**体感外眼设计文档**

曹XX 0XXXXXX（队伍编号）

薛XX 0XXXXXX （队伍编号）

董XX 0XXXXXX（队伍编号）

**第一部分 设计概述**

* 1. 设计目的
  2. 应用领域
  3. 主要技术特点
  4. 关键性能指标
  5. 主要创新点

**第二部分 系统组成及功能说明**

* 1. 整体介绍



图 2-1 系统总体结构

本设计是一个以硬件加速引擎为中心的、基于Cortex-M3核和AMABA总线的SoC，之所以谓之以硬件加速引擎为中心，是因为本设计采用的全部算法均由所设计的硬件加速模块执行，Cortex-M3核在本设计中只起到软件驱动和功能配置的作用，更具体地来说，图2-1中的ACC（算法加速引擎）、Resizer（图像预处理引擎）和DMAC（DMA控制器）承担了几乎所有的数据计算和搬运功能，而Cortex-M3负责配置ACC、Resizer、DMAC以及由SoC互联的所有外设接口，即LCD、Camera、UART、GPIO和TIMER。

本系统用于边缘端的穿戴合规性检测，针对的是目前市面上主流的分类检测算法，如SSD、YOLO等，其执行流程分为图像预处理、CNN（卷积神经网络）推理和NMS（非极大值抑制）三个步骤。

在一次算法执行过程中：

1. Camera首先采集图像数据，由DMAC传送至DDR和LCD显示；
2. Resizer从DDR中读取图像数据并进行预处理，将其转化为DNN能够识别的特征图并写回DDR；
3. ACC读取该特征图并进行逐层计算推理；
4. ACC读取推理结果进行NMS处理；
5. CM3核从ACC中读取最终结果并交至LCD显示。



图 2-2 算法在系统中的执行过程

* 1. 各模块介绍
     1. ACC



图 2-3 ACC总结构

ACC用于执行算法的主体部分，即DNN推理和NMS处理，由DNN加速器、NMS加速器、bias ROM和配置寄存器堆组成。

其中，DNN加速器是本设计的核心组成部分，它由全局缓存和DNN计算阵列两部分构成，全局缓存分为8KB的输入特征图（ifmap）缓存和4KB的权重（filter）缓存，均以SRAM的形式实现，用于缓存从片外DDR读取的输入特征图数据和权重数据，其中前者可以存储2个输入通道，而后者可以存储64个卷积核。DNN计算阵列由64个卷积池化单元（CPE）组成，输入特征图缓存的数据通过一个multicast网络传输至所有CPE，权重缓存的数据通过一个unicast网络传输至所有CPE，这使得输入特征图的数据在两个输入通道之间实现了并行计算，以及对64个卷积核的数据复用。每个CPE能够实现MAC（乘累加）、pooling（池化）、padding（补0）和ReLU（激活）功能，并可以根据用户的配置使能其中一个或多个功能，每个CPE内部设有一个Inter-R Pad，以SRAM的形式存储计算的中间结果，并被不同计算过程的结果数据分时复用。CPE功能的多样性以及Inter-R Pad的复用性提升了系统的运行效率，这是因为数据被读入一次可以实现多次不同的运算，省去了多次往返耗费的时间，极大减少了片外访存的次数。



图 2-4 DNN加速器以及CPE结构

NMS加速器用于实现对DNN推理结果的NMS处理，之所以需要NMS处理，是因为一般情况下DNN的推理结果中包含很多组坐标及分类结果，需要使用NMS算法从这些众多的结果中挑选出用于标记每个目标的最终结果。NMS加速器在结构上包括数据过滤单元、硬件排序单元、回归解码单元和NMS处理单元。其中前三者分别用于结果置信度的阈值比较与筛选、基于置信度的结果重排序和结果回归矩形坐标解码，最后的NMS处理单元用于执行核心的NMS算法，在该算法的执行过程中，每次从RAM中取出一个最优结果并根据IoU准则剔除无效结果，反复迭代，直到得到一个最优结果集。



图2-5 NMS处理过程



图 2-6 NMS加速器结构

bias ROM用于存放DNN的bias（偏置）参数。

配置寄存器堆用于DNN加速器和NMS加速器的功能配置，共包含14个32bit的寄存器。

* + 1. Resizer
    2. DMAC
    3. Cortex-M3核

本次设计中Cortex-M3核来自于CMSDK(Cortex-M System Design Kit)评估版。CM3(Cortex-M3)是一个 32位处理器内核，其内部的数据路径、寄存器以及存储器接口都是 32 位的，所以可支持4GB内存空间，可以满足大多数设计需求。CM3 采用了哈佛结构，拥有独立的指令总线和数据总线，因而取指与数据访问可以并行不悖，大大提升了性能。而且CM3采用Thumb-2指令集，同时支持16位和32位指令，不需要在ARM状态和Thumb状态之间来回切换，简化了软件开发和代码维护，提高了代码的移植性。Thumb-2指令集为编程带来了更多的灵活性。许多数据操作现在能用更短的代码搞定，这意味着 CM3的代码密度更高，也就对存储器的需求更少。

图示

描述已自动生成

图2-7 CM3简化结构图

由图2-7可以看到，CM3内核暴露的总线接口有四种：指令总线I-Code、数据总线D-Code、系统总线以及私有外设总线。在本次设计中主要使用指令总线、数据总线和系统总线，其中指令总线和数据总线负责访问代码存储区（ITCM），前者负责取值，后者专注查表等。为了安全起见，系统总线可以访问除ITCM外的任何内存空间，即对片外RAM和各种外设的访问由系统总线来完成。

CM3核内建了嵌套向量中断控制器（NVIC），可支持240条外部中断输入。中断的向量化不需要通过软件去判断中断源，大大减少了中断延迟。中断的嵌套也是在硬件水平上实现的，不需要软件代码来实现。本设计的软件部分主要基于中断实现，各外设的工作可以很大程度上保持并行，大大提高了系统性能。

* + 1. AMBA总线与地址映射

AMBA是由ARM公司研发推出的一种高级微控制器总线架构(Advanced Microcontroller Bus Architecture)。其中AMBA包含了四种不同的总线标准，分别是：AHB、ASB、APB和AXI。在本设计中主要用到AHB和APB这两种总线标准。

AHB是高级高性能总线，通常用于CPU和高性能设备之间的交互，类比于现代PC的北桥。而APB是高级外设总线，通常通过AHB2APB桥和AHB相连接，来访问一些低速的外设。

在本设计中由于AHB Master有CM3核、ACC和Resizer，所以整个系统架构基于总线矩阵实现，如图2-1所示。其中总线矩阵AHB Bus Matrix中主从设备之间的互联关系如图2-8所示。



图2-8 AHB Bus Matrix映射关系

图中SI表示Slave Interface，连接Master；MI表示Master Interface，连接Slave。另外，APB Bridge代表的是AHB转APB的协议桥，DMA Reg、ACC Reg和Resizer Reg表示DMA、ACC和Resizer配置寄存器的接口。

* + 1. ITCM和DTCM
    2. LCD Driver
    3. Camera Driver
    4. DDR Interface

**第三部分 完成情况及性能参数**

**第四部分 总结**

* 1. 可扩展之处
  2. 心得体会

**第五部分 参考文献**

**第六部分 附录**

重要代码、推导过程等不便于在正文中体现的内容