■気づいたら

7 月．2019 年も後半戦.

■ Intel PAC で Signal Tap 使う

PAC で Siganl Tap 使う練習.
Accelerator Functional Unit (AFU) Developer ’s Guide for Intel(R) Programmable Acceleration Card with Intel(R) Arria(R) 10 GX FPGA を参考に，サンプル nlb_mode_0_stp を動かしてみる.
まずは,

```
% cd $OPAE_PLATFORM_ROOT/hw/samples/nlb_mode_0_stp/
```

に移動.

```
% afu_synth_setup --source hw/rtl/filelist_mode_0_stp.txt build_synth
% cd build_synth
% rtl_src_config --qsf --rel build ../hw/rtl/filelist_mode_0_stp.txt >hw/afu.qsf # 変更してなければ不要な気が
% ${OPAE_PLATFORM_ROOT}/bin/run.sh
```

としてビルド.
ここで，filelist_mode_0_stp.txt をみてみると,

```
+define+INCLUDE_REMOTE_STP
C:filelist_mode_0.txt
QI:../par/${OPAE_PLATFORM_FPGA_FAMILY}/extra_tcl-0_stp.tcl
../par/nlb_0_stp.sdc
```

と，INCLUDE_REMOTE_STP のバリアントがついていることが確認できる.
参照されている，hw/rtl/filelist_mode_0.txt の中身は,

```
+define+NLB400_MODE_0
C:filelist_base.txt
```

で，さらに，hw/rtl/filelist_base.txt の中身は,

```
+define+BIST_AFU
nlb_400.json
test_sw1.sv
test_rdwr.sv
test_lpbk1.sv
requestor.sv
nlb_lpbk.sv
nlb_csr.sv
nlb_C1Tx_fifo.sv
ccip_std_afu.sv
ccip_interface_reg.sv
ccip_debug.sv
arbiter.sv
nlb_gram_sdp.v
pipeline.sv
platform/${OPAE_PLATFORM_FPGA_FAMILY}/local_mem.sv
resync.v
altera_std_synchronizer_nocut.v

QSYS_IPs/${OPAE_PLATFORM_FPGA_FAMILY}/RAM/req_C1TxRAM2PORT.qsys
QSYS_IPs/${OPAE_PLATFORM_FPGA_FAMILY}/RAM/lpbk1_RdRspRAM2PORT.qsys

include_files/common
../par/stp_basic.stp
../par/nlb_0_stp.sdc
```

となっていて，これが実体っぽい．
また，filelist_mode_0_stp.txt に書いてある par/${OPAE_PLATFORM_FPGA_FAMILY}/extra_tcl-0_stp.tcl には，

```
set_global_assignment -name ENABLE_SIGNALTAP ON
set_global_assignment -name USE_SIGNALTAP_FILE ../hw/par/A10/stp_basic.stp
```

と書いてあった．このファイルは，build_synth/hw/afu.qsf の中で，

```
source ../../hw/par/A10/extra_tcl-0_stp.tcl
```

と参照される．
推測だけど，afu_synth_setup の --source に指定するファイルでは，

```
C: foo
```

って書くと foo が，afu_synth_setup 時に展開されて，

```
QI: bar
```

って書くと bar が，build_synth/hw/afu.qsf で参照されるて Quartus で使われる，の，かな．

ビルドがおわったら nlb_400.gbs っていうのができたので，

```
% sudo fpgaconf nlb_400.gbs
```

でコンフィグレーション．

ソフトウェアは，$OPAE_PLATFORM_ROOT/hw/samples/nlb_mode_0_stp/sw/README にも書いてあるように
$OPAE_PLATFORM_ROOT/sw/opae-1.1.2-1/samples/ の下．

```
% gcc -std=c99 hello_fpga.c -lopae-c -luuid
```

とかして，コンパイル．

リモートデバッグには，

- $OPAE_PLATFORM_ROOT/hw/remote_debug/mmlink_setup_profiled.tcl
- $OPAE_PLATFORM_ROOT/hw/remote_debug/remote_debug.sof

を使う．これはインストールされたものを使う．
一つターミナルを開いて

```
% sudo mmlink -P 3333
```

を実行.
別のターミナルで

```
%export PATH= どこか /intelFPGA_pro/quartus/sopc_builder/bin:$PATH
```

(自分の環境だと)

```
% export PATH=/home/miyoshi/data/inteldevstack/intelFPGA_pro/quartus/sopc_builder/bin:$PATH
```

として,

```
% cd $OPAE_PLATFORM_ROOT/hw/remote_debug
% system-console --rc-script=mmlink_setup_profiled.tcl remote_debug.sof localhost 3333
```

とか.

Quartus でメニューの File で stp を指定すると Signal Tap が開く.
JTAG Chain configuration に,
"system-console on localhost"っていうのがみえてるのでそれを指定.
とりあえず，JTAG Ready になった.

$OPAE_PLATFORM_ROOT/sw/opae-1.1.2-1/samples/ の下でコンパイルした a.out を

```
./a.out -s
```

として実行 (FPGA_OPEN_SHARED をつけて fpgaOpen を呼ぶ) するとプログラムは実行できた.

... けど，Ready to acquire にならないな

インストール時に展開されたであろう,

- ./hw/par/A10/stp_basic.stp
- ./bin/nlb_mode_0_stp.gbs

なら Ready to acquire になるなあ.