PILOT/SIGNATURE PATTERN BASED MODULATION TRACKING

6.1 TRANSMITTER AND RECEIVER

Each modulated signal is preceded by a unique ‘N’ bit pilot sequence (Manton, JH 2001). A switch in the transmitter shown in Figure 6.1 selects one of the modulated signals which have to be transmitted. AM signal waveform along with its pilot sequence is selected by the switch 1. The FM signal is selected by the switch 2. The PSK signal is selected by the switch 3. The selected signal is sent through the communication channel.

At the receiver shown in Figure 6.2, the received signal is given to the pilot tracker which automatically identifies the type of modulation by which the transmitted signal is modulated. The pilot tracker also uses the same type of mechanism used during the generation of Pilot Sequence at the transmitter side to generate the Pilot Sequence. The incoming pilot sequence and the generated pilot sequence are compared by the tracker and accordingly the modulation is identified. The correlation is done between the received Pilot Sequence and the generated Pilot Sequences at the pilot tracker. The pilot sequence which
correctly matches with the recovered pilot sequence helps to find the type of modulation done for the received signal.

Accordingly, the tracked signal is sent to its respective demodulators. The AM demodulator, demodulates the received signal and gives out the AM signal. The FM Demodulator gives out the FM signal and the data is obtained at the output of the PSK demodulator.

**6.2 ADAPTIVE CLUSTERING**

The MANET is a wireless network, where nodes communicate with each other using multi hop method in an infrastructure less environment and relies on a code division access scheme for data transfer. In this network architecture, nodes are organized into non-overlapping clusters. The clusters are controlled independently and as the nodes move, they are reconfigured dynamically (Li, CR & Gerla, M 1997).

The advantage of this network architecture includes;
1. Due to node clustering, spatial reuse of the bandwidth is possible.
2. Bandwidth can be shared or reserved in each cluster.
3. The clustering method enables the architecture to be stable even during topological changes caused by node motion, node failure and node insertion/removal.

This proposed architecture also helps in providing an efficient and stable infrastructure for the integration of different types of traffic in a dynamic radio network.
Figure 6.3 Conventional Cellular networks (Single-hop)

Figure 6.3 shows, the single hop cellular model wireless networks where, A, B, C and D are four fixed base stations which are connected by a wired backbone. Nodes 1 through 8 are all mobile nodes. A mobile node is a node which is only one hop away from a base station. Communication between two mobile nodes takes place through fixed base station and the wired backbone.

Multi hopping is defined as the ability of the radios to relay packets from one node to another without the use of base stations. It is mainly employed during disaster management in situations such as fire, earthquakes etc. Multi hopping through wireless repeaters reduces the battery power and increase network capacity.

Figure 6.4 Multi-hop situation when base station B fails occurs

In a single hop network, if a base station fails, a mobile node may not be able to access the wired network. But, in multi hop network as shown in Figure 6.4, if a base station, say, B fails, node 4 will access the base stations A or C through node 2 or node 5 which in turn acts as wireless multi-hop repeaters.
6.3 CLUSTER MAINTENANCE IN THE PRESENCE OF MOBILITY

In the dynamic radio network, usually topological changes occur when a node disconnects from or connects to all or part of its neighbours and this modifies the cluster structure. System performance is affected by frequent cluster changes, for that reason, it is important to design a cluster maintenance scheme to keep the cluster infrastructure stable. The cluster maintenance scheme was intended to minimize the number of node transitions from one cluster to another.

In this research work, as shown in Figure 6.5(a), there are 5 nodes in the cluster and the hop distance between the nodes should not be more than 2. Due to mobility of nodes, the topology changes with respect to the configuration as shown in Figure 6.5(b). Now it can be calculated as, \( d(4,1) = 3 \) and \( d(2,1) = 3 \) which is greater than 2, where \( d(i, j) \) is the hop distance between node \( i \) and \( j \). So the cluster needs to be reconfigured. In this experiment the highest connectivity node and its neighbors are allowed to stay in the original cluster, and the other nodes are removed. Each node only keeps the information of its one and two hop neighbours.

The following steps are required to maintain the cluster architecture:
Step 1: Check if any member of the cluster has moved out of its locality.
Step 2: If step 1 is successful, decide whether it should change cluster or remove the nodes that is not its locality from its cluster.
Consider the examples in Figure 6.5(b). Node 3 is the highest connectivity node. Thus node 3 and its neighbours \{2, 4, 5\} do no change cluster. However, node 1 should either join another cluster or form a new cluster. If a node intends to join a cluster, it has to check first if all the members of this cluster are in its locality and only it can join the cluster.

6.4 SCHEDULER BASED IMPLEMENTATION

In this work, the tracking algorithm (section 3.8 of this thesis) is implemented to suit realtime and includes a priority handler. Based on the priority of a node, the availability of the modulation scheme is varied.

For example:
1) Total number of slots= N
2) Total number of nodes= M
3) Different available modulation= K
   where the possible ‘K’ schemes are known to receiver.
4) Priority Table in Table 6.1 shows the priority in the transmission of ‘M’ nodes which are dynamic.

<table>
<thead>
<tr>
<th>Node</th>
<th>Priority during time ‘t1’</th>
<th>Priority during time ‘t2’</th>
<th>Priority during time ‘t3’</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>04</td>
<td>03</td>
<td>02</td>
</tr>
<tr>
<td>2</td>
<td>01</td>
<td>04</td>
<td>02</td>
</tr>
<tr>
<td>3</td>
<td>02</td>
<td>01</td>
<td>02</td>
</tr>
<tr>
<td>4</td>
<td>03</td>
<td>02</td>
<td>01</td>
</tr>
</tbody>
</table>

6.5 RULE ALLOCATION TABLE

The priority handler implements the following tracking algorithm to find the priority (in percentage) for the given nodes for various time intervals.
Algorithm

During interval ‘t₁’,
- Allocate 40% of N for node 1
- Allocate 30% of N for node 4
- Allocate 20% of N for node 3
- Allocate 10% of N for node 2

During interval ‘t₂’,
- Allocate 40% of N for node 2
- Allocate 30% of N for node 4
- Allocate 20% of N for node 3
- Allocate 10% of N for node 1

During interval ‘t₃’,
- Allocate 10% of N for node 4
- Allocate 30% of N for node 1
- Allocate 30% of N for node 2
- Allocate 30% of N for node 3

6.6 IMPLEMENTATION OF AUTOMATIC MODULATION IDENTIFICATION

The transmitter and receiver hardware architecture for transmitting various digitally modulated signals and receiving the particular modulated signal is discussed in the following topics.

6.6.1 Transmitter

Figure 6.6 shows the block diagram of the transmitter implemented in SDR. The Modulation selection switch selects a particular type of Digital Modulator to modulate the carrier signal generated by the sine/square wave carrier generator circuit using the information signal. The selection is based on the provided information about the dynamic channel after the mobility of nodes take place. The Digitally modulated signal before being transmitted is sent through the digital to analog converter, so that the signal can be sent as analog signal through the channel. Whenever there is a change in the node location and
thereby a change in the channel, the type of modulation concerned with that channel is chosen.

6.6.2 Receiver

At the receiver (Figure 6.7), the signal is received and sent to the analog to digital converter to convert the analog signal back to Digital Modulated signal. The noise present in the signal is filtered out using the Noise filtering circuit. The signal is then given to its respective Digital Demodulator, which demodulates the signal back to the original information signal. The respective demodulator is determined implementing the Automatic Modulation Identification technique.
6.6.3 Hardware Implementation

This FL 2440 ARM9 Embedded development board is used for designing and implementing the Automatic Modulation Identification technique.
Data from one node (Personal Computer) is sent to the board in a serial manner using the USB RS 232 cable. The program helps in identifying a modulation type and the data is modulated using that type and sent to the next node which is connected to the receiving node. The next node helps in processing the header and finding the exact modulation technique. Figure 6.8 shows the FL2440 ARM9 Embedded development board.

6.6.4 Snapshots showing the Hardware output in choosing the modulation/demodulation type

The following screenshots shows the operation of the hardware in selecting a particular modulation/demodulation type and performing the chosen modulation successfully.

Figure 6.9 Snapshots showing the process of hardware operation in choosing the QAM256 modulation type
Figure 6.9 shows the hardware process involved in choosing QAM256 modulation. The steps involved are choosing whether the modulation or demodulation action has to be performed, the amount of noise in the signal, choosing a particular modulation technique which is to be implemented and the successful performance of the modulation technique.

Figure 6.10 shows the hardware process involved in choosing QAM16 modulation.

Figure 6.10 Snapshots showing the process of hardware operation in choosing the QAM16 modulation type

Figure 6.11 shows the hardware process involved in choosing BASK modulation which is to be implemented and the successful performance of the modulation technique.
Figure 6.11 Snapshots showing the process of hardware operation in choosing the BASK modulation type

6.7 CLUSTERING MODEL IMPLEMENTATION

6.7.1 Setting Up Network Connections

The systems should be interconnected with Ethernet cable and the CRO terminals have to be connected to the audio input of the CPU to view the waveforms. The stepwise details to set up Network connections are listed below:

Step 1:
Execution Code set up:

i. Channel parameter files
channel_param_BASK.txt, channel_param_CPFSK.txt, channel_param_DQPSK.txt, channel_param_QASK.txt and channel_param_QAM16.txt.

i. Distance file distance.txt and IP address file ipaddress.txt

iii. Input files packet.txt or in.jpg

**Step 2:**

Userspace: python

Code Execution

Code Flow

It provides 3 options:

a. Configure Network
b. Send message
c. Exit

6.7.2 Configure Network

i. Select the first option in the main menu to configure the network, configuring the network enables the user to enter the own node or source node and then the user has to enter the total number of nodes in the network.

ii. All the channel parameters and IP addresses are loaded from text files to the respective arrays.

iii. The UDP port has to be opened for communication in asynchronous mode.

6.7.3 Send message

The second option from the main menu is then selected to send a message,

1. According to this option, the user has to enter the destination node to receive the message.

2. Next all the possible paths from source node to the destination node is found, and their corresponding weights depending on the channel parameters of each modulation technique is also noted.
3. Then print all the possible clusters with their corresponding weights. Also print the cluster that has the shortest path to the destination.
4. The best cluster is found depending on the weight.
5. Then modulate the input file and send the modulated file (ModulatedFile.bin) over the best cluster path.
6. The file is formed as packets of size 512 bytes and sent over the UDP.
7. Some details such as sending time, modulation technique and modulated file size are stored in the log file (temp1.txt) which will be present in the present working directory

6.7.4 Receiving message in asynchronous mode

1. The received file (RecModulatedFile.bin) will be stored in the present directory
2. Check if the received node is the destination node, if yes, then receive the message else forward the message to the next node depending on the best cluster path.
3. After the file is received, demodulate the file (DemodulatedFile.jpg or DemodulatedFile.txt) and store it in the present working directory.
4. If the file needs to be forwarded, the demodulated file (DemodulatedFile.jpg or DemodulatedFile.txt) will be again modulated and the modulated file (ModulatedFile.bin) will be sent to the next node.
5. Some of the details such as sending time, modulation technique and modulated file size, Total Network Throughput and node throughput are stored in the log file (ShortestLogFile.txt) in the present working directory of the destination node.
6. The forwarding node will have the log file (tempRec1.txt) which will contain the sending time, modulation technique and modulated file size.
7. The log files created will be named according to the own node number.
temp<own node number >.txt and tempRec<own node number >.txt
Example: if ownnode number is 2, the log files created will be temp2.txt for the sending message or tempRec2.txt for receiving message.
After every modulation and demodulation, two log files Receiver_Log.txt and Transmitter_Log.txt will be created.
6.8 TOPOLOGY DESCRIPTION USING ADAPTIVE CHANNEL MATRIX

Each node within a group of nodes which are placed in the network layer can modulate the data and demodulate the same within the group. The communication technique involves analysing the channel conditions and accordingly choosing that modulation technique to the neighbouring node and combining the routing with shortest path algorithm. This process is continuously followed by all the nodes whenever they receive data. The metrics involved include the network layer metrics, wireless channel based metrics, modulation based aspects etc.

6.8.1 Optimising Network Layer Routing Algorithms

The network layer is optimized with the shortest path routing between the source node and the destination node. This reduces the variability in delay which is also called as Jitter.

Assume, Data rate = B (in Mbps)
Number of switching stages = N
And, if utilization of each slot is given as ‘λ’, then, for a transmission waiting time equal to τT (0<τ<1), where ‘T’ is the individual stage transmission time, the jitter is calculated as,

\[ BJ = \lambda N [\tau T + T] + (1-\lambda) NT \]

\[ BJ = \lambda NT \tau T + \lambda NT + (1-\lambda) NT \]

Therefore, Jitter \( J = \left[ NT \left[ \lambda \tau + 1 \right] \right] / B \) (6.1)

The effect of jitter is more important than trying to minimize it. Once jitter effect is calculated, suitable schemes can be created using buffers wherever it is necessary to enable continuous transfer of packets. In this work, the effect of variation in each parameter of Eqn. 6.1 is observed and plotted in Figure 6.12 for adaptive \( \lambda \), adaptive B and adaptive T. The graph in Figure 6.12 (a) shows the linear increase in jitter as utilization of each slot is increased. In Figure 6.12 (b) jitter reduces as the data rate increases and Figure 6.12 (c) shows that by choosing the shortest path algorithm in the network layer, the individual stage transmission time ‘T’ is reduced which thereby reduces the jitter.
Figure 6.12(a) Jitter effects and its variations w.r.t Utilization

Figure 6.12(b) Jitter effects and its variations w.r.t Data rate

Figure 6.12 (c) Jitter effects and its variations w.r.t Transmission time for N stages
6.8.2 Configuration of Network layer routing algorithm

This work involves a Wireless LAN designed comprising of 9 nodes arranged in a Basic Service Set (BSS) without Access Point (AP) i.e., in an Ad-Hoc architecture. Each node is located in an assigned XY position with no roaming capability. Best effort Type of Service (TOS) is provided for the regulation of traffic. This network uses 802.11g physical standard which operates at 24 Mbps data rate and employs DSSS Modulation. The transmission power consumption is 0.005 W with packet reception Power threshold allocated as -95 and fragmentation threshold is assigned as 256 Bytes. The channel setting is also specified with 7 short retry limit and 4 long retry limit with a buffer size of 4608000 bits. Large packets if any will be dropped to facilitate smooth transfer of data without congestion.

6.8.3 Captured simulation results for Jitter analysis

While initializing the network, values are assigned to various attributes for each node belonging to the Wireless network. Figure 6.13 shows the basic attributes for node 2 of the designed wireless LAN. The position of the node in the network, time when the node is created in the network, destination node, type of service, MAC address are some of the basic features assigned.

![Figure 6.13 Basic network attributes and their values](image)
Figure 6.14 Wireless LAN parameters with their values

Figure 6.14 and 6.15 shows the values assigned for various Wireless LAN parameters. Some of them include how the BSS identifier is assigned, whether Access Point is initialized or not, data rate in bits per seconds, transmitting power in Watts, packet reception, RTS and fragmentation thresholds. Also, entries are made for parameters like retry limits, AP Beacon interval in seconds, maximum receive lifetime, buffer size, roaming capability of the node, large packet processing.

Figure 6.15 Assigning values for attributes
Figure 6.16 shows the captured results for no data dropped (bits/sec) due to buffer overflow for the overall network. Figure 6.17 shows the Graphical analysis is made for no data dropped (bits/sec) due to retry threshold exceeded which occurs when acknowledgments are not received for packets until the retry timer limit has exceeded. End to end delay (sec) of all packets due to MAC reception of all fragments individually is shown in Figure 6.18. The total load variation (bits/sec) by all nodes in the wireless network is shown in Figure 6.19. According to Figure 6.20, the delay variation of packets caused due to the media access (MAC) limitations such as collisions, back off periods, delayed acknowledgments etc., are shown. Figure 6.21 illustrates the overall network load variation which is measured for each BSS between the source node and the AP via the shared medium. He output captured and shown in Figure 6.22 illustrates the number of retransmission attempts by the network until the packet is successfully transmitted or the packet is discarded as the retry timer limit is reached and Figure 6.23 explains the total number of bits delivered to the destination per second.
Figure 6.17 Captured result showing data dropped due to exceeded Retry Threshold

Figure 6.18 Captured result showing delay in the considered wireless LAN

No data dropped due to exceeded Retry Threshold

Delay variations of packets (Jitter) with respect to time
Figure 6.19 Captured result showing Load in the considered wireless LAN

Figure 6.20 Captured result showing Media Access delay in the wireless LAN
Figure 6.21 Captured result showing Network Load in the wireless LAN

Figure 6.22 Captured result showing No. of Packets involved in retransmission attempts
Figure 6.23 Captured result showing Throughput in the wireless LAN

Figure 6.24 shows the captured results for the delay encountered by the data packets transmitted by Node 2 which is the source in the wireless network. The delay is mainly due to queuing and MAC delay. The result shown in Figure 6.25 illustrates the number of bits successfully received by the source Node 2 in the wireless LAN.

Figure 6.24 Captured result showing delay for (source) node 2 in the wireless LAN
Figure 6.25 Captured result showing throughput for (source) node 2 in the wireless LAN

Figure 6.26 Captured result showing delay for (destination) node 8 in the wireless LAN

No. of bits successfully received by source node per second

Delay variations of packets (Jitter) at the destination node
Figure 6.26 shows the captured results for the delay encountered by the data packets transmitted by Node 8 (destination) in the wireless network. The delay is mainly due to queuing and MAC delay. The result shown in Figure 6.27 illustrates the number of bits successfully received by the destination Node 8 in the wireless LAN. It is to be noted that no data is dropped either by buffer overflow or exceeded retry threshold. Also due to the network behaviour, the total load in all nodes and Network load at each BSS fluctuate abruptly with time thereby causing variable number of retransmission attempts.

6.8.4 Shortest path algorithm

Consider a network with four nodes in which a node has processing capability with which it performs the ability to identify the shortest neighbour, detect the corresponding channel conditions, perform suitable modulation to maintain the required SNR, etc. The concept of shortest path between two nodes can be illustrated by considering the source node as 1 and destination node as 3 as shown in Figure 6.28. The different paths that can...
exist between node 1 and node 3 are (i)1,4,3 (ii)1,2,4,3 (iii)1,2,3 (iv)1,4,2,3. For each path, based on channel conditions, the modulation that is best suited to maintain a high SNR is determined. For example: out of these four paths if the shortest path is 1-4-2-3 then the set of modulations needed for this path based on the channel 1 to 4, channel 4 to 2 and channel 2 to 3 is determined. In this work, five different modulation schemes are available at each node. The novelty of this work is the capability provided at each node to perform any of the five different modulations, transmit the same through the shortest path from the transmitter, independently detect the modulation and subsequently demodulate the data at the receiver.

![Figure 6.28 Topology for illustration with four nodes](image-url)

To design such an algorithm, the channel matrix and distance matrix is required. In channel matrix input is negative i.e (-1) when nodes are not connected. The channel conditions are given in the matrix for different modulation and are dynamically updated to include a time varying channel. The distance matrix is symmetric, diagonal elements are zero, if nodes are connected the value is non-negative and if input is not connected entries are assigned a large value.

6.8.5 Channel behaviour Vs Modulation Type

The Channel Parameter for 5 different modulations are shown for each link connecting the nodes are shown in Table 6.2.
Table 6.2 Channel behavior Vs Modulation type

<table>
<thead>
<tr>
<th></th>
<th>i)ASK</th>
<th>ii)BPSK</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.1 0.9 0.1 0.0 0.0</td>
<td>0.0 0.8 0.2 0.4 0.1</td>
</tr>
<tr>
<td></td>
<td>0.2 0.2 0.9 0.9 0.1</td>
<td>0.1 0.2 0.7 0.0 0.2</td>
</tr>
<tr>
<td></td>
<td>0.3 0.3 0.2 0.2 0.2</td>
<td>0.2 0.5 0.3 0.3 0.5</td>
</tr>
<tr>
<td></td>
<td>0.7 0.7 0.1 0.3 0.8</td>
<td>0.8 0.7 0.1 0.2 0.4</td>
</tr>
<tr>
<td></td>
<td>0.0 0.2 0.0 0.5 0.1</td>
<td>0.5 0.4 0.0 0.2 0.1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>iii) QAM</th>
<th>iv) QASK</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.4 0.6 0.0 0.5 0.7</td>
<td>0.2 0.6 0.1 0.2 0.9</td>
</tr>
<tr>
<td></td>
<td>0.0 0.7 0.1 0.1 0.3</td>
<td>0.5 0.9 0.4 0.5 0.0</td>
</tr>
<tr>
<td></td>
<td>0.3 0.3 0.5 0.8 0.7</td>
<td>0.0 0.0 0.6 0.7 0.6</td>
</tr>
<tr>
<td></td>
<td>0.9 0.9 0.2 0.3 0.0</td>
<td>0.1 0.2 0.8 0.4 0.3</td>
</tr>
<tr>
<td></td>
<td>0.6 0.1 0.7 0.2 0.1</td>
<td>0.3 0.1 0.0 0.1 0.7</td>
</tr>
</tbody>
</table>

v) SUNDAE

<table>
<thead>
<tr>
<th></th>
<th>0.8 0.2 0.4 0.6 0.3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.0 0.8 0.0 0.0 0.5</td>
</tr>
<tr>
<td></td>
<td>0.1 0.3 0.0 0.2 0.2</td>
</tr>
<tr>
<td></td>
<td>0.3 0.1 0.9 0.7 0.1</td>
</tr>
<tr>
<td></td>
<td>0.7 0.2 0.9 0.2 0.0</td>
</tr>
</tbody>
</table>

It has to be noted that the maximum value indicates good Channel Support.

6.9 AUTOMATIC MODULATION DETECTION BASED ON FEATURE EXTRACTION

Based on the observations, 20 parameters are extracted from each of the 10 modulated signals. The simulation is done using verilog coding and the algorithm extracts 20 features from a digitally modulated signal. These features can be collected by examining the statistical properties of both the signal and its normalized one. These features contain the necessary information to distinguish the signal between different modulations. The following snapshot as shown in figure 6.14 shows the detection of mod3 signal.
Signal mod3 is detected

Based on the 20 parameters mod3 signal is detected

Signal mod5 is detected

Detection based on the 20 parameters

Figure 6.29 Snapshot showing the automatic detection of mod3 signal

Figure 6.30 Snapshot showing the automatic detection of mod5 signal
From Figure 6.29 it can be seen that analysing the 20 parameters, the value obtained for the modulated signal is 110101011001010001. Based on the number, the corresponding modulated signal mod3 is enabled. Similarly, Figure 6.30 shows the detection of mod5 signal based on the 20 parameters analysed for the given signal. The value obtained after analysing the modulated signal is 101101011001010101. The corresponding modulated signal enabled is mod5 signal.

6.10 MODULATION CLASSIFICATION IN THE PRESENCE OF NOISE

The work deals with using a particular modulation technique which can be very much feasible for transmitting the signal in a low noise location. But, there is every possibility that noise can affect the transmitted modulated signal while transmitting through a communication channel. So, the channel has to be made a reliable information transmitter. This is achieved by implementing a channel coding technique to encode the signal at the transmitter and decode it at the receiver. The efficient and simple coding technique implemented in this work is Hamming encoding Technique which performs error detection and correction.

6.10.1 Error Detection and Correction

Hamming codes fall under the category of FEC which uses a “Block Parity” i.e, sending a block of parity bits to enable correction of single bit errors. Each parity bits are computed on different combination of bits from the data. In this work, the Hamming code introduces 3 redundancy bits to a 4-bit data stream. These redundancy bits are interspersed at bit positions 4, 5 and 7 with the original data bits to form a total of 7 bit encoded data stream.

In this program, the positions of the data bits are represented as:

<table>
<thead>
<tr>
<th>P_1</th>
<th>P_2</th>
<th>P_3</th>
<th>P_4</th>
<th>P_5</th>
<th>P_6</th>
<th>P_7</th>
</tr>
</thead>
<tbody>
<tr>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

128
After error correction, if any, the data bits have to be reassembled by removing the redundancy bits.

6.10.2 Captured results for the generation of 7 bit Hamming code (7,4)

The captured verilog output shows the operation of the Hamming encoder. The Encoder converts the 4 bit data bits into its respective 7 bit encoded bit stream. The results are taken for the inputs starting from 0000 (0\text{H} corresponding hexadecimal value) to 1111 (F\text{H} corresponding hexadecimal value).

Figure 6.31 Captured result shows the Hamming Encoder inputs for values 0\text{H} to 5\text{H}

In Figure 6.31, the Hamming encoded 7 bit output data for the 4 bit input data for values starting from 00\text{H} to 05\text{H} is shown. Figure 6.32 shows, the Hamming encoded 7 bit output data for the 4 bit input data for values starting from 06\text{H} to 0B\text{H}. Similarly, the Hamming encoded 7 bit output data for the 4 bit input data for values starting from 0B\text{H} to 0F\text{H} is shown in Figure 6.33.
Figure 6.32 Captured result shows the Hamming Encoder outputs for values $6_H$ to $A_H$

Figure 6.33 Captured result shows the Hamming Encoder outputs for values $B_H$ to $F_H$
Consolidating the results obtained for all the 16 input values, the following Table 6.3 is prepared. The table shows the 7 bit Hamming coded values for its corresponding 3 bit input values to the Hamming coder.

### Table 6.3 Contents of the file used for Hamming encoding

<table>
<thead>
<tr>
<th>4 bit input to Hamming encoder</th>
<th>Hexadecimal value of the input</th>
<th>7 bit output from the Hamming encoder</th>
<th>Hexadecimal value of the output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>0</td>
<td>0000000</td>
<td>00</td>
</tr>
<tr>
<td>0001</td>
<td>1</td>
<td>0000111</td>
<td>07</td>
</tr>
<tr>
<td>0010</td>
<td>2</td>
<td>0011001</td>
<td>19</td>
</tr>
<tr>
<td>0011</td>
<td>3</td>
<td>0011110</td>
<td>1E</td>
</tr>
<tr>
<td>0100</td>
<td>4</td>
<td>0101010</td>
<td>2A</td>
</tr>
<tr>
<td>0101</td>
<td>5</td>
<td>0101101</td>
<td>2D</td>
</tr>
<tr>
<td>0110</td>
<td>6</td>
<td>0110011</td>
<td>33</td>
</tr>
<tr>
<td>0111</td>
<td>7</td>
<td>0110100</td>
<td>34</td>
</tr>
<tr>
<td>1000</td>
<td>8</td>
<td>1001011</td>
<td>4B</td>
</tr>
<tr>
<td>1001</td>
<td>9</td>
<td>1001100</td>
<td>4C</td>
</tr>
<tr>
<td>1010</td>
<td>A</td>
<td>1010010</td>
<td>52</td>
</tr>
<tr>
<td>1011</td>
<td>B</td>
<td>1010101</td>
<td>55</td>
</tr>
<tr>
<td>1100</td>
<td>C</td>
<td>1100001</td>
<td>61</td>
</tr>
<tr>
<td>1101</td>
<td>D</td>
<td>1100110</td>
<td>66</td>
</tr>
<tr>
<td>1110</td>
<td>E</td>
<td>1111000</td>
<td>78</td>
</tr>
<tr>
<td>1111</td>
<td>F</td>
<td>1111111</td>
<td>7F</td>
</tr>
</tbody>
</table>
6.11. PERFORMING HAMMING BASED ERROR DETECTION AND CORRECTION

6.11.1 Captured results showing Hamming based Error Detection and Correction

Figure 6.34 Captured result shows the input and output of the Hamming decoder for values $0_H$ to $5_H$

Figure 6.35 Captured result shows the input and output of the Hamming decoder for values $6_H$ to $C_H$
Figure 6.36 Captured result shows the input and output of the Hamming decoder for values $D_H$ to $F_H$

Figure 6.34 shows the Hamming decoder outputs 00, 07, 19, 1E, 28 and 2D for the inputs 01, 07, 19, 1E, 28 and 2D. The decoder detects the error in the 1st and 5th inputs and corrects them. Also in Figure 6.35, the inputs to the decoder are 34, 1B, 4C, 52, 55 and 61 and the outputs from the decoder are 34, 1B, 4C, 52, 55 and 61. The detector detects no error for the given inputs. Similarly in Figure 6.36, the inputs to the decoder and the outputs from the decoder are shown. The detector detects no error. All the inputs and outputs from the Hamming decoder are listed below in Table 6.4. The status of the error is also shown in the table.

Table 6.4 Contents of the input and output of the Hamming decoder

<table>
<thead>
<tr>
<th>Sl. No.</th>
<th>7 bit input to the Hamming decoder</th>
<th>Hexadecimal value of the input</th>
<th>7 bit output from the Hamming decoder</th>
<th>Hexadecimal value of the output</th>
<th>Status of error</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>00000001</td>
<td>01</td>
<td>00000000</td>
<td>00</td>
<td>Error in position 1</td>
</tr>
<tr>
<td>2</td>
<td>00001111</td>
<td>07</td>
<td>00001111</td>
<td>07</td>
<td>No Error</td>
</tr>
<tr>
<td>3</td>
<td>0011001</td>
<td>19</td>
<td>0011001</td>
<td>19</td>
<td>No Error</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0011110</td>
<td>1E</td>
<td>0011110</td>
<td>1E</td>
<td>No Error</td>
</tr>
<tr>
<td>5</td>
<td>0101000</td>
<td>28</td>
<td>0101010</td>
<td>2A</td>
<td>Error in position 2</td>
</tr>
<tr>
<td>6</td>
<td>0101101</td>
<td>2D</td>
<td>0101101</td>
<td>2D</td>
<td>No Error</td>
</tr>
<tr>
<td>7</td>
<td>0110011</td>
<td>33</td>
<td>0110011</td>
<td>33</td>
<td>No Error</td>
</tr>
<tr>
<td>8</td>
<td>0110100</td>
<td>34</td>
<td>0110100</td>
<td>34</td>
<td>No Error</td>
</tr>
<tr>
<td>9</td>
<td>1001011</td>
<td>4B</td>
<td>1001011</td>
<td>4B</td>
<td>No Error</td>
</tr>
<tr>
<td>10</td>
<td>1001100</td>
<td>4C</td>
<td>1001100</td>
<td>4C</td>
<td>No Error</td>
</tr>
<tr>
<td>11</td>
<td>1010010</td>
<td>52</td>
<td>1010010</td>
<td>52</td>
<td>No Error</td>
</tr>
<tr>
<td>12</td>
<td>1010101</td>
<td>55</td>
<td>1010101</td>
<td>55</td>
<td>No Error</td>
</tr>
<tr>
<td>13</td>
<td>1100001</td>
<td>61</td>
<td>1100001</td>
<td>61</td>
<td>No Error</td>
</tr>
<tr>
<td>14</td>
<td>1100110</td>
<td>66</td>
<td>1100110</td>
<td>66</td>
<td>No Error</td>
</tr>
<tr>
<td>15</td>
<td>1111000</td>
<td>78</td>
<td>1111000</td>
<td>78</td>
<td>No Error</td>
</tr>
<tr>
<td>16</td>
<td>1111111</td>
<td>7F</td>
<td>1111111</td>
<td>7F</td>
<td>No Error</td>
</tr>
</tbody>
</table>

6.12 CHAPTER CONCLUSION

The main essence of this thesis of performing the automatic modulation identification technique is explained in this chapter by tracking the Pilot sequence appended in the modulated signal. The work is implemented using hardware for 10 modulation types on FL 2440 ARM9 core. Moreover, simulated outputs are obtained to illustrate Automatic modulation detection based on feature extraction. Additionally, the scheduler based implementation is also demonstrated in this chapter to perform the node tracking analysis using the priority allocation for each node. Chapter 6 also deals with network layer routing algorithm to find the shortest path based on channel conditions and clustering of the mobile nodes to enable establishment and safeguarding of the path for continuous transmission of data from the source to the destination. In an adaptive SDR wireless network, the longevity of the nodes in the transmission path is controlled by the power consumed and the bandwidth provided by the link between the nodes to handle the data transfer.
Subsequently, Chapter 7 deals with results obtained from calculating energy saving and bandwidth efficiency. The energy saving has a significant improvement of achieving up to 75% when adaptive modulation is used. The graphical result in Figure 7.3 shows constant effective bandwidth efficiency throughout the distance for Adaptive modulation. The energy consumption factor and bandwidth efficiency with channel conditions are explained in Section 3.3 to determine the best modulation for the particular channel.