■ GateMate

Cologne Chip の GateMate FPGA という FPGA を知ったので評価ボード遊んでみた.
ツールチェインの大部分にオープンソースソフトウェアが使われている，というのが面白い.
ツールチェインの詳細は，Technology Brief of GateMate FPGA Technology にある.
とりあえずは pre-build 版を使ってみる．ダウンロードするツール以外に,

```
sudo apt install iverilog gtkwave
sudo apt install libhidapi-hidraw0
```

などとして，必要なツールとライブラリをインストールしておく.

サンプルの合成と P&R

ダウンロードしたら，展開して,

```
cd cc-toolchain-linux/workspace/blink
```

でサンプルディレクトリに移動する.

```
make synth_vlog
```

で Verilog ファイルの合成ができる．synth_vlog の定義は，../config.mk にあって

```
$(YOSYS) -qql log/synth.log -p 'read -sv $^; synth_gatemate -top $(TOP) \
-nomx8 -vlog net/$(TOP)_synth.v'
```

となっている．TOP には blink が定義してあって，実際に実行されるコマンドは

```
../../bin/yosys/yosys -qql log/synth.log -p 'read -sv src/blink.v; \
synth_gatemate -top blink -nomx8 -vlog net/blink_synth.v'
```

これで，net/blink_synth.v が生成される.

```
make synth_vhdl とすると,
```

VHDL な src/blink.vhd をソースとして

```
../../bin/yosys/yosys -ql log/synth.log \
-p 'ghdl --warn-no-binding -C --ieee=synopsys src/blink.vhd -e blink; \
synth_gatemate -top blink -nomx8 -vlog net/blink_synth.v'
```

と，同様に net/blink_synth.v が生成される.
Verilog から生成した net/blink_synth.v と VHDL から生成したものを比べると,
微妙に wire が違う程度で，同じような構造が生成されるのが面白い.
シミュレーションは，生成した net/blink_synth.v をターゲットに実行される.

```
make synth_sim
```

とすると，iverilog が使われてシミュレーションされる．
P&R は，

```
make impl
```

とする．コマンドは，

```
../../bin/p_r/p_r -i net/blink_synth.v -o blink -ccf src/blink.ccf > log/impl.log
```

だった．./blink_00.cfg.bit ができあがった．

実機で動かしてみる

FPGA の書き込みは

```
make jtag
```

らしい ... が，手元の Ubuntu 20.04.6 では，

```
../../bin/openFPGALoader/openFPGALoader  -b gatemate_evb_jtag blink_00.cfg
  ../../bin/openFPGALoader/openFPGALoader:     /lib/x86_64-linux-gnu/libstdc++.so.6 :     version
`GLIBCXX_3.4.29' not found (required by ../../bin/openFPGALoader/openFPGALoader)
 ../../bin/openFPGALoader/openFPGALoader: /lib/x86_64-linux-gnu/libc.so.6: version `GLIBC_2.32' not
found (required by ../../bin/openFPGALoader/openFPGALoader)
 ../../bin/openFPGALoader/openFPGALoader: /lib/x86_64-linux-gnu/libc.so.6: version `GLIBC_2.33' not
found (required by ../../bin/openFPGALoader/openFPGALoader)
 ../../bin/openFPGALoader/openFPGALoader: /lib/x86_64-linux-gnu/libc.so.6: version `GLIBC_2.34' not
found (required by ../../bin/openFPGALoader/openFPGALoader)
make: *** [../config.mk:39: jtag] Error 1
```

と言われてしまって動かなかった．

OpenFPGALoader のビルドと再挑戦

というわけで，OpenFPGALoader はビルドすることにした．

```
sudo apt install libftdi1-2 libftdi1-dev libhidapi-hidraw0 libhidapi-dev \
libudev-dev zlib1g-dev cmake pkg-config make g++
pushd /tmp
git clone https://github.com/trabucayre/openFPGALoader.git
cd openFPGALoader
mkdir build
cd build
cmake ..
make -j
sudo make install
popd
```

で，../config.mk の OFL のパスを，/usr/local/bin/OpenFPGALoader に書き換えて

```
make jtag
```

FPGA ボードの mini USB コネクタとホスト PC を接続しておくと
bit ファイルが FPGA に書き込まれて，ボード上の LED D1 の点滅が確認できた．

ちょっと変更してみる

ちょっと変更，というわけで点滅させる LED を増やしてみる．
src/blink.v の

```
                output wire led
```

を

```
                output wire [7:0] led
```

に，

```
        assign led = counter[26];
```

を

```
        assign led = counter[26:19];
```

に変更．
出力 LED を増やした分，ピン配置も変更する必要がある．src/blink.ccf の

```
    Pin_out   "led"   Loc = "IO_EB_B1"; # LED D1
```

を

```
    Pin_out   "led[0]"   Loc = "IO_EB_B1"; # LED D1
    Pin_out   "led[1]"   Loc = "IO_EB_B2"; # LED D2
    Pin_out   "led[2]"   Loc = "IO_EB_B3"; # LED D3
    Pin_out   "led[3]"   Loc = "IO_EB_B4"; # LED D4
    Pin_out   "led[4]"   Loc = "IO_EB_B5"; # LED D5
    Pin_out   "led[5]"   Loc = "IO_EB_B6"; # LED D6
    Pin_out   "led[6]"   Loc = "IO_EB_B7"; # LED D7
    Pin_out   "led[7]"   Loc = "IO_EB_B8"; # LED D8
```

に変更．
で，合成と P&R をやりなおして，FPGA に書き込む．

```
    make synth_vlog
    make impl
    make jtag
```

これで，D8 から D1 にカウンタを出力することができた．
評価ボードの CCF は，GateMateTM FPGA Evaluation Board の Evaluation Board Master CCF constraints file からダウンロードできる．
複数ビットな信号とピン配置の設定がどうなってるかわかりさえすれば，あとはなんとかなるかな．
ダウンロードできる ccf だとクロックまたぎの設定はよくわからないな．