In [1]:
```python
from pynq import Overlay
from pynq import allocate
import numpy as np
```

In [2]:
```python
overlay = Overlay("./cnn.bit")
```

In [3]:
```python
dma0 = overlay.dma0     # DMA for sending input image
dma1 = overlay.dma1     # DMA for receiving output probabilities
```

In [4]:
```python
image_in = allocate(shape=(30*30*3,), dtype=np.float32)  # Your image input
probability_out = allocate(shape=(43,), dtype=np.float32)  # 43 classes
class_out = allocate(shape=(1,), dtype=np.float32)  # Final class output
```

In [5]:
```python
custom_image = np.array([
    [[[236, 240, 255]], [[230, 231, 255]], [[119, 125, 169]], [[71, 77, 95]], [[42, 45, 50]],
    [[[235, 238, 255]], [[233, 235, 255]], [[119, 126, 165]], [[63, 70, 89]], [[27, 32, 41]],
    [[[242, 241, 255]], [[237, 239, 255]], [[123, 132, 164]], [[57, 62, 80]], [[28, 28, 37]],
    [[[230, 232, 254]], [[243, 243, 255]], [[138, 138, 163]], [[71, 72, 87]], [[28, 28, 36]],
    [[[228, 233, 254]], [[247, 247, 255]], [[142, 144, 161]], [[77, 81, 94]], [[28, 32, 38]],
    [[[227, 232, 252]], [[247, 251, 255]], [[130, 148, 160]], [[72, 87, 102]], [[54, 63, 74]]
    [[[234, 237, 255]], [[240, 254, 255]], [[119, 179, 182]], [[48, 88, 89]], [[78, 82, 89]],
    [[[232, 235, 255]], [[247, 255, 255]], [[125, 160, 149]], [[142, 165, 133]], [[115, 116,
    [[[227, 228, 248]], [[255, 255, 255]], [[255, 255, 252]], [[255, 255, 245]], [[255, 255,
    [[[198, 182, 201]], [[239, 233, 241]], [[255, 239, 246]], [[232, 209, 219]], [[194, 172,
    [[[76, 71, 87]], [[77, 75, 99]], [[88, 80, 107]], [[77, 75, 99]], [[65, 69, 98]], [[77, 7
    [[[80, 84, 100]], [[76, 84, 113]], [[93, 95, 124]], [[75, 87, 106]], [[49, 73, 91]], [[48
    [[[79, 85, 100]], [[95, 103, 123]], [[90, 86, 106]], [[87, 94, 110]], [[49, 72, 88]], [[3
    [[[48, 55, 69]], [[54, 61, 72]], [[58, 65, 77]], [[51, 69, 86]], [[52, 82, 101]], [[35, 4
    [[[49, 74, 83]], [[74, 96, 104]], [[98, 113, 130]], [[109, 124, 155]], [[113, 128, 152]],
    [[[101, 149, 182]], [[119, 161, 174]], [[157, 177, 189]], [[198, 202, 227]], [[128, 120,
    [[[116, 136, 172]], [[105, 121, 135]], [[88, 100, 107]], [[73, 77, 94]], [[66, 63, 83]],
    [[[37, 43, 52]], [[41, 46, 52]], [[51, 55, 61]], [[61, 62, 69]], [[63, 63, 78]], [[40, 41
    [[[43, 43, 49]], [[50, 50, 56]], [[53, 52, 58]], [[52, 51, 57]], [[50, 49, 58]], [[41, 43
    [[[30, 29, 34]], [[31, 28, 34]], [[30, 27, 33]], [[28, 27, 32]], [[31, 32, 36]], [[37, 38
    [[[29, 28, 33]], [[30, 28, 33]], [[29, 27, 32]], [[28, 27, 32]], [[28, 28, 30]], [[40, 38
    [[[30, 28, 33]], [[30, 28, 32]], [[28, 28, 32]], [[28, 27, 32]], [[29, 27, 31]], [[35, 32
    [[[32, 30, 36]], [[32, 30, 35]], [[31, 31, 35]], [[31, 31, 36]], [[32, 31, 35]], [[33, 32
    [[[31, 30, 36]], [[31, 29, 34]], [[30, 28, 33]], [[30, 28, 34]], [[30, 30, 34]], [[30, 30
    [[[30, 28, 33]], [[30, 28, 32]], [[29, 27, 31]], [[29, 28, 30]], [[30, 30, 32]], [[29, 29
    [[[31, 29, 33]], [[31, 29, 33]], [[30, 29, 33]], [[29, 28, 31]], [[30, 29, 32]], [[30, 29
    [[[31, 30, 35]], [[31, 31, 34]], [[31, 31, 35]], [[34, 32, 36]], [[35, 32, 35]], [[34, 32
    [[[45, 40, 44]], [[45, 40, 41]], [[44, 39, 40]], [[46, 40, 42]], [[48, 41, 42]], [[45, 41
    [[[43, 33, 35]], [[45, 34, 34]], [[47, 37, 36]], [[44, 35, 33]], [[40, 31, 31]], [[39, 32
    [[[56, 44, 51]], [[51, 37, 38]], [[56, 46, 43]], [[55, 48, 44]], [[48, 43, 44]], [[51, 41
], dtype=np.float32)

# Remove the extra singleton dimension (1)
custom_image = np.squeeze(custom_image, axis=2)  # Now (30, 30, 3)

# Flatten it to 1D array (2700 values)
custom_image_flattened = custom_image.reshape(-1)
```

In [6]:
```python
image_in[:] = custom_image_flattened
```

In [7]:
```python
cnn_ip = overlay.cnn
```

In [8]:
```python
import time
cnn_ip.write(0x00, 0x04)  # Reset
time.sleep(0.01)            # Small delay
cnn_ip.write(0x00, 0x00)  # Clear reset
```

In [9]:
```
dma0.sendchannel.transfer(image_in)

# Prepare receiving buffer
dma1.recvchannel.transfer(probability_out)

# Start the CNN IP
cnn_ip.write(0x00, 0x01)  # Set ap_start=1

# Wait for DMA transfer to complete
dma0.sendchannel.wait()
dma1.recvchannel.wait()
```

In [10]:
```
predicted_class = np.argmax(probability_out)
print(f"Predicted Class: {predicted_class}")

print("\nClass Probabilities:")
for i, prob in enumerate(probability_out):
    print(f"Class {i}: {prob:.4f}")
```

```
Predicted Class: 15

Class Probabilities:
Class 0: 0.0000
Class 1: 0.0000
Class 2: 0.0000
Class 3: 0.0000
Class 4: 0.0000
Class 5: 0.0000
Class 6: 0.0000
Class 7: 0.0000
Class 8: 0.0000
Class 9: 0.0000
Class 10: 0.0000
Class 11: 0.0000
Class 12: 0.0000
Class 13: 0.0000
Class 14: 0.0000
Class 15: 1.0000
Class 16: 0.0000
Class 17: 0.0000
Class 18: 0.0000
Class 19: 0.0000
Class 20: 0.0000
Class 21: 0.0000
Class 22: 0.0000
Class 23: 0.0000
Class 24: 0.0000
Class 25: 0.0000
Class 26: 0.0000
Class 27: 0.0000
Class 28: 0.0000
Class 29: 0.0000
Class 30: 0.0000
Class 31: 0.0000
Class 32: 0.0000
Class 33: 0.0000
Class 34: 0.0000
Class 35: 0.0000
Class 36: 0.0000
Class 37: 0.0000
Class 38: 0.0000
Class 39: 0.0000
Class 40: 0.0000
Class 41: 0.0000
Class 42: 0.0000
```