東南大學

毕业设计(论文)报告

题 目： 基于 BNN 的 ECG 信号检测神经网络加速器的设计

学　　号： 06219105

姓　　名： 蒲宁昊

学　　院： 电子科学与工程学院

专　　业： 电子科学与技术

指导教师： 刘昊

起止日期： 2022.12.1 至 2023.6.10

东南大学毕业（设计）论文独创性声明

本人声明所呈交的毕业（设计）论文是我个人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得东南大学或其它教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

论文作者签名：_____ 日期：_____年_____月_____日

东南大学毕业（设计）论文使用授权声明

东南大学有权保留本人所送交毕业（设计）论文的复印件和电子文档，可以采用影印、缩印或其他复制手段保存论文。本人电子文档的内容和纸质论文的内容相一致。除在保密期内的保密论文外，允许论文被查阅和借阅，可以公布（包括刊登）论文的全部或部分内容。论文的公布（包括刊登）授权东南大学教务处办理。

论文作者签名：_____ 导师签名：_____

日期：_____年_____月_____日 日期：_____年_____月_____日

摘　　要

有效快速地检测生理信号以进行心律失常识别对于人类的身体健康有重大意义。目前对于生理信号的检测涌现了大量的先进算法，这些算法虽然有着较高的准确率，但随之而来的是复杂的模型与动辄百万级别的模型权重，不可避免地提升了在可穿戴设备等硬件上实现的难度、功耗与资源占用。

基于前述问题，本课题采取二值化算法，针对心律失常提出了一种硬件友好型的二值化卷积分类网络，通过将普通卷积神经网络的权重与激活值量化为 0 与 1，将模型的大小极限压缩，采用 MIT-BIH 数据库对模型进行训练与验证。所设计的 BP（Better Performance）模型在 5 分类任务的准确率达到了 96.90%，在 17 分类任务的准确率达到了 97.50%，与近年的轻量化检测算法相比具有一定优势。

本课题选取设计模型中的 LP（Low Power）模型，为其设计专属的硬件加速器，并进行了功能仿真与指标评估，最终加速器的存储占用仅为 3.92KB，达到了 28.48 倍的存储空间压缩率，50MHz 的条件下，执行一次识别需 89.150μs，功耗为 0.708W，吞吐率为 133.19GOPS，能效为 188.12GOPS/W。将 LP 模型从软件移植到硬件没有精度损失，加速器准确率为 91.60%，满足了预期的设计指标，为利用二值化卷积分类网络对生理信号分类提供了一种更为精巧高效的解决方案。

关键词：心律失常，二值化卷积分类网络，硬件友好，硬件加速器

ABSTRACT

The effective and rapid detection of physiological signals for identifying cardiac arrhythmias holds significant importance for human physical well-being. Currently, numerous advanced algorithms have emerged for the detection of physiological signals. While these algorithms boast high accuracy, they come with complex models and weight parameters reaching the scale of millions, inevitably increasing the difficulty, power consumption, and resource utilization in hardware implementations such as wearable devices.

Addressing the aforementioned challenges, this study adopts a binarization algorithm and proposes a hardware-friendly binary convolutional classification network specifically tailored for cardiac arrhythmia. By quantizing the weights and activations of a conventional convolutional neural network to 0 and 1, respectively, the model's size is drastically compressed. The MIT-BIH database is utilized for training and validating the model. The designed Better Performance (BP) model achieves an accuracy of 96.90% in the 5-class classification task and 97.50% in the 17-class classification task, exhibiting certain advantages compared to recent lightweight detection algorithms.

Within the scope of this study, the Low Power (LP) model is selected for design, and a dedicated hardware accelerator is developed. Functional simulation and performance evaluation are conducted, resulting in a storage occupancy of merely 3.92KB for the accelerator. This corresponds to a compression ratio of 28.48 times in storage space. Under conditions of 50MHz, a single recognition execution requires 89.150μs, with a power consumption of 0.708W. The throughput reaches 133.19GOPS, and the energy efficiency amounts to 188.12GOPS/W. The migration of the LP model from software to hardware is achieved without any loss in accuracy, as the accelerator achieves an accuracy of 91.60%, meeting the expected design targets. This provides a more refined and efficient solution for utilizing binarized convolutional classification networks in the classification of physiological signals.

KEY WORDS: Arrhythmias, Hardware-friendly, Binary convolutional classification network, Hardware accelerator

目　　录

第一章 绪论

1.1 课题背景和意义

心血管疾病（CVDS）是全球的头号死因，每年死于心血管疾病的人数多于任何其他原因死亡的人数[1]。心律失常是一种非常常见的 CVD 类型，被认为是每年发生的大多数心源性猝死的主要原因[2]。自 2019 年 COVID-19 病毒的全球性爆发以来，病毒的传播与变异从未停止，心律失常是 COVID-19 患者心血管疾病的常见表现，在中国住院患者的临床病例系列研究中，在 44.4%的重症监护病房患者中，心悸是 7.3%的初始首发症状，16.7%的病例报告了心律失常[3]，而心电图目前在心脏病的临床诊断中起着不可替代的作用，可以用于帮助医护人员发现心律失常的具体类型[4]。

由于心电图（Electrocardiogram, ECG）信号的复杂性，大量的研究表明人工对 ECG 信号的解释充满困难与不确定性，不同心脏病专家对同一 ECG 信号的解释也有着较大差异，甚至同一心脏病专家在不同场合对同一 ECG 信号的解释也会有很大不同[5]。随着深度学习的发展，设计卷积神经网络来完成心律检测分类任务拥有更好的精度表现[6][7]。其中最先进的卷积神经网络分类精度已超过人类专家水平[8]。

卷积神经网络是计算密集型和访存密集型任务，其中包含大量重复简单的并行运算，这些运算不适合在 CPU 上执行。虽然 GPU 具有高度并行性，非常适合神经网络计算，但是其不适合部署在嵌入式平台上[9]。由于神经网络的算法更新迅速，而 ASIC 的设计流程过长，导致 ASIC 难以利用最新的神经网络成果。相比之下，FPGA 可编程的灵活性使其可以定制化地针对用户进行设计。因此，在嵌入式系统中使用 FPGA 平台来部署深度学习神经网络是非常合适的选择。

但是 FPGA 平台的资源与带宽受限，实现较为大型的神经网络于其上较为困难。因此如何减少神经网络的参数量和计算量值得我们研究。目前为了为减少神经网络的计算量和参数量，主流方法是设计新的轻量化网络结构和神经网络压缩。设计新的轻量化网络结构构建了新的卷积算法，如分层卷积等，它们和其他相关算法被应用于 MobileNet[9]，ShuffleNet[10]等，降低了卷积层的参数量和计算量；神经网络压缩涉及到剪枝、量化和编码三种算法[11]。其中，剪枝通过删除网络中的权重或卷积核来稀疏化网络，而编码则通过聚类方法将网络参数简化为有限的几类，这两种方法都会在网络中引入不规则性，对硬件不够友好。相比之下，量化可以通过压缩网络参数的位宽来直接有效地降低网络的参数量和计算量，便于在硬件上实现。典型的量化方法是将预训练网络的浮点权重定点化，映射

为整型权重，称之为训练后动态量化。训练后静态量化则是在动态量化的基础上进一步对激活值进行量化。这两种方法都是对已训练完成的网络进行量化，无法在训练中学习弥补量化带来的损失，相对于原网络，则可能会导致精度下降。因此在保证精度的前提下，通常只能将数据量化到 8bit，难以进一步压缩网络位宽。近年来，越来越多的网络采用训练中量化技术将网络压缩到更低的位宽。增量量化[12]采用分批量化的方法，通过更新未量化的权重来弥补精度。它将网络的权重映射到非线性的幂指数，可以将权重量化到 4 位而只造成很少的精度损失。量化感知训练则更进一步，同时维护浮点参数和量化参数，在训练中通过更新浮点参数来更新量化参数，从而实现在训练过程中进行量化。由于能够更新量化权值来弥补精度，量化感知训练能够将网络压缩到极低的位宽，例如三值化[13]和二值化[14]网络。其中，二值化神经网络（Binary Neural Network, BNN）能够最大程度地压缩网络的位宽，非常适合在资源有限的 FPGA 平台上进行部署。

综上，基于二值化的生理信号检测算法实现于嵌入式设备上有多重意义：

便携性：嵌入式设备通常较小，可轻松携带。因此，将生理信号检测算法实现于嵌入式设备上，可以让用户随时随地检测他们的健康状况，而不必频繁地去医院或诊所。

实时性：嵌入式设备通常具有较高的计算能力，可以实时处理数据。因此，将生理信号检测算法实现于嵌入式设备上，可以使用户立即了解他们的生理状况，及时采取行动。

节省成本：嵌入式设备通常价格相对较低，因此将生理信号检测算法实现于嵌入式设备上，可以降低检测成本。此外，由于用户可以在家中进行检测，可以避免医疗机构的开销。

保护隐私：将生理信号检测算法实现于嵌入式设备上，可以避免将数据发送到云端或其他服务商进行处理，从而更好地保护用户的隐私。

1.2 研究现状

1.2.1 传统 ECG 检测算法

模式识别被广泛用于心律失常的检测[15][16][17][18][19]，通过前期的信号预处理，如滤波[20]、小波变换[21]等方法，再通过特征提取[22]后送入分类器进行分类，这种方法往往可以取得令人满意的精度，但是特征提取算法中的滤波、小波变换等实现于可穿戴设备需要额外的资源，且有关特征提取的算法普遍会带来较大的计算量与延迟，使得模型的实时性差[18]。算法性能高度依赖特征也较大程度降低了模型的泛化能力[23]。

随着深度学习的兴起，应用神经网络对心律失常进行分类的研究逐渐增多。神经网络的特点是将特征提取与分类融合，可直接由原始 ECG 数据推理得到其类别，目前主流的心律失常分类均基于 MIT-BIH 数据集，值得注意的是由于 MIT-BIT 数据集的高度不平衡，也有大量基于深度学习的方法对于原始数据进行了数据增强。Acharya 等人[24]提出了一个 9 层的 CNN 模型来自动识别 ECG 中 5 类（N、S、V、F、Q）不同的心跳，作者对原始数据进行了高频降噪与数据增强，并在测试集用到了增强数据。Mousavi 等人[25]对前者提出了质疑，提出了 CNN+Encoder-Decoder 模型来识别 ECG 中 4 类心跳（N、S、V、F），对样本数较少的类别进行过采样，最后在真实数据样本上评估模型。Ádám 等人[26]采用了 2 层卷积层与 1 层全连接层对 ECG 进行 2 分类（正常与不正常），并利用网格搜索方法查找网络的最佳超参数。Le M D 等人[27]提出了一个多模块循环卷积神经网络（RCNN），1D CNN 与 RNN 负责提取 1D-ECG 序列的时空信息，2D CNN 与 RNN 负责学习 ECG 的频谱图，还有一个模块将患者的年龄性别矢量化，最后通过转换编码器融合 3 个模块的信息后经由全连接层对 ECG 进行 6 分类（N、L、R、A、V、Other）。Xu 等人[28]将 CNN 与双向长短时记忆网络（Bidirectional Long Short-Term Memory, biLSTM）融合，对原始数据去噪后，运用了迁移学习将 ECG 信号分为 5 类（N、S、V、F、Q）。Rexy 等人[29]对原始数据降噪后，应用 biLSTM 算法根据 ANSI-AAMI EC57 标准类别将心电图分为 5 类（N、S、V、F、Q）。Kumar 等人[30]将原始 ECG 数据降噪与增强后，利用 CNN 提取特征，最后利用模糊聚类算法（Fuzzy Clustering Algorithm）将 ECG 由 ANSI-AAMI EC57 标准分为 5 类。Ali 等人[31]利用迁移学习方法，将 LSTM、AlexNet、VGG-16、ResNet-50 用于 ECG 数据的 5 分类与 2 分类，取得了相当不错的结果。Lu 等人[32]对 ECG 数据进行 17 分类，引入 Focal Loss[33]解决 ECG 分类中的数据不平衡问题并使用可分离卷积减少所提出的 CNN 架构的参数。

1.2.2 轻量级 ECG 检测算法

随着轻量级模型的发展与对 ECG 分类算法以低功耗低计算量实现于可穿戴设备的重视，涌现了越来越多的轻量级 ECG 检测网络。Li 等人[41]以基于贪心算法的分层量化将 ECG 以 17 种不同的心律失常类别进行分类，不同网络层的位宽因其重要性而异。Rizqyawan 等人[42]将 Acharya 等人[24]的心律失常分类 CNN 框架以 Jacob 等人[43]的量化方法进行量化，即激活值和权重均为 8bit，并部署在了实时边缘设备（Real-Time Edge Device）。Farag 等人提出了一种轻量级的独立短时傅立叶变换（STFT）卷积神经网络（CNN）模型[44],利用 1D CNN 提取 ECG 信号频谱图，将输出重塑为 2 维热图（Heatmap）并输入 2D-CNN 进行分

类，最后作者将模型量化为 8bit 位宽，在云端与树莓派上进行了测试。Scrugli 等人[45]设计了一种自适应传感器节点架构，提出的 CNN 模型经由 PyTorch 量化为 8bit 位宽后仍有着不错的精度。Sivapalan 等人[46]利用 PCA 等方法提取 ECG 特征，结合 LSTM 与 MLP 将 ECG 信号分为正常和异常心律（Normal and Abnormal beats）两类，最后将浮点运算转为 16bit 与 32bit 定点运算并将实现于嵌入式平台。Sun 等人[47]利用 ALQ（Adaptive Loss-aware Quantization）算法对在 MIT-BIH 数据集上的 17 种不同心律失常类别进行分类。Janveja 等人[48]提出了基于 DNN 的低功耗心律失常分类器并以 180nm Bulk CMOS 技术实现，对 ECG 分别进行了 2 分类与 5 分类，同时将权重与偏差转化为 16 位定点数。Wu 等人[49]设计了具有 9 层 1D 卷积与 1 个全连接层的全精度 CNN 并将其二值化，在 PhysioNet/CinC AF Classification Challenge 2017 数据集上取得了不错的成绩。Wong 等人[50]设计了 1 个卷积层+1 个池化层+2 个全连接层的二值化 CNN 用来将 ECG 信号分为 V 类与非 V 类并于 FPGA 上实现。Wang 等人[51]设计了 15 层的 CNN 以 AAMI 标准将 ECG 信号分为 5 类并将模型二值化。Yun 等人[52]将用于根据 ECG 分类心理健康的 CNN+LSTM 模型二值化，大大降低了推理延迟与能耗。

综上所述，在轻量级 ECG 检测网络方面已有大量研究，也有少数工作对 ECG 检测网络进行二值化，但 Wong 等人[50]的 BNN 加速器仅仅实现了 2 分类，难以满足实际应用中医患双方的需求；Wang 等人[51]的 BNN 算法未能在硬件上实现，且对于批量归一化层未作优化，其权重仍旧为 32bit 浮点数；Yun 等人[52]的模型以 47KB 的存储占用实现了 87%的准确率。故在基于 BNN 的 ECG 检测网络方向还有很大的优化与研究空间。

本设计在支持分类的类别数方面，可通过修改最后一层网络块的超参数，替换权重以实现 5 分类与 17 分类的轻易转换；在网络结构方面，保证高精度的同时去除了卷积神经网络中最为常见也是权重占比最多的全连接层；在硬件设计方面，运用了算子融合，从数学层面简化批量归一化层等复杂运算，以 1bit 同或运算代替乘法，极大程度降低了硬件设计的难度的同时提升了加速器的能效，具有一定的实用价值。

1.3 本文研究内容

1.3.1 研究内容

本课题的目标是利用 PyTorch 框架设计一个高精度，硬件友好型的 ECG 检测分类网络，在保证精度的前提下，实现较高的精度与较为先进的模型存储占用。接着以此模型与 Xilinx

ZYNQ 平台为基础，使用 Verilog 设计硬件加速器，在 vivado 中对所设计的加速器进行仿真与功能验证。

本课题的主要研究内容包括：

(1) 设计了一种基于卷积神经网络的 ECG 分类模型，通过二值化算法将模型的权重与激活值限制为 1bit，实现了高精度、硬件友好的二值化神经网络来分类 ECG 信号。并与其他轻量级 ECG 信号分类算法进行对比。

(2) 基于二值化神经网络，为其设计专门的硬件架构。以逐比特同或代替卷积中的乘法，大幅度降低运算单元的资源占用；设计脉冲阵列，以流水线实现卷积的运算，最大化复用网络的权重与输入；将批量归一化层与激活层等层融合为阈值函数，最大程度消除网络中的非二值运算，降低它们的硬件资源占用。

(3) 针对设计好的BNN硬件架构，利用硬件描述语言Verilog实现，并基于Xilinx ZYNQ平台进行功能仿真以及资源、功耗等指标评估。

1.3.2 设计指标

本课题通过二值化算法，在保证高准确率的前提下，将分类 ECG 信号的卷积神经网络的权重与激活值量化为 1bit，再通过结构与算子上的优化，使其成为硬件友好的 BNN，接着为此 BNN 设计专属的硬件加速器。

通过 1.2 节的研究现况介绍与相关文献的阅读，现存的工作中大都通将 ECG 信号分为 5 类或 17 类，并主要以准确率为指标评估分类模型的好坏。本设计的 BNN 软件算法也应考虑 5 分类任务与 17 分类任务，以 Scrugli 等人的工作[45]与 Sun 等人的工作[47]为参考，制定算法准确率指标如表 1.1 所示：

表 1.1 ECG 分类模型准确率对比表

	本设计	Scrugli[45]	Sun[47]
分类数目	5、17	5	17
数据位宽（bit）	1	8	混合精度
准确率（%）	5 分类：96.90 17 分类：97.50	96.98	95.84
数据集	MIT-BIH	MIT-BIH	MIT-BIH

混合精度是指神经网络的每一层的参数都由特定的优化算法根据其重要程度被量化为不同位宽。表 1.1 中本设计的准确率为 BP（Better Performance）模型的准确率，根据表 1.1 可得，数据位宽为 1bit 后，准确率仅与 Scrugli 等人位宽为 8 的工作相差 0.90%；与 Sun 等人混合精度的工作相比，本设计的准确率则提高了 1.66%。

上面的指标为软件算法，而对于加速器，则应以模型存储占用（KB）、吞吐率（GOPS）和能效（GOPS/W）来表示。模型的存储占用即为加速器运行所需的权重大小，吞吐率为每秒加速器计算的乘法与加法数量，能效即为吞吐率除以加速器功耗。本文的硬件加速器采用设计的 LP（Low Power）模型来实现 5 分类 BNN，以 Wei 等人的工作[69]和 Lu 等人的工作[70]为参考，制定加速器性能指标如表 1.2 所示：

表 1.2 ECG 检测加速器性能指标对比表

	本设计	Wei[69]	Lu[70]
分类数目	5	5	5
数据位宽（bit）	1	16	16
存储占用（KB）	3.92	21.61	21.61
吞吐率（GOPS）	133.19	26.6	25.7
能效（GOPS/W）	188.12	33.67	—
时钟频率（MHz）	50	200	200

由上表可见，由于 BNN 中权重与激活值为 1bit 的特性，本设计的加速器在存储占用、吞吐率、能效上均有着一定优势。

(1) 在 PyTorch 框架与硬件加速器上，设计的 BNN 模型在 MIT-BIH 数据集上的心律失常识别准确率不低于 90%。

(2) 模型应实现 25 倍以上的压缩，大小应控制在 5KB 以下。

(3) 加速器电路功耗不高于 1W，吞吐率应高于 100GOPS，能效应高于 120GOPS/W。

1.4 论文组织结构

本文共分为 6 个章节，论文的组织结构如下：

第一章节为绪论，首先介绍了利用 ECG 检测心律失常的背景与意义，以及检测算法的研究现状，将研究现状分为传统 ECG 检测算法与轻量级 ECG 检测算法叙述。最后阐述

了本文的研究内容、设计指标以及组织结构。

第二章为卷积神经网络原理与二值化方法综述，首先回顾了传统 ECG 检测算法的优劣，接着引入卷积神经网络的结构、原理与优势，最后对二值化算法及其近年的发展做了详细综述。

第三章为 ECG 检测算法与二值化方法设计，首先介绍了软件算法所使用的数据集与评估指标，接着通过实验，对 CNN 的结构与网络层构成进行了探索与确定，最后在二值化部分通过实验与公式推导，确定了所要设计 BNN 的具体结构、优化算法，并将设计的模型与近年先进的轻量级 ECG 检测分类模型做了详细的性能对比。

第四章为二值化神经网络的硬件加速器的结构设计，首先介绍了加速器的整体架构与数据流，接着介绍了加速器的数据如何存储，以及针对数据存储专门设计的优化方法，最后选取加速器中的重要模块进行了其各自的设计理念、数据流、架构的详尽介绍。

第五章为加速器的仿真与结果分析，首先介绍了仿真测试台的设置，接着截取重要模块的仿真波形图，以第四章各个模块的数据流为基础，详细介绍了各个信号的功能，证明了所设计模块的正确性，接着从 4 个方面分析了加速器的准确率，最后对加速器的存储占用、资源、功耗、推理时间等硬件指标做了介绍与分析，并与相关加速器论文作了对比。

第六章为总结与展望，系统性地总结了文章中所作的工作，分析了工作中的不足之处，指出了今后可改进与研究的方向。

第二章 卷积神经网络原理与二值化算法综述

本章主要介绍卷积神经网络（CNN）的原理以及神经网络二值化方法。首先回顾以往的 ECG 检测算法的优缺点；接着介绍卷积神经网络相较于传统方法所带来的优势并对其原理与结构作简要介绍；最后，介绍了多比特量化压缩原理，并详细讲解了二值化算法的原理、发展以及专为该算法设计的数学公式和硬件。

2.1 传统 ECG 信号检测方法

传统的 ECG 信号检测方法主要分为 4 个步骤：数据预处理，分割，特征提取与分类。

数据预处理的主要目的为降噪。直接从医学检测仪器上获取的 ECG 信号往往会具有噪声，因此设计各种各样的滤波器来去除噪声成为研究人员的首选方案，这些滤波器从高通、带通、低通滤波器、卡尔曼滤波器到基于神经网络的自适应滤波器，均对 ECG 信号的识别精度有了一定改善。

分割，即通过合适的算法来检测 ECG 中的 QRS 复合波。QRS 复合波对应于心脏右、左心室的除极和大心室肌肉的收缩，是 ECG 信号中最显著的特征，因此，它的检测对于后续的 ECG 信号分析至关重要。

特征提取是利用 ECG 对心律失常分类的关键。它可以分为两种主要的类型：时域和频域。时域上，可以通过分割得到的 QRS 复合波群来获取一个信号的 RR 间隔，或者利用波群计算 QRS 复合波的持续时间等，得到的数据即为 ECG 的时域特征。在频域上，可通过离散小波变换、傅里叶变换等对分割后的 ECG 信号进行频谱分析，得到 ECG 的频域特征。

最后即为分类，研究人员可以根据特定的检测任务，采取机器学习中的支持向量机（SVM）、k 近邻算法（kNN）、随机森林（RF）等算法，构建特定的模型来对提取到的特征进行分类。

综上所述，传统的 ECG 信号检测方法步骤较多且极为复杂，对于哪一步更为重要，需要着重优化，目前也没有定性的结论。对于降噪，往往在医学检测仪器上就有相关的滤波器来实现，而分割、特征提取，则需要研究人员有着一定的医学基础，这些算法实现于嵌入式硬件又需要特定的设计，消耗了额外的资源。相比之下，深度学习可以直接输入未经分割与提取特征的 ECG 信号，自动学习、提取信号中的特征并分类，实现端到端的 ECG 信号检测。ECG 信号为时序信号，可使用循环神经网络（RNN）或卷积神经网络（CNN）进行分类，CNN 较 RNN 而言，具有更好的并行性与较低的时序依赖性，更易部署到嵌入

式设备，它也成为了大多数利用深度学习分类 ECG 信号的研究人员的首要选择。

2.2 卷积神经网络基本原理

卷积神经网络最初是为处理图像数据而生的神经网络，它的灵感来自于视觉神经元。对于一个区域的视觉神经元，单个神经元负责感应一小部分区域的外来光刺激，而千千万万个神经元的感应区域相互交叠，则为动物提供了一个完整清晰的视野。1998 年 LeCun 第一次提出了卷积神经网络的概念，设计的 LeNet-5[34]网络被美国邮政局用来自动识别手写邮政号码，也被用来识别写在支票上的数字。然而由于计算机算力在当时较低，卷积神经网络的发展一直较为缓慢。2012 年，AlexNet[35]凭借英伟达 GPU 的高算力与吞吐量，训练了具有千万参数级别的 CNN，凭借巨大的优势获得了 2012 年 ImageNet 图像分类大赛的冠军，自此以后卷积神经网路成为了各个领域研究的焦点。当今几乎所有的图像识别、目标检测或语义分割相关的研究都以此为基础。

典型卷积神经网络包括卷积层、激活层、池化层、批量归一化层与全连接层等，本节将对这些层作简要介绍。

2.2.1 卷积层

卷积操作是通过卷积核（也称为滤波器或卷积矩阵）对输入图像进行卷积计算得到输出特征图的过程，为了实现一次完整的卷积操作，需要设置以下参变量:

（1）卷积核

卷积核是卷积操作的关键，它通常是一个小的矩阵，其大小通常为正方形，例如 3x3、5x5 或 7x7 等。卷积核的值在卷积过程中被用来计算输出特征图的每个像素的值。

（2）填充（Padding）

在卷积操作中，为了避免输入图像的边缘像素在卷积核滑动时被忽略，通常会在输入图像周围添加一定数量的零值像素，这个过程称为填充。

（3）步长（Stride）

卷积核在输入图像上滑动的像素数称为步长，步长的大小决定了输出特征图的大小。较大的步长可以减少输出特征图的大小，而较小的步长则可以保留更多的信息。

（4）卷积操作

卷积操作是通过将卷积核与输入图像逐元素相乘，然后将所有乘积的结果相加得到输出特征图的每个像素的值。在这个过程中，卷积核从输入图像的左上角开始滑动，每滑动

一个步长，就会产生一个输出像素值。

图 2.1 示意了一个具有 3 个通道的输入特征图与 3 个卷积核的卷积过程。每一个卷积核的通道数与输入特征图的通道数相等；卷积核的个数对应于输出特征图的通道数；输出特征图的大小可由（2.1）求得。式中的p即为填充大小；f为卷积核的大小；s为步长。在图 2-1 中，它们分别为：1、3、1。故最后求得输出特征图的大小为 3。

$$Size = \frac{n + 2p - f}{s} + 1 \tag{2.1}$$

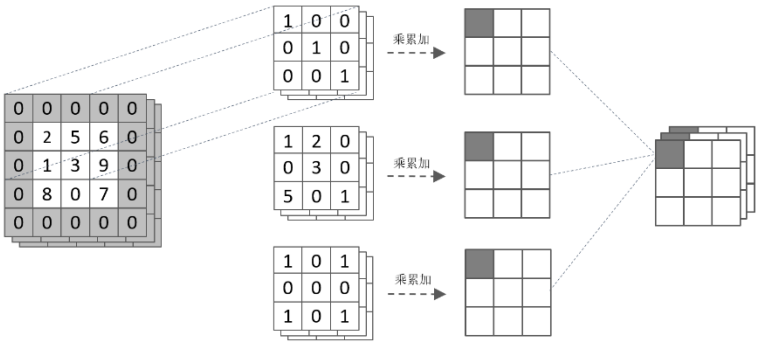

图 2-1 卷积过程示意图

2.2.2 激活层

激活层是 CNN 中的一种关键组件，它通常跟在卷积层之后，在卷积层的输出上应用一个非线性函数。激活函数的作用是对输入信号进行非线性变换，以使得网络可以捕捉到更加丰富的特征，同时也有助于缓解梯度消失的问题。本小节介绍 2 种常用的激活函数：ReLU 与 PReLU。

（1）ReLU

ReLU 函数的数学公式极为简单，同时它也是最为常用的激活函数。其表达式为$f(x) = max(0, x)$，即小于 0 的输入变为 0，而大于 0 的输入不变。

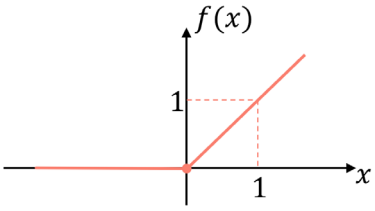

图 2-2 ReLU 示意图

（2）PReLU

PReLU 是 ReLU 的改良版本，它会将小于 0 的输入乘以一个可学习的参数，而不是全部归零。PReLU 的公式与示意图如下所示：

$$f(x)=\begin{cases}x & \text{if } x \geq 0\\ ax & \text{otherwise}\end{cases} \tag{2.2}$$

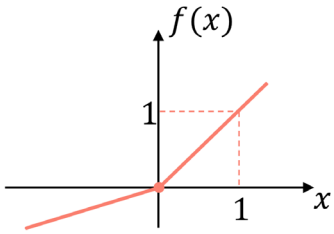

图 2-3 PReLU 示意图

2.2.3 池化层

池化层用于减小特征图的大小。在池化层中，通常通过对输入特征图中的局部区域进行聚合操作来得到输出特征图。它通常有两种类型：最大池化和平均池化。

（1）最大池化（Max Pooling）：

最大池化是一种常用的池化方法，它的聚合操作是对输入特征图中的每个局部区域（也称为池化窗口）取最大值。

最大池化的主要作用是减小特征图的空间大小。在卷积神经网络中，最大池化通常与卷积层交替使用，可以有效地减小特征图的大小，从而减少模型的计算量和内存占用，并且可以增加模型的稳定性和泛化能力。

（2）平均池化（Average Pooling）：

平均池化是另一种常用的池化方法，它的聚合操作是对输入特征图中的每个局部区域取平均值。在平均池化中，池化窗口通常是非重叠的，并且在特征图的每个通道上分别进行聚合操作。平均池化除了减小特征图的空间大小，还可以增加特征图的平滑性。

综上，池化层是卷积神经网络中非常重要的一部分，它可以通过聚合操作来减小特征图的大小和参数数量，并且可以增加模型的稳定性、泛化能力和平滑性。在实际应用中，由于平均池化涉及计算，故最大池化使用颇多。

2.2.4 批量归一化层

批量归一化是一种在卷积神经网络中广泛使用的技术，可以加速训练过程并提高模型

的泛化能力。它通过对每个神经网络层的输入进行归一化处理，从而缓解了神经网络层输入数据分布变化对训练的影响，使神经网络层的学习更加稳定。其公式如（2.3）所示：

$$f(x)=\frac{x-\mu}{\sqrt{\sigma^2+\varepsilon}}\gamma+\beta \tag{2.3}$$

μ与σ分别为输入x的均值与标准差，ε则为防止分母为 0 而设置的一极小正数。γ与β为可学习的参数，前者称为缩放参数，后者称为平移参数，用于调整标准化后的数据输出范围和偏移量。

2.2.5 全连接层

全连接层之所以在本小节的最后介绍，是因为卷积神经网络的全连接层（Fully Connected Layer）通常是网络的最后一层，卷积层和池化层输出的特征图展开成一维向量后，全连接层将其与权重矩阵进行矩阵乘法和加法运算，得到最终的分类结果。

全连接层的输出通常会经过一个激活函数，如 ReLU，来增加网络的非线性，同时还常通过 Dropout 来随机地不更新一部分权重的值以防止过拟合。全连接层的公式可简单地表示为式（2.4）：

$$f(x)=Wx+b \tag{2.4}$$

式（2.4）中W为全连接层的权重矩阵，b为全连接层的偏置向量。

2.3 神经网络量化算法

1998 年的 LeNet-5[34]仅需约 60,000 个参数；2012 年 AlexNet[35]用凭借 GPU 的高算力与吞吐量，可以训练 61.1 百万个参数；2015 年的 ResNet 中的 ResNet-50[36]约有 25 百万个参数；2018 年的 ShuffleNet V2[37]结构比以往的网络更加高效率，参数更少，但也需要 136 万个参数。庞大的参数量同时也带来了大量的浮点计算，这对于在低成本低功耗低资源的可穿戴设备部署是致命的。

为了解决模型巨大难以部署的问题，业界通常采用量化策略，将在 PyTorch、Tensorflow 等框架中训练的 float32 数据量化为 8bit，这使得存储张量内存开销减少 4 倍，矩阵乘法的计算成本减少 16 倍[38]。更进一步还有 4bit 量化，Nogami 等人[39]对 AlexNet 与 VGG16 的 4bit 量化策略仅仅带来了约 2%的精度损失。最为极端的量化是二值化[40]，由 Matthieu Courbariaux 团队于 2016 年提出并开放了源码，这一方法最大幅度地减少了存储权重的内存开销，并且可以让 DNN 中的浮点乘法被 1bit 的 XNOR-POPCOUNT 运算代替，完美契合硬件的底层运算逻辑。

2.3.1 多比特量化压缩原理

神经网络量化是指网络中的权重和激活值从 32 位浮点数转换为较低位宽与精度的定点数，以减少存储空间和计算量。目前的量化算法主要可分为两大类：训练后量化（Post-Training Quantization, PTQ）和量化感知训练（Quantization-Aware-Training, QAT）。

在 PTQ 中，训练好的浮点数模型被转换为可以在硬件上更高效运行的低比特定点数或整数模型。通过将模型的权重和激活值从浮点数转换为定点数或整数，可以大大减少模型的存储空间和计算需求，从而加速模型的推理过程。

在 QAT 中，模型在训练过程中被量化为低比特定点数或整数，并在量化后的模型中进行训练。通过在训练期间模拟量化过程，QAT 可以帮助模型适应量化过程中可能引入的误差和不确定性。这使得模型在后续量化时具有更好的鲁棒性，并且可以保持较高的推理准确度。

无论哪一种量化算法，都需要将浮点数转为定点数或整数，这一转换过程可视为一种映射，目前主流的映射方法为均匀仿射量化（Uniform affine quantization）、对称均匀量化（Symmetric uniform quantization）以及二次幂量化（Power-of-two quantization）：

（1）均匀仿射量化

均匀仿射量化是非对称量化，在确定浮点数x_{float}的量化位宽b后，再由x_{float}的最大值与最小值引入尺度因子s如式(2.5)：

$$s = \frac{Max(x_{float}) - Min(x_{float})}{2^b - 1} \tag{2.5}$$

接着为了考虑实数 0 被正确映射到整数以使 ReLU 等函数不引起量化误差，且反量化后的整数在q_{min}~q_{max}范围内（q_{min}与q_{max}是指b位宽的数可表达的无符号整数最小值与最大值），引入零点z如式（2.6）所示：

$$z = round\left(q_{max} - \frac{r_{max}}{s}\right) \tag{2.6}$$

最后可得均匀仿射量化的映射公式为（2.7）：

$$x_{int} = clamp\left(round\left(\frac{x_{float}}{s}\right) + z; 0, 2^b - 1\right) \tag{2.7}$$

上式中x_{int}即为量化后的整数，round 为一取整函数，它会将计算结果取整到离其最近的整数，clamp 为截断函数，它将 round 函数计算的结果限制在[0~$2^b - 1$]，这正是位宽为b的多比特数可以表示的无符号整数范围。均匀仿射量化可由图 2-4 表示如下：

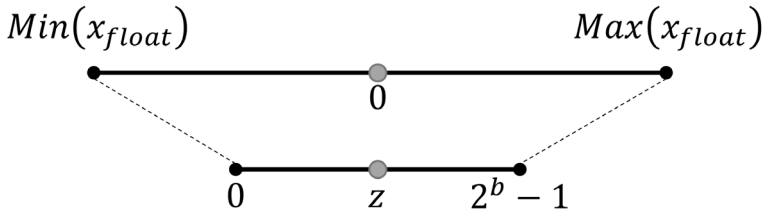

图 2-4 均匀仿射量化示意图

（2）对称均匀量化

对称均匀量化较上文的均匀仿射量化而言更加简便，它的特点是零点为 0，这使得量化之前不需要额外有关零点的计算，量化后也无需考虑零点的存储。因为量化后的有符号整数关于 0 点对称，在计算s时，就要令浮点数的范围为$|Max(x_{float})|$~$|Max(x_{float})|$，s的计算公式如式（2.8）所示：

$$s=\frac{2\times|Max(x_{float})|}{2^b-1} \tag{2.8}$$

由于z为 0，则直接可得对称均匀量化的映射公式（2.9）：

$$x_{int}=clamp\left(round\left(\frac{x_{float}}{s}\right);-2^{b-1},2^{b-1}-1\right) \tag{2.9}$$

由于没有考虑$Min(x_{float})$与z，因此对称均匀量化较均匀仿射量化具有更小的计算量与更大的误差，对称均匀量化的示意图如图 2-5 所示：

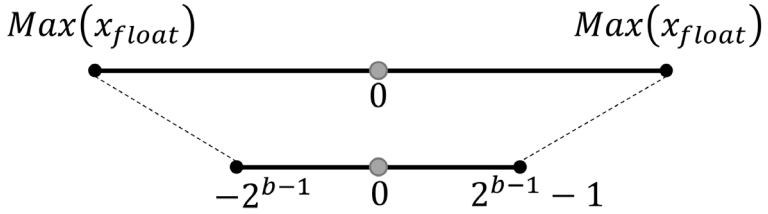

图 2-5 对称均匀量化示意图

（3）二次幂量化

上文中的 2 种量化方法s皆为浮点数，二次幂量化是对称量化的特殊情况，它令$s=2^{-k}$，这样就可以让除以浮点数变为硬件上的移位操作，大幅度提升了硬件的量化操作执行效率，但2^{-k}表达能力有限，它作为尺度因子数值跨度太大，因此也不可避免地带来了量化误差。

2.3.2 二值化算法基本概念

浮点运算计算代价昂贵，且不为大多数嵌入式硬件支持。BNN 则是将卷积、全连接层

的浮点乘累加（MAC）转为 1bit 数的同或（XNOR）与位 1 计数（POPCOUNT），这最大化降低了计算复杂度与权重的内存占用。我们通过式（2.10）将权重与激活值转换为{+1, -1}。

$$x^b = Sign(x) = \begin{cases} +1 \; if \;\; x \geq 0 \;, \\ -1 \; otherwise \end{cases} \tag{2.10}$$

式（2.10）中，x为神经网络的输入与权重，它经过符号函数，被量化为{+1, -1}，在神经网络训练时，运算以{+1, -1}进行，但是在利用计算出的损失值通过反向传播来对权重更新时，则更新的是权重在转换前的浮点值。这就引入了一个新的问题：符号函数$Sign(x)$在 0 处的梯度为∞，在其他处梯度为 0，这使得基于链式法则的梯度计算无法正常进行。因此最普遍的方法是采用直通估计器（Straight-Through Estimator, STE），其等效于在反向传播时视$Sign(x)$为$Htanh(x)$以计算梯度。基于上述基本方法，二值神经网络（BNN）得以被正常训练。

$$Htanh(x) = \begin{cases} +1 \; if \;\; x \geq +1 \;, \\ -1 \; if \;\; x \leq -1 \;, \\ x \quad otherwise \end{cases} \tag{2.11}$$

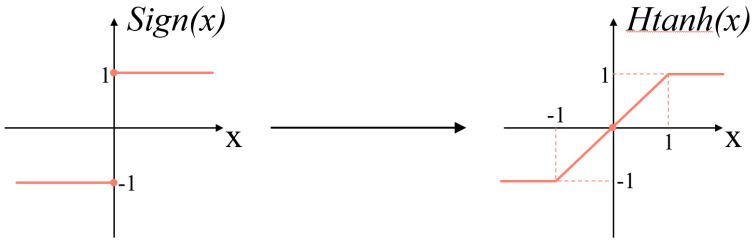

图 2-6 $Sign(x)$与$Htanh(x)$示意图

2.3.3 二值化算法研究近况

自 2016 年 BNN[40]发表后，各种 BNN 改良方案层出不穷。

XNOR-Net[54]为了提高普通 BNN 的精度，在每个 1bit 卷积后需要乘以 2 个尺度因子（K与α）以修正结果，其中α可根据二值化后的权重计算得出，即训练完毕后可根据模型存储的 1bit 权重提前计算并存储在硬件中，而K则需要在推理中求得。K的计算与卷积完成后乘以尺度因子这两个因素造成了延迟的增加与额外的计算资源，并影响着硬件加速代码的设计；XNOR-Net++[55]考虑了在 1bit 卷积后乘以一个可学习的尺度因子来提高精度，同样引入了额外的多 bit 数相乘，以硬件的能耗换取了量化误差的减小；Tang 等人[56]则使用 PReLU 代替 ReLU，将前面方法中的尺度因子吸收到了激活函数中；Choi 等人[57]的 PACT 模型将$Sign(x)$函数的阈值 0 更改为一个可学习的参数，从硬件上来看使得激活函数从只需

判断符号位的比较器变成了一个多 bit 减法器；Liu 等人的 Bi-Real Net[58]采取了类似 ResNet 的结构，引入 shortcut，即将输入加到输出上，以此来减少信息的损失，因此 BNN 网络在经过批归一化后的结果需要与网络块的输入相加再通过$Sign(x)$二值化，这使得在 BNN 中被广泛使用的算子融合技术失效；Geng 等人[59]针对此问题提出了相应的硬件改良。

也有一些研究认为 STE 代替$Sign(x)$函数的梯度是导致精度下降的来源之一，Bi-Real Net 不仅引入 shortcut，也将 STE 改良为分段函数（称之为 ApproxSign）以尽可能模拟$Sign(x)$的导数；Qin、Ding、Xu 等人[60][61][62]也分别以各自提出的近似函数代替了 STE。改进 STE 只会影响反向传播，不会影响正常的模型推理，也就是说不会在硬件实现时占用额外的资源。

在激活值的角度，Ding 等人[63]通过添加额外的损失函数以正则化激活值的分布；在权重的角度，Lin 等人[64]考虑了二值化权重与全精度权重的角度误差，提出了 RBNN，在训练时将全精度权重旋转以减少角度误差后再二值化；因$Htanh(x)$导致绝对值大于 1 的权重梯度为 0，不会被更新，Xu 等人[65]将其称之为死权重并提出了 RECU 将其恢复，以此来减少量化误差。这 3 个工作是硬件友好的，虽然训练时引入了一些额外计算，但不会对硬件推理产生影响。

2.3.4 二值化算法有关的数学公式与专用硬件

在 2.3.1 节的介绍中，我们注意到$Sign(x)$函数仅仅会将权重与激活值量化为{+1, -1}，而要实现真正的 1bit，它们应为{1, 0}。{+1, -1}与{1, 0}的乘累加运算，可由式（2.12）等效：

$$Result = 2 * p - k \tag{2.12}$$

式（2.12）中，p代表着所有值均为{1, 0}的权重与激活向量逐比特同或（XNOR）后进行位 1 计数（POPCOUNT）的结果，k则代表向量的长度，在卷积运算中，即为卷积核的大小。图 2-7 展示了一个简单的示例，上半部分为普通的卷积，乘累加后结果为-5，下半部分是硬件中的等效卷积，5 个 1 与 5 个 0 逐比特同或后为 5 个 0，因此 POPCOUNT 的结果为 0，图中可见k为 5，因此计算结果为-5。可见通过式（2.12），{+1, -1}之间的卷积计算可以等效映射硬件上的 1bit 运算。

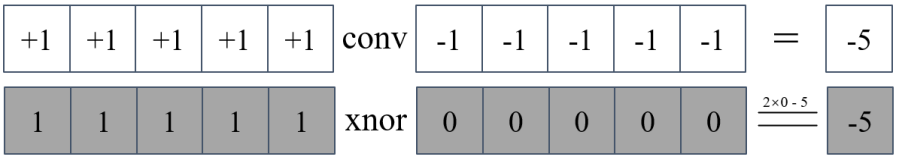

图 2-7 普通卷积与逐比特 XNOR+POPCOUNT 的等效卷积

对于 1bit 同或，在 FPGA 上，可由 1 个二输入查找表（LUT）实现，在 ASIC 上，则可由 4 个 MOS 管实现，如图 2-8 所示：

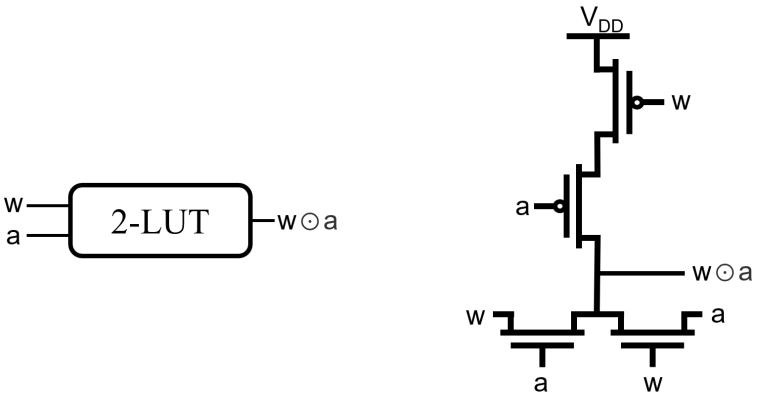

图 2-8 XNOR 在 FPGA 与 ASIC 上的实现方式

2.4 本章小结

本章首先介绍了传统 ECG 检测方法的 4 个步骤，分析其缺点并引入深度学习中的卷积神经网络分类方法，然后介绍了卷积神经网络的基本概念，包括其最常见的各个运算层的相关公式与作用，最后，以近年来深度学习发展的简要陈述，引入了神经网络的二值化算法，并介绍了它的基础概念、研究近况以及在硬件上实现的相关公式与电路。为下一章的 ECG 检测网络与二值化算法设计打下理论基础。

第三章 ECG 检测算法与二值化方法设计

本章主要为实验设计介绍。分为四个部分，第一部分介绍了实验所用的 MIT-BIH 数据集，第二部分介绍了要用到的评估指标，第三部分介绍 ECG 检测算法的设计思路、实验结果。第四部分则以是以前三部分奠定的模型，结合近年先进的二值化算法，对设计的模型进行修改与优化，第五部分将二值化后的实验结果与近年轻量级的 ECG 检测算法作对比。

3.1 MIT-BIH 数据集简介

本实验的网络对 ECG 信号进行心律失常 17 分类与 5 分类。所有数据均来源于的 MIT-BIH Arrhythmia[66]数据库。

5 类别数据集包含了 7740 个 10s 片段。分类遵循 AAMI EC57 标准，将所有数据分为 ventricular ectopic（V）, beat against other classes such as non-ectopic（N）, supraventricular ectopic（S）, fusion（F） and unknown（Q）5 类。其具体分布如表 3.1 所示。

表 3.1 心律失常 5 类数据集样本分布

序号	类别	数量
1	V	1890
2	N	5186
3	S	19
4	F	545
5	Q	1890
合计		7740

17 类别数据集与 Pławiak 等人的工作[67]所用相同，包括了正常窦性心律、起搏器心律等 15 种类型的心律失常（每种类型至少 10 个），共有 1000 个 10s 的 ECG 信号片段用于分析，其分布如表 3.2 所示：

表 3.2 心律失常 17 类数据集样本分布

序号	类别	数量
1	Normal sinus rhythm	283
2	Atrial premature beat	66
3	Atrial flutter	20
4	Atrial fibrillation	135
5	Supraventricular tachyarrhythmia	13
6	Pre-excitation (WPW)	21
7	Premature ventricular contraction	133
8	Ventricular bigeminy	55
9	Ventricular trigeminy	13
10	Ventricular tachycardia	10
11	Idioventricular rhythm	10
12	Ventricular flutter	10
13	Fusion of ventricular and normal beat	11
14	Left bundle branch block beat	103
15	Right bundle branch block beat	62
16	Second-degree heart block	10
17	Pacemaker rhythm	45
合计		1000

无论是 5 类还是 17 类数据，我们都使用了来自一根导联 MLII 的信号，它们以 360HZ 在 10s 内取 3600 个样本记录，并将作为 ECG 分类网络的输入。

关于数据集的预处理，值得注意的是，因为 MIT-BIH 数据集的高度不平衡，绝大多数研究都对数据集利用各种方法作了数据增强、过采样等处理。我们则对心律失常 17 分类数据集做了标准化处理，对 5 类数据集作了中值滤波与标准化处理。关于滤波，通过购置较好的设备，使用其内在的降噪功能也可完成操作。因此实际应用中在嵌入式硬件中仅需对采样到的数据作标准化处理，这相比于特征提取更为硬件友好。

3.2 评估指标简介

合理多元的评估指标利于我们从各个方向评估所设计网络的性能。本设计使用准确率：accuracy（ACC），灵敏度：sensitivity（SEN），特异度：specificity（SPE），精确度：precision（PPR），F1 分数（F1-score）来综合评估模型性能。令 TP, TN, FP, FN 分别为真阳性（true positive），真阴性（true negative），假阳性（false positive），假阴性（false negative），则各个分类指标的公式如下：

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.1}$$

$$Sensitivity = \frac{TP}{TP + FN} \tag{3.2}$$

$$Specificity = \frac{TN}{TN + FP} \tag{3.3}$$

$$Precision = \frac{TP}{TP + FP} \tag{3.4}$$

$$F1 - \text{score} = \frac{2 \times \text{TP}}{2 \times (\text{TP} + \text{FP} + \text{FN})} \tag{3.5}$$

在 ECG 检测神经网络设计中，为简化设计流程，我们以准确率为导向来设计评估网络，得到最终二值化的模型后，计算其余 4 个指标，并与近年最先进的轻量级网络作对比。

3.3 ECG 检测算法设计

二维卷积神经网络普遍被用于进行图像分类、目标检测等工作，近年来取得了非常好的成绩。而一维卷积神经网络在处理序列数据方面，也有着极好的优势：

（1）捕捉局部特征：生理信号通常包含具有局部相关性的信号段，例如心电图中的 QRS 波群。1D CNN 能够通过卷积操作捕捉这些局部特征，从而提高分类准确性。

（2）参数共享：1D CNN 使用的卷积操作具有参数共享的特点，即在不同的时间步上使用相同的卷积核，这有效地减少了网络中需要训练的参数数量，避免了过拟合问题。

（3）平移不变性：生理信号在时间上具有平移不变性，即信号在时间轴上的平移不会改变信号的意义。1D CNN 的卷积操作可以在不同时间步之间共享卷积核，因此具有平移不变性，从而提高了分类准确性。

（4）快速训练和推理速度：1D CNN 具有简单的结构和较少的参数，可以在较短的时间内进行训练，并且在推理时具有较快的速度，适用于实时应用场景。

因此，1D CNN 在处理生理信号分类任务方面有着较好的性能，被广泛应用于心电图分类等生理信号处理任务中。本课题的 ECG 检测算法设计便是基于 1 维卷积神经网络开展。

3.3.1 ECG 检测网络结构探索

在设计一维卷积 ECG 检测网络时，必须秉持硬件友好原则。因为最开始的设计是基于 PyTorch 等高级编程语言，它提供了许多库函数来帮助我们实现卷积、池化、全连接等种种复杂的模块，并利用 Nvidia 的 CUDA 来加速它们的计算。而将卷积神经网络中的种种操作使用 Verilog 等硬件描述语言实现，一个简单的乘法操作都要在电路设计层面为其考虑代码，若网络结构过于复杂多样，则会在很大程度上增加后期的硬件实现难度。

首先确定卷积神经网络的构成。为将网络快速部署于嵌入式硬件，卷积网络应由多个基本块构成，每个基本块内含网络层相同，均由卷积、池化、激活等层构成，由多个基本块可以轻易构建不同深浅的网络结构。

图 3-1 基本块构建

如图 3-1 所示，每个基本块由 4 个网络层构成，输入特征图首先经过卷积提取特征，输出特征图经过最大池化层减小大小后，经过激活层使数值变得非线性，最后经由批量归一化层重新调整数据分布，结果最终再输入其他基本块。

以基本块为基础，我们可以构建 ECG 检测网络的整体结构如图 3-2。本结构与经典结构的不同在于，我们没有引入全连接层来对网络的特征分类，而是引入了全局累加池化

（Global Sum Pooling）与比较层作为分类结构。

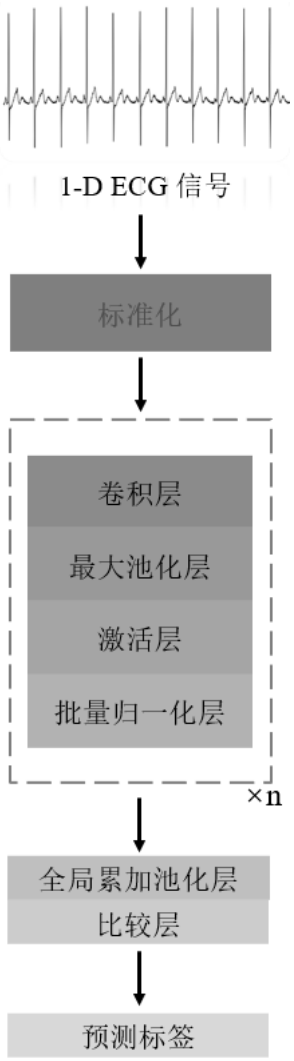

图 3-2 ECG 检测网络结构

网络的总体结构确定后，就要进行网络超参数的选择，包括但不限于：基本块的个数；每个基本块内卷积层的输出通道数、卷积核大小、卷积步长、填充大小；最大池化层的池化窗口大小、池化步长等。

上文提到的超参数，如输出通道数、步长等，可取常用的经验参数。CNN 的通道数通常从较小的数字，如 2 的 3 次方开始逐渐递增，这是因为较小的通道数可以在更早的阶段捕捉生理信号的基本特征，如心电图信号中的 QRS 波群等，而更大的通道数可以在后续阶段捕捉更复杂的特征和更高层次的语义信息，通道数为 2 的次方有利于硬件实现中用移位来计算卷积核个数等参数。因此我们的通道数将以 8、16、32 逐渐增加到 64。最后一个基本块的输出通道为我们要分类的数目，在本课题中为 5 或者 17，全局累加池化将批量归一化层的输出张量每一个通道上的所有值相加，以此消去输出的最后一个维度，比较层将全

局累加池化层输出的张量在最后一个维度比较，输出最大值的索引，这索引即为输入 1-D ECG 信号的类别。

关于卷积层的步长，在第一个基本块设置为 2，这是因为每一个 ECG 信号有 3600 个样本，设置为 2 降低了两倍输出特征图的尺寸，可以减少后续层的计算量，也能有效地减小过拟合，因为它强制使每个卷积核在输出特征图上的采样位置之间有更大的间隔，从而降低了特征图之间的相关性。后续基本块的卷积步长设置为 1，不仅保留了信号更多的细节，利于捕获它们的局部特征，也可以使得我们引入移位寄存器来将存储在 RAM 中的输出特征图逐节拍打出，更加高效地实现数据的传输。卷积的填充根据经验设置为步长减去 2，以最大程度防止边缘信息的缺失。关于最大池化层，它的步长统一设置为 2，以进一步减少输出特征图的尺寸，以此降低模型的复杂度与计算量，加快模型训练。池化窗口则与卷积核大小相同，利于软硬件代码的编写与调试。对于激活层，选取被各种研究广泛使用的 ReLU。

经过上述考虑后，ECG 检测网络结构的超参数仅剩下卷积核的大小与基本块的个数。首先，我们固定基本块个数为 5，对卷积核的大小进行了探索。卷积核的大小影响着模型的参数量与性能，当卷积核的大小增加，1D CNN 可以捕捉更长的序列特征，因此对于处理长序列数据（如音频、文本）时可能会更有效。但如果卷积核过大，可能会导致过拟合和计算资源消耗过大的问题。而如果卷积核过小，则可以捕获更多的局部特征，但是也会丧失更多的全局特征。我们对卷积核大小的实验结果如图 3-3 所示：

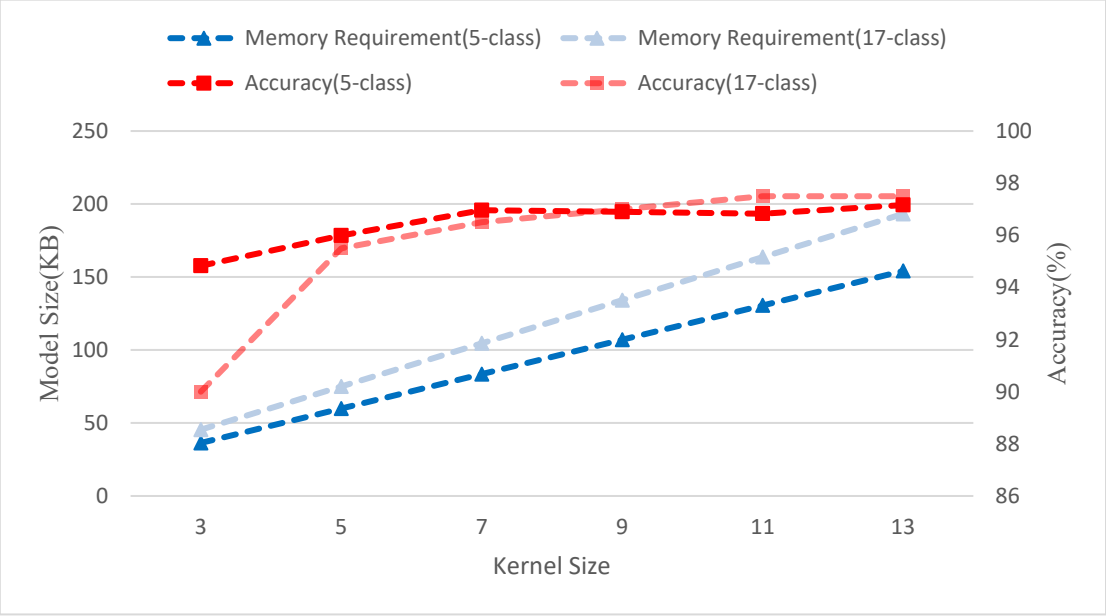

图 3-3 卷积核大小对模型大小与准确率影响

由图 3-3 可见，随着卷积核的增大，准确率整体呈现上升趋势，模型大小则与卷积核大小近乎成正比。在卷积核大小超过 7 以后，增大卷积核带来的准确率增益变得不再明显，模型大小也逐渐向着 100KB 以上靠拢。因此，折中考虑，ECG 检测网络结构的卷积核统一设置为 7。

确定卷积核大小后，最大池化的窗口大小与卷积之前的填充（Padding）大小也随之确定，分别为 7 与 5。接下来，未确定的超参数为基本块的数量，对基本块个数的实验数据如图 3-4 所示：

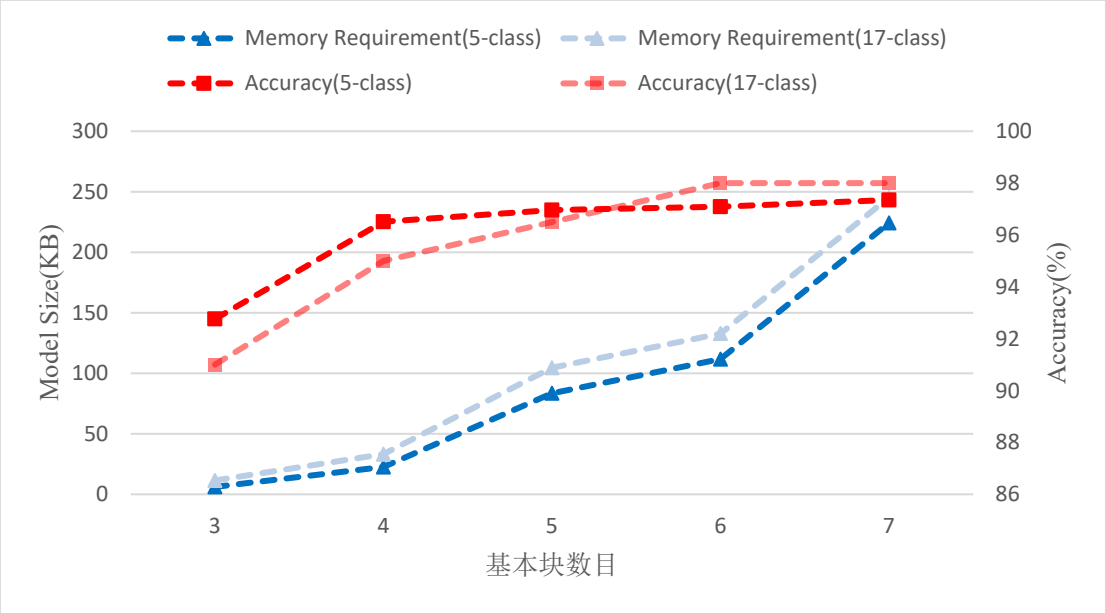

图 3-4 基本块数目对模型大小与准确率影响

由图 3-4 可见，随着基本块数目的增多，ECG 检测网络的准确率呈现上升趋势，但是当基本块数目从 6 增至 7 时，已无法对准确率产生明显的影响，模型大小反而增长了 2 倍。因此设置基本块数目为 6。

综上所述，我们的模型超参数已完全确定，由此获得的 ECG 5 分类模型的大小为 111.71KB，准确率为 97.07%；ECG 17 分类模型的大小为 132.81KB，准确率为 98.00%。网络详细参数如表 3.3 所示；假设网络的输入张量形状为（1, 3600）而非训练时的（batch size, 1, 3600），则张量的形状变化如表 3.4 所示。

表 3.3 ECG 检测网络参数表

	输入通道	输出通道	卷积层填充大小	卷积核大小及步长	池化窗口大小及步长	参数量
基本块 1	1	8	5	7,2	7,2	73
基本块 2	8	16	5	7,1	7,2	929
基本块 3	16	32	5	7,1	7,2	3649
基本块 4	32	32	5	7,1	7,2	7233
基本块 5	32	64	5	7,1	7,2	14465
基本块 6	64	5 or 17	5	7,1	7,2	2251 or 7651
合计						28600 or 34000

表 3.4 张量在网络中的形状变化

	输出形状
标准化	(1, 3600)
基本块 1	(8, 898)
基本块 2	(16, 448)
基本块 3	(32, 223)
基本块 4	(32, 111)
基本块 5	(64, 55)
基本块 6	(5 or 17, 27)
全局累加池化层	(5 or 17)
比较层	(1)

3.4 二值化算法设计与实验

3.4.1 激活层的选取

Tang 等人的工作中[56]，PReLU 代替 ReLU 使得 BNN 取得了更为令人满意的结果，但 PReLU 是否在二值化后的 ECG 检测网络中有效还需要通过实验来验证。我们以 2.3.1 节介

绍的二值化算法中最为基本的直通估计器（STE）作为被量化为 BNN 的 ECG 检测网络的梯度估计器，对比了不同激活层对 BNN 准确率的影响，实验结果如表 3.5 所示：

表 3.5 ReLU、PReLU 对 BNN 准确率的影响

	5 分类模型准确率(%)	17 分类模型准确率(%)
ReLU	96.51	97.00
PReLU	96.71	97.50
无激活层	96.06	96.00

根据表 3.5 所示的实验结果，使用 PReLU 于 5 分类 BNN 模型可以有 0.20%的准确率提升，对于 17 分类 BNN 模型有 0.50%的准确率提升。因此在 BNN 模型中，采取 PReLU 代替 ReLU 是合适的选择。

3.4.2 阈值融合

二值化神经网络（BNN）中的批量归一化层是必不可少的，它可以对下一个基本块中二元激活函数的输入张量进行移位缩放，这可以最大限度地减少精度下降。二值化算法通常将卷积核权重与卷积后输出的激活值量化，很少对批量归一化层做额外的二值化处理，其若转换为定点数并在 FPGA 中存储，则会在加速器的前向推理中引入整数运算，并消耗大量硬件资源 DSP，引入额外的能耗与延迟。若在 PyTorch 等框架中将原本的 float32 浮点数量化为 8bit 定点数存储，则会不可避免地造成精度上的损失。合理地将批量归一化层与其他层相融合，使它们成为一个阈值函数，可以最大程度解决资源浪费与精度损失等问题。

根据 3.3 节中设计好的基本块与前一小节的实验结果，批量归一化层前面为 PReLU 激活层，除最后一个基本块，批量归一化层后面为卷积层，而卷积的输入应被$Sign(x)$量化，因此相当于批量归一化层后为$Sign(x)$函数。我们可以将他们融合为阈值函数，式（2.2）与（2.3）可融合为式（3.6）：

$$f(x)=\begin{cases} kx+b & \text{if } x \geq 0 \\ akx+b & \text{otherwise} \end{cases} \tag{3.6}$$

式（3.6）中的a为 PReLU 在 y 轴左侧直线的斜率，k、b分别可由（3.7）与（3.8）表示：

$$k=\frac{\gamma}{\sqrt{\sigma^2+\varepsilon}} \tag{3.7}$$

$$b = \beta - \frac{\mu\gamma}{\sqrt{\sigma^2 + \varepsilon}} \tag{3.8}$$

其中的γ、β、σ、ε、μ均会在训练完毕 BNN 后被固定并存储在模型当中。

紧接着，式（3.6）要通过$Sign(x)$，经过简单的数学变换后，我们也可以将它们融合。首先介绍一个新的阈值函数$Sign_\delta(x)$：

$$f(\mathrm{x}) = Sign_\delta(x) = \begin{cases} +1, & x \geq \delta \\ -1, & \text{otherwise} \end{cases} \tag{3.9}$$

式（3.9）不同于之前的符号函数$Sign(x)$，它并非根据符号判断是否将输入x变为{+1, -1}，而是根据阈值δ，倘若$x \geq \delta$则输出为+1，否则输出-1。式（3.6）与$Sign(x)$可融合为式（3.10），式中的$\delta_+ = \frac{-b}{k}$，$\delta_- = \frac{-b}{ak}$。

$$f(x) = \begin{cases} sign_{\delta_+}(x) & \text{if } x \geq 0 \\ sign_{\delta_-}(x) & \text{otherwise} \end{cases} \tag{3.10}$$

需要额外注意的是，式（3.6）中的k与ak变为$-b$的分母，是在不等式两边经过除法运算的，若他们小于 0，则阈值比较的不等号需要改变，带来的直接影响为$Sign_\delta(x)$需变为$-Sign_\delta(x)$。

综上所述，经过一系列数学上的等效变换，我们可以将基本块 1 至基本块 5 中原本包含乘除的浮点运算转化为了比较两个数的大小，这可以大幅度降低硬件资源的占用以及加速器的能耗。对于基本块 6，由于批量归一化层之后为全局累加池化层，没有$Sign(x)$函数供其融合，故其仅与 PReLU 融合为式（3.6）。

3.4.3 二值化算法实验

第二章的 2.3.2 节中介绍了近年来各种先进的二值化算法及其改进策略。对于参数在 30,000 级别的小型 CNN，我们更倾向于尝试不影响正常推理的改进方法，因此我们考虑了 Tang 等人提出的 PReLU 代替 ReLU[56]，Liu 等人在其论文中提出的分段函数 ApproxSign[58]，Qin 等人在其工作中提出的误差衰减估计器（Error Decay Estimator, EDE）[60]，Ding 等人设计的信息增强估计器（information enhanced estimator, IEE）[61]，Xu 等人考虑频域提出的频域近似估计器（frequency domain approximation, FDA）[62]，Ding 等人考虑激活值分布设计的正则化激活值的损失函数 DistributionLoss[63]，Lin 等人从权重的角度偏差考虑设计的算法 RBNN[64]，Xu 等人用来更新“死权重”的算法 ReCU[65]。表 3.6 为采取上述方法后的实验结果。

表 3.6 各种算法、优化方法对 BNN 准确率的影响

	5 分类模型准确率(%)	17 分类模型准确率(%)
Bi-Real Net[58]	96.25	97.00
IR-Net[60]	96.71	97.00
IE-Net[61]	96.90	97.00
FDA-Net[62]	96.90	97.00
Disloss-Net[63]	96.06	97.50
RBNN[64]	96.38	96.50
ReCU[65]	96.71	97.00

上述所有算法中，FDA-BNN 并未开源算法代码，而是开源了训练好的模型文件，因此我们复现时以 1 级傅里叶级数的梯度来替换$Sign(x)$的梯度（级数增高均会出现精度的明显下降）。

根据表 3.6，可见 IE-Net 可以让心律失常 5 分类 BNN 达到 96.90%的准确度，仅比全精度降低了 0.19%。对于心律失常 17 分类 BNN，精度总是在 96.50%~97.50%浮动，各种方法无法对其精度作进一步提升。我们称精度为 96.90%与 97.50%的模型为 BP（Better Performance）模型，他们的混淆矩阵如图 3-5 与图 3-6 所示。

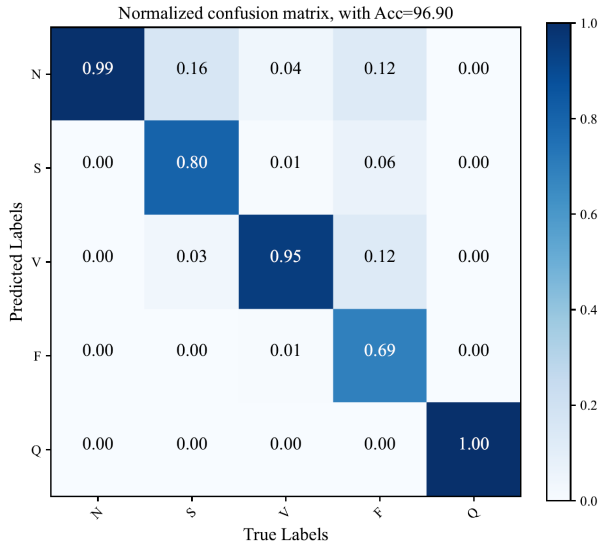

图 3-5 混淆矩阵（5 分类模型）

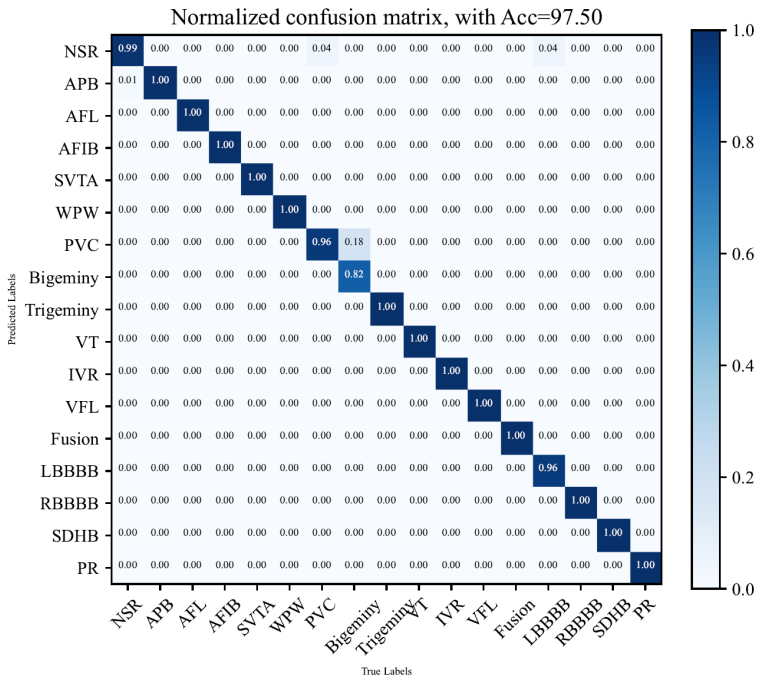

图 3-6 混淆矩阵（17 分类模型）

在基于 BNN 的各种研究中，往往对第一个基本块的卷积层和最后一个基本块的卷积层、全连接层不进行量化，以最大程度保留网络输入端与输出端的信息，获得较高的准确率。本课题的 BNN 不同于以往的设计，我们的 BP 模型二值化了第一个基本块卷积层的权重，并对最后一个基本块的卷积进行了完全二值化。也就是说，网络的所有卷积层皆为同或运算（第一个卷积层为多 bit 输入与 1bit 权重同或，其他则均为 1bit 输入与 1bit 权重同或）。但最为理想的 BNN 网络应是将输入数据也进行二值化，以达到所有卷积运算均可在硬件上体现为逐比特 XNOR+POPCOUNT，这也可以很大程度的降低第一个基本块的功耗。输入生理信号被二值化的模型称之为 LP（Low Power）模型。实验结果显示，即使在心电图信号被二值化为{+1, -1}，损失了大量的信息后，5 分类模型准确率仍可达到 91.60%，17 分类模型准确率则为 92.00%。可见我们的 BNN 具有极好的鲁棒性。

3.5 与近年轻量级 ECG 检测分类模型性能对比

表 3.7 从各个角度比较了所提出的 BP 模型与最先进的轻量级模型的性能。注意到由于缺乏对 ECG 的 17 分类轻量级模型的研究，为了保证客观公正的比较结果，我们在表 II 中包含了 Tuncer 等人的全精度工作[68]。由于页面大小限制，我们将未能在表 3.7 列出的剩余 3 个工作列于表 3.8。

表 3.7 模型性能对比 1

指标	本项目	Scrugil [45]	Sivapalan[46]	Sun[47]	Wu[49]
数据位宽 (bit)	1	8	16、32	混合精度	1
分类数目	5、17	5	2	17	4
准确率(%)	5 分类：96.9 17 分类：97.5	96.98	97	95.84	—
灵敏度(%)	5 分类：88.7 17 分类：97.9	95.35	87	94.19	—
特异度(%)	5 分类：98.5 17 分类：99.8	—	98	99.67	—
精确度(%)	5 分类：91.3 17 分类：96.4	85.17	87	—	—
F1 分数(%)	5 分类：89.9 17 分类：96.5	91.12	87	—	86.8
模型大小 (KB)	5 分类：3.92 17 分类：4.64	—	—	13.54	266.24

由表 3.7 与表 3.8 可得，在数据位宽方面，我们的模型为 1bit，在模型中权重相等的前提下，1bit 的位宽可以使模型大小较工作[45]小 8 倍，比工作[46]小 16 到 32 倍。

在 ECG 信号的分类数目方面，我们的模型可以在 5 分类与 17 分类之间切换，而[46]和[50]仅能进行心电图的二分类。

在评估方法方面，我们对模型使用了准确率、灵敏度、特异度等一共 5 种评估指标，弥补了工作[49]中的评估指标过少的缺点。更进一步，本设计提出的 5 分类模型特异度在 1bit 模型中最高，为 98.5%，17 分类模型的准确率、特异度的精密度在所有 17 分类模型中最高，分别为 97.5%、99.8%和 96.4%。

表 3.8 模型性能对比 2

指标	本项目	Wong [50]	Wang[51]	Tuncer[68]
数据位宽 (bit)	1	1	1	32
分类数目	5、17	2	5	17
准确率(%)	5 分类：96.9 17 分类：97.5	97.3	95.67	96.60
灵敏度(%)	5 分类：88.7 17 分类：97.9	91.3	94.8	98.51
特异度(%)	5 分类：98.5 17 分类：99.8	98.1	—	—
精确度(%)	5 分类：91.3 17 分类：96.4	86.7	96.2	95.18
F1 分数(%)	5 分类：89.9 17 分类：96.5	88.9	—	96.69
模型大小 (KB)	5 分类：3.92 17 分类：4.64	8.02	10.62	—

最后，我们的模型存储占用属于已知的最低大小，5 分类模型仅需 3.92KB 的存储空间，17 分类模型仅需 4.64KB 的存储空间，实现了显著的存储压缩比率，分别为 28.48 倍和 28.62 倍。

3.6 本章小结

本章详细介绍了实验所用的 MIT-BIH 数据集以及各种评估指标，接着介绍了全精度下的 ECG 检测算法设计，包括网络结构的探索以及超参数的实验等，在确定了全精度最优模型后，即开始二值化算法的设计与实验，首先对网络层进行了改良，将激活层替换为 PReLU，接着以此为基础，以不影响 BNN 前向推理为前提，进行了大量实验，分析了实验结果，确定了 ECG 检测网络的 BP（Better Performance）模型与 LP（Low Power）模型，并在第五节将 BP 模型的各项指标与近年最先进的轻量级模型作比较并作总结分析。

第四章 二值化神经网络硬件加速器结构设计

基于第三章得到的基于 BNN 的 ECG 检测网络模型，本设计针对其中的 Low Power 模型设计心律失常 5 分类 BNN 专用的硬件加速电路。本章将介绍所设计加速器的整体架构与数据流，接着介绍了数据存储设计，最后介绍 PE（Processing Element）阵列模块、控制模块、最大池化和阈值比较模块的设计，并以此为基础总结了加速器运算的整体流程。

4.1 加速器总体架构与数据流

将权重与激活为 32 位浮点数的 CNN 二值化为 BNN，虽然增大了前期的软件框架调试难度，但也降低了后期的硬件架构设计周期。BNN 硬件加速器无需考虑复杂的定点数运算，且可以将仅 4KB 左右的权重均存储于片上内存，无需考虑片外内存的存取操作。本该是多 bit 数的乘累加转变为 1bit 向量的逐位同或与位 1 计数，这也大幅度降低了实现卷积所需的硬件资源消耗。

4.1.1 加速器架构设计

本课题设计的加速器整体框架图如图 4-1 所示，为了更好地了解与分析每一个基本块的数据流，我们为每一个基本块编写了其各自独立的加速单元，并将它们互相连接，组成了最终的二值化神经网络硬件加速器。对于基本块 1，由于其输入为 3600 个值，我们将输入 ECG 数据进行了特殊的排布处理，并因此用到了 Output Buffer 来存储最大池化与阈值比较模块的输出，图 4-1 对基本块 1 的内部架构做了特写。对于其他基本块，基本块 2~基本块 6 与基本块 1 相比，将 ECG Buffer 换为了 Input Buffer，新增了 IB Addr Ctrl 模块，并去除了 Output Buffer；基本块 6 作为最后一个基本块，其结构与之前的有所不同，如图 4-2 所示：

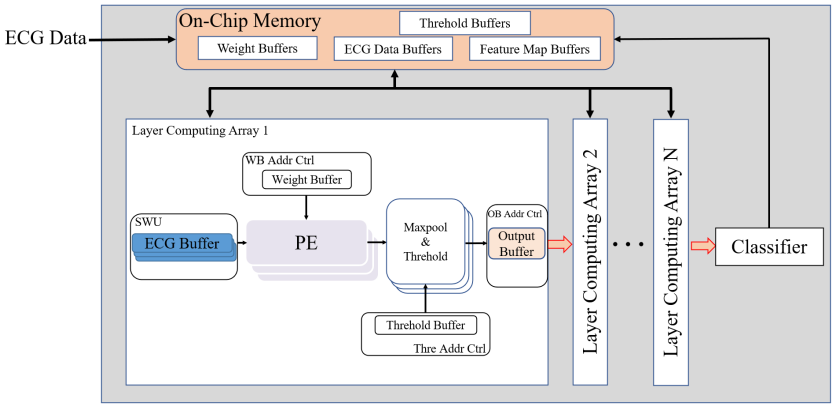

图 4-1 加速器整体框架图

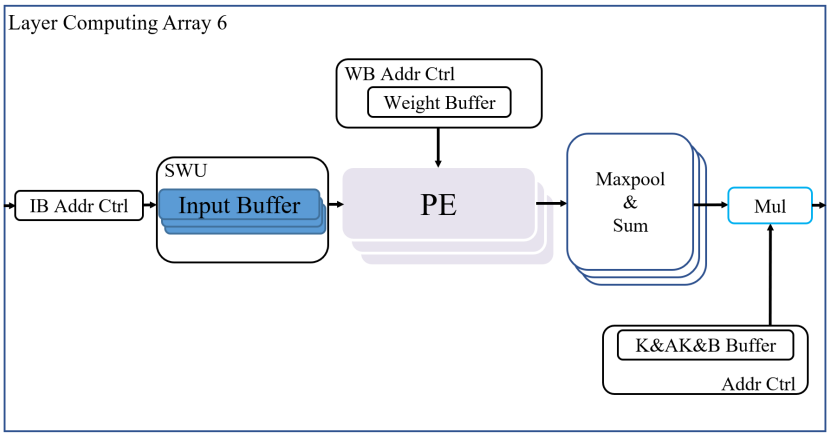

图 4-2 基本块 6 内部架构

加速器的各个模块功能如表 4.1 所示：

表 4.1 加速器主要模块及其功能表

模块	模块功能
ECG Buffer	有 4 个，用于存放被分为 4 段的 ECG 信号
Weight Buffer	存放其所在基本块的权重数据
Threhold Buffer	存放其所在基本块的阈值数据
K&AK&B Buffer	存放最后一个基本块完成式（3.6）计算所需的 k、ak、b
Output Buffer	仅基本块 1 含此模块，存储基本块 1 的输出特征图
SWU	本质上是地址控制器，将其内的 Buffer 存储的数据按特定顺序输出给 PE 阵列
WB/Thre Addr Ctrl	控制 Weight Buffer、Threhold 内存储的数据按序传输给 PE、比较模块
IB Addr Ctrl	将上一个基本块输出的 1bit 值整合为 32bit 后存储到 Input Buffer 中
PE	脉冲卷积运算阵列，计算 BNN 的卷积输出
Maxpool & Threhold	进行对 PE 阵列输出卷积结果的最大池化与阈值比较
Maxpool & Sum	对最后一个基本块的卷积输出进行累加
Mul	进行式（3.6）的乘法运算
Classifier	将基本块 6 的多个输出结果求其最大值，最大值索引即为 ECG 信号的类别

4.1.2 数据流分析

加速器整体运行流程如下：

在基本块 1 内，Weight Buffer 将权重传输给 PE 阵列，接着 4 个 ECG Buffer 将存储的

ECG 数据按序向 PE 阵列输入，PE 阵列将并行计算这些数据。由于 BNN 的乘法可被同或门代替，其具有低资源消耗性，因此我们采用的 PE 阵列为全映射，它的输出即为卷积输出，无须考虑部分和。卷积结果边计算边输出到 Maxpool&Threhold 模块内，进行最大池化，最大池化的结果一经输出，立即进行阈值比较，得出其对应的 1bit 值，1bit 的值累计达到 32 个后，就将这 32bit 值存入基本块 1 的 Output Buffer 或下一个基本块的 Input Buffer 存储，存储完毕后继续被当前基本块的 SWU 按序输入给 PE 阵列，以此往复，直到基本块 6，经由最大池化的数据通过相加与相乘降维，对于 5 分类加速器，经此操作仅剩 5 个数据，这 5 个数据仅由 Classifier 模块比较，得到最大值的索引，索引即为 ECG 信号的类别。

综上所述，加速器整体数据流为从 ECG Buffer 输出数据，经由 PE 阵列计算卷积结果，最大池化减少特征图数量，阈值比较将最大池化的输出量化为 1bit 后，传输至下一个基本块进行运算，直至输出分类结果。

4.2 数据存储设计

CNN 作为计算密集型任务需要处理大量的数据，有效的数据存储方案可以显著影响 CNN 加速器的性能和功耗。因此，在设计加速器时，数据存储设计非常重要。

4.2.1 ECG 数据存储设计

ECG 数据由 3600 个样本组成，在本设计的低功耗（LP）模型中，它们均为 0、1。因此，需 3600bit 的存储空间，通常做法是以 32bit 宽，深度为 113 的 BRAM 存储数据。

然而对于第一个基本块，根据式（2.1）与表 3.3 可得，输出特征图的每个通道有 1802 个数据，也就是说采取全映射需要至少 1802 个时钟周期才能计算得到所有输出特征图。但若利用 4 个相同的 PE 阵列并行计算，每个 PE 阵列计算输出特征图的 1/4，则仅需约 451 个时钟周期。因此 3600bit 的 ECG 数据可分为 4 份存储在 4 个 BRAM 中，并行输出数据给 PE 阵列以达到将基本块 1 的计算周期减少 4 倍的效果。

4.2.2 卷积权重存储设计

由于本设计的卷积核大小均为 7，因此对于一个输入通道为 N，输出通道为 M 的基本块，其存储权重需要$N \times M \times 7\ bit$，由表 3.3 可得卷积权重存储所需 bit 数，其具体数据如表 4.2 所示。无论哪一个基本块，其N与M总有一个为 4 的倍数，故每个基本块的权重均为 28 的倍数，我们可用宽度为 28bit 的 BRAM 存储权重。

表 4.2 权重存储空间分析

结构	权重大小(bit)	Width×Depth
基本块 1	1×8×7	28×2
基本块 2	8×16×7	28×32
基本块 3	16×32×7	28×128
基本块 4	32×32×7	28×256
基本块 5	32×64×7	28×512
基本块 6	64×5×7	28×80
合计		28×1010

4.2.2 阈值存储设计

根据第三章式（3.6）~式（3.10）的分析，在基本块 1~基本块 5，模型需要存储的阈值为$\delta_+ = \frac{-b}{k}$，$\delta_- = \frac{-b}{ak}$，另外还需记录每一个δ对应的ak、k的正负值。对于式（3.10）的输入，其均为被最大池化后的卷积结果，而卷积是通过逐比特同或与位 1 计数后，由第二章介绍的公式$Result = 2 * p - k$得来的，其计算的结果一定为整数。本设计的软件部分均在 PyTorch 深度学习框架下完成，所有的参数，如b、ak、k均为 32 位浮点数，浮点数的精度极高，经实验统计，b、ak、k均为小数，即使数值极低，也只能无限接近于 0，如1×10^{-15}，不会出现绝对为 0 的数。因此，阈值一定为小数，若将他们存储在硬件中，通常的做法是将量化为 INT8 的整数或者 8bit 定点数存储于硬件中，再编写量化与反量化模块，以让最大池化的输出能顺利与量化后的阈值比较，并输出正确的 0、1。

阈值比较本是将复杂的批量归一化计算融合到其他函数，以简化硬件设计。在 BNN 加速器中，若引入量化反量化模块对作为小数的阈值进行额外处理，无疑又复杂化了硬件设计，因此本课题提出的优化方案如下：

基于上文讨论，与小数阈值作比较的均为整数，例如：-1、0、1，假设阈值δ为-0.239，根据式（3.9）的$Sign_\delta(x)$, 大于阈值δ的应为 1，小于阈值δ的应为 0，它们在数轴上的位置关系如图 4-3 所示：

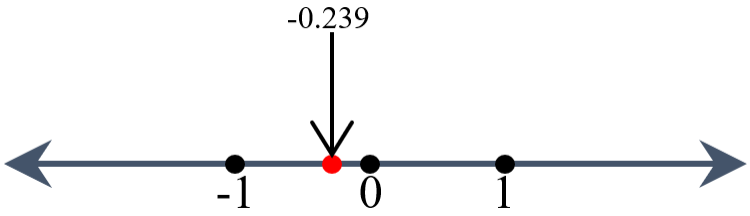

图 4-3 阈值比较示意图

阈值存储优化策略的第一步是将阈值向上取整后减去0.5，使其位于距离自己最近的两个整数正中间。由图 4-3 可见，无论-0.239 的值在-1~0 之间如何变化，它与-1、0、1 的大小关系都是固定的，也就是说，$Sign_{\delta}(x)$的输出并不会被影响。在本例中，-0.239 将变为-0.5。无论阈值精度有多高，小数点后有多少位，其均可被化为$x.5$的格式，x代表阈值的整数部分。

阈值存储优化策略的第二步是将所有变为$x.5$格式的阈值乘以 2。硬件中的$Sign_{\delta}(x)$函数的本质即为$x \geq \delta$ 为 1，小于为 0。这与$Sign_{2\delta}(2x)$是等效的。以-1 与-0.239 比较为例，他们的比较过程可等效演化为式（4.1）。

$$-1 < -0.239 \Leftrightarrow -1 < -0.5 \Leftrightarrow -2 < -1 \tag{4.1}$$

阈值存储优化策略的第三步是考虑k、ak的正负导致$Sign_{\delta}(x)$变为$-Sign_{\delta}(x)$的问题。由第二章的讨论，$\delta_+ = \frac{-b}{k}, \delta_- = \frac{-b}{ak}$。若$k$为负，则（3.10）中的$Sign_{\delta_+}(x)$ 需变为$-Sign_{\delta_+}(x)$；若ak为负，则$Sign_{\delta_}(x)$ 需变为$-Sign_{\delta_}(x)$。因此，可提前在软件内确定k、ak的正负，若正则记为 1，负则记为 0，与其对应的阈值一起存储于硬件。

综上所述，式（3.10）中的$Sign_{\delta_+}(x)$与$Sign_{\delta_}(x)$的阈值最终被转换为整数，经 Python 中转为多 bit 数后，与代表k、ak正负的 0、1 拼接，最后以 Xilinx 软件可识别的 coe 格式存储。阈值存储策略如图 4-4 所示：

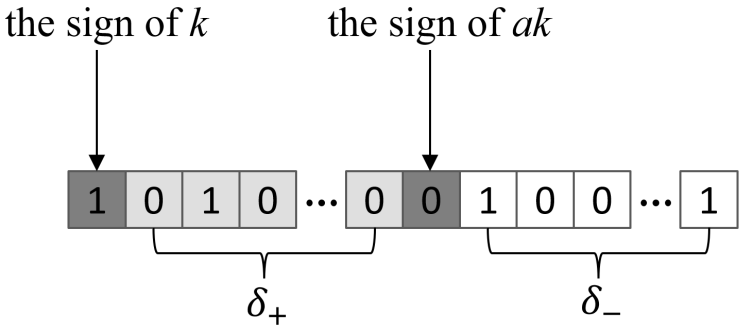

图 4-4 阈值存储示意图

经简单的实验分析，确定各个基本块的阈值统一转化为 12bit 整数即可涵盖所有阈值，

以图 4-4 所示方式存储阈值，2 个阈值需 24bit 的存储空间，又每一个输出通道对应 2 个阈值，因此阈值存储空间分析如表 4.3 所示。

表 4.3 阈值存储空间分析

结构	输出通道数	Width×Depth
基本块 1	8	24×8
基本块 2	16	24×16
基本块 3	32	24×32
基本块 4	32	24×32
基本块 5	64	24×64
合计		24×152

4.2.3 基本块 6 的k、ak、b存储设计

根据 3.4.2 节末的讨论，因为基本块 6 的批量归一化层后面是全局累加池化层与比较层，无$Sign(x)$函数供其融合为上一节所述的阈值格式。因此最大池化层的输出特征图各个元素均需经过式（3.6）计算。由表 3.4，基本块 6 内经最大池化层输出的张量维度为（5，27），这 5 个通道的 27 个值经过 PReLU 激活层与批量归一化层后，经由全局最大池化在最后一维相加，降维至（5），再经由比较层比较，得出最大值索引后，将索引值作为预测标签输出。以上过程中，PReLU 层与批量归一化层融合为（3.6）式，接着通过全局最大池化层在最后一维相加，此三层可通过伪代码 4.1 融合如下：

伪代码 4.1：

$$\boldsymbol{ge\_sum} = Sum(relu(\boldsymbol{x}), dim = -1)$$

$$\boldsymbol{le\_sum} = Sum(-relu(-\boldsymbol{x}), dim = -1)$$

$$\boldsymbol{Y} = \boldsymbol{ge\_sum} \times \boldsymbol{k} + \boldsymbol{le\_sum} \times \boldsymbol{ak} + 27 \times \boldsymbol{b}$$

其中，ge_sum表示维度为（5，27）的张量x中，5 个通道内 27 个值中所有大于 0 的元素之和。le_sum则表示每个通道中所有小于 0 的元素之和。Y、k、ak、b、ge_sum、le_sum皆为形状为（5）的张量。

综上所述，在硬件实现中，可通过判断最大池化层输出结果的符号位，将大于 0 的数与小于 0 的数分别累加，供每个通道的k、ak相乘。本加速器于基本块 6 有 5 个通道，每个通道都有其各自的k、ak、b，因此需要存储 15 个值，它们皆为浮点数。因为这里无法和上一节一样，将浮点数的小数位转为 0.5 并乘以 2 转化为整数，因为计算结果Y的最大值

索引即为预测类别，若随意的对乘数、被乘数进行取整等操作，会引起结果相对大小的变化，进而直接影响预测类别。

通过观察伪代码 4.1 的第三行，对于 5 分类网络来说，$ge_sum \times k + le_sum \times ak$共需 10 次乘法，$27 \times b$则完全可以通过移位、累加完成。为 10 次乘法专门设计量化、反量化模块不符合设计加速器时硬件友好的初衷，又由网络数据流可知，我们仅仅需要的是确定Y的 5 个值中哪一个最大，而不需要准确的浮点乘法结果，因此应争取将浮点乘法转为由 FPGA 内部的 10 个 DSP 完成的整数乘法。故对基本块 6 的k、ak、b存储设计优化策略如下：

将k、ak、b共 15 个值作为一组，将他们以相同的尺度因子放大为n bit 整数，n 值可由实验确定。当这 15 个值被相同的尺度因子放大后，他们的相对大小不会发生变化，也就不会影响到Y中最大值的索引，即网络的预测类别。实验结果如图 4-5 所示，当 n 为 14 时准确率不会有任何损失，为 91.60%。

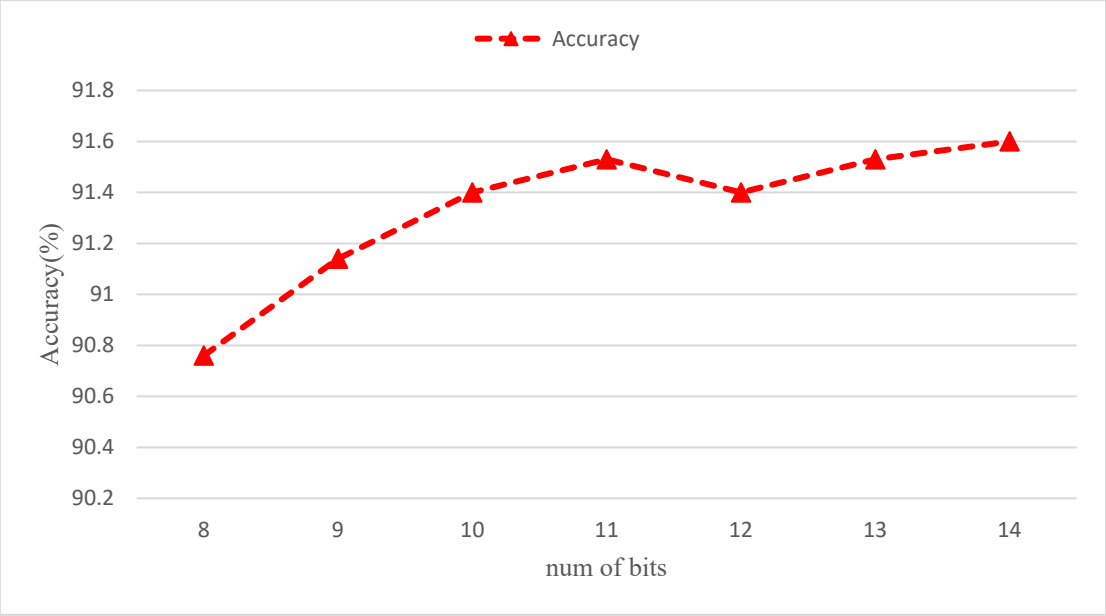

图 4-5 比特数对于准确率的影响

因此，基本块 6 的k、ak、b存储设计为将其量化为 15 个 14bit 整数，因仅有 15 个值，可直接作为基本块 6 的内置变量存储。

4.3 PE 阵列设计

本设计的 PE 阵列采用脉动阵列来实现卷积运算。在硬件加速器中，脉动阵列（SA）是一个流水线的二维 PE 阵列，具有非常高效的本地数据移动能力。它非常适合加速卷积

运算，并且在工业界广泛部署。脉冲阵列具有如下优点：

（1）高速计算：卷积运算可以通过并行处理的方式在一次计算中完成，因此其相较于软件而言，可以更加快速的实现卷积运算。

（2）低功耗：由于脉冲阵列可以可利用内置的寄存器固定权重，并将输入逐级传递，达到数据复用的效果，可极大降低数据存取引起的功耗。

（3）可拓展性：脉冲阵列可以增添行数或者每一行的运算单元数来提高处理的数据规模，实现处理更多输入输出通道的数据。

脉冲阵列由多个相同的处理单元（PE）组成，每一个 PE 计算上游 PE 传递给他们的数据，并存储或输出计算结果，同时将上游传递给自己的数据又接着传递给下游 PE，以实现流水线运行。

4.3.1 处理单元（PE）设计

如上文所述，脉冲阵列由多个相同的 PE 构成，因此 PE 是脉冲阵列中最为重要的部分。基于 BNN 的硬件友好性与式（2.12），本课题设计的 PE 结构如图 4-6 所示，结构图中只列出了部分较为重要的数据信号，而时钟信号 clk、复位信号 rst_n 等则被隐去。

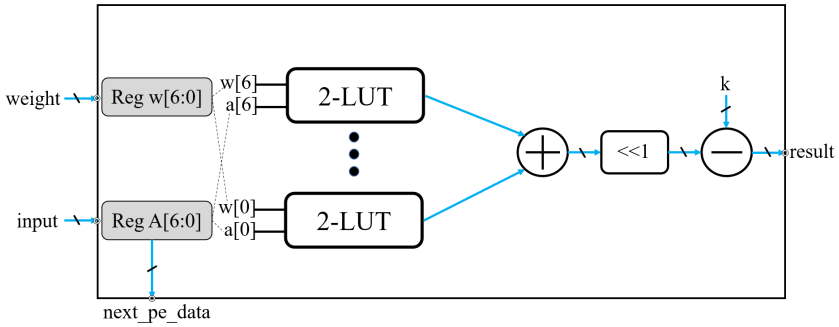

图 4-6 PE 结构图

PE 内的数据流为：第一行的 PE 接收 SWU 的输出并计算，其余每个 PE 接收上一行 PE 给它的数据，在 7 个二输入查找表（2-LUT）中将输入与权重逐比特同或后，计算所有结果中 1 的数目，根据式（2.12），通过右移一位将此结果乘以 2，再减去卷积核的大小k，所得即为计算结果$result$。计算完成后，PE 将保存在 Reg A 中的数据传给下一个 PE 令其计算，以此往复直至所有卷积运算结束。

4.3.2 PE 阵列数据流

由于本设计的加速器为全映射加速器，因此 PE 的数目等于输入通道×输出通道。假设

某一卷积层的输入通道为 4，输出通道为 8，那么此卷积层的 PE 阵列如图 4-7 所示：

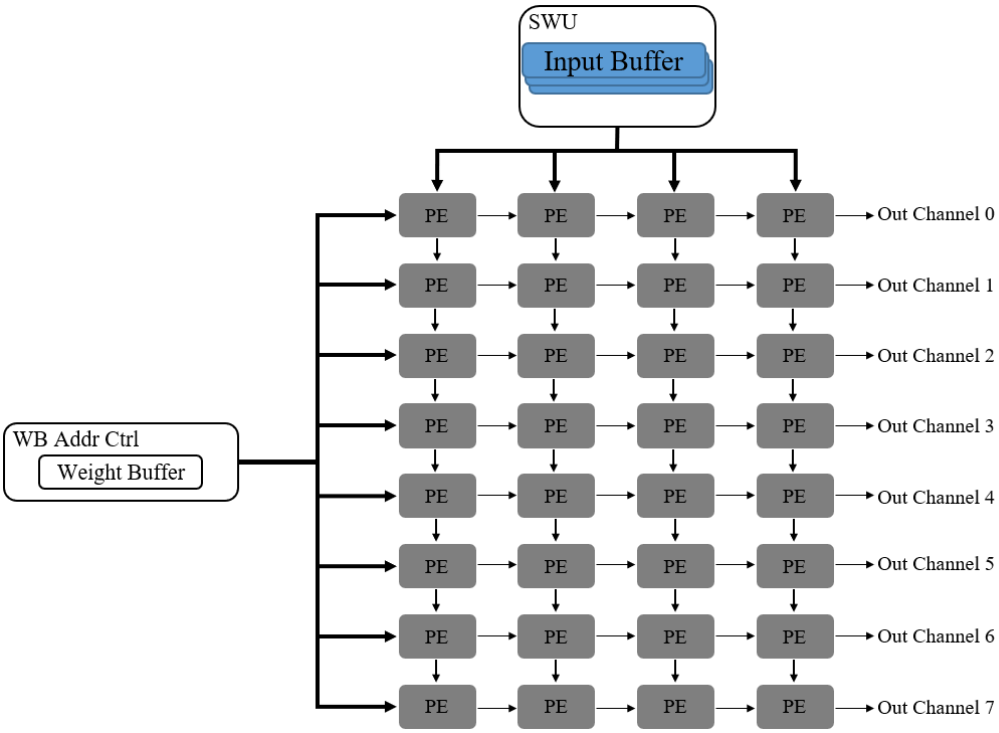

图 4-7 PE 阵列示意图

首先需要明确的是 4×8 PE阵列的分工。对于输入通道为4，输出通道为8的卷积层，其权重共有$8 \times 4 \times 7$ bit，即有 8 个卷积核，每个卷积核有 4 个权重，每个权重有 7bit。PE 阵列每一行负责一个卷积核的计算，每一列负责一个输入通道的计算。

此 PE 阵列的数据流为：SWU 将 Input Buffer 内存储的 4 个通道的值以规定步长和数量逐时钟周期取出，将其传输给第一行的 4 个 PE。第一个时钟周期，输出通道 0 的所有部分和计算完成，它们相加得到输出通道 0 的第一个特征值，而使用结束的输入数据传给下一行的 PE；第二个时钟周期，第一行的 PE 接收新的数据，计算输出通道 0 的第二个特征值，第二行的 4 个 PE 则计算输出通道 1 的第一个特征值，计算结束后继续将数据向下游传输……以此往复，直至所有输出通道的结果计算完成。由此设计的 PE 阵列固定了权重，以流水线运作，不断复用 SWU 的输入数据，最大程度减少了对权重缓存与输入缓存的读取次数，降低访存功耗的同时也减少了计算延迟。

4.3.3 加速器算存分析

（1）计算量

CNN 的计算量主要由卷积层与全连接层构成，它是评估模型性能，设计加速器的重要

指标。在确定了 CNN 模型具体参数后，计算量也随之固定，对于一维卷积网络的全连接层、卷积层，它们的 MACs 分别由式（4.2）与式（4.3）计算，MACs 是一层卷积或全连接执行乘累加的次数。

$$Conv_{MAC_s} = C_i \times K \times W_o \times C_o \tag{4.2}$$

$$FC_{MAC_s} = C_o \times C_i \tag{4.3}$$

上面两式中，C_i是输入通道，C_o是输出通道，K是卷积核大小，W_o是输出特征图的宽，本设计的加速器仅使用了卷积层，故其 MACs 可统计如表 4.4：

表 4.4 加速器 MACs 统计

结构	MACs
基本块 1	100912
基本块 2	808192
基本块 3	1619968
基本块 4	1627136
基本块 5	1648640
基本块 6	132160
合计	5937008

（2）访存次数

通常，CNN 于硬件上实现需要进行切片操作，将输入通道、输出通道分为T_n、T_m份，并将输出特征图也进行分片，在一维 CNN 中，可分为T_w份。这是因为加速器的 PE 阵列通常处理的是多 bit 定点数，执行多 bit 乘法的硬件资源有限，PE 阵列规模受限，导致仅能使用小部分权重与激活计算生成对应输出特征图的部分和后进行实时存储，待所有部分和计算完毕后累加得最终结果。基于切片操作，卷积的输入数据（即激活值）访存量可由式（4.4）计算：

$$Memacc_{activation} = ceil\left(\frac{C_o}{T_m}\right) \times ceil\left(\frac{C_i}{T_n}\right) \times ceil\left(\frac{W_o}{T_w}\right) \times T_n \times (T_w + K - 1) \tag{4.4}$$

上式中的 ceil 函数为向上取整。卷积层的权重访存量可由式（4.5）计算：

$$Memacc_{weight} = ceil\left(\frac{C_o}{T_m}\right) \times ceil\left(\frac{C_i}{T_n}\right) \times ceil\left(\frac{W_o}{T_w}\right) \times T_m \times T_n \times K \tag{4.5}$$

接着即为输出特征图的访存次数，它可由式（4.6）计算：

$$Memacc_{out} = ceil\left(\frac{C_o}{T_m}\right) \times ceil\left(\frac{W_o}{T_w}\right) \times T_m \times T_w \tag{4.6}$$

对于采取分片策略的加速器，即可根据式（4.4）、（4.5）、（4.6）求得激活值、权重、输出特征图的访存次数，进而求出总访存次数。对于 BNN 加速器，因其 PE 无需用到在 FPGA 里资源较少的 DSP，运算仅为逐比特的同或、计数、移位与减法运算，耗费资源较少，本设计采取的全映射策略，可最大程度减少访存次数，以此来从访存层面降低功耗。

全映射下无需做任何分片，因此与含$ceil$的各项均为 1，以基本块 4 为例，若加速器内每个基本块的 PE 阵列固定皆为4×8，则相当于令$T_n = 4$，$T_m = 8$。式（4.4）的分母T_m无法约去，故增加了$\frac{C_o}{T_m} = 4$倍的激活值访存量；同理，式（4.5）内的T_w无法约去，访存量进一步激增；输出访存量则固定，一直为$C_o \times W_o$。

结合上文，所设计的全映射加速器访存次数可统计如下表：

表 4.5 加速器访存次数统计

结构	激活值访存	权重访存	输出访存
基本块 1	904	56	7184
基本块 2	3632	896	7168
基本块 3	3664	3584	7136
基本块 4	3744	7168	3552
基本块 5	1952	14336	3520
基本块 6	2112	2240	135
合计	16008	28280	28695

4.4 控制模块设计

加速器结构为每个基本块、每个模块均单独设计，它们在运算到合适时间产生相应使能信号令下一个模块开始运行，加速器的控制模块主要由 IB Addr Ctrl、SWU 以及 WB/Thre Addr Ctrl 这类地址控制单元构成。

4.4.1 IB Addr Ctrl

IB Addr Ctrl 模块存在于基本块 2~基本块 6 内，负责将上一个基本块中输出的 1bit 特征图收集 32 个，而后输出 Input Buffer 的使能信号、地址以及存储的 32bit 数等信息，送

入 Input Buffer 存储。Input Buffer 实际为单口 RAM，它的输入为地址（ADDR）、数据（DIN）、读写信号（WE）和使能信号（EN），EN 为 1 则可以运行，WE 为 1 就将 DIN 存储于地址为 ADDR 的空间，为 0 就将地址为 ADDR 的空间内存储的数据输出到端口 DOUT。以基本块 2 为例，其 IB Addr Ctrl 模块结构图如图 4-8 所示：

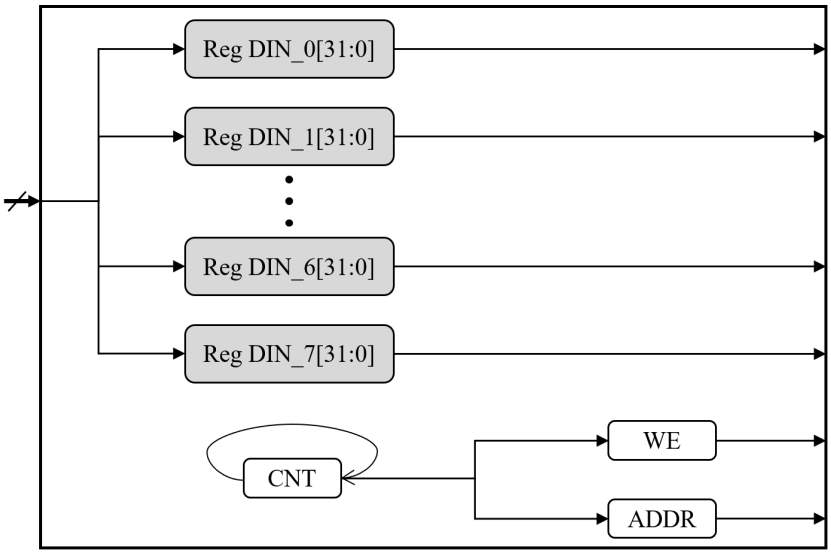

图 4-8 IB Addr Ctrl 模块结构图

上图中，DIN_x 皆为移位寄存器，上一个基本块的输出特征值逐比特移入 DIN 寄存器，CNT 不断计数，当计数满 31 时，DIN_x 输出，ADDR 加 1，WE 跳变为 1，输出的数据存入对应的 Input Buffer 中。因为 IB Addr Ctrl 模块仅控制 Input Buffer 的写入，当 WE 为 1 时，EN 也为 1，即二者是相同的，故图 4-8 中隐去了 EN 信号。

4.4.2 SWU

SWU 在基本块 1 的主要功能即为将 ECG 数据按照步长为 2，窗口大小为 7 输出至 PE 阵列，其功能可由伪代码 4.2 实现。

伪代码 4.2：

1 $Slidedata = data[31 - 2 \times cnt \ -: 7]$

$Slidedata$即为要输入给 PE 阵列的数据，$data$为从 ECG_Buffer 中的一个地址取出的 32bit 数，cnt为一计数器，等号右边表示从$31 - 2 \times cnt$计算所得的 bit 位起，取 7bit 的数据。因此，cnt每增加 1，就相当于以步长为 2，取了 7bit 数据赋值于$Slidedata$，$Slidedata$则会立马传入 PE 阵列进行计算。在基本块 1 之所以不用移位寄存器是因为基本块 1 的卷积步长为 2，用移位寄存器会导致 PE 阵列相隔一时钟周期的输出才有效，会较大程度影响

PE 阵列的流水线计算效率。对于基本块 2~6，SWU 则是使用一个 38bit 的移位寄存器来实现$Slidedata$的逐时钟周期输出，其结构如图 4-9 所示：

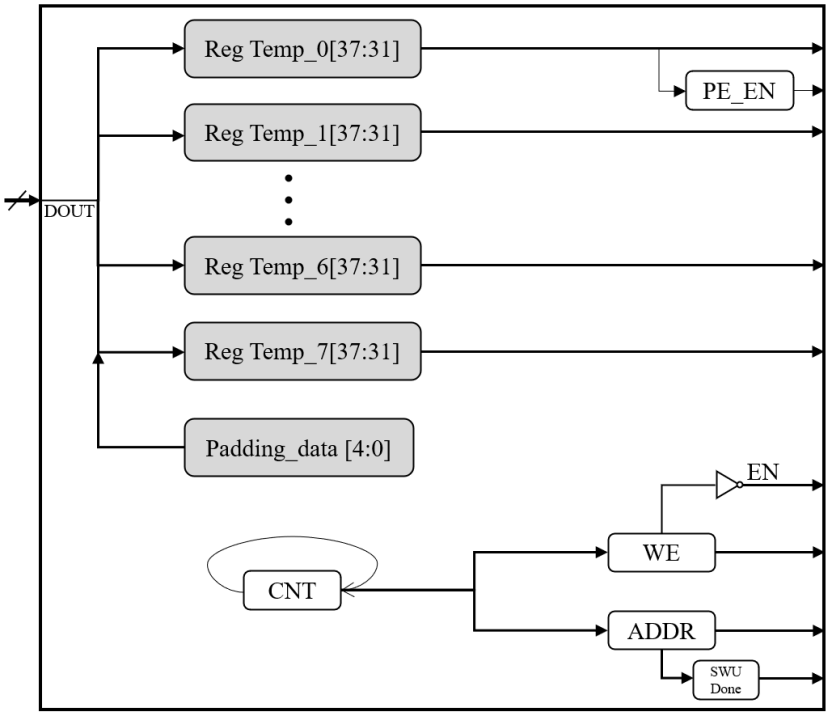

图 4-9 SWU 结构图

SWU 的输入是 Input Buffer 的输出，为 32bit 的值，当不断地取这 32bit 中的 7bit 值时，因位于末尾的 6bit 的值要与下一个 Input Buffer 地址中的 32bit 值组合才能得到完整的 7bit $Slidedata$，故有了图中的 Temp_x，Temp_x 寄存器共$32 + 6 = 38$ bit，存储的是上一个地址的末 6bit 值与本地址的 32bit 值。$Slidedata$则是取 Temp_x 靠近 MSB 的 7bit 值，即第 37 位到第 31 位，当$Slidedata$输出给 PE 阵列后，Temp_x 便可以左移 1bit，因此下一个时钟周期的 Temp_x[37:31]即相当于步长为 1 的卷积输入，一旦当$Slidedata$开始输出，则 PE_EN 跳变为 1，示意 PE 阵列可开始运行。以此往复，当 Temp_x 寄存器内的有效值只剩 6bit 时，WE 为 0，EN 为 1，ADDR 加 1，表示从 Input Buffer 的下一个地址取特征图，接着这 32bit 的新特征图与剩余的 6bit 拼接，存入 Temp_x，循环往复直至 ADDR 为最终地址，所有$Slidedata$输出完毕后，输出 SWU Done 信号。

4.4.3 WB/Thre Addr Ctrl

WB Addr Ctrl 与 Thre Addr Ctrl 模块功能相同，都是从 Weight Buffer 与 Threhold Buffer 中读取数据，并将数据提前存入 PE 阵列与阈值比较模块，以供模块正常运行。以基本块 2

为例，其 WB Addr Ctrl 模块如下图：

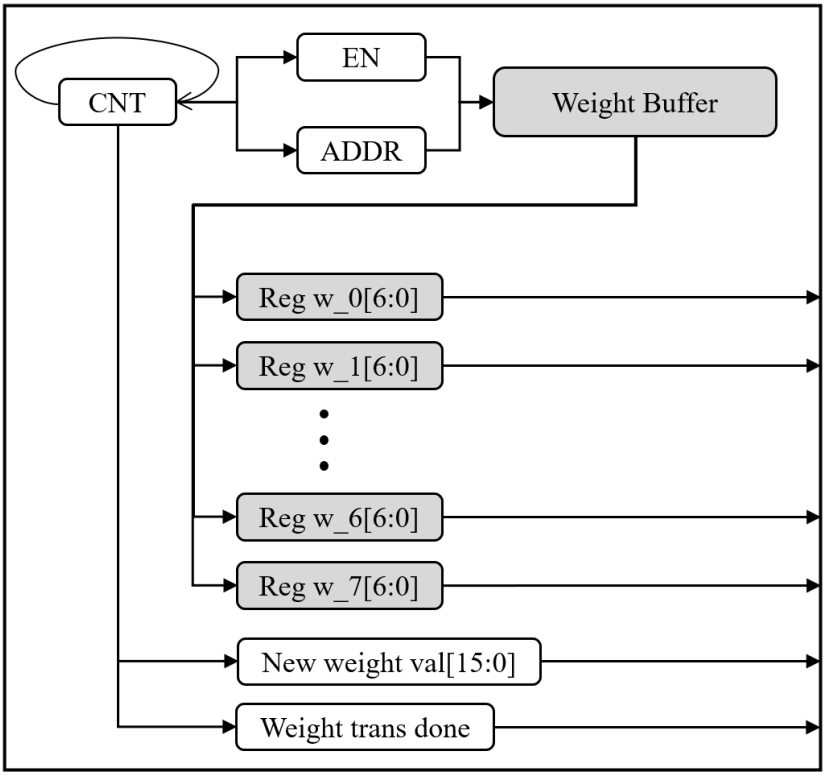

图 4-10 WB Addr Ctrl 模块

Weight Buffer 每一个地址仅存有 4 个 weight 的数据，而基本块 2 的 PE 阵列为8×16，即第一行要 8 个 weight，因此，第一个时钟周期读取的数据存入 w_0~w_3 内，第二个时钟周期读取的数据存入 w_4~w_7 内，此时 New weight val 信号的第 0 位立即为 1，以此示意 PE 阵列第 0 行的 8 个权重已准备就绪并开始传输。以此往复，直至 New weight val 寄存器内所有值均为 1，代表 PE 阵列的所有权重已传送完毕，Weight trans done 由 0 跳变为 1。

Thre Addr Ctrl 模块实际并未单独存在，其图中的相关信号与模块均位于 Maxpool&Threhold 模块中，它的结构与 WB Addr Ctrl 模块类似，每个地址的阈值从 Threhold Buffer 读出后送入阈值比较模块，初始化其内的阈值寄存器，将基本块 2 的阈值控制信号抽离出来可作 Thre Addr Ctrl 模块结构图如 4-11 所示：

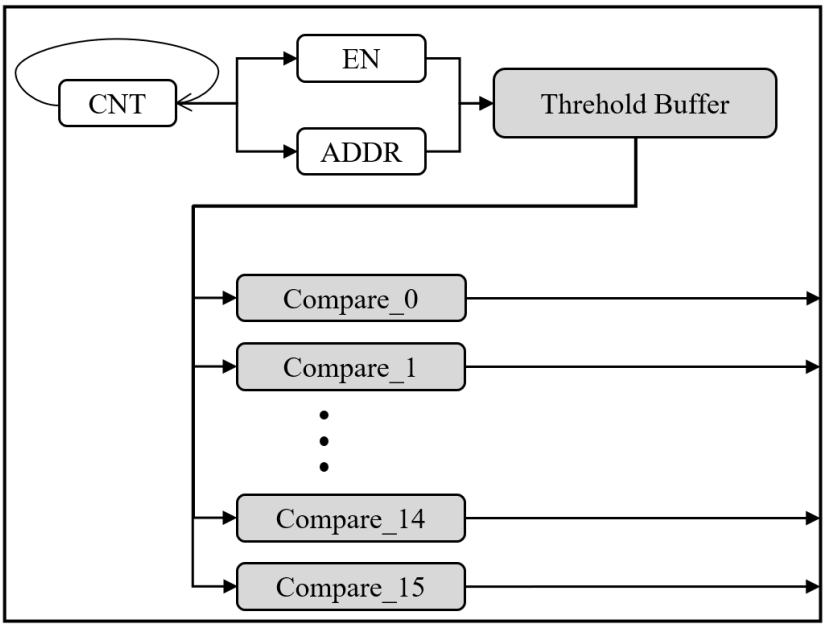

图 4-11 Thre Addr Ctrl 模块

基本块 2 有 16 个输出通道，每一个通道对应一个地址内的 24bit 阈值，ADDR 递增，遍历结束 Threhold Buffer 内的所有数据并存入对应的 16 个 Compare 模块后，EN 为 0，此模块功能结束。

4.5 最大池化与阈值比较模块设计

因 PE 阵列可直接计算出输出通道的所有部分和并相加得到输出通道特征值，而最大池化的池化窗口统一为 7，步长统一为 2，故每个基本块的最大池化模块可以统一，以输出通道数为 8 举例，其最大池化与阈值比较模块结构如图 4-12 所示：

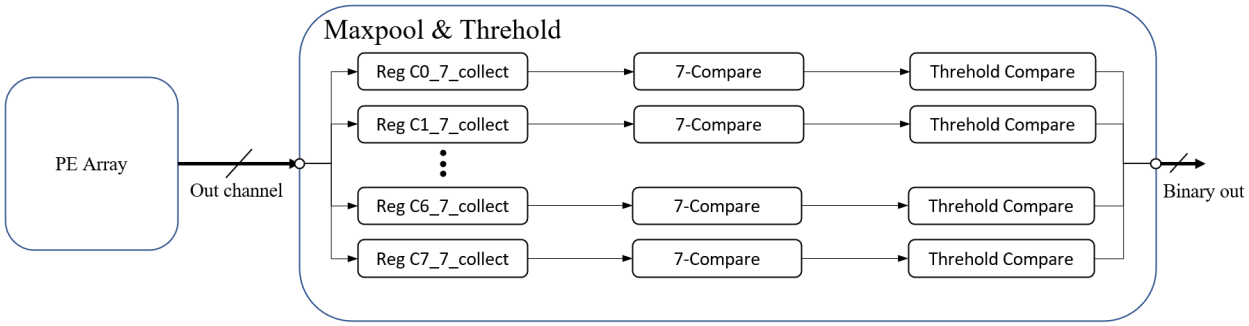

图 4-12 最大池化与阈值比较模块示意图

因为 PE 阵列的输出特征图为 n bit 整数，n 与每一行 PE 的个数有关。若直接使用 $7 \times n\ bit$ 输入的比较器，则扇入过大，会造成面积、功耗、延迟的增加。因此 7-Compare 模块的设计策略为：使用 3 个 3 输入比较器。前两个三输入比较器比较得出 6 个值的 2 个

最大值，接着这两个最大值与剩余的 1 个值进行比较，得到 7 个输入的最大值。

上述模块的数据流为：使用一个寄存器不断存储 PE 阵列逐时钟周期计算得到的特征值，待存储 7 个后，使用比较模块来得到最大值，接着此值输入阈值比较模块进行比较，得到 1bit 的输出。最大池化与阈值比较模块是以流水线运行的，最开始的寄存器采用移位策略，将存储的 7 个值送入比较模块后，将靠近 MSB 的 1 个特征值移出，接收 PE 阵列的下一个特征值至寄存器并继续将寄存器内的所有特征值向比较模块传递，持续得到最大值并经由阈值比较模块得到 1bit 输出，以此往复，每个时钟周期均有被量化完毕的 1bit 值输出。若直接将所有时钟周期的 1bit 输出存储，这实际是池化步长为 1 的二值数据。要得到步长为 2 的二值化结果，需存储每隔一个时钟周期的输出数据，这仅需令 RAM 的写使能信号每隔一个时钟周期有效即可。

4.6 加速器运算过程

前几节从整体到局部介绍了加速器的架构与各个模块设计，本节将以前面小节的设计为基础，详细介绍加速器的整个运算流程。

加速器的输入仅有时钟信号与复位信号，当复位信号失效后， 基本块 1 开始运行：WB Addr Ctrl 模块开始向 PE 阵列输出权重，输送完毕即发出信号令 SWU 模块使能信号跳变为 1；ECG 数据存储于基本块 1 的 Input Buffer 中，其内有 4 个 ROM，将 ECG 数据分为 4 段存储，SWU 的使能信号为 1 后，4 个 ROM 的数据以步长为 2，大小为 7bit 逐个时钟周期并行输出，当数据开始输出时，SWU 内的 PE 使能信号由 0 变为 1；基本块 1 的 PE 阵列本应为1×8，为了加速运算，特令其为4×8，每一列接收一个 ECG ROM 输出的数据，每一行计算出 4 个特征值，这些特征值为 4bit 有符号整数，每 8 个拼接为 32bit，存入基本块 1 独有的 Output Buffer 内，Output Buffer 内有 8 个 RAM 来分别存储每个通道的特征值，存入结束后使能 Maxpool&Threhold 模块，将 Output Buffer 内的数据读出，每 7 个比较出最大值，接着最大值经由阈值比较模块，被量化为 1bit 的 Binary_x，模块内的二值化输出有效信号也跳变为 1，基本块 1 的输出 Binary_0~Binary_7 并行输出至基本块 2。

基本块 2 接收到基本块 1 的 Binary_0~Binary_7 与二值化输出有效信号后，二值化输出有效信号使能 IB Addr Ctrl 与 Thre Addr Ctrl 模块，IB Addr Ctrl 模块开始收集基本块 1 的输出特征图并将其存入基本块 2 的 Input Buffer 之中，Input Buffer 内有 8 个单口 RAM 来存储基本块 1 的输出；Thre Addr Ctrl 模块在一开始便将 Threhold Buffer 内存储的阈值提前传给阈值比较模块；Input Buffer 存储结束后发出使能信号，令 WB Addr Ctrl 模块将存储的权

重数据传输给8×16的 PE 阵列；传输完毕后，WB Addr Ctrl 模块使能 SWU 模块，SWU 模块控制 Input Buffer 输出大小为 7bit，步长为 1 的数据给 PE 阵列，PE 阵列有 16 行，每一行计算出基本块 2 的一个特征图；当 PE 阵列开始输出特征图，此时其内 Maxpool&Threhold 模块的使能信号跳变为 1，Maxpool&Threhold 模块也因此开始工作，以步长为 2，窗口为 7 来比较最大值并通过阈值函数输出二值化后的特征图 Binary_0~Binary_15 与二值化输出有效信号。

基本块 3 收到基本块 2 的特征图与有效信号后，其各个模块执行顺序与基本块 2 完全一致。至此，数据不断传播，直到基本块 6。

基本块 6 内的 Maxpool&Threhold 模块变为了 Maxpool & Sum&Mul 模块，其每个通道的数据被最大池化模块输出后，逐个根据条件累加为 10 个有符号整数值，接着与存储的k、ak、b按照 4.2.4 内的方式相乘相加，得到 5 个有符号整数。最后，将这 5 个有符号整数作比较，得到最大值的索引即为 ECG 数据的类别。至此，加速器运行结束。

4.7 本章小结

本章首先介绍了加速器的总体架构以及其总体数据流，然后介绍了数据的存储方式以及基于硬件优化而考虑的数据处理策略与实验，接着详细介绍了硬件加速器中最为重要的 PE 阵列的设计，举例并详细介绍了其数据流并做了存算分析，最后介绍了控制模块、最大池化与阈值比较模块的硬件结构、设计思路以及加速器的整体运算流程。

第五章 仿真与结果分析

基于前面章节的讨论，本章对本课题提出的基于二值神经网络的 ECG 分类加速器中各个模块进行了功能验证，最后通过 vivado 综合出来的模型对加速器的相关指标，如功耗、模型大小等进行了分析。

5.1 功能仿真

根据上一章的讨论，加速器所用到的所有权重、数据均可妥善存储于 BRAM 中；BNN 中的批量归一化层、激活层、全局最大池化层可以融合，比较层功能单一、代码简单，故它们均可一起归纳于基本块 6 中。因此，本加速器的测试平台结构如图 5-1 所示，测试台文件仅产生时钟信号与复位信号，并接收由基本块 6 输出的分类结果。

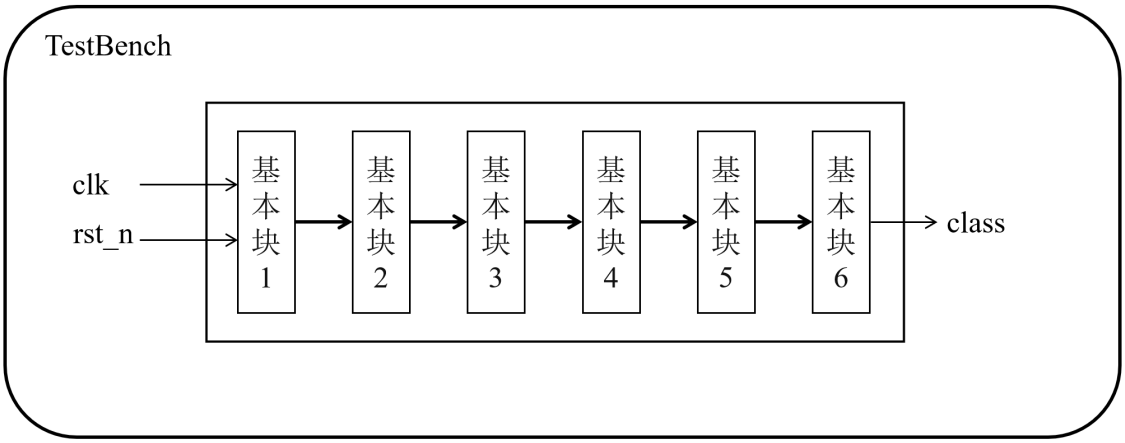

图 5-1 加速器测试平台结构

5.1.1 Weight_Control 功能仿真

每个基本块内均有 PE 阵列，它们接收 SWU 的输出并对其执行卷积操作，而这一切的前提是 PE 阵列内早已存储了有效权重。因此，Weight_Control 模块的功能为将自己所在基本块的所有卷积权重传输给 PE 阵列的每个单元。基本块 1 内仅有$1 \times 8 \times 7bit$的权重，波形图较为单调，故截取基本块 2 的波形图如图 5-2 所示：

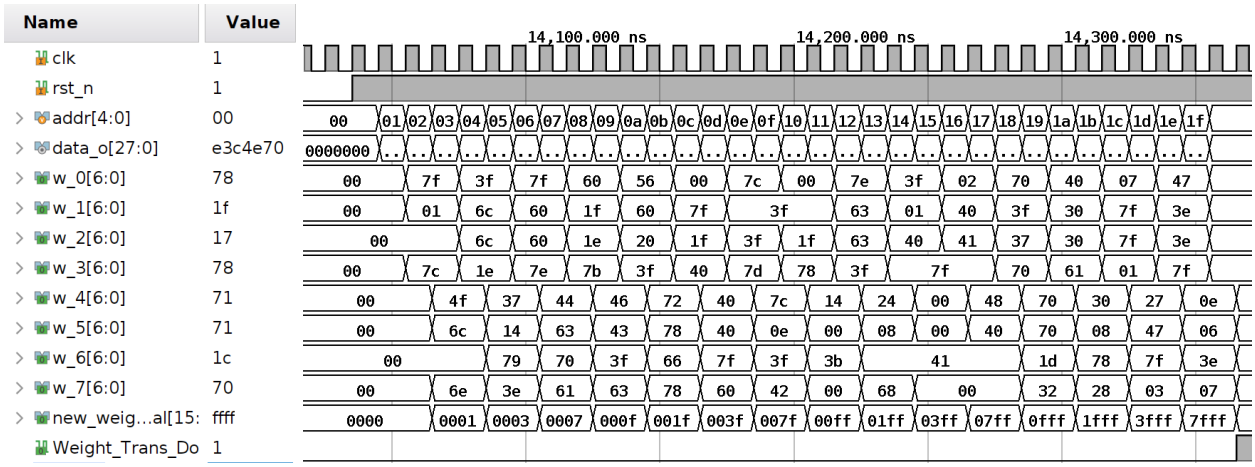

图 5-2 基本块 2 权重控制模块波形图

Weight_Control 模块的 rst_n 信号并非 TestBench 产生的全局复位信号，而是基本块 1 的所有输出数据传输至基本块 2 的 Input Buffer 后，Input Buffer 的地址控制模块产生的，它令 Weight_Control 模块工作并产生 PE 阵列的权重；rst_n 信号有效后，指向 Weight Buffer 的地址 addr 递增，data_o 即为其输出的权重数据；由于每个权重为 7bit，而 Weight Buffer 每个地址存放有 28bit 数据，即每一次只能读出 4 个权重，而基本块 2 的输入通道为 8，也就是说每个卷积核有 8 个 7bit 数据存放于8×16 PE 阵列的第一行，因此需两个时钟周期 w1~w7 方可得到第一行 8 个 PE 的权重并传给 PE 阵列；关于如何判断某一时钟周期的 w1~w7 寄存器究竟输出的是哪一行 PE 的权重，使用 16bit 的 new_weight_val 寄存器，从 LSB 起，其第 i 位即对应 PE 阵列的第 i 行，即当 new_weight_val 的第 0 位变为 1 时， PE 阵列的第 0 行存储 w1~w7 内的数据。最后，new_weight_val 的 16 bit 均为 1，说明所有权重均已传输完毕，Weight_Trans_Done 跳变为 1，它将作为 SWU 的 rst_n 信号令其开始工作。

5.1.2 SWU 功能仿真

滑动窗口单元（Slide Window Unit）负责调度 ECG Buffer 与 Input Buffer，将它们存储的数据以 PE 阵列想要的步长与大小逐个时钟周期输出。对于基本块 2~基本块 6，它们的 SWU 模块功能几乎一致，假设基本块的输入通道数为 N，则有 N 个 Input Buffer，每一个存储一个输入通道的数据，SWU 负责将这些 Input Buffer 的数据利用移位寄存器以大小为 7bit，步长为 1 并行输出。基本块 1 的 SWU 则是做了额外处理，将 ECG 数据分为 4 份存储于 ECG Buffer 中，待复位信号为 1 后，这 4 份数据并行输出。基本块 1 的 SWU 模块波形图如图 5-3 所示：

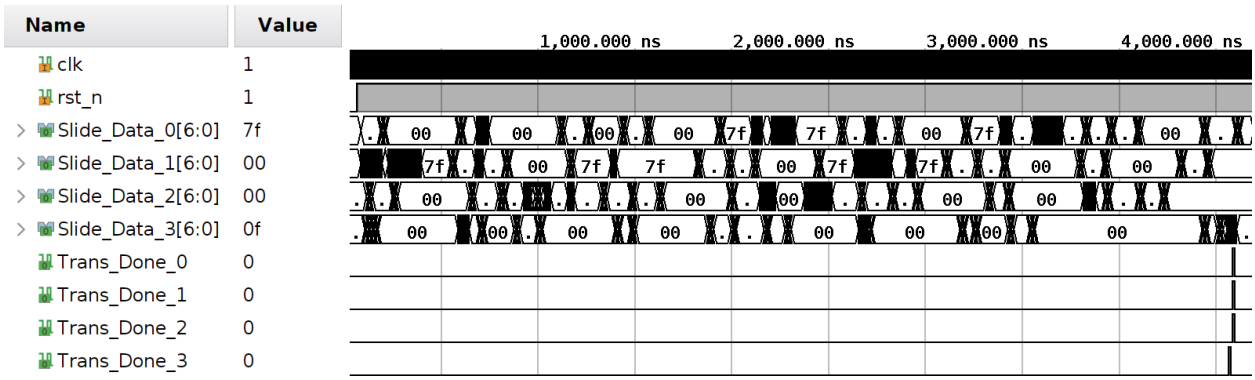

图 5-3 基本块 1 滑动窗口单元波形图

可见当所有结果均输出后，其对应 Trans_Done 跳变为 1。PE 阵列也因此由原本的1×8扩充为4×8，如此以来，可将基本块 1 的计算周期缩短为原来的 4 倍，充分利用 PE 阵列的并行性降低了整体加速器的计算延迟。

5.1.3 PE 阵列功能仿真

PE 阵列负责以流水线计算每个基本块的卷积结果并输出给最大池化、阈值比较等模块。以基本块 2 为例，其 PE 阵列的波形图如 5-4 所示：

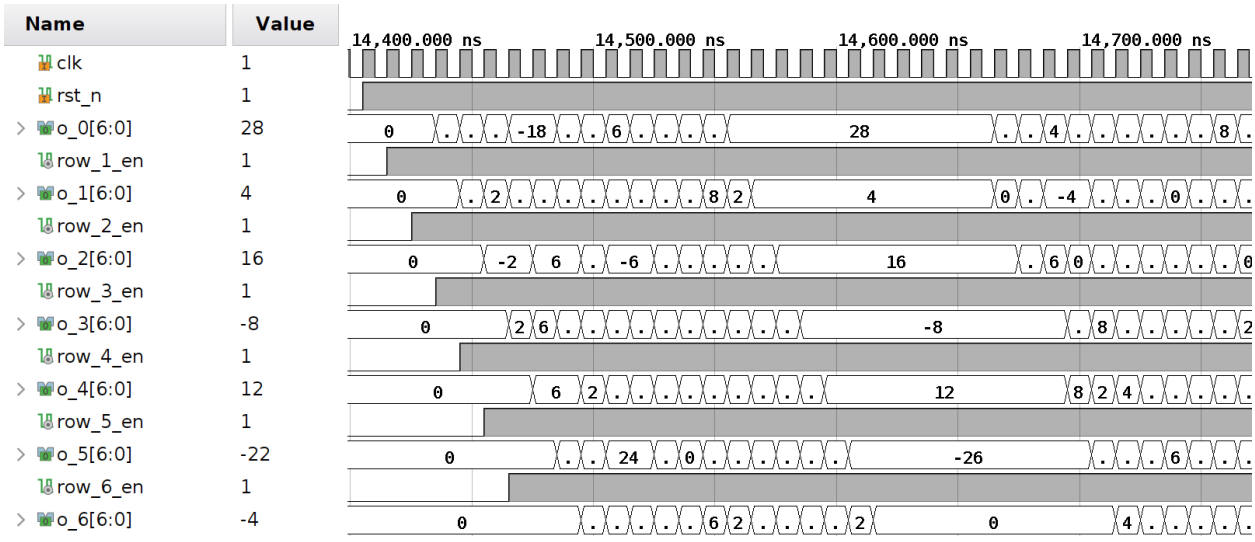

图 5-4 基本块 2 PE 阵列输出及使能信号波形图

上图仅截取了基本块 2 的 7 个输出通道的特征值与使能信号。PE 阵列的 rst_n 由 SWU 模块提供，当 SWU 模块开始输出有效值给第一行 PE 时，rst_n 为 1，代表 PE 阵列的第一行开始运作，由于每一行涉及 8 个 PE 的运算以及他们运算结果的累积，故在使能信号有效后 3 个时钟周期才输出特征值，接着开始流水线运作，每个周期均输出特征值，输出的

特征值与 PyTorch 框架下的值经过比对，完全符合。

当每一行的 PE 计算一个时钟周期后，此时其已使用完毕当前的输入数据，故输出下一行 PE 的使能信号 row_x_en，并将使用过的数据传递至下一行 PE。以第一列的前 6 行 PE 为例，数据传递波形图如图 5-5 所示：

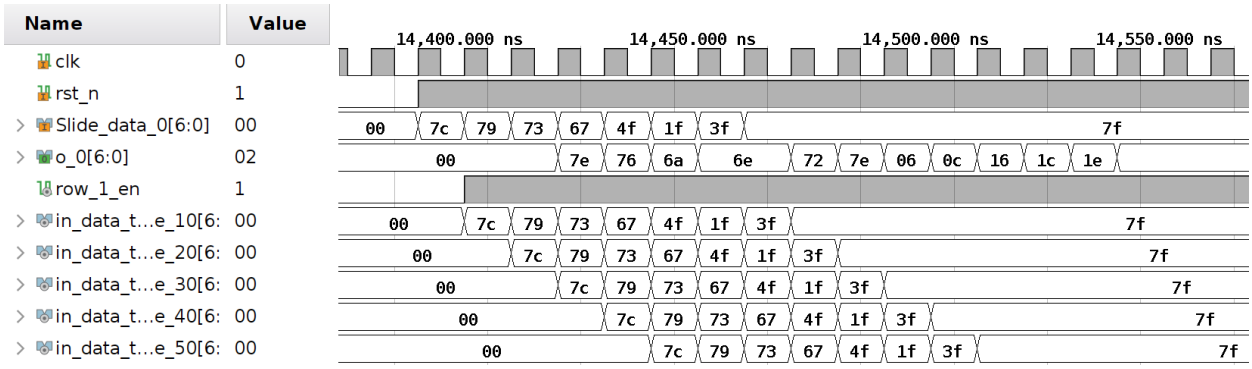

图 5-5 基本块 2 PE 阵列数据传递波形图

当 rst_n 为高电平时，Slide_data_0 由初始的 0 变为 7bit 的有效数据，它在 rst_n 为高电平的第一个时钟周期为第一行第一列的 PE 所用，下一个时钟周期便通过此 PE 的输出端口 in_data_to_next_pe_10 输送给第一行第零列的数据，供其计算，以此往复。SWU 输出的一个 7bit 数据，如波形图中的 7c，可不断地被传递与使用 16 次，在每个 PE 权重固定的条件下，实现了输入数据的高效复用与权重的不断重用，由波形图可见此模块功能无误。

5.1.4 最大池化与阈值比较模块功能仿真

对于基本块 1 至基本块 5，最大池化与阈值比较均在一个模块中，还是以基本块 2 为例，截取其输出通道 0 的部分波形图：

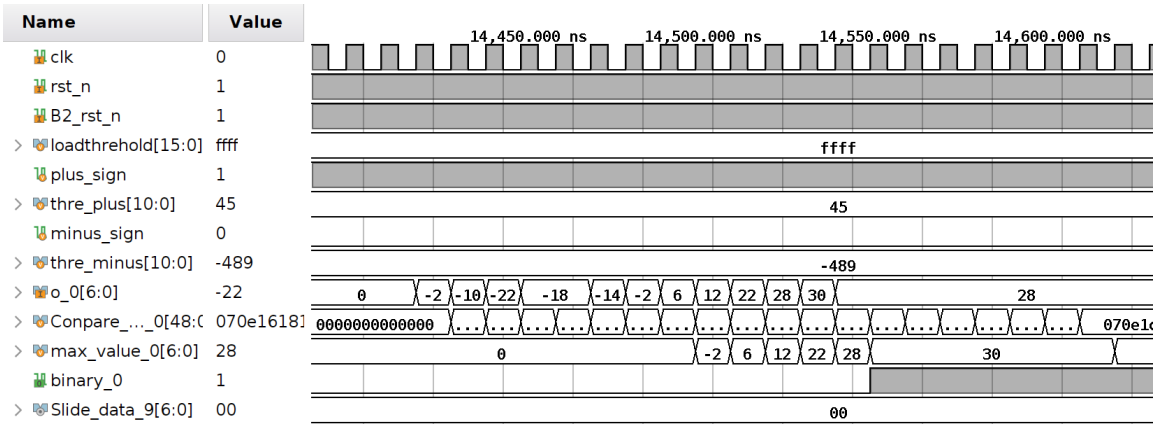

图 5-6 基本块 2 最大池化与阈值比较模块仿真波形图

在图 5-6 中，B2_rst_n 信号作为将阈值读取的标志信号，在基本块 2 的复位信号无效后，令阈值比较模块以与前面小节中 Weight_Control 模块类似的地址控制方式，将 Threhold Buffer 中阈值数据提前传输给阈值比较模块的寄存器组存储，位宽 16 的 loadthrehold 信号均为 f，说明负责处理 16 个输出通道的阈值比较模块均已成功加载阈值数据；而对于加载的 24bit threhold 值，可拆为波形图中的 plus_sign、thre_plus、minus_sign、thre_minus 4 部分，其中为 plus_sign、minus_sign 分别为k、ak的符号标志位，为 1 代表其对应的值为正，为 0 即为负，thre_plus、thre_minus 均为存储的 11bit 有符号整数阈值，用于与最大池化得出的最大值比较；o_0 是 PE 阵列的输出特征值，它被一个个打入 Compare_in_seven_0 寄存器中，此寄存器装满值则开始进行比较。由波形图可见输出的第一个最大值为-2，-2 小于 0，应与阈值 thre_minus 比较，$-2\times 2>-489$，binary_0 本该为 1，可是 minus_sign 为 0，即ak为负值，$Sign(x)$应为$-Sign(x)$，因此 binary_0 输出为 0；当 max_value 为 28 时，$28\times 2>45$，又 plus_sign 为 1，故下一个时钟周期的 binary_0 输出为 1。可见此模块功能正确无误。

5.1.5 IB Addr Ctrl 模块仿真

根据 4.4.1 节的分析，IB Addr Ctrl 模块负责收集上一个基本块输入的二值化激活值并产生相应的地址信号、地址使能信号给 Input Buffer 模块，令其存储 IB Addr Ctrl 模块收集的数据。以基本块 2 为例，IB Addr Ctrl 模块波形图如下：

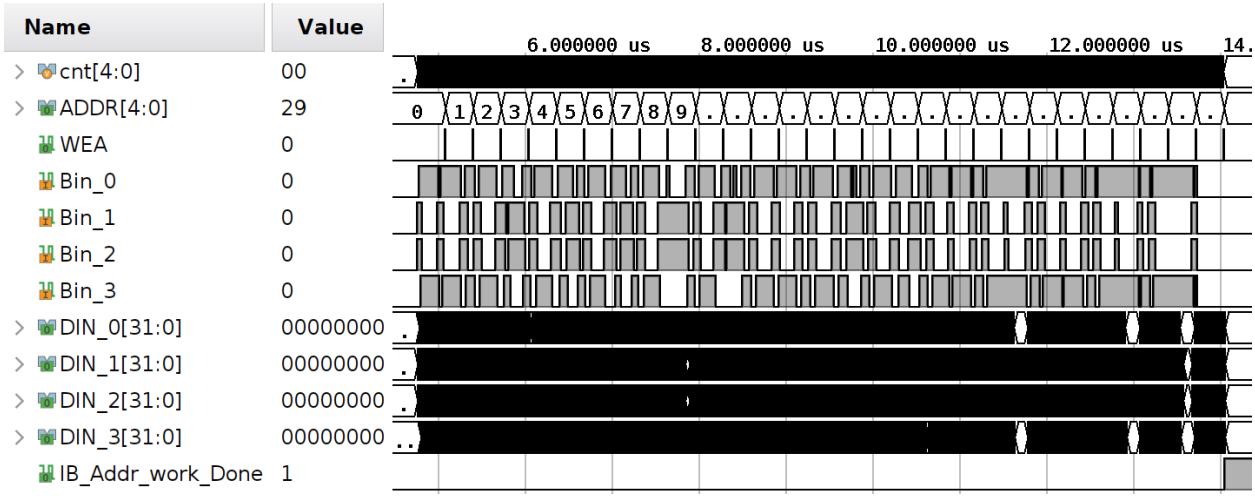

图 5-7 IB Addr Ctrl 模块仿真波形图

上图中的 Bin_0~Bin_3 即为基本块 1 输入给基本块 2 的 1bit 特征图，这里由于空间原因未截 Bin_4~Bin_7 有关的信号。可见随着复位信号失效，cnt 开始计数，DIN_x 寄存器开

始将对应的 Bin_x 信号逐渐从 LSB 打入，当攒够 32bit 时，WEA 信号跳变为 1，开启了 Input Buffer 内 RAM 的存储使能，DIN_x 内的 32bit 信号立即存入 RAM 的 ADDR 地址，以此往复，直到所有数据存储完毕。

5.1.6 Sum、Mul、Max 模块仿真

根据 4.2.4 节的分析，批量归一化层、激活层、全局累加池化层融合为 Sum 与 Mul 模块，将最大池化模块的输出累加后，执行乘加运算，其波形图如图 5-8 所示：

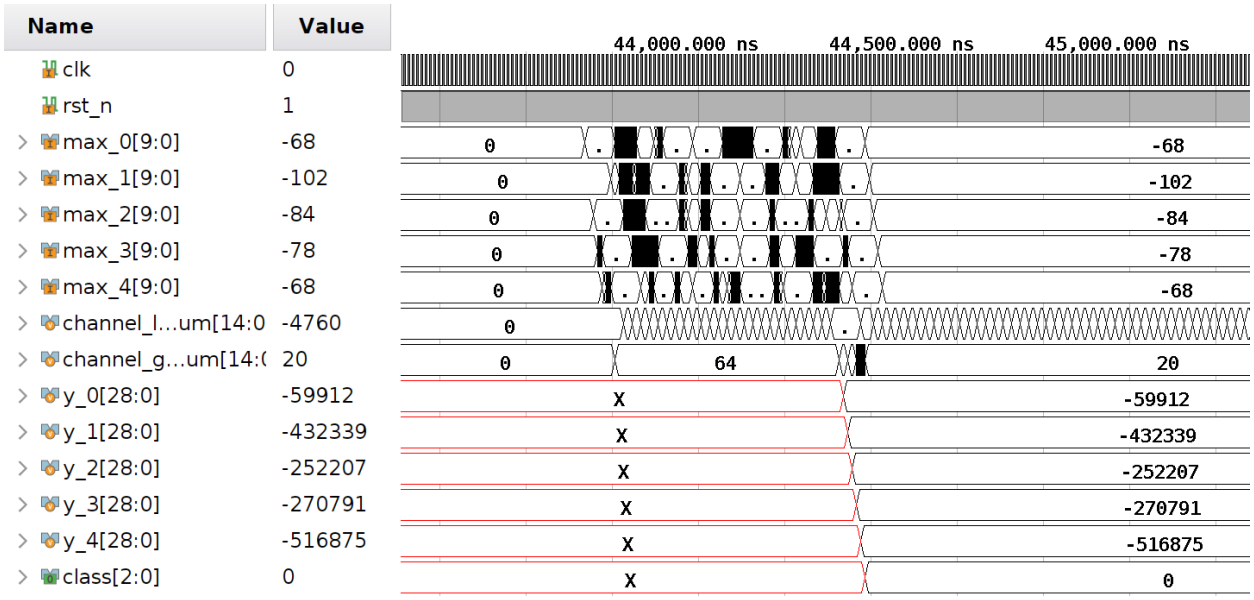

图 5-8 Sum、Mul、Max 模块仿真波形图

根据上图，当最大池化模块开始输出基本块 6 的 5 个通道的特征值时，每个通道便有其各自的 channel_lex_sum 与 channel_gex_sum 来分别存储通道 x 的所有大于 0 与小于 0 的数之和，由于空间原因图 5-7 仅列出了通道 0 的累加存储寄存器波形。由表 3.4 可知，共需在每个通道的 27 个特征值内判断正负并累加。累加结束后则执行伪代码 4.1 中的 $ge\_sum \times k + le\_sum \times ak + 27 \times b$ 计算，计算得到的值即为波形图中的 y_0~y_4，最终通过比较器得到最大值的索引，在此波形图中即为 0，也就是说 ECG Buffer 中存储的信号为第 0 类，至此基于二值神经网络的 ECG 分类加速器运算结束。y_0~y_1、class 与软件算法中的对应值完全一致，也证明了本硬件加速器的功能与软件上的结果没有偏差。

5.2 加速器准确率评估

通常情况下，将使用 PyTorch、TensorFlow 等框架实现的软件算法映射至硬件电路，可能会导致准确率下降，这是因为 FPGA 等硬件平台通常有不同的架构和计算资源限制。

与常规的 CPU 或 GPU 不同，这些硬件平台可能具有更少的存储容量、计算单元数量或带宽限制，从而影响算法在硬件上的性能表现。而为了进行算法的移植，就需要在已有算法的基础上对其进行网络结构的调整、数据流的处理以及浮点数据的量化、对不适合硬件的算法与函数的优化，以上 4 个方面对精度的影响总结如下：

（1）网络结构调整：在将算法从软件框架转移到硬件实现时，可能需要对网络结构进行调整以适应硬件平台的要求。这可能包括修改卷积核大小、网络深度、卷积核数量等参数。这些调整可能会对模型的泛化能力产生影响，导致准确率下降。

（2）数据流处理：FPGA 等硬件平台通常采用并行处理的方式，可以同时处理多个数据流。然而，某些算法在软件框架中可能不是设计为并行处理。将这些算法移植到硬件平台时，需要进行并行化的调整。如果并行处理策略不当，可能会导致数据依赖性问题，从而影响准确率。

（3）量化误差：为了适应硬件实现的需求，需要对模型进行量化，将浮点数参数转换为定点数，这种量化过程引入了量化误差，会导致模型的性能下降。过度的量化操作可能会严重影响准确率。

（4）不适合硬件实现的函数与算法：一些深度学习优化算法在软件框架中的实现可能不适用于硬件平台，它们可能过于依赖于特定的计算结构或内存布局，而这些结构在硬件上可能不存在或过于耗费资源。因此，在硬件实现中可能需要重新选择、调整或开发新的优化算法，这也可能导致准确率下降。

本课题设计的加速器采用了易于硬件实现的一维卷积神经网络而非循环神经网络 RNN，RNN 虽然因其独特的循环结构具有较强的“记忆性”，常被用于处理时间序列，但是这一循环结构也造成了硬件的难以实现性，因 RNN 中当前的输出结果可能与前几个时钟周期的输出结果相关，也就是说在计算出网络的特征图后，若将其存于 RAM 中，过一段时间就又需取出来进行下一个特征图的计算，即使是单独设置寄存器组来进行结果的缓存，也需要耗费额外的硬件资源，而 CNN 具有局部性、高并行性与高计算密度，更易于向硬件移植。

对于网络结构调整，在第三章 3.3 节设计全精度网络时便进行了提前考虑，将所有卷积核调整为 7，卷积步长第一个基本块为 2，其余基本块均为 1；也对网络深度进行了实验，权衡准确率与模型大小，选取了基本块个数为 6；同时统计了网络参数量，通过网络结构的调整将参数量控制在 3 万左右。以上这些探索网络结构的措施与实验均以高准确率作为约束。因此在软件设计阶段，我们的加速器便最大程度避免了网络结构调整对于准确率的

影响。

对于数据流处理，本设计的加速器将卷积层的数据流全映射至 PE 阵列，最大池化、阈值比较、PE 阵列的计算均采取流水线并行处理，提高了加速器的吞吐量，降低了加速器的延迟，加速器未在数据流方面产生精度下降。

对于量化误差，本加速器基于二值化算法将 ECG 数据、卷积层的权重与激活量化为 1bit 的 0、1。将 ECG 数据量化为 1bit 损失了大量信息，但加速器的准确率仍在 91%以上，若不量化 ECG 数据，则 BNN 的准确率相较于全精度 CNN 仅仅下降了不到 1%，模型大小却压缩了 29.5 倍以上，这是完全可以接受的。

对于不适合硬件实现的函数与算法，如批量归一化层，本课题设计的加速器将其融合到阈值函数或者多 bit 数乘法中。将激活层、批量归一化层、$Sign(x)$通过数学公式的推导融合，对于加速器的准确率不会有任何影响。而将激活层、批量归一化层融合为$kx+b$、$akx+b$函数，图 4-5 的实验已证明并不会对准确率造成任何损失。

从软件算法到硬件电路映射，由于二值化算法的硬件友好性以及前 4 章对软件算法所作的优化，基本块 6 的最大池化模块以前的所有模块运算均基于 1bit 的 0、1 以及多 bit 的有符号整数，没有任何的浮点数参与，唯一有浮点数参与的是基本块 6 的 Mul 模块，然而经过 4-2 节的实验与分析，将所有浮点数转为了 14bit 有符号整数后，精度并不会下降，仍旧保持 91.60%。

综上所述，本课题为 LP 模型设计的加速器 ACC 与软件算法的完全相同，并未下降，为 91.60%。

5.3 加速器设计指标评估

根据第四章的数据存储介绍内容，加速器中的权重缓存、阈值缓存以及k、ak、b缓存的存储占用空间统计如表 5.1：

表 5.1 加速器存储占用

	存储空间(bit)
权重缓存	28280
阈值缓存	3648
k、ak、b	210
合计	32138

可见加速器的存储占用为$32138/1024/8 \approx 3.92\,KB$。根据表 3.3，5 分类 ECG 检测算法共在 PyTorch 中共有 28600 个参数，每个参数需以 32bit 浮点数在软件框架中存储。因此可求得加速器的压缩率为$\frac{28600\times32}{32138} \approx 28.48$。

设计完成的加速器在 vivado2022.2 软件中进行综合，得出功耗与资源占用报告，具体整理为表 5.2：

表 5.2 加速器所用资源与功耗

	查找表	寄存器	DSP	功耗(W)
基本块 1	3597	3088	0	0.033
基本块 2	3180	4705	0	0.041
基本块 3	9024	14221	0	0.053
基本块 4	19055	26471	0	0.1
基本块 5	34757	47620	0	0.282
基本块 6	12432	16663	10	0.095

表 5.2 的功耗为加速器以 50Mhz 运行时的动态功耗，涵盖时钟、信号、逻辑等多方面功耗。而加速器的器件还有静态功耗，它主要由 MOSFET 在关断状态下的漏电电流引起，综合考虑，加速器的总体功耗如图 5-9 所示：

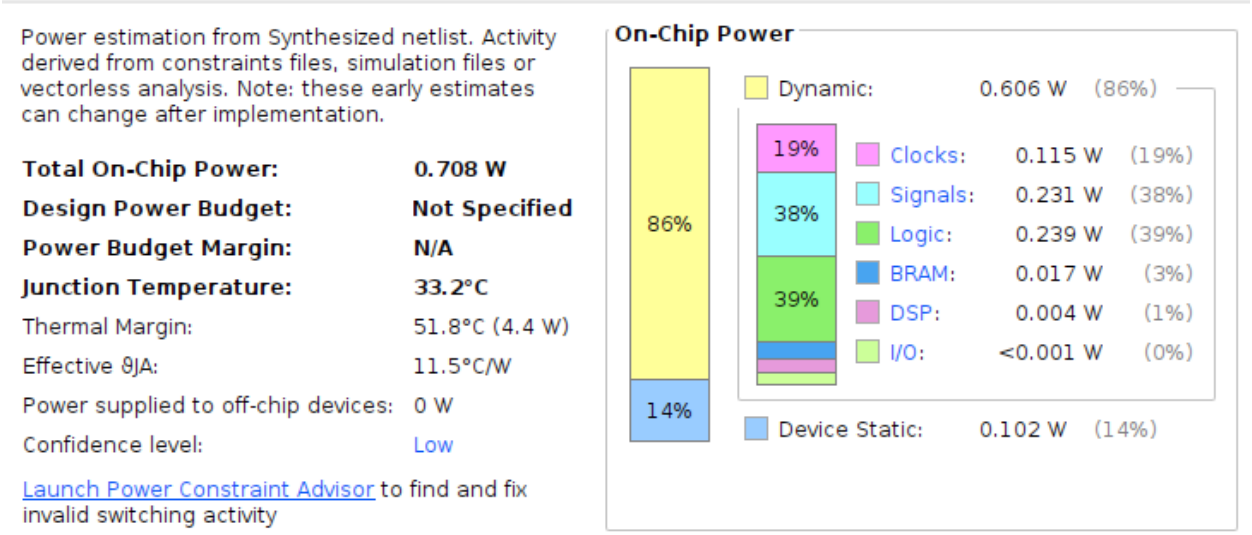

图 5-9 加速器功耗

识别一次 ECG 数据所需时间如图 5-10 所示，需 89.150μs。由$1\ MACs = 2\ OPs$，可得加速器共有$0.011874GOPs$。则吞吐率为 133.19GOPS，能效为 188.12GOPS/W。

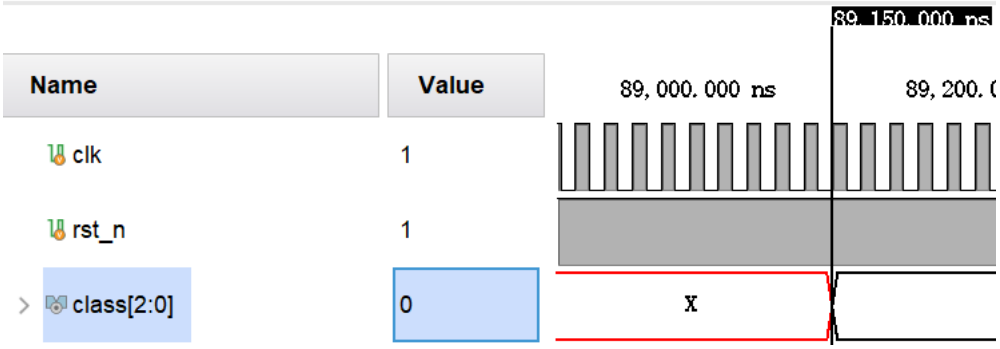

图 5-10 识别时间

综上所述，设计的加速器准确率为 91.60%，需 3.92KB 的存储空间，达到了 28.48 倍的内存压缩，仅使用了 10 个数字信号处理器（DSP），在 50MHz 的时钟频率下总体功耗为 0.708W，推理时间为 89.150μs，吞吐率为 133.19GOPS，能效为 188.12GOPS/W，达到了第一章 1.3.2 节的设计标准。

5.4 加速器性能对比

经过以上小节的分析后，最终的硬件指标对比如表 5.3 所示，我们以 Wong 的工作[50]，Wei 的工作[69]，Lu 的工作[70]，以及 Yang 的工作[71]为参考进行比较。

表 5.3 ECG 检测加速器性能指标对比表

	本设计	Wei[69]	Lu[70]	Wong[50]	Yang[71]
分类数目	5	5	5	2	10
数据位宽（bit）	1	16	16	1	1
存储占用（KB）	3.92	21.61	21.61	8.02	2355.2
DSP 个数	10	96	80	0	53
吞吐率（GOPS）	133.19	26.6	25.7	82.4	-
功耗(W)	0.708	0.79	—	227.3μ	2.4
能效（GOPS/W）	188.12	33.67	—	362516.50	-
时钟频率（MHz）	50	200	200	0.1	143

Yang[71]使用 BNN 来对 CIFAR-10 数据集进行分类，其他三个工作均为使用 MIT-BIH 数据库分类 ECG 信号。由上表可见，BNN 加速器相较于 16bit 位宽的加速器，带来了最低的存储占用，仅 3.92KB，且吞吐率与能效要比 Wei 与 Lu 的工作高 6 倍左右，可见 BNN 加速器可实现较普通加速器更高的吞吐率与能效，也能节省更多的片上存储空间。对于 Wong 的工作，因其时钟频率极低，又仅将 ECG 信号分为 2 类，所设计的加速器功耗仅有 227.3μW，故有着极高的能效，但即便在如此情况下本设计的加速器存储占用与吞吐率也均高与 Wong 的工作。Yang 的工作虽为 BNN，但是用到了 53 个 DSP，功耗为 2.4W，高于本设计的加速器。综上，可见本设计的加速器具有极低的存储占用，较高的性能与一定的实用价值。

5.5 本章小结

本章首先详细介绍了加速器中重要模块的功能仿真波形图，解释了波形图中重要信号的具体意义并分析了波形图结果，接着从硬件实现角度，基于网络结构、数据流、量化误差与算法，论证了加速器的准确率并未因算法移植而下降，最后评估了加速器的模型大小、资源占用以及功耗，并与其他加速器做了详细对比，证明加速器已达到最初的设计指标，具有一定的实用价值。

第六章 总结与展望

6.1 工作总结

目前基于 BNN 来分类 ECG 信号以进行心律失常检测的工作较少，本文设计了基于 BNN 的 ECG 检测分类网络，并利用 BNN 的硬件友好特性为其设计了专用的加速器。所做的工作可总结为以下几点：

（1）通过将激活值和权重量化为 0，1、固定网络结构参数、进行算子融合、替换 softmax 层、抛弃全连接层，我们设计了具有硬件友好性的，基于 BNN 的通用型轻量级 ECG 检测分类网络。提出的 BP（Better Performance）模型在 ECG-5 分类任务上的准确率达到了 96.90%，ECG-17 分类任务上达到了 97.50%，较全精度 CNN 准确率仅分别下降了 0.19%与 0.50%，与其他轻量级 ECG 检测工作相比，设计的算法具有一定的优势。

（2）基于 LP（Low Power）模型，使用 Verilog 代码搭建了其专用的 BNN 加速器，设计了加速器的地址控制、权重存储、阈值存储、计算阵列、最大池化、阈值比较等模块，确定了加速器的工作流，并为加速器专门设计了优化方法，去除了加速器中几乎所有的定点数、浮点数运算，真正达到了硬件友好的目的。LP 模型将 ECG 数据量化为 0、1，损失大量信息情况下仍具有 90%以上的准确率，证明了模型具有很好的抗干扰能力。

（3）对加速器的整体架构、重要模块的设计理念与细节进行了详细的阐述与说明，分析了加速器的整体数据流与各个模块的数据流，基于全映射设计的 PE 阵列减少了对权重与阈值缓存的存取次数，无需设置专门的缓存来存取累加部分和，在访存方面降低了加速器的功耗。

（4）对加速器进行了功能仿真与设计指标评估，所设计的加速器模型大小为 3.92KB，达到了 28.48 倍的存储空间压缩率，在 50MHz 的时钟下，最终功耗为 0.708W，单次识别需 89.150 μs，吞吐率为 133.19GOPS，能效为 188.12GOPS/W，加速器准确率为 91.60%。

6.2 工作展望

本文设计了基于 BNN 的 ECG 信号检测神经网络加速器，有着良好的准确率的与模型存储占用表现，加速器的相关指标达到了与预期相符合的结果。不过本论文的工作仍有待改进与完善之处，主要可列为以下几点：

（1）LP（Low Power）模型将富含信息的 ECG 信号量化为 0、1，虽然极大地简化了模型的设计，降低了功耗，但这是以牺牲约 5%的准确率为代价的，后续的工作中应尝试 BP（Better Performance）模型的加速器设计，高准确率可以更好地满足医患需求。

（2）ChatGPT 近期掀起了社会各界对 AI 行业的重视，其良好的模型表现主要原因之一是利用了互联网上几乎所有的优质文本资源来对模型进行训练，约为 TB 级别。ECG 信号不同于文本资源，它需要研究心血管疾病的资深专家来标注信息，反复论证以确认类别，因此可供使用的权威公开数据集较少，约为 0.5GB 左右，这导致了训练出的模型泛化性较低。相信随着数据集在 AI 行业的重要性不断被提高与重视，会有更多的生理信号数据集涌现。

（3）本课题设计的加速器为全映射，对 FPGA 中的 LUT 与寄存器资源使用颇多，今后应设计更为先进的结构，如尝试固定 PE 阵列大小，考虑部分和缓存的引入，以推理时间换取更多的闲置资源等。

（4）由于加速器的存储占用仅为 3.92KB，因此本设计是纯 PL 端的，并未考虑利用 FPGA 的 PS 端进行数据的调度，后续工作应考虑为 FPGA 开发板接上心率探测器，将心电图数据传输至 PS 端处理过后，送入 PL 内存储与计算，实现一个完整的心律失常检测流程。

以上内容将作为本课题的研究重点，在今后三年的研究生生涯中继续进行更深入的学习与实现。

参考文献

[1] World Health Organization. Cardiovasculardiseases (CVDs) [R/OL]. (2017-05-17) [2023-04-30]. https://www.who.int/zh/news-room/fact-sheets/detail/cardiovascular-diseases-(cvds).

[2] Oresko J J, Jin Z, Cheng J, et al. A wearable smartphone-based platform for real-time cardiovascular disease detection via electrocardiogram processing[J]. IEEE Transactions on Information Technology in Biomedicine, 2010, 14(3): 734-740.

[3] Wang D, Hu B, Hu C, et al. Clinical characteristics of 138 hospitalized patients with 2019 novel coronavirus–infected pneumonia in Wuhan, China[J]. Jama, 2020, 323(11): 1061-1069.

[4] Shaker A M, Tantawi M, Shedeed H A, et al. Generalization of convolutional neural networks for ECG classification using generative adversarial networks[J]. IEEE Access, 2020, 8: 35592-35605.

[5] Hill A C, Miyake C Y, Grady S, et al. Accuracy of interpretation of preparticipation screening electrocardiograms[J]. The Journal of pediatrics, 2011, 159(5): 783-788.

[6] Acharya U R , Fujita H , Lih O S , et al. Automated detection of coronary artery disease using different durations of ECG segments with convolutional neural network[J]. Knowledge-Based Systems, 2017, 132(sep.15):62-71.

[7] Nurvitadhi E, Sheffield D, Sim J, et al. Accelerating binarized neural networks: Comparison of FPGA, CPU, GPU, and ASIC[C]//2016 International Conference on Field-Programmable Technology (FPT). IEEE, 2016: 77-84.

[8] Hannun A Y, Rajpurkar P, Haghpanahi M, et al. Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network[J]. Nature medicine, 2019, 25(1): 65-69.

[9] Howard A G, Zhu M, Chen B, et al. Mobilenets: Efficient convolutional neural networks for mobile vision applications[J]. arXiv preprint arXiv:1704.04861, 2017.

[10] Zhang X, Zhou X, Lin M, et al. Shufflenet: An extremely efficient convolutional neural network for mobile devices[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 6848-6856.

[11] Han S, Mao H, Dally W J. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding[J]. arXiv preprint arXiv:1510.00149, 2015.

[12] Zhou A, Yao A, Guo Y, et al. Incremental network quantization: Towards lossless cnns with low-precision weights[J]. arXiv preprint arXiv:1702.03044, 2017.

[13] Li F, Zhang B, Liu B. Ternary weight networks[J]. arXiv preprint arXiv:1605.04711, 2016.

[14] Courbariaux M, Bengio Y, David J P. Binaryconnect: Training deep neural networks with binary weights during propagations[J]. Advances in neural information processing systems, 2015, 28.

[15] Pławiak P. Novel methodology of cardiac health recognition based on ECG signals and evolutionary-neural system[J]. Expert Systems with Applications, 2018, 92: 334-349.

[16] He L, Hou W, Zhen X, et al. Recognition of ECG patterns using artificial neural network[C]//Sixth international conference on intelligent systems design and applications. IEEE, 2006, 2: 477-481.

[17] Xiang K, Chen J. Fast screening algorithm for electrical alternans in ECG based on dynamical pattern recognition framework[C]//2009 7th Asian Control Conference. IEEE, 2009: 734-739.

[18] Jin F, Liu J, Hou W. The application of pattern recognition technology in the diagnosis and analysis on the heart disease: Current status and future[C]//2012 24th Chinese Control and Decision Conference (CCDC). IEEE, 2012: 1304-1307.

[19] Yang H, Wei Z. Arrhythmia recognition and classification using combined parametric and visual pattern features of ECG morphology[J]. IEEE Access, 2020, 8: 47103-47117.

[20] Gotchev A, Nikolaev N, Egiazarian K. Improving the transform domain ECG denoising performance by applying interbeat and intra-beat decorrelating transforms[C]//ISCAS 2001. The 2001 IEEE International Symposium on Circuits and Systems (Cat. No. 01CH37196). IEEE, 2001, 2: 17-20.

[21] Wang L, Sun W, Chen Y, et al. Wavelet transform based ECG denoising using adaptive thresholding[C]//Proceedings of the 2018 7th International Conference on Bioinformatics and Biomedical Science. 2018: 35-40.

[22] Marinho L B, de MM Nascimento N, Souza J W M, et al. A novel electrocardiogram feature extraction approach for cardiac arrhythmia classification[J]. Future Generation Computer Systems, 2019, 97: 564-577.

[23] Parvaneh S, Rubin J, Babaeizadeh S, et al. Cardiac arrhythmia detection using deep learning: A review[J]. Journal of electrocardiology, 2019, 57: S70-S74.

[24] Acharya U R, Oh S L, Hagiwara Y, et al. A deep convolutional neural network model to classify heartbeats[J]. Computers in biology and medicine, 2017, 89: 389-396.

[25] Mousavi S, Afghah F. Inter-and intra-patient ecg heartbeat classification for arrhythmia detection: a sequence to sequence deep learning approach[C]//ICASSP 2019-2019 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE, 2019: 1308-1312.

[26] Ádám N, Vaľko D, Havrilla M. Using Neural Networks for ECG Classification[C]//2022 IEEE 20th Jubilee World Symposium on Applied Machine Intelligence and Informatics (SAMI). IEEE, 2022: 000367-000372.

[27] Le M D, Rathour V S, Truong Q S, et al. Multi-module Recurrent Convolutional Neural Network with Transformer Encoder for ECG Arrhythmia Classification[C]//2021 IEEE EMBS International Conference on Biomedical and Health Informatics (BHI). IEEE, 2021: 1-5.

[28] Xu X, Jeong S, Li J. Interpretation of electrocardiogram (ECG) rhythm by combined CNN and BiLSTM[J]. Ieee Access, 2020, 8: 125380-125388.

[29] Rexy J, Velmani P, Rajakumar T C. HEART BEAT CLASSIFICATION IN MIT-BIH ARRHYTHMIA ECG DATASET USING DOUBLE LAYER BI-LSTM MODEL[J].

[30] Kumar S, Mallik A, Kumar A, et al. Fuzz-ClustNet: Coupled fuzzy clustering and deep neural networks for Arrhythmia detection from ECG signals[J]. Computers in Biology and Medicine, 2023: 106511.

[31] Ali O M A, Kareem S W, Mohammed A S. Comparative Evaluation for Two and Five Classes ECG Signal Classification: Applied Deep Learning[J]. JOURNAL OF ALGEBRAIC STATISTICS, 2022, 13(3): 580-596.

[32] Lu Y, Jiang M, Wei L, et al. Automated arrhythmia classification using depthwise separable convolutional neural network with focal loss[J]. Biomedical Signal Processing and Control, 2021, 69: 102843.

[33] Lin T Y, Goyal P, Girshick R, et al. Focal loss for dense object detection[C]//Proceedings of the IEEE international conference on computer vision. 2017: 2980-2988.

[34] LeCun Y, Bottou L, Bengio Y, et al. Gradient-based learning applied to document recognition[J]. Proceedings of the IEEE, 1998, 86(11): 2278-2324.

[35] Krizhevsky A, Sutskever I, Hinton G E. Imagenet classification with deep convolutional neural networks[J]. Communications of the ACM, 2017, 60(6): 84-90.

[36] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 770-778.

[37] Ma N, Zhang X, Zheng H T, et al. Shufflenet v2: Practical guidelines for efficient cnn architecture design[C]//Proceedings of the European conference on computer vision (ECCV). 2018: 116-131.

[38] Nagel M, Fournarakis M, Amjad R A, et al. A white paper on neural network quantization[J]. arXiv preprint arXiv:2106.08295, 2021.

[39] Nogami W, Ikegami T, Takano R, et al. Optimizing weight value quantization for cnn inference[C]//2019

International Joint Conference on Neural Networks (IJCNN). IEEE, 2019: 1-8.

[40] Courbariaux M, Hubara I, Soudry D, et al. Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1[J]. arXiv preprint arXiv:1602.02830, 2016.

[41] Li Z, Li H, Fan X, et al. Arrhythmia classifier using a layer-wise quantized convolutional neural network for resource-constrained devices[C]//Proceedings of the 1st International Symposium on Artificial Intelligence in Medical Sciences. 2020: 38-44.

[42] Rizqyawan M I, Munandar A, Amri M F, et al. Quantized convolutional neural network toward real-time arrhythmia detection in edge device[C]//2020 International conference on radar, antenna, microwave, electronics, and telecommunications (ICRAMET). IEEE, 2020: 234-239.

[43] Jacob B, Kligys S, Chen B, et al. Quantization and training of neural networks for efficient integer-arithmetic-only inference[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 2704-2713.

[44] Farag M M. A Self-Contained STFT CNN for ECG Classification and Arrhythmia Detection at the Edge[J]. IEEE Access, 2022, 10: 94469-94486.

[45] Scrugli M A, Loi D, Raffo L, et al. An adaptive cognitive sensor node for ECG monitoring in the Internet of Medical Things[J]. IEEE Access, 2021, 10: 1688-1705.

[46] Sivapalan G, Nundy K K, Dev S, et al. ANNet: A lightweight neural network for ECG anomaly detection in IoT edge sensors[J]. IEEE Transactions on Biomedical Circuits and Systems, 2022, 16(1): 24-35.

[47] Sun H, Wang A, Pu N, et al. Arrhythmia Classifier Using Convolutional Neural Network with Adaptive Loss-aware Multi-bit Networks Quantization[C]//2021 2nd International Conference on Artificial Intelligence and Computer Engineering (ICAICE). IEEE, 2021: 461-467.

[48] Janveja M, Parmar R, Tantuway M, et al. A DNN-based low power ECG co-processor architecture to classify cardiac arrhythmia for wearable devices[J]. IEEE Transactions on Circuits and Systems II: Express Briefs, 2022, 69(4): 2281-2285.

[49] Wu Q, Sun Y, Yan H, et al. ECG signal classification with binarized convolutional neural network[J]. Computers in biology and medicine, 2020, 121: 103800.

[50] Wong D L T, Li Y, John D, et al. An energy efficient ECG ventricular ectopic beat classifier using binarized CNN for edge AI devices[J]. IEEE Transactions on Biomedical Circuits and Systems, 2022, 16(2): 222-232.

[51] Wang A, Xu W, Sun H, et al. Arrhythmia classifier using binarized convolutional neural network for

resource-constrained devices[C]//2022 4th International Conference on Communications, Information System and Computer Engineering (CISCE). IEEE, 2022: 213-220.

[52] Yun M, Hong S, Yoo S, et al. Lightweight End-to-End Stress Recognition using Binarized CNN-LSTM Models[C]//2022 IEEE 4th International Conference on Artificial Intelligence Circuits and Systems (AICAS). IEEE, 2022: 270-273.

[53] Goldberger A L, Amaral L A N, Glass L, et al. PhysioBank, PhysioToolkit, and PhysioNet: components of a new research resource for complex physiologic signals[J]. circulation, 2000, 101(23): e215-e220.

[54] Rastegari M, Ordonez V, Redmon J, et al. Xnor-net: Imagenet classification using binary convolutional neural networks[C]//Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV. Cham: Springer International Publishing, 2016: 525-542.

[55] Bulat A, Tzimiropoulos G. Xnor-net++: Improved binary neural networks[J]. arXiv preprint arXiv:1909.13863, 2019.

[56] Tang W, Hua G, Wang L. How to train a compact binary neural network with high accuracy?[C]//Proceedings of the AAAI conference on artificial intelligence. 2017, 31(1).

[57] Choi J, Wang Z, Venkataramani S, et al. Pact: Parameterized clipping activation for quantized neural networks[J]. arXiv preprint arXiv:1805.06085, 2018.

[58] Liu Z, Wu B, Luo W, et al. Bi-real net: Enhancing the performance of 1-bit cnns with improved representational capability and advanced training algorithm[C]//Proceedings of the European conference on computer vision (ECCV). 2018: 722-737.

[59] Geng T, Wang T, Wu C, et al. LP-BNN: Ultra-low-latency BNN inference with layer parallelism[C]//2019 IEEE 30th International Conference on Application-specific Systems, Architectures and Processors (ASAP). IEEE, 2019, 2160: 9-16.

[60] Qin H, Gong R, Liu X, et al. Forward and backward information retention for accurate binary neural networks[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2020: 2250-2259.

[61] Ding R, Liu H, Zhou X. IE-Net: Information-Enhanced Binary Neural Networks for Accurate Classification[J]. Electronics, 2022, 11(6): 937.

[62] Xu Y, Han K, Xu C, et al. Learning frequency domain approximation for binary neural networks[J]. Advances in Neural Information Processing Systems, 2021, 34: 25553-25565.

[63] Ding R, Chin T W, Liu Z, et al. Regularizing activation distribution for training binarized deep networks[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019: 11408-11417.

[64] Lin M, Ji R, Xu Z, et al. Rotated binary neural network[J]. Advances in neural information processing systems, 2020, 33: 7474-7485.

[65] Xu Z, Lin M, Liu J, et al. Recu: Reviving the dead weights in binary neural networks[C]//Proceedings of the IEEE/CVF international conference on computer vision. 2021: 5198-5208.

[66] Moody G B, Mark R G. The impact of the MIT-BIH arrhythmia database[J]. IEEE engineering in medicine and biology magazine, 2001, 20(3): 45-50.

[67] Pławiak P. Novel methodology of cardiac health recognition based on ECG signals and evolutionary-neural system[J]. Expert Systems with Applications, 2018, 92: 334-349.

[68] Tuncer T, Dogan S, Plawiak P, et al. A novel Discrete Wavelet-Concatenated Mesh Tree and ternary chess pattern based ECG signal recognition method[J]. Biomedical Signal Processing and Control, 2022, 72: 103331.

[69] Wei L, Liu D, Lu J, et al. A low-cost hardware architecture of convolutional neural network for ECG classification[C]//2021 9th International Symposium on Next Generation Electronics (ISNE). IEEE, 2021: 1-4.

[70] Lu J, Liu D, Liu Z, et al. Efficient hardware architecture of convolutional neural network for ECG classification in wearable healthcare device[J]. IEEE Transactions on Circuits and Systems I: Regular Papers, 2021, 68(7): 2976-2985.

[71] Yang L, He Z, Fan D. A fully onchip binarized convolutional neural network fpga impelmentation with accurate inference[C]//Proceedings of the International Symposium on Low Power Electronics and Design. 2018: 1-6.

致　　谢

大学四年太过短暂，眨眼间 4 年结束，已经快要毕业。回想这 4 年的生活，收获感触颇多，遇到了很多优秀的人，他们给予了我莫大的支持与帮助，与此同时也深感自身阅历尚浅，许多能力仍需锻炼。

首先要感谢父母与家人，是他们给予我精神与经济上的支持，爷爷奶奶时常嘘寒问暖，父母对我大一时转专业的选择也予以认同与鼓励，这也间接促成了我在喜欢的专业与方向不断地学习与成长。

然后我要感谢我的导师刘昊，大二时我参与到刘昊老师的课外研学项目中进行相关知识的学习，刘老师为人和蔼可亲，为学生着想，为我指明了今后学习的方向，在他的指导下我成功完成了一篇英文论文并被主办方接收，系统与完整的科研训练让我对研究方向充满了兴趣。

同时感谢吴中行、刘子劲学长在加速器软硬件有关问题上的指点迷津；感谢范雪梅学姐对论文格式的纠正；感谢孙寒石与汪奥在课外研学项目的对我的帮助与支持；感谢室友在课业与学习上的互帮互助；感谢青年志愿者协会的工作安排与理解，他们使得我在九龙湖生活格外充实而有意义。

最后，感谢各位评审对我提出指导与建议，让我更清晰地了解自己的不足。学无止境，希望我能在今后三年的学习中不断提升自己的知识水平与专业素养，不断进取，为自己的研究生生活交上满意的答卷。