

# **Assignment 3**

Name – Sati, Ankit

Date – 10/14/2021

Section – 001

SID – as14128

Total in points (Maximum 100 points)–

Professors Comments –

Affirmation of Independent Effort – Ankit Sati

## Answers

1.

- a. SMTP uses the TCP protocol in the transport layer.

Why ? - If you lose several packets in the middle of the message the recipient might not even receive the message and if they do they might be missing key information. This makes TCP more appropriate because it ensures that every packet is delivered. UDP is NOT a reliable ordered data stream channel.

- b. Both the applications use a combination of protocols in order to **deliver instant messages in real time** . Some of the protocols that are used are mentioned below.

### Whatsapp

- open standard Extensible Messaging and Presence Protocol (**XMPP**) - Extensible Messaging and Presence Protocol is an open communication protocol designed for instant messaging, presence information, and contact list maintenance.
- **Instant messaging (IM)** technology is a type of online chat allowing real-time text transmission over the Internet or another computer network. Messages are typically transmitted between two or more parties, when each user inputs text and triggers a transmission to the recipient(s), who are all connected on a common network.
- **SIMPLE** - the Session Initiation Protocol for Instant Messaging and Presence Leveraging Extensions, is an instant messaging (IM) and presence protocol suite based on Session Initiation Protocol (SIP) managed by the Internet Engineering Task Force.[1] Contrary to the vast majority of IM and presence protocols used by software deployed today, SIMPLE is an open standard like XMPP.
- **TLS** – This is used for encryption and security by almost all the applications that use the real time messages.

### iMessage

- The iMessage protocol is based on **the Apple Push Notification service (APNs)—a proprietary, binary protocol**. It sets up a Keep-Alive connection with the Apple servers. Every connection has its own unique code, which acts as an identifier for the route that should be used to send a message to a specific device. The connection is encrypted with TLS using a client-side certificate, that is requested by the device on the activation of iMessage.
- **SIMPLE** - the Session Initiation Protocol for Instant Messaging and Presence Leveraging Extensions, is an instant messaging (IM) and presence protocol suite based on Session Initiation Protocol (SIP) managed by the Internet Engineering Task Force.[1] Contrary to the vast majority of IM and presence protocols used by software deployed today, SIMPLE is an open standard like XMPP.
- **TLS** – This is used for encryption and security by almost all the applications that use the real time messages.

- c. The two different server placements philosophies that are adopt by CDNs are shown below.

### 1. Enter deep

- It is pioneered by Akamai. It deploys server clusters in access ISPs (ISPs direct accessing end users) all over the world.

- The goal of Enter deep is to get close to end users, thereby improving user-perceived delay and throughput by decreasing the number of links.
- Routers between the end user and then CDN cluster from which it receives content. Because of this highly distributed design, the task of maintaining and managing the clusters become challenging.

## 2. Bring home

- It can be taken by Limelight and other CDN companies; it brings the ISPs home by building large clusters at a smaller number of key locations and connecting these clusters using a private high-speed network.
- Instead of getting inside the access ISPs, these CDNs typically place each cluster at a location that is simultaneously near the point of presence of many tier-1 ISPs.
- Compared with the enter-deep design, the bring-home design typically results in lower maintenance and management overhead, possibly at the expense of higher delay and lower throughput to end users.

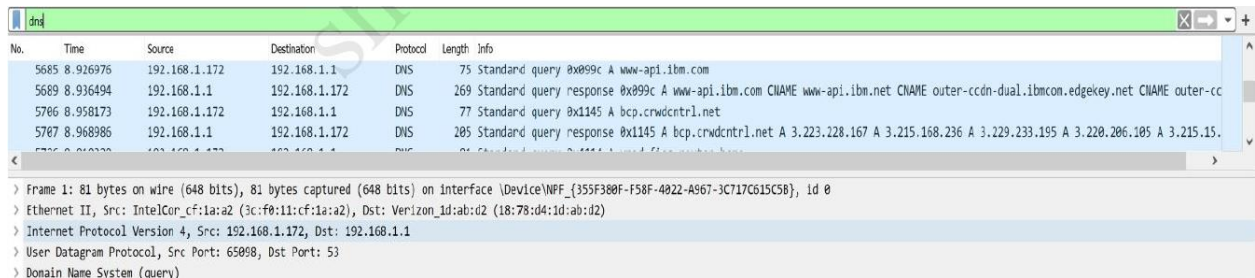
### d. HTTP response codes

- **201 – Created** - The request has been fulfilled and resulted in a new resource being created. The newly created resource can be referenced by the URI returned in the entity of the response, with the most specific URI for the resource given by a Location header field. The response should include an entity containing a list of resource characteristics and location from which the user or user agent can choose the one most appropriate. The entity format is specified by the media type given in the Content-Type header field.
- **404 - The HTTP 404** Not Found client error response code indicates that the server can't find the requested resource. Links that lead to a 404 page are often called broken or dead links and can be subject to link rot. A 404 status code does not indicate whether the resource is temporarily or permanently missing.
- **500** - The HyperText Transfer Protocol (HTTP) **500** Internal Server Error server error response code indicates that the server encountered an unexpected condition that prevented it from fulfilling the request. This error response is a generic "catch-all" response. It means that the server encountered an unexpected condition that prevented it from fulfilling the request. This error is usually returned by the server when no other error code is suitable.

### e. Key differences between **Download and delete** & **Download and keep** in POP3.

- **Download and delete** downloads the message from the server to the local mail reader program, and then deletes the message from the server; the only copy of the message is now on your local computer.
- **Download and keep** (a setting usually called "leave messages on server") downloads the message from the server to the local mail reader, and does not delete the message from the server; the message exists on the server as well as the local computer.
- In the download-and- delete mode, client receives messages from a POP, then delete the messages.
- In the download-and-keep mode, client receives messages from a POP and store messages, never deleted messages.

- f. Advantages of P2P over Client server architecture.
- No need for a network administrator.
  - Network is fast and inexpensive to setup and maintain.
  - Each computer can make backup copies of its data to other computers for security.
- g. Protocols used.
- Application Layer protocols – HTTP, DNS.
  - Transport Layer protocols – TCP, UDP.



No.	Time	Source	Destination	Protocol	Length	Info
5685	8.926976	192.168.1.172	192.168.1.1	DNS	75	Standard query 0x899c A www-api.ibm.com
5689	8.936494	192.168.1.1	192.168.1.172	DNS	269	Standard query response 0x899c A www-api.ibm.com CNAME www-api.ibm.net CNAME outer-ccdn-dual.ibmcom.edgekey.net CNAME outer-cc
5766	8.958173	192.168.1.172	192.168.1.1	DNS	77	Standard query 0x1145 A bcp.crowdctrl.net
5767	8.968986	192.168.1.1	192.168.1.172	DNS	285	Standard query response 0x1145 A bcp.crowdctrl.net A 3.223.228.167 A 3.215.168.236 A 3.229.233.195 A 3.220.206.105 A 3.215.15.

> Frame 1: 81 bytes on wire (648 bits), 81 bytes captured (648 bits) on interface \Device\NPF\_{355F380F-F58F-4022-A967-3C717C615C5B}, Id 0  
 > Ethernet II, Src: IntelCor\_cf:1a:a2 (3c:f0:11:cf:1a:a2), Dst: Verizon\_1d:ab:d2 (18:78:d4:1d:ab:d2)  
 > Internet Protocol Version 4, Src: 192.168.1.172, Dst: 192.168.1.1  
 > User Datagram Protocol, Src Port: 65088, Dst Port: 53  
 > Domain Name System (query)

2. True or False
- False – To call a specific process in a host, we also need the port and other specific details.
  - False
  - False – UDP does not ensure the packet delivery for every packet.
  - True – This means that the application can afford to loose a packet here or there but should not loose many packets.
3. Number of RTT images needed when
- Non persistent HTTP - **2 RTT per image = 2 + 10 = 12RTT** in this case.
  - Persistent HTTP –
    - Without pipelining – 2 + 5 = 7RTT
    - With pipelining – 2 + 1 = 3RTT.**
4. HTTPS request format
- Definitions
    - **Method** - HTTP request methods specify the action to perform through the request. **GET is the method that is requested here.** This is a request sent by the user to the host.
    - **Path** - The path identifies the resource on the server. if none is present in the original URI, it MUST be given as **"/" (the server root).**
    - **Accepted response** - This is called a **relative quality factor**. It specifies what language the user would prefer, on a scale of 0 to 1, as can be seen from the HTTP/1.1 Specification, §14.4. **text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,\*/\*.**
    - **Multiple Users** –
      - The User-Agent lets servers and network peers identify the application, operating system, vendor, and/or version of the requesting user agent. **Ex** - Gecko/geckotrail indicates that the browser is based on Gecko. (On the desktop, geckotrail is always the fixed string 20100101.)

- So naturally when a HTTP packet is transmitted around the internet it can be received at any type of client via different browser applications hosted by various operating systems.
- In order to ensure compatibility across different browser platforms around different operating systems, multiple user agents are listed for the request to be processed successfully without facing any issues for retrieving, rendering and facilitating end user interaction consisting of web content.

b. Response specific queries.

- h. Was it Successful ? - Yes it was because the status returned 200 (OK) which means that the request was completed.
- i. **UTF 8** - tells the web browser or email application how to interpret the text characters in your HTML. **charset=UTF-8** stands for Character Set = Unicode Transformation Format- 8. It is an octet (8-bit) lossless encoding of Unicode characters.
- j. **Set-Cookie** - The Set-Cookie HTTP response header is used to send a cookie from the server to the user agent, so that the user agent can send it back to the server later. To send multiple cookies, multiple Set-Cookie headers should be sent in the same response.
- k. **NO**. date modified header shows the last modified document date. **Date** HTTP header contains the date and time at which the message was generated. It is supported by all the browsers.

5. Average size = **250Kbits**

Average Request = **60 Req/sec**

Average data rate to browsers = **100 Mbps**

Access link rate = **15 Mbps**

RTT to any origin server = **2 sec**

a. Traffic intensity at LAN =

$$(\text{Avg req rate} * \text{Object size}) / 100 =$$

$$(60 * 0.25) / 100 = 15/100 = \mathbf{0.15}$$

b. Traffic Intensity at access Link

$$(\text{Avg req rate} * (\text{Object size} / \text{Access link rate})) =$$

$$(60 * (0.25/15)) / 100 = 1$$

c.

- 1) local cache which uses web cache. This is a cheaper operation and comparatively easier to achieve.
- 2) A fatter access link which increases access link speed. This is comparatively expensive but will increase the speed comparatively.

- d. Total Delay –  
 Cache hit rate – **0.4**  
 Sec Delay – **0.01**  
 Therefore, 40% requests satisfied at cache, 60% requests satisfied at origin  
 Access Link Utilization: 60% of requests use access link  
 Data Rate to Browsers Over Access Link =  $0.6 * 1.50 \text{ Mbps} = .9 \text{ Mbps}$   
 Utilization =  $0.9 / 1.50 = .60$   
 Total Delay =  $0.6 * (\text{delay from origin servers}) + 0.4 * (\text{delay when satisfied at cache})$   
 =  $0.6 (2.00 \text{ sec}) + 0.4 (0.01 \text{ secs}) = \mathbf{1.204 \text{ secs}}$

6. Different distributions.

File size  $F = 25 \text{ Gbits}$   
 (Assuming  $1 \text{ Gbit} = 1000 \text{ Mbits}$ )  
 $U_s = 50 \text{ Mbps} = 0.05 \text{ Gbps}$  (server upload rate)  
 $D_i = 5 \text{ Mbps} = 0.005 \text{ Gbps}$  (Peer download rate)  
 $N = 10, 100, 1000$   
 $u = 250 \text{ Kbps}, 750 \text{ Kbps}, 2.5 \text{ Mbps} = 0.25 \text{ Mbps}, 0.75 \text{ Mbps}, 2.5 \text{ Mbps}$

**server transmission:**

time to send one copy:  $F/U_s = 25/0.05 = 500 \text{ secs}$   
 time to send  $N$  copies =  $N * F/U_s$   
 i)  $N = 10 : 10 * 500 = 5000 \text{ secs}$   
 ii)  $N = 100 : 100 * 500 = 50,000 \text{ secs}$   
 iii)  $N = 1000 : 1000 * 500 = 500,000 \text{ secs}$

**client download:**

time to download one copy =  $F/D_i = 25/0.005 = 5000 \text{ secs}$

**1) Client-Server distribution:** (Files will be sent sequentially)

**time to distribute  $F$  to  $N$  clients using client-server approach:**  $D_{C-s} > \text{Max}\{NF/u_s, F/D_i\}$

- i)  $N = 10 : D_{C-s} = \text{Max}\{5000, 5000\} = \mathbf{5000 \text{ secs}}$   
 ii)  $N = 100 : D_{C-s} = \text{Max}\{5000, 50000\} = \mathbf{50,000 \text{ secs}}$   
 iii)  $N = 1000 : D_{C-s} = \text{Max}\{5000, 500000\} = \mathbf{500,000 \text{ secs}}$

**2) P2P distribution:** (Will download aggregated)

**Minimum Distribution Time (T):**

- i)  $N = 10, u = 250 \text{ Kbps} = 5000 \text{ seconds}$   
 ii)  $N = 10, u = 750 \text{ Kbps} = 5000 \text{ seconds}$   
 iii)  $N = 10, u = 2.5 \text{ Mbps} = 5000 \text{ seconds}$   
 iv)  $N = 100, u = 250 \text{ Kbps} = 33333 \text{ seconds}$   
 v)  $N = 100, u = 750 \text{ Kbps} = 20000 \text{ seconds}$

- vi)  $N = 100$ ,  $u = 2.5\text{Mbps} = 8334\text{ Seconds}$
- vii)  $N = 1000$ ,  $u = 250\text{Kbps} = 83334\text{ seconds}$
- viii)  $N = 1000$ ,  $u = 750\text{Kbps} = 31250\text{ seconds}$
- ix)  $N = 1000$ ,  $u = 2.5\text{Mbps} = 9804\text{ seconds}$ .

## 7. Algorithms for wave functions.

### a) NRZI

```

For i = 1 to input_length
  If input(i) is one {
    If output(i-1) is "+"; output(i) = "-"
    Else output = "+"
  } else output(i) = output(i-1)

```

### b) Bipolar AMI

```

For i = 1 to input_length
  If input(i) is one {
    If previous is "-"; output(i) = "+"; previous = "+"
    Else output(i) = "-"; previous = "-"
  } else output(i) = "neutral"

```

### c) Pseudo Ternary

```

For i = 1 to input_length
  If input(i) is zero {
    If previous = "-"; output(i) = "+"; previous = "+"
    Else output(i) = "-"; previous = "-"
  } else output(i) = "neutral"

```

### d) Manchester

```

For i = 1 to input_length
  If input(i) is zero; output(2*i-1) = "+"; output(2*i) = "-"
  Else output(2*i-1) = "-"; output(2*i) = "+"

```

### e) Diff Manchester

```

If output(i) is one; output(1) = "+"; output(2) = "-"
Else output(1) = "-"; output(2) = "+"
For i = 2 to input_length
  If input(i) is one; output(2*i-1) = output(2*i-2); output(2*i) = output(2*i-3)
  Else output(2*i-1) = output(2*i-3); output(2*i) = output(2*i-2)

```

**f) B8ZS**

```
counter = 0;
x((i-1)*n+1:i*n) = -lastbit;
lastbit = -lastbit;

for i = 1:length(t)
    if t(i)>counter
        counter = counter + 1;
        if x(i)==lastbit
            result(counter:counter+4) = 0;
            counter = counter + 4;
        else
            if(x(i)==0)
                result(counter) = 0;
            else
                result(counter) = 1;
            lastbit = -lastbit;
        end
    end
end
```

**g) HDB3**

```
for i = 1:length(t)
    if t(i)>counter
        counter = counter + 1;
        if x(i)==lastbit
            result(counter-3:counter) = 0;
        else
            if(x(i)==0)
                result(counter) = 0;
            else
                result(counter) = 1;
            lastbit = -lastbit;
        end
    end
end
```

**h) NRZL**

```
For i = 1 to input_length
    If input(i) is zero; output(i) = "+"
    Else output(i) = "-"
end
```



8. We are given the following information

a. Numbers are encoded as :

$$c_m = b_m - b_{m-1}$$

Numbers are received as :

$$a_m = c_m \bmod 2$$

We can deduce the below proof from above.

It is given that:

$$\Rightarrow c_m = b_m - b_{m-1}$$

$$\Rightarrow b_m - b_{m-1} = (a_m + b_{m-1}) - b_{m-1} = a_m$$

**Combining the above equation, we get:**

$$c_m = a_m$$

Therefore we can conclude that we have not lost any of the signal and the **received values are equal to the transmitted values.**

**b. Bipolar AMI (Alternative Mark Invasion) is a return-to-zero (RZ) type.**

It is a return-to-zero (RZ) type encoding, where two non zero values are used. In other words in this coding technique the signal spends equal amount of time in 0 and non 0 parts.

9. Yes, this is true. This is a major side effect however the receiver would not the difference. However the sender will have a very good idea about the difference in signal.

$$+ - 0 + - 0 - +$$

The two ways in which this signal could have been sent is mentioned below

a.  $+ - 0 + - \text{+} - + = + - 0 + - \text{0} - +$

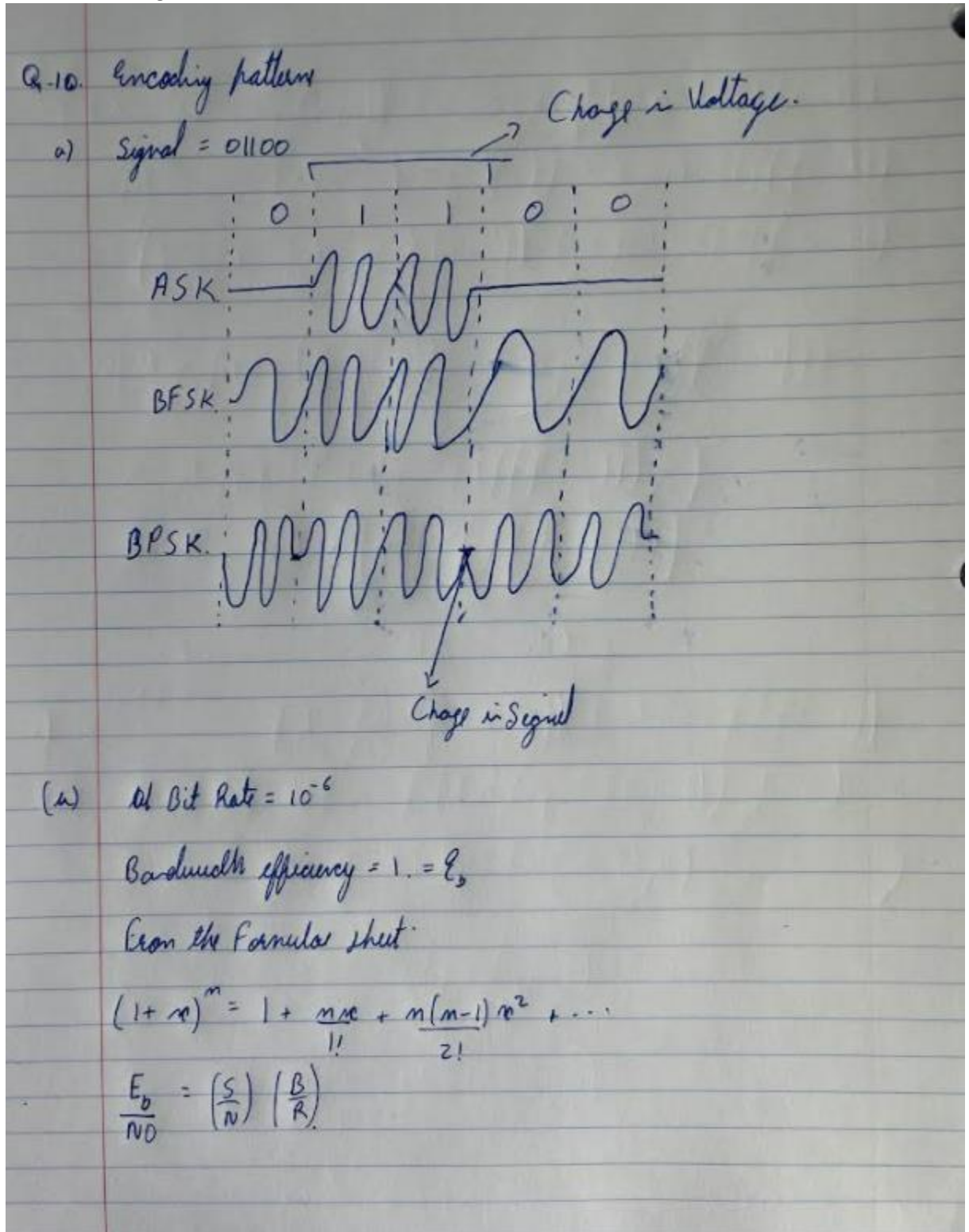
In this case the + has been converted to the Zero. (Error +)

b.  $+ - 0 + - \text{0} 0 - + = + - 0 + - \text{-} 0 - +$

This has been converted to the -. (Error 0)

10. Encoding patterns

- Attached in image below
- Attached in image below



$$\left(\frac{S}{N}\right)_{dB} = \left(\frac{E_b}{N_0}\right)_{dB}$$

(i) For ASK.

Signal to Noise ratio

$$\frac{E_b}{N_0} = 13.5 \text{ dB.} \quad \text{--- (1)}$$

(ii) For FSK.

It will be the same as ASK.

$$\frac{10^{-6}}{N_0} \text{ dB} = 13.5 \text{ dB.} \quad \text{--- (2)}$$

(iii) PSK.  $\left(\frac{E_b}{N_0}\right)_{dB} = 10.5$  ( $E_b = 10^{-6}$ )

$$\left(\frac{S}{N}\right)_{dB} = 10.5 \text{ dB.} \quad \text{--- (3)}$$

(iv) QPSK. (effective bandwidth is halved.)

$$\frac{R}{B} = 2.$$

$$\left(\frac{2}{B}\right)_{dB} = 3.$$

$$\left(\frac{S}{N}\right)_{dB} = 3 + 10.5 = 13.5 \text{ dB.} \quad \text{--- (3)}$$

c. More on signal to noise ratio.

i. Given

$$\text{SNR} = 35 \text{ db}$$

Also we know that  $\text{SNR} = 6.02n + 1.76 \text{ db}$ ,

$N = 5.59$  (ceiling value 6 bits (loosely upper bound))

$$\text{Number of quantization levels} = 2^n = 2^6 = \mathbf{64}$$

ii. Required data rate.

The required rate = (bits per sample \* sample per second)

$$6 \text{ bits/sample} * 7000 \text{ samples/seconds}$$

$$6 * 7000 = \mathbf{42000 \text{ bps}}$$