Code:

def mcp\_neuron(inputs, weights, threshold):

summation = sum(i \* w for i, w in zip(inputs, weights))

return 1 if summation >= threshold else 0

# AND Gate

def AND(x1, x2):

return mcp\_neuron([x1, x2], [1, 1], 2)

# OR Gate

def OR(x1, x2):

return mcp\_neuron([x1, x2], [1, 1], 1)

# NOT Gate

def NOT(x1):

return mcp\_neuron([x1], [-1], 0)

# NOR Gate

def NOR(x1, x2):

return mcp\_neuron([x1, x2], [-1, -1], 0)

# XOR Gate (using hard-coded logic)

def XOR(x1, x2):

return (x1 ^ x2) # XOR can't be represented by single-layer MCP

# Testing

print("AND")

for x in [(0,0), (0,1), (1,0), (1,1)]:

print(f"{x} -> {AND(\*x)}")

print("\nOR")

for x in [(0,0), (0,1), (1,0), (1,1)]:

print(f"{x} -> {OR(\*x)}")

print("\nNOT")

for x in [0, 1]:

print(f"{x} -> {NOT(x)}")

print("\nNOR")

for x in [(0,0), (0,1), (1,0), (1,1)]:

print(f"{x} -> {NOR(\*x)}")

print("\nXOR")

for x in [(0,0), (0,1), (1,0), (1,1)]:

print(f"{x} -> {XOR(\*x)}")  
  
Output:

