Sinhgad Institutes

Sinhgad Technical Educational Society's
SINHGAD COLLEGE OF ENGINEERING
VADGAON PUNE-41
Department of Electronics & Telecommunication

Experiment No. _ 01

Subject: - Mobile Computing

Name of the Stude

Date: _____ **Marks & Signature: -**

Subject Teacher

TLE:

implement a basic function of Code Division Multiple Access (CDMA) to test the orthogonally & tocorrelation of a code to be used for CDMA operation.

M:

asic function of Code Division Multiple Access (CDMA).

BJECTIVES:

understand function of CDMA used to testorthogonally and autocorrelation of a code

OFTWARE & HARDWARE REQUIREMENTS:

OS.: Unix or windows 7/8/10,

Processor: i3/i5/i7

Software: Python (Jupyter Notebook) or java

THEORY-CONCEPT

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

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

How does CDMA work?

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)
```

CONCLUSION:

The experiment successfully showcased the principles of CDMA, demonstrating orthogonality & auto-correction of codes for multiple access through chip multiplication & inner product calculation, it is effectively transmitted data bits over a common channel validating the efficiency of CDMA in maintaining privacy & enabling secure communication.

QUESTIONS:

1. Write a note on MAC Protocol?

A MAC (Medium access control) protocol is a set of rules that govern how devices share a single communication channel in a network. It prevents data collision & ensures efficient data transmission by regulating access to share medium

- Collission avoidance: Devices avoid transmitting simultaneously to prevent data corruption
- Fairness: Ensures all devices have a chance to transmit data without being unfairly blocked by others
- efficiency: Maximizes channel utilization to minimize wasted bandwidth & transmission time.

Common MAC protocols include Carrier sense multiple access with collision detection (CSMA/CD) used in ethernet networks & wifi CSMA/CA (Carrier sense multiple access with collision avoidance).

2. Write down difference between FDMA TDMA and CDMA?

Feature	FDMA	TDMA	CDMA
Resource Sharing	(Frequency division multiple access) frequency bands	(Time division multiple access) Timeslots	(Code division multiple Access) Entire frequency spectrum
Code Requirement	Not Required	Not Required	Required for each user
Synchronization	Not Required	Required for all users	Not required
Data Rate	lower	Medium	Higher
Cell capacity	lower	lower	higher

Date

MR – Marks for Regularity, MP – Marks for Presentation, MU – Marrk for Understanding

main.py

Save Run

```
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)

```

Shell Clear

```
Enter the data bits :
Enter D1 :1
Enter D2 :1
Enter D3 :1
Enter D4 :1
Resultant Channel [4 0 0 0]
Enter the station to listen for C1=1 ,C2=2, C3=3 C4=4 : 1
Inner Product [4 0 0 0]
Data bit that was sent 1.0
>
```