第二次作业

author: PB19061272 金桥

[2-22] 结合 3D 图形生成的应用，分析所需要的“算力”具有什么特征。

- 图像数据往往为较大量的形式统一的浮点数，且对图像的处理往往涉及到超越函数的计算。
- 所以“算力”应擅长计算超越函数与浮点数乘法，可并行运算。

[2-24] 试分析 2006 年 Nvidia 发布“Unified Graphics and Computing Architecture”的意义。

- 实现了顶点和片元两种计算单元的统一架构，平衡了工作负载，大幅提升了工作效率。
- GPU 开始由单纯的渲染转向通用计算领域。

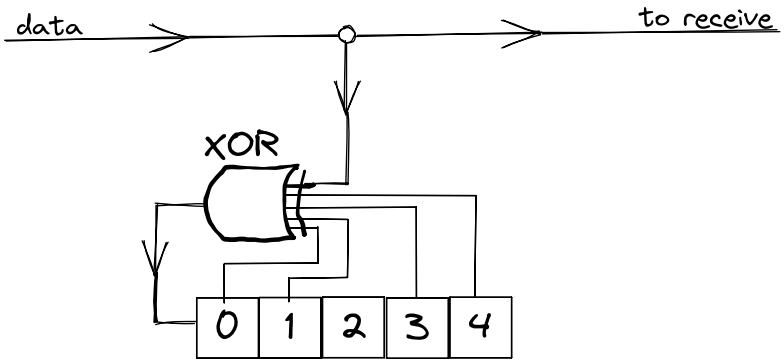
[2-26] 列举英特尔 CPU（以第 10 代酷睿为例）中支持的的硬件视频编解码器。

- JPEG
- MJPEG
- MPEG-2 (H.262)
- MPEG4 AVC1 (H.264) 8-bit 4K 4:2:0
- MVC (H.265)
- HEVC1,2 8 bit (H.265) 8K 4:2:0
- HEVC1,2 10 bit (H.265) 8K 4:2:0
- VC-1
- VP8
- VP9 8 bit 8K 4:2:0
- VP9 10 bit 8K 4:2:0

[2-27] 用 C 或 Python 语言实现 USB3.0 中的扰码计算（扰码多项式为：G(X)= X16+X5+X4+X3+1。

```
code1 = 0b10000000000111001        #扰码
code2 = int(input('输入信号: '), 2)
code3 = bin(code1^code2)           #按位异或
print('扰码结果为: {}'.format(code3))
```

[2-28] 若生成多项式为 G(x)= x4+x3+x1+1，给出基于 LFSR 对应的 CRC 电路。



[2-31] 列举三种以上在通信网络中使用的 8B/10B 控制码。

- K.28.0
- K.28.1
- K.28.2

[2-33] 扰码电路和 CRC 电路均基于 LFSR，区别何在?

- CRC 校验中，最终输出的结果是各个触发器中的比特。
- 而扰码电路是随着时钟的节拍每个时钟周期输出一个比特。

[2-34] 试从 AES 的加密/解密流程分析英特尔 AES-NI 扩展指令可能的调用方式。

- 加密过程应是多次 AESDEC 加上 AESDECLAST，解谜过程为多次 AESENC 加上 AESENCLAST。
- AESDEC、AESDECLAST、AESENC、AESENCLAST 都需要对应的密钥，所以这 4 个指令应该都会调用 AESKEYGENASSIST。
- 解密过程需要逆列混淆操作，所以 AESENC 和 AESENCLAST 应该会调用 AESIMC。

[2-36] FFT 算法对计算电路有什么要求？如果设计一款 FFT 算法专用电路，该电路需要包含哪些要素?

- FFT 算法需要对处理数据进行重新排序，计算电路应能实现位反转寻址。
- 电路应包含逆进位的加法器。

[2-37] 试分析 FFT 算法中的乱序操作 C 语言软件实现和硬件实现的差异。

- 软件实现（Rader 算法）：
  - 先从位序二进制的最高位往最低位数查找，碰到第一个 0，这个 0 左边（高位）的 1 都变成 0，即依次向低位进位，进到低位出现 0 为止。
  - 再把碰到第一个 0 所在的那个位置变成 1。
  - 需要对位序进行遍历，代价较高
- 硬件实现：
  - 位序为 1 的反序二进制为 0000b
  - 反序二进制与 1000b 逆进位相加即得到下一个反序二进制
  - 相当于普通的加法运算，代价较低

[2-38] 结合英伟达 Tensor Core，简述对混合精度计算的理解。

- 混合精度计算能在底层硬件算子层面，使用半精度作为输入和输出，使用全精度进行中间结果计算。
- 混合精度计算能根据精度的降低动态调整算力，在保持准确性的同时提高吞吐量。
- 混合精度计算牺牲了溢出的精度换来了算力的提升。

[2-39] 试对比英伟达 GPU 中 FP32、TF32、FP16、FP8 所能表示的数的范围。

- FP32：$-(2-2^{-23})\times 2^{127}\le x\le -2^{-126}$、$0$、$2^{-126}\le x\le (2-2^{-23})\times 2^{127}$
- TF32：$-(2-2^{-10})\times 2^{127}\le x\le -2^{-126}$、$0$、$2^{-126}\le x\le (2-2^{-10})\times 2^{127}$
- FP16：$-(2-2^{-10})\times 2^{15}\le x\le -2^{-14}$、$0$、$2^{-14}\le x\le (2-2^{-10})\times 2^{15}$
- FP8：
  - E5M2：$-(2-2^{-5})\times 2^{2}\le x\le -2^{-1}$、$0$、$2^{-1}\le x\le (2-2^{-5})\times 2^{2}$
  - E4M3：$-(2-2^{-4})\times 2^{4}\le x\le -2^{-3}$、$0$、$2^{-3}\le x\le (2-2^{-3})\times 2^{4}$

[2-40] 简述英特尔 AVX-512 VNNI（Vector Neural Network Instructions）执行快速 INT8 卷积操作的思路。

- 在 VNNI 之前，我们需要做两个 8 位向量乘加得到 16 位，使用基础 AVX-512，对于 16 位，这可以使用两条指令实现 VPMADDWD 用于将两个 16 位对相乘并将它们加在一起，然后将 VPADDD 添加累加值。
- 利用基础 AVX512 需要 3 条指令，而 VNNI 只需要 1 个周期就可以完成。通过将三条指令融合为一条，可以最大化利用计算资源，提升 cache 利用率及避免潜在的带宽瓶颈。

[2-39] 试分析英特尔 AMX(Advanced Matrix Extensions) 加速矩阵运算的思路。

- 使用了一组新的寄存器，可用 Tiles 声明状态。
- 使用了基于 Tiles 的 TMUL 指令集加速矩阵运算。

[2-45] 提升 AI 算法的执行效率有几类可行途径?

- 算法本身的优化
- 电路结构优化
- 混合精度计算
- FPGA 加速 AI 计算