

| **AY**  **CLASS** | **: 2023\_24**  **: BE E&TC** | **Sem**  **DATE** | **: II**  **:** |
| --- | --- | --- | --- |
| **SUBJECT ROLL NO.** | **: Mobile Computing**  **: 42314** | **EXPT. No. BATCH** | **: 01**  **: R8** |

**TITLE**: CODE DIVISION MULTIPLE ACCESS

**OBJECTIVE**: Simulate and analyze the operations of multiple access techniques for CDMA

**SOFTWARE USED:** OS: Unix or windows 7/8/10,

Processor: i3/i5/i7,

Software: Python (Jupyter Notebook) or java.

# THEORY:

CDMA stands for Code Division Multiple Access. It is a digital cellular standard that utilizes spread-Spectrum Technology. It spreads the signal over a fully available spectrum or over multiple channels through division. It is a channelization protocol for Multiple Access, where information can be sent simultaneously through several transmitters over a single communication channel.

It is achieved in below steps: A signal is generated which extends over a wide bandwidth. The code which performs this action is called spreading code. Later, a specific signal can be selected with a given code even in the presence of many other signals. It is mainly used in mobile networks like 2G and 3G. It is a more secure and private line. It has good voice and data communication capabilities.

# PROCEDURE OR WORKING:

* 1. The station encodes its data bit as follows. If bit = 1 then +1

If bit = 0 then -1

No signal (interpreted as 0) if station is idle

* 1. Each station is allocated a different orthogonal sequence (code) which is N bit long for N stations
  2. Each station does a scalar multiplication of its encoded data bit and code sequence.
  3. The resulting sequence is then stored on the channel.
  4. Since the channel is common, amplitudes add up and hence resultant channel sequence is the sum of sequences from all channels.
  5. If station 1 wants to listen to station 2, it multiplies (inner product) the channel sequence with code of station S2.
  6. The inner product is then divided by N to get data bit transmitted from station 2.



# THEORETICAL CALCULATIONS:

To see how CDMA works, we must understand orthogonal sequences (also known as chips).Let N be the number of stations establishing multiple access over a common channel. Then the properties of orthogonal sequences can be stated as follows:

An orthogonal sequence can be thought of as a 1xN matrix. Eg: [+1 -1 +1 -1] for N = 4.

Scalar multiplication and matrix addition rules follow as usual. Eg: 3.[+1 -1 +1 -1] = [+3 -3 +3 -3]

Eg: [+1 -1 +1 -1] + [-1 -1 -1 -1] = [0 -2 0 -2]

Inner Product: It is evaluated by multiplying two sequences element by element and then adding all elements of the resulting list.

Inner Product of a sequence with itself is equal to N [+1 -1 +1 -1].[+1 -1 +1 -1] = 1 + 1 + 1 + 1 = 4

Inner Product of two distinct sequences is zero [+1 -1 +1 -1].[+1 +1 +1 +1] = 1-1+1-1 = 0

# CODE:

import numpy as np c1=[1,1,1,1]

c2=[1,-1,1,-1]

c3=[1,1,-1,-1]

c4=[1,-1,-1,1]

rc=[]

print("Enter the data bits :") d1=int(input("Enter D1 :")) d2=int(input("Enter D2 :")) d3=int(input("Enter D3 :")) d4=int(input("Enter D4 :")) r1=np.multiply(c1,d1) r2=np.multiply(c2,d2) r3=np.multiply(c3,d3) r4=np.multiply(c4,d4) resultant\_channel=r1+r2+r3+r4;

print("Resultant Channel",resultant\_channel)

Channel=int(input("Enter the station to listen for C1=1 ,C2=2, C3=3 C4=4 : ")) if Channel==1:



rc=c1

elif Channel==2:

rc=c2

elif Channel==3:

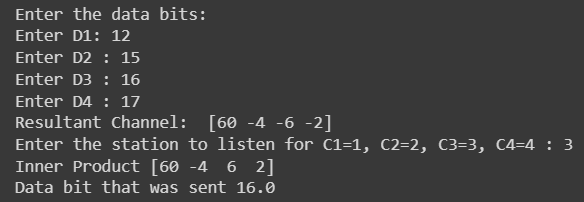
rc=c3

elif Channel==4:

rc=c4 inner\_product=np.multiply(resultant\_channel,rc) print("Inner Product",inner\_product) res1=sum(inner\_product) data=res1/len(inner\_product)

print("Data bit that was sent",data)

# OUTPUT



**CODE:**

import numpy as np

# Function to generate Walsh matrix def generate\_walsh\_matrix(n):

if n == 1:

return np.array([[1, 1], [1, -1]]) else:

prev\_matrix = generate\_walsh\_matrix(n-1)

upper = np.concatenate((prev\_matrix, prev\_matrix), axis=1) lower = np.concatenate((prev\_matrix, -prev\_matrix), axis=1) return np.concatenate((upper, lower), axis=0)

# Function to encode data using CDMA technique def cdma\_encode(data\_bits, walsh\_matrix):

encoded = np.zeros\_like(walsh\_matrix[0]) for i, bit in enumerate(data\_bits):

encoded += bit \* walsh\_matrix[i] return encoded

# Function to decode a specific channel in CDMA

def cdma\_decode(encoded\_data, walsh\_matrix, channel):

decoded = np.dot(encoded\_data, walsh\_matrix[channel]) / len(walsh\_matrix[channel]) return decoded

# Get number of data bits from user

num\_bits = int(input("Enter the number of data bits: "))

# Get data bits from user data\_bits = []

print("Enter the data bits:") for i in range(num\_bits):

bit = int(input(f"Enter bit {i + 1}: ")) data\_bits.append(bit)

#Defining x=1

while (2\*\*x<num\_bits):

x=x+1

# Generate Walsh matrix

walsh = generate\_walsh\_matrix(x) np.set\_printoptions(threshold=np.inf) # Set printing options print("\nGenerated Walsh Matrix:")

print(walsh)

# Encode data using CDMA technique encoded\_data = cdma\_encode(data\_bits, walsh)

# Get channel to decode from user

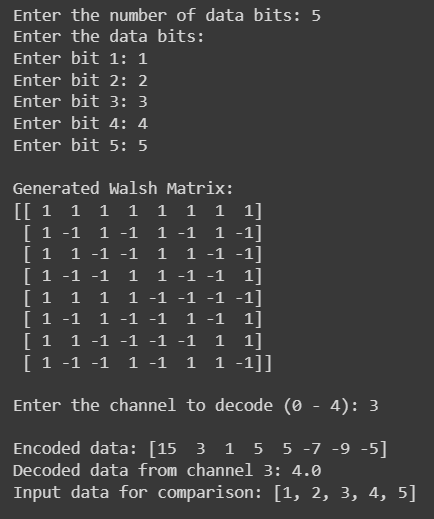
decode\_channel = int(input(f"\nEnter the channel to decode (0 - {num\_bits - 1}): "))

# Decode the selected channel

decoded\_channel = cdma\_decode(encoded\_data, walsh, decode\_channel) print(f"\nEncoded data: {encoded\_data}")

print(f"Decoded data from channel {decode\_channel}: {decoded\_channel}") print(f"Input data for comparison: {data\_bits}")

# OUTPUT:



1. **CONCLUSION:**

In the experiment described above, we explored Code Division Multiple Access (CDMA) and its applications in spread spectrum technology within cellular networks. We implemented CDMA using two techniques: In the first technique, we utilized PN sequences to implement CDMA, allowing us to encode and decode 4 bits of data. In the second technique, we employed the Walsh Matrix, offering flexibility for users to choose the number of bits for encoding and decoding. We ran the code, observing the generation of matrices and the encoding-decoding process.

By experimenting with these two access techniques, we effectively simulated and analyzed multiple access methods for CDMA.

# SIGNATURE

**REFERENCES**:

* 1. “Mobile Communications” – Jochen Schiller.