ECE 479 - Homework 3

Ahmadreza Eslaminia

Ae15

Question 1: GPU and GPU programming

1. differences between GPU and CPU:

   Function: CPUs are general-purpose processors optimized for single-threaded performance, while GPUs are specialized processors optimized for parallel processing of visual data.

   Architecture: CPUs have a few cores and shared cache memory, while GPUs have many cores and dedicated VRAM memory.

   Application: CPUs are used for running applications, managing the operating system, and executing complex algorithms, while GPUs are used for accelerating visual data rendering, scientific simulations, machine learning, and cryptocurrency mining

2. Parts of GPU

Grid	(G)
Block	(H)
Warp	(A)
Thread	(B)
Tensor Core	(D)
NVLink	(I)
Shared Memory	(F)
Global Memory	(C)

3. Blurring algorithm Calculation

a) The grid dimensions are calculated as (38, 25) using the ceiling function with the given values of 600/16 and 400/16.

b) The BLUR_SIZE value does not impact the grid dimensions. It only affects the output value of each pixel after the blurring algorithm is applied. The kernel value is used to determine which pixels around the current pixel should be used to calculate the average pixel value. The grid dimension is not affected by this process as it determines the output size.

c) There are three different values, which are 9, 15, and 25.

4. True False

   a) It is false that GPUs are always faster than CPUs for sequential code as CPUs can be 10+X faster in some cases.

   b) The statement is true.

   c) The statement is true.

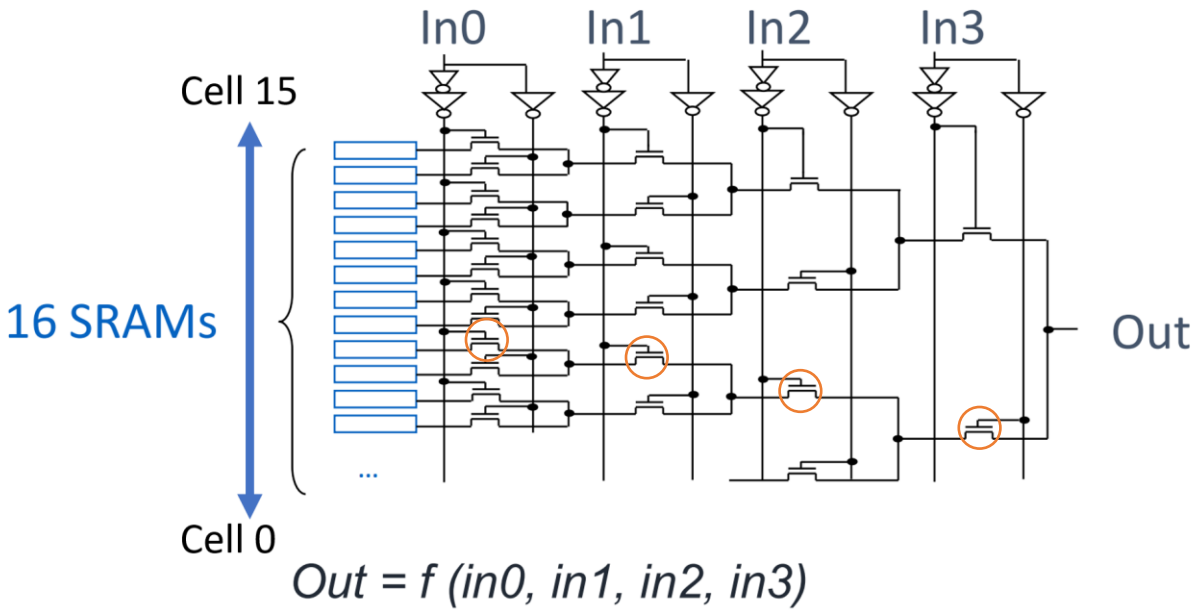
   d) The statement is true.

   e) The statement is false as CPU and GPU share memory, and thus data does not need to be moved between them.

Question 2: FPGA Basics and High-Level Synthesis

1. 4-bit look-up table
   a) There are four inputs, resulting in a total of 16 unique possible combinations. This number matches the number of SRAMs in the 4-bit lookup table, as well as the number of values in a truth table. Therefore, the output for all 16 possible combinations of the logic function can be precomputed and stored in the SRAMs, allowing for the simulation of the logic function.

b) in3 in2 in1 in0 – 1 0 0 0



2. HLS C code

For the for loop in line 23, we can apply Loop bounds optimization by setting the bounds to the maximum value, which is the value of 'boundary' in this case. By conditionally executing the loop body, we can increase optimization opportunities and further improve the performance of the code.

We can improve the performance of the code by using Loop Fusion on the two loops in line 16 and line 19. This can be achieved by using an if statement on the i variable to improve data locality, shorten latency, and better share resources.

3. HLS C code for a convolution

a. The Initiation Interval (II) represents the time between successive new inputs to the pipeline. The current code cannot achieve an II of 1 due to the write operation performed on 'acc' in line 9 and the two reads (for kernel and feature) and a write operation (for acc) in line 14. As there is only a single port for read and a single port for write (total of two ports), it will take two cycles to complete the

write operation to 'acc', read the values in 'kernel' and 'feature' for the addition operation, and write the result to 'acc'.

b. To achieve an II of 1, Loop Perfection needs to be performed. Loop sinking can be carried out on line 9, and loop unrolling can be performed on line 14

As we know, a low SSE indicates that the data points are tightly clustered around their respective centroids, while a high SSE indicates that the data points are widely spread out, and the clustering solution is not very effective. So, in most cases we should use the elbow method to define the best number of K. The elbow method is one commonly used technique for selecting the optimal number of clusters based on the SSE. As we can see from the SSE-K diagram we can see the elbow is on the K=5. After K=5 the SSE decrease not significantly and before that the decrease is significant. Therefore the k=5 is optimal number for the number of classes. In addition, we can see from the data that k=5 for the cluster number is a reasonable choice.

Question 3: Efficient DNN Accelerator

1. Roofline Model:

   a) Regions 3 and 4 suggest a feasible design, whereas region 2 is not feasible as it lies above the Bandwidth Roof.

   b) The graph levels off because it reaches the Computation Roof, which represents the maximum number of operations per execution cycle. This limit is hardware-specific and cannot be exceeded.

   c) The main challenge faced by this design is the requirement of a high bandwidth for the Off-Chip Bus that connects to the External Memory. The solution to this challenge is Data Reuse

2. Depth wise-pointwise Convolution:
   a) dj * di * k * k * hi * wi

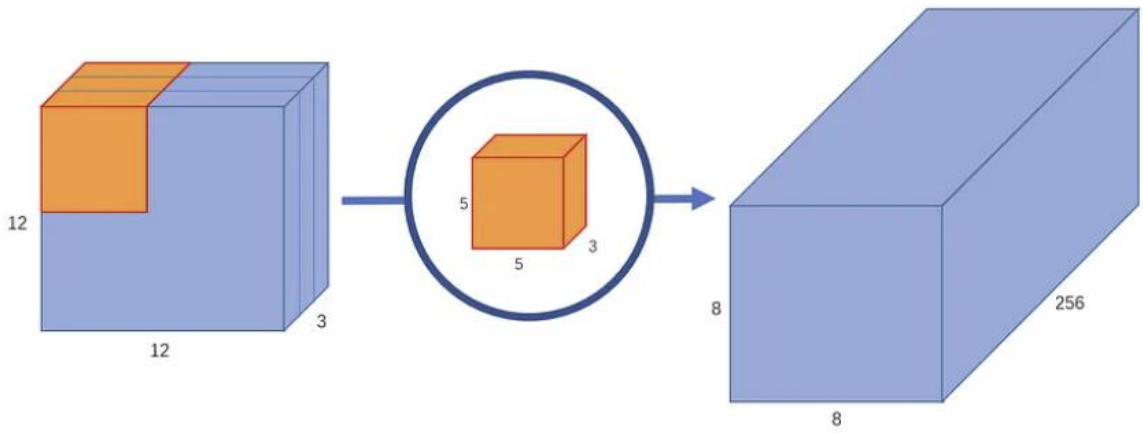


Image 5: A normal convolution with 8×8×256 output

b) di * k * k * hi * wi

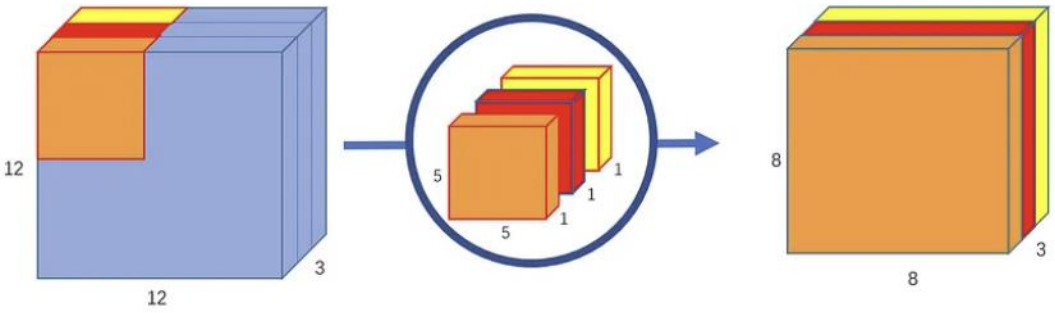


Image 6: Depthwise convolution, uses 3 kernels to transform a 12×12×3 image to a 8×8×3 image

c) dj * di * hi * wi

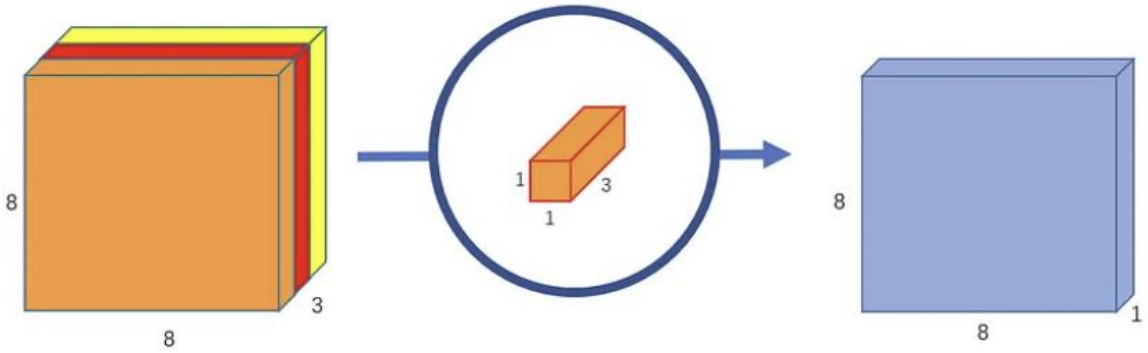


Image 7: Pointwise convolution, transforms an image of 3 channels to an image of 1 channel

d) Lecture 19

For calculating the number of multiplications, the computer has to do in the original convolution. There are 256 5x5x3 kernels that move 8x8 times. That's 256x3x5x5x8x8=1,228,800 multiplications.

What about the separable convolution? In the depth wise convolution, we have 3 5x5x1 kernels that move 8x8 times. That's 3x5x5x8x8 = 4,800 multiplications. In the pointwise convolution, we have 256 1x1x3 kernels that move 8x8 times. That's 256x1x1x3x8x8=49,152 multiplications. Adding them up together, that's 53,952 multiplications.

$$ratio = \frac{standard}{Deptwise + pointwise} = \frac{1228800}{53952} = 22.77$$

The ratio of standard convolution to the combined depthwise and pointwise convolution is about 23, meaning that standard convolution requires about 23 times more multiplications. In standard convolution, the image is transformed 256 times, each transformation requiring 4,800 multiplications. In contrast, separable convolution only transforms the image once during the depth wise convolution and then elongates it to 256 channels. This reduces the number of transformations drastically, resulting in a significant decrease in the number of required multiplications.

e) (1) When designing accelerators for resource-constrained embedded devices like small FPGAs, the use of depth wise and pointwise convolutions can present certain challenges. Specifically, the two separate convolutions may require different hardware implementations, leading to increased complexity and cost. Additionally, combining the two convolutions may lead to a higher memory requirement, as the intermediate feature maps must be stored in memory before being passed to the pointwise convolution.

(2) Depth wise and pointwise convolutions can have better or worse accuracy compared to normal convolutions, depending on the specific use case. In general, depth wise convolutions are less expressive and less powerful than normal convolutions, which may result in lower accuracy. However, pointwise convolutions can compensate for this by increasing the number of channels and adding non-linearity. Moreover, depth wise and pointwise convolutions have a lower number of parameters than normal convolutions, which can reduce overfitting and improve accuracy on smaller datasets. Overall, it is difficult to make a definitive statement about the accuracy of depth wise and pointwise convolutions compared

to normal convolutions, as it depends on the specific network architecture and training data.

Question4 : Attention is all you need

1. Computer vision Vs NLP:
   One similarity between computer vision tasks and natural language processing (NLP) tasks is that they both involve processing large amounts of data, such as images or text, and extracting meaningful patterns and relationships from it using machine learning algorithms.

   One difference between computer vision tasks and NLP tasks is the nature of the input data. Computer vision tasks involve processing image or video data, which is often high-dimensional and spatially structured. In contrast, NLP tasks involve processing textual data, which is often sequential and has a natural language structure. Additionally, computer vision tasks often involve pre-processing steps like image normalization and data augmentation, whereas NLP tasks may involve pre-processing steps like tokenization and word embedding.

2. Q-k MATRIX
   a) After the head-split, the Query tensor Q with dimensions (64, 1024, 25) is split into 32 parts, one for each head, along the second dimension (channel depth). Each split tensor has a shape of (64, 32, 25) because the number of channels is divided by the number of heads (1024/32=32).
   b) In each attention module, the dimension of the Q-K matrix in each attention module is (64, 25, 25) of (batch size, sequence length, sequence length), representing the attention weights for each position in the sequence for each item in the batch.
   c) Multi-head attention is necessary to capture complex relationships between different parts of the input sequence and improve generalization. Using a single attention head may not capture these complex relationships, but using multiple attention heads allows the model to attend to different parts of the input sequence based on different patterns, improving its ability to learn generalizable representations in NLP tasks.

3. BERT:

a) BERT training differs from GPT in that BERT uses a bidirectional transformer while GPT uses a unidirectional transformer, and BERT is trained using a masked language model while GPT is trained using a left-to-right language model.
b) The "Segment Embeddings" in BERT help the model to understand the relationship between different segments of input text, such as different sentences or paragraphs within a document, by encoding them as distinct vectors. This enables BERT to better capture context and meaning in language, and improves its ability to perform tasks such as question answering and text classification.
c) Transformer models need "Position Embeddings" because they don't have any inherent notion of word order or position in the input sequence. The self-attention mechanism used in Transformers allows the model to attend to any position in the sequence, but it needs some way to distinguish between the positions. Position embeddings provide the model with a representation of the relative positions of the words in the input sequence, allowing it to effectively model the order of the words and the dependencies between them. Without position embeddings, the Transformer model would not be able to capture the sequential nature of natural language text.

4. Bonus:

Question5: IoT Security and Cybersecurity

1. ARM Trust Zone,

   ARM Trust Zone is a hardware-based security solution that creates a secure environment, or "secure world," isolated from the regular, or "normal world," on a device. The isolation strategy is implemented through a combination of hardware and software techniques, including a secure boot process, secure memory management, and a secure monitor that controls access to the secure world. This allows for sensitive operations and data to be kept separate from less trusted software and helps to protect against a variety of security threats, including malware, hacking, and unauthorized access.

2. (a) Denial-of-Sleep attack affects an IoT edge device by continuously sending fake or malicious traffic to the device, preventing it from entering sleep mode and conserving energy.
   (b) As the attacker, one can conduct a Denial-of-Sleep attack by sending a large number of encrypted packets to the device, forcing it to decrypt and process each packet, consuming significant energy and preventing the device from entering sleep mode. This attack can be sustained by sending a continuous stream of packets, making it difficult for the device to conserve energy and leading to a faster battery drain.

3. Cyber DDoS
   a) A DDoS (Distributed Denial-of-Service) attack is a type of cyber attack in which a large number of compromised devices, or "zombies," are used to flood a target server or network with an overwhelming amount of traffic, rendering it inaccessible to legitimate users. Unlike a regular DoS (Denial-of-Service) attack, which is launched from a single source, a DDoS attack is distributed across many sources, making it more difficult to mitigate and trace back to the source. Additionally, because the attack traffic is coming from multiple sources, it can be harder to filter out the malicious traffic from legitimate traffic.

   b) Mirai takes advantage of Internet of Things (IoT) devices, such as routers, cameras, and DVRs, that have weak or default passwords and are connected to the internet. Mirai infects these devices by exploiting known vulnerabilities and then uses them to launch DDoS attacks on a

target server or network. Mirai is particularly effective against IoT devices because they often have limited computing power and security features, making them easier to compromise and control.

c) Mirai was able to infect a large number of devices because of the poor state of IoT security in 2018, which left many devices with default or weak passwords that Mirai could exploit. Despite being poorly written, Mirai was effective at compromising heterogeneous hardware by using stateless scanning to locate and infect vulnerable devices. While Mirai did not use any new or complex techniques, its ability to exploit known vulnerabilities and rapidly infect a large number of devices made it a significant threat to IoT security.