FPGA人工知能のポテンシャルを探る

第8回　量子化ニューラル・ネットワークをFPGAで動かしてみる

鈴木 量三朗

リスト1　QNNを回路化してみる（最適化なし）

```
#include <cassert>
#include "yolo_l0.h"

int
yolo_l0(pixel_t in_img[INPUT_HEIGHT][INPUT_WIDTH]
                                    [INPUT_CHANNEL_N],
        pixel_t weight[OUTPUT_CHANNEL_N]
              [INPUT_CHANNEL_N][KERNEL_N][KERNEL_N],
        pixel_t bias[OUTPUT_CHANNEL_N],
        pixel_t out_img[OUTPUT_HEIGHT]
                  [OUTPUT_WIDTH][OUTPUT_CHANNEL_N])
{
#pragma HLS INTERFACE m_axi port=in_img offset=direct
#pragma HLS INTERFACE m_axi port=weight offset=direct
#pragma HLS INTERFACE m_axi port=bias offset=direct
#pragma HLS INTERFACE m_axi port=out_img offset=direct
#pragma HLS INTERFACE ap_ctrl_chain port=return

  for( unsigned int out_ch_i = 0 ; out_ch_i <
                    OUTPUT_CHANNEL_N; out_ch_i++ ) {
    for( unsigned int y = 1; y < INPUT_HEIGHT
                                        - 1; y++ ) {
      for( unsigned int x = 1; x < INPUT_WIDTH
                                        - 1; x++ ) {
        pixel_t p = 0;
        for( unsigned int in_ch_i = 0 ;
          in_ch_i < INPUT_CHANNEL_N ; in_ch_i++ ) {
          for(unsigned int k_y = 0 ; k_y
                                < KERNEL_N; k_y++ ) {
            for( unsigned int k_x = 0 ;
                          k_x < KERNEL_N; k_x++ ) {
              p += in_img[y - 1 + k_y]
                         [x - 1 + k_x][in_ch_i]
                          * weight[out_ch_i]
                           [in_ch_i][k_y][k_x];
            }
          }
        }
        p += bias[out_ch_i];
        out_img[y][x][out_ch_i] = p;
      }
    }
  }
  return 0;
}
```

ザイリンクスの量子化ニューラル・ネットワーク（QNN[注1]：Quantized Neural Network）を取り上げています．参照している実装（QNN）ではPythonを使って浮動小数点で畳み込みの演算をしています．

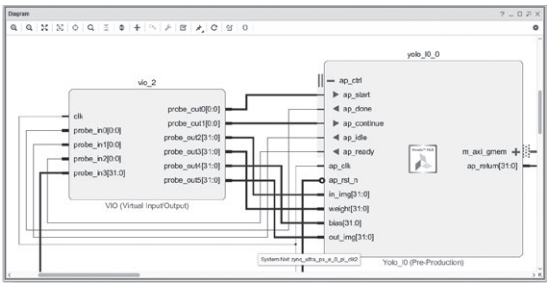

図1　合成された回路ブロックをVivadoで読み込む

FPGA化のための第一歩として，今回はFPGAで動作可能（合成可能）なCのプログラムを書いていきます．

FPGA用の回路をC言語で記述する

● ステップ1…最適化せずに回路化してみる

最初に，簡単に何の並列化もしないCのプログラムを書いてみましょう．

FPGA向け高位合成ツールVivado HLSで合成可能なコードの例を**リスト1**に示します．これをコンパイルし，FPGA開発ツールのVivadoへ組み込みます．

何の最適化もしておらず，また，FPGAが苦手な浮動小数点を使っているため（ここでは`pixel_t`は`float`）残念ながら処理速度はソフトウェアより遅くなります．あくまでも，FPGA上で動作可能であるという検証用のための記述です．

実際にコンパイル（動作合成）してVivadoに組み込めました（**図1**）．インターフェースはVIO（Virtual Input/Output）[注2]を使っています．これによりソフトウェアの介在なしにテストができます．

● ステップ2…Darknetを試す

Darknet[(3)]はCとCUDAで書かれたニューラル・ネットワークのオープンソースのフレームワークで

注1：QNNと言えば量子ニューラル・ネットワーク（Quantum Neural Network）を指すことが多い．ザイリンクスのQNNは意味が異なるので注意．

注2：Vivadoで，FPGAの内部信号をリアルタイムにモニタ／駆動するための回路ブロック．

第2回　FPGA向けニューラル・ネットワークBNNの動作をまずはPCで確かめる（2018年7月号）
第3回　FPGA向けニューラル・ネットワークBNNのハードウェア化（2018年8月号）
第4回　Zynqボードでカメラを使えるようにする（2018年9月号）