

- [BNN_on_fpga](#)
 - [项目概述](#)
 - [网络结构](#)
 - [硬件架构](#)
 - [卷积模块](#)
 - [最大池化层](#)
 - [全连接模块](#)
 - [top模块](#)
 - [项目总结](#)

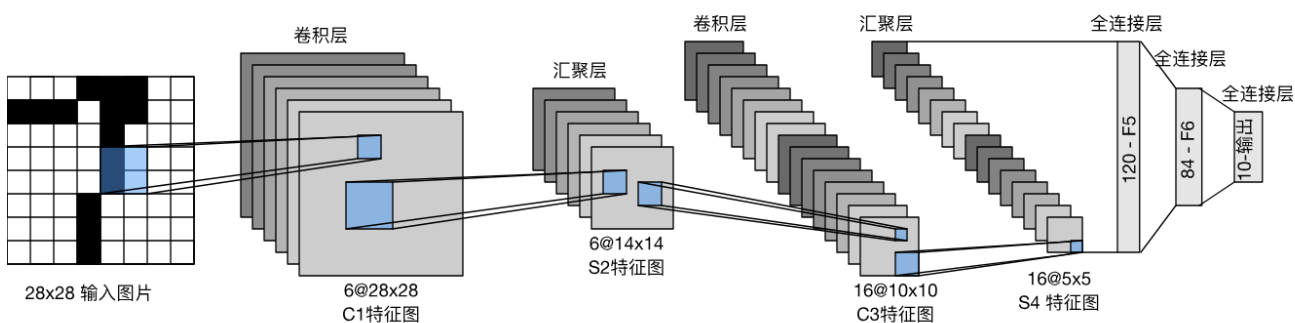
BNN_on_fpga

项目概述

这个项目是一个基于 FPGA 的二值神经网络（Binary Neural Network, BNN）加速器，旨在加速 MNIST 数据集的手写数字识别任务，通过训练不同数据集，也可以用于其他任务，如fashionMNIST, cifar10（需要改通道数，重新调整时序）等。

二值化神经网络（BNN）是指仅使用+1和-1两个值来表示权重的神经网络，相比于全精度的神经网络，它可以使用XOR+popcount这样的简单组合代替fp32的乘加来实现卷积操作，从而大大节省内存，减少计算开销。

网络结构



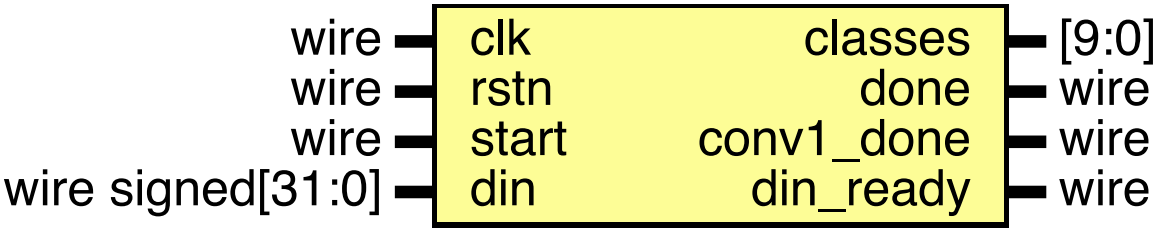
上图为经典的LeNet5网络，本项目为了减小复杂度，将三个全连接减少为一个，实际测试中并没有准确率。 lenet 2conv3fc

1bit量化，权重只为1或者-1

LeNet(

```
(conv1): BinaryConv2d(1, 6, kernel_size=(5, 5), stride=(1, 1), bias=False)
(pool1): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
(rel1): ReLU()
(conv2): BinaryConv2d(6, 12, kernel_size=(5, 5), stride=(1, 1), bias=False)
(pool2): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
(rel2): ReLU()
(fc1): BinaryLinear(in_features=192, out_features=10, bias=False)
)
```

硬件架构



conv1的输入是 (1, 28, 28)，输出通道为6，所以直接将并行度设置为6，第二个卷积层的输入通道为6，输出通道为12，所以一共是13次卷积操作。

卷积模块

首先是卷积的滑窗模块，由于卷积核尺寸是55，所以滑窗模块预存的是285=140个像素，需要支持两种模式，第一种模式的输入是2828，第二种是在经过第一个卷积层和第二个池化层过后，输入变成了1212，所以预存的像素点变成了12*5=60个，由于对于每五行，从上到下，从左到右读入mem中，所以第一个读入的点应该位于mem的最高位。

接下来是对该模块进行仿真，需要将图片转化为txt文件，每一行一个像素，用八位二进制无符号数表示。

接下来是卷积模块的核心部分，卷积相乘的部分，由于所有的权重都已经量化为了1bit，所以我们人为规定，1代表权重为1，0代表权重为-1，转化为硬件电路里面，便是为1时

加上这个激活值，为0时减掉这个激活值。激活值与权重对应相乘结束过后，进行流水线操作进行结果相加，一共是7级流水线。

执行流程如下

- 加载权重矩阵，一共是150个周期，因为并行度是6，所以是 $6 \times 5 = 150$
- 在滑窗模块加载完过后，还需要4个周期，流水线的第一次运行还需要7个周期，一共就是161个周期。
- 在卷积计算完成的时候，因为输出是2424，所以在这里需要移位2424次，同时滑窗模块换行的时候，需要进行同步操作，一共换行23次，所以是 23×4 次，加起来就是829

仿真部分，硬件模拟软件，首先用pytorch生成一个卷积，权重全为-1，

```
conv = nn.Conv2d(1, 1, 5, 1, 0, bias=False)
conv.weight.data.fill_(-1)
output = conv(image)
```

卷积完的结果如下

```
torch.Size([1, 24, 24])
tensor([[[[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
           0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
           0, 0, 0, 0],
         [ 0, 0, 0, 0, 0, 0, 0, 0, -3, -21,
          -39, -57, -183, -316, -473, -481, -629, -758, -869, -821,
          -795, -629, -374, -127],
         [ 0, 0, 0, 0, -30, -66, -160, -314, -487, -728,
          -963, -1140, -1365, -1581, -1710, -1637, -1785, -1903, -1956, -1747,
          -1549, -1130, -633, -191],
         [ 0, 0, 0, -49, -317, -606, -953, -1360, -1737, -1993,
          -2228, -2405, -2630, -2844, -2813, -2569, -2546, -2467, -2308, -2006,
          -1726, -1225, -672, -191],
         [ 0, 0, 0, -67, -554, -1096, -1696, -2356, -2968, -3258,
          -3438, -3544, -3763, -3965, -3681, -3239, -3034, -2708, -2308, -2006,
          -1726, -1225, -672, -191],
         [ 0, 0, 0, -67, -634, -1332, -2039, -2952, -3817, -4232,
          -4267, -4266, -4275, -4378, -3889, -3436, -3231, -2862, -2308, -2006,
          -1726, -1225, -672, -191],
         [ 0, 0, 0, -67, -634, -1346, -2054, -3121, -4236, -4723,
          -4726, -4706, -4435, -4152, -3416, -2955, -2602, -2104, -1439, -1185,
          -931, -596, -298, -641]]]])
```

同时用硬件进行仿真，结果打印在控制台上面，如下图所示

可以看到，结果是一样的，说明卷积模块的功能是没有问题的。

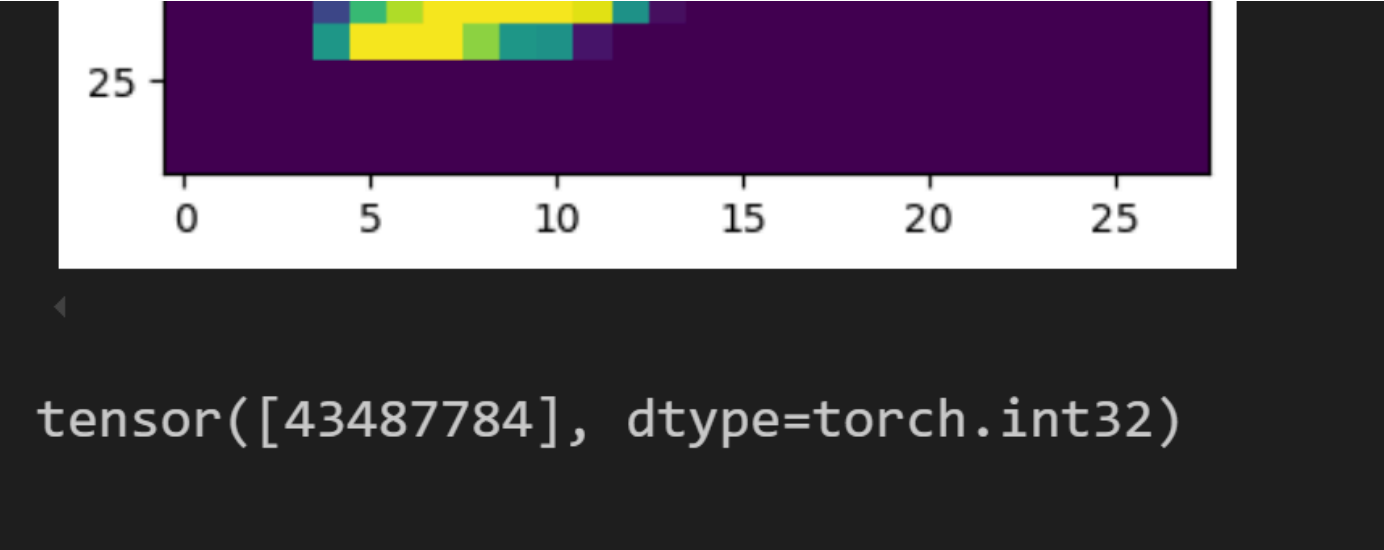
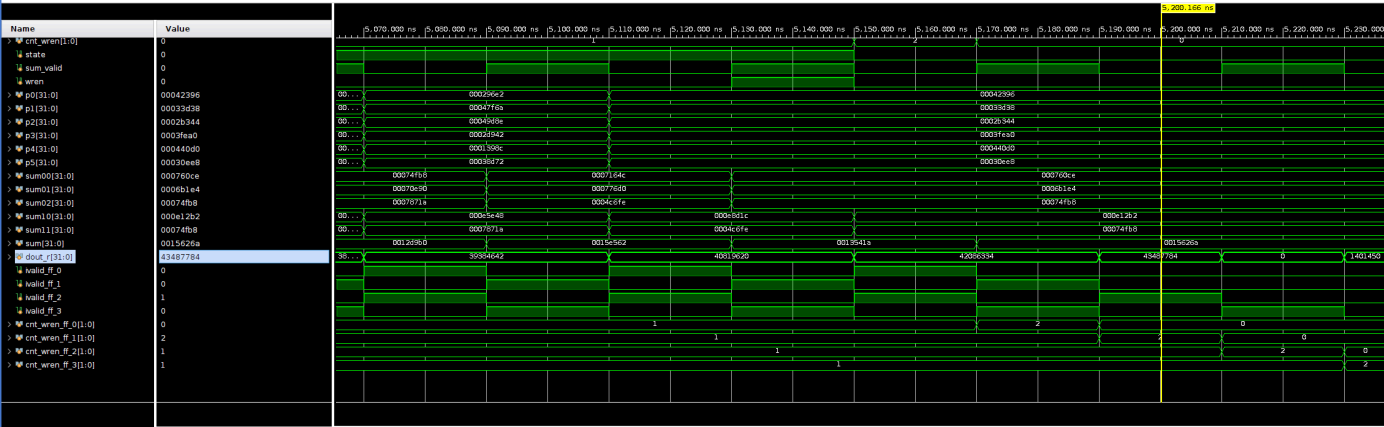
最大池化层

仿真方法与卷积模块结果一样

```
INFO: [Wavedata 42-604] Simulation restarted
run 30 us
picture read over
maxpooling complete
 0 :      0      0      0      0      0      0      0      0      0      0      0
 1 :      0      0     -30     -160     -487     -963     -1365     -1637     -1785     -1747     -1130     -191
 2 :      0      0     -554     -1696     -2968     -3438     -3763     -3239     -2708     -2006     -1225     -191
 3 :      0      0     -604     -1894     -4144     -4207     -3079     -1799     -959     -259      -95      0
 4 :      0      0     -80     -358     -1899     -2936     -1753     -466     -155      0      0      0
 5 :      0      0      0      0     -628     -1975     -2286     -919     -145      0      0      0
 6 :      0      0      0      0      0     -35     -867     -2189     -2589     -1267     -214      0
 7 :      0      0      0      0      0      0     -412     -1779     -3758     -2796     -736      0
 8 :      0      0      0      0     -24     -398     -1327     -2623     -4034     -2496     -704      0
 9 :      0     -18     -431     -1240     -2437     -3586     -3431     -2266     -966     -182      0
10 :    -191     -1113     -2497     -3439     -3422     -2404     -1082     -2290     -2      0      0
11 :    -191     -1095     -2066     -2004     -1177     -404     -11      0      0      0      0
12 :$finish called at time : 11560 ns : File "/home/curry/code/curry_code_summay/rtl_works/BNN_on_fpga/rtl/maxpool_tb.v" Line 109
```

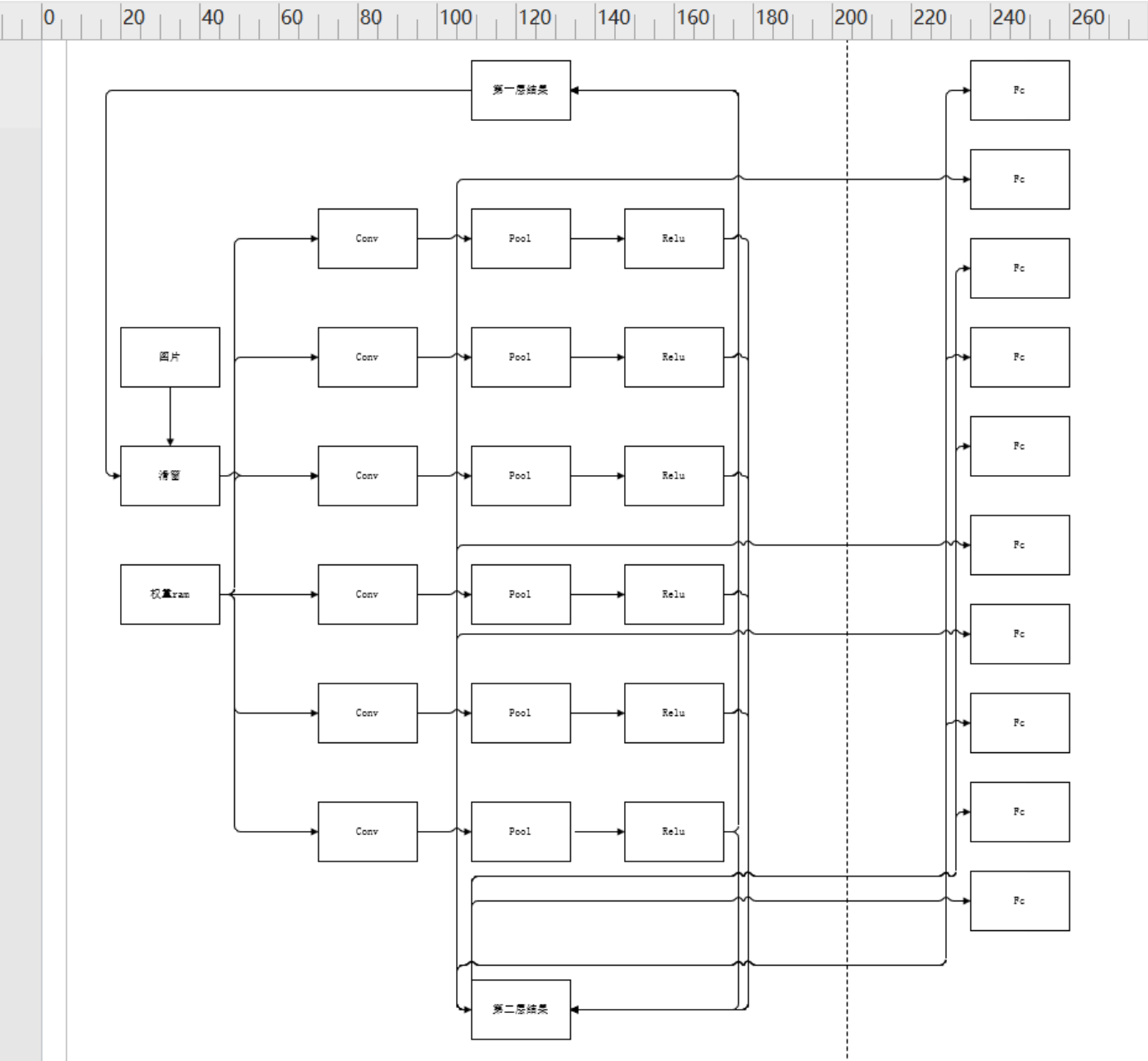
全连接模块

由于网络第二层一共有12个输出通道，feature map大小是16，所以全连接层一共接收12个输入，流水线级接受16次，最终4级累加到最后结果。



top模块

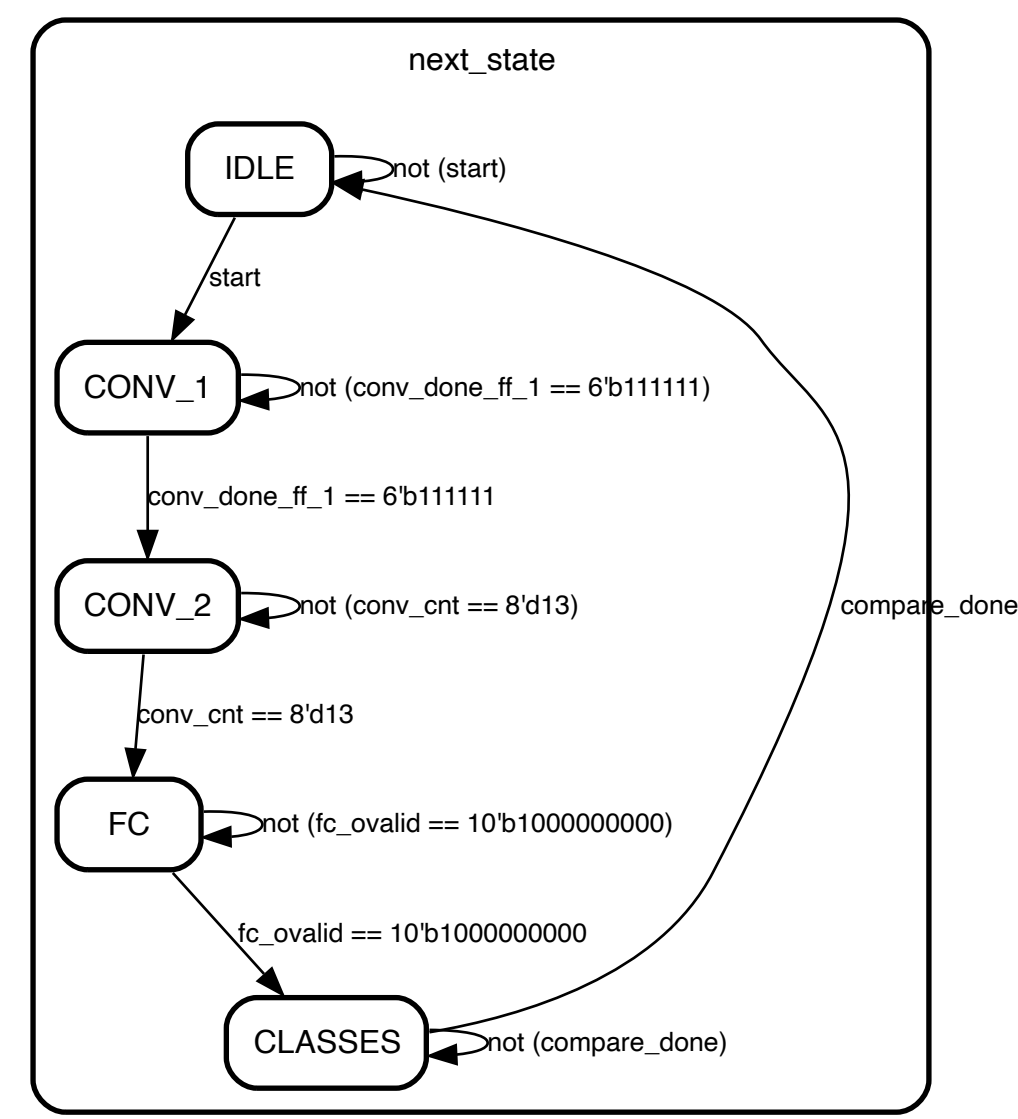
最初版本的top将所有模块直接例化，如下图所示。



在实际过程中调度过于复杂，波形信号有几百条，最终决定做一个化繁为简的操作，将卷积模块，池化模块，ReLU模块进行合并，通过再多添加流水线实现卷积后的池化和激活操作。

第二版例化六个融合卷积模块，10个全连接，最后再通过五级流水线输出最大值以及对应类别。

运算过程的状态机如图所示。



项目总结

这个项目verilog代码一共有7000多行，已经是一个很大的工程了，在