-kvs \
  -s3_dcp_key=axonerve-kvs-20190503 \
  -s3_logs_key=axonerve-kvs-20190503
```

- *_afi_id.txt を開いて FpgaImageId を確認
- AFI イメージの作成をまつ (↓コマンドを実行して State が available になるのを待つ)

```
aws ec2 describe-fpga-images --fpga-image-ids 確認した FpgaImageId
```

- axonerve_kvs.exe と binary_container_1.awsxclbin を手元にコピーする
- オプション : ビルドディレクトリ (/home/centos/build 以下) をダウンロード
    - たとえば, ↓とか

```
ssh -i pem ファイル centos@ リモート IP tar zcpvf - build | tar zxpvf -
```

AWS-F1 で実行

- AWS-F1 インスタンスを作成
    - AMI で FPGA を検索. 今度は f1.2xlarge で作成.
    - 起動したら aws configure を 実行
    - /home/centos の下で git clone https://github.com/aws/aws-fpga.git $AWS_FPGA_REPO_DIR
    - cd $AWS_FPGA_REPO_DIR; source sdaccel_setup.sh
- axonerve_kvs.exe と binary_container_1.awsxclbin をアップロードする
- 実行する
    - sudo -s
    - source $AWS_FPGA_REPO_DIR/sdaccel_runtime_setup.sh
    - ./axonerve_kvs.exe binary_container_1.awsxclbin