CHAPTER 4

AUTOMATIC MODULATION RECOGNITION
(FREQUENCY AND PHASE)

Conventionally, modulation identification is performed by (i) Decision Theoretic (DT) approach (ii) Statistical Pattern (SP) approach (iii) Spectral processing (iv) Instantaneous amplitude, phase and frequency histogram techniques (v) Hybrid scheme (vi) Universal demodulators. The DT and pattern recognition approaches are characterized in Table 4.1.

Table 4.1 Scope of the various approaches

<table>
<thead>
<tr>
<th>S. No.</th>
<th>Approach type</th>
<th>Implementation</th>
<th>Limitations</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Decision Theoretic</td>
<td>1. Uses probabilistic likelihood function</td>
<td>High computational complexity, difficulty in implementation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2. probabilistic and hypothesis testing arguments based recognition problem</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>classification</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Pattern recognition</td>
<td>1. Two fold application</td>
<td>Poor tolerance to model mismatch</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2. Feature extraction and Recognition</td>
<td></td>
</tr>
</tbody>
</table>
In this work, the modulation identification technique uses a pre-processing module and performs (i) rejection of noise (ii) normalization (iii) carrier frequency estimation (iv) recovery of complex envelope etc. The equalization module mitigates the channel effect. The feature extraction module extracts the training sequence (which helps to identify the type of modulation used) from the transmitted sequence. The typical characteristics of different estimators are shown in Table 4.2. The architecture of a third order modulation estimator is shown in Figure 4.1.

**Table 4.2 Comparison between Different Estimators**

<table>
<thead>
<tr>
<th>S. No.</th>
<th>3rd Order</th>
<th>Conventional</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>(i)</td>
<td>$E[\Phi_e]=0$</td>
<td>$E[\Phi_e] \neq 0$</td>
<td>Advantage in 3rd order</td>
</tr>
<tr>
<td>(ii)</td>
<td>$\text{Sgn}(R_{\text{acc}})=\delta(KT)$, $</td>
<td>^F-F</td>
<td>^2&gt;E_{\text{Th}}$</td>
</tr>
</tbody>
</table>

**Figure 4.1 Third order modulation estimator**

4.1 EXPERIMENTAL SETUP

When a data is to be transmitted over the wireless network, the modulator selector selects the appropriate modulation scheme from the different kinds of
modulations such as ASK, FSK, PSK, QPSK and BPSK. Individual data can choose their own appropriate modulation best suited for transmission through wireless medium. The transmitter block diagram is shown in Figure 4.2. The data must be transmitted error free even at low signal to noise ratio.

When a data is to be received over the wireless network, the modulator tracker selects the appropriate demodulation scheme from the different kinds of demodulations such as ASK, FSK, PSK, QPSK and BPSK. The error free data is received by the receiver. The receiver block diagram is shown in Figure 4.3.

### 4.2 Phase Compensators Design

A trade-off in the form of an increase in SNR improvement is needed to compensate for any phase mismatch error in QPSK. This necessitates the requirement for
a phase error estimator or compensator as an integral part of an SDR unit for improved performance. The effect of phase error on system performance is shown in Table 4.3.

**Figure 4.3 Block diagram of Receiver**

**Table 4.3 Phase error effect in QPSK**

<table>
<thead>
<tr>
<th>S. No.</th>
<th>SNR</th>
<th>$\Delta \phi = 0^\circ$</th>
<th>SNR</th>
<th>$\Delta \phi = 10^\circ$</th>
<th>SNR</th>
<th>$\Delta \phi = 20^\circ$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Probability Of Error</td>
<td></td>
<td>Probability Of Error</td>
<td></td>
<td>Probability Of Error</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>$10^{-1.5}$</td>
<td>2</td>
<td>$10^{-1.4}$</td>
<td>2</td>
<td>$10^{-1.2}$</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>$10^{-2.0}$</td>
<td>4</td>
<td>$10^{-1.8}$</td>
<td>4</td>
<td>$10^{-1.4}$</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>$10^{-2.8}$</td>
<td>6</td>
<td>$10^{-2.5}$</td>
<td>8</td>
<td>$10^{-2.0}$</td>
</tr>
<tr>
<td>4</td>
<td>8</td>
<td>$10^{-4.0}$</td>
<td>10</td>
<td>$10^{-4.0}$</td>
<td>12</td>
<td>$10^{-4.0}$</td>
</tr>
<tr>
<td>5</td>
<td>10</td>
<td>$10^{-5.6}$</td>
<td>12</td>
<td>$10^{-6.0}$</td>
<td>14</td>
<td>$10^{-5.4}$</td>
</tr>
</tbody>
</table>
4.3 FREQUENCY MEAN BASED MODULATION ESTIMATION

By comparing frequency estimate with frequency mean, the values of modulation phase and frequency can be calculated. The frequency estimate value reveals an absolute alignment with the actual frequency value.

4.3.1. CHANNEL ESTIMATOR

The communication channel is modeled as a nonlinear amplifier. A MLP Neural Network architecture is used to “undo” the channel nonlinearities by performing the mapping that is an inverse of the one introduced by the channel. The input sequence used for the training is selected so that it spans the entire amplitude range of the input signal. Otherwise, the network may be driven into saturation and this can be a cause of even more severe distortion than the one introduced by the channel itself. This is avoided by data normalization. Higher nonlinearity in the channel is taken care by increasing the number of hidden neurons. Two test signals are chosen to test the estimator and they are given in equation (4.1) and (4.2).

\[ s1 = 0.8*\sin(2*\pi*n/10)+0.25*\cos(2*\pi*n/25); \]  
\[ s2 = \text{randn}(1,100); \]

4.3.1.1 TRAINING PHASE

These two test signals undergo data normalization and the network produce the equalized channel output. In Figure 4.4 Comparison of the desired and actual channel output (of the equalized channel) is plotted to the training signal.
4.3.1.2 TESTING PHASE

Comparison of the desired and actual channel output of the equalized channel to the test signal $s_1(n)$ and $s_2(n)$ is shown in Figure 4.5 and 4.6 respectively.

Figure 4.4 Comparison of the desired and actual channel output of the equalized channel to the training signal.

Figure 4.5 Comparison of the desired and actual channel output of the equalized channel to the test signal $s_1(n)$
Figure 4.6 Comparison of the desired and actual channel output of the equalized channel to the test signal $s_2(n)$

4.3.2 NOISE FILTER AT RECEIVER

The receiver receives the original data sequence from the transmitter and the noise filter at the receiver side plays a vital role. The data may be sometimes corrupted by the interference signal which is shown in Figure 4.8. Using the trained adaline network, the corrupted data is filtered as a function of the number of adaline neurons. The power of the difference between the original and filtered signal as a function of number of adaline neurons is shown in Figure 4.10

Figure 4.7 Original data sequence

Figure 4.8 Data corrupted with interference signal
4.4 DESIGN OF GAIN BOOSTING CHARGE PUMP

Charge-pump based Phase-Locked Loops (CPPLL) is broadly used as clock generators in a most of the applications like microprocessors, wireless receivers and serial link transceivers and so on. CP in PLL provides the theoretical zero static phase offset, and given the simple and robust design platforms. The CPLL also provides large frequency and capture range and it can also be used for discrete time analysis.

A typical implementation of the CPPLL consists of a Phase Frequency Detector (PFD), a CP, a passive Loop Filter (LF) and a VCO. The basic PLL structure is shown in Figure 4.11. The PFD commonly generates a pair of digital pulses corresponding to the Phase/Frequency error between the reference clock and the VCO output by comparing
the positive (or negative) edges of the two inputs. The CP then converts the digital pulses into an analog current that is converted into a voltage via the passive loop filter network and this control voltage drives the VCO. The negative feedback of the loop produces the zero phase/frequency error. Like any feedback system, a CPPLL has to be designed with a proper consideration for stability.

![Figure 4.11 Basic structure of PLL](image)

Figure 4.11 Basic structure of PLL

Phase-Locked Loops (PLLs) consists of a Phase/Frequency Detector (PFD), a CP, a Loop Filter (LP) and a VCO, whose output is feedback to the PFD. The PFD compares an external reference signal and the VCO output signal and produces two digital signals (Up and Down), with the width of these two signals being determined by their frequency and phase, the CP converts PFD output signal into a current that is fed into the LF, which determines the output LF voltages. The LF output voltages cause the VCO to generate a single frequency signal. Any fluctuation in the LF output voltage due to current mismatch in the CP causes proportional variations in the VCO signal.

The drain to source voltage of the n-channel MOS (nMOS) and p-channel MOS (pMOS) in the conventional CP can vary depending on the latter one’s output voltage, thereby causing the magnitude of the currents in the pMOS (Up) and nMOS (Down) to
differ. The LF output voltage fluctuations due to the current mismatch in the CP when the PLL is in the locking state; this causes the VCO output signal to have a large amount of phase noise with spurs.

Phase detectors may exhibit a dead zone, resulting in enlarged jitter. A common design technique to avoid a dead zone is to make sure that both Up and Down output signals are fully activated before shutting them both off. This is implemented by generating a reset signal with an AND operation of Up and Down output and introducing a delay before feeding back this signal to reset the phase detector. If the charge sharing in the CP is not perfectly cancelled or if there is a mismatch of and there will always be some current compensation, leading to phase offset and loop filter ripple. A longer reset delay results in a longer period during which the VCO is running at a different frequency due to the compensation current.

![Figure 4.12 Schematic diagram of CPPLL](image)

Therefore, the reset delay should be minimized under the constraint that it has to be longer than the response time of the PFD with some additional design margin to avoid a dead zone. Figure 4.12 shows the schematic diagram of a charge-pump PLL.
The PFD checks the reference and feedback signals and controls the CP to produce a current Ip, which adjust the control voltage Vc through a Loop Filter (LF). The voltage Controlled Oscillator (VCO) oscillates at a frequency that varies with Vc. The feedback signal is the output of VCO, possibly charge that has been pumped to the CP from the PFD. In both the figure, time (t) plotted in X axis and Up current is compared with reference (ref) signal and the Down current is compared with the feedback signal (div) which is obtained from divider network (N).

The amount of charge that is being pumped is given by the equation;

\[
|Q| = \int_{t}^{t+T} I_p(\tau) d\tau = \frac{\Delta \varphi T_{\text{ref}}}{2\pi} \quad (4.3)
\]

\[
I_M = \frac{|Q|}{T_{\text{ref}}} = I_p \frac{\Delta \varphi}{2\pi} \quad (4.4)
\]

4.5 SIMULATION RESULTS

The CP with error amplifier and the conventional CP are simulated by cadence tool 180nm, 1.8-V CMOS parameters. This CP shows the current matching characteristics around 5% of the sourcing/sinking current difference.
Figure 4.13 shows the Up and down signal generated from the Phase/ Frequency detector. Figure 4.14 shows the transient signals during charge pumped to the charge pump. Figure 4.15 to 4.18 shows the stability, gain of VCO output and the CP current with reference and feedback signal.

Figure 4.15 VCO Control Voltages when Reference Signal Leads Feedback Signal (Conventional Charge Pump PLL)

Figure 4.16 VCO Control Voltages when Reference Signal Leads Feedback Signal (Proposed Charge Pump PLL)
Figure 4.17 VCO Control Voltage Reference Signal and Feedback Signal Phase and Frequencies are Equal (Conventional Charge Pump PLL)

The graph has been plotted for both conventional and CP with error amplifier PLL. Good current matching characteristics are observed over the CP output voltage ranges 0-1.8V CMOS process.

Figure 4.18 VCO Control Voltage Reference Signal and Feedback Signal Phase and Frequencies are Equal (Proposed Charge Pump PLL)
The locking phenomena of the PLL is designed and verified for its locking range from 75MHz to 275MHz. Whenever the frequencies are below 75MHz and above 275MHz, the PLL loses its tracking frequency. This experiment is carried out by varying the different frequency of the incoming signal with the designed value for the VCO frequency. The perfect locking will take place at \( \cos \phi = 90 \) and it moves on both sides by 90 degree. The simulated waveform shows the different locking conditions with the reference voltage refer Table 4.4

### Table 4.4 Different locking conditions Vs Reference Voltage

<table>
<thead>
<tr>
<th>Input</th>
<th>75MHz</th>
<th>125 MHz</th>
<th>150 MHz</th>
<th>175 MHz</th>
<th>200 MHz</th>
<th>275 MHz</th>
<th>Above 280 MHz</th>
</tr>
</thead>
<tbody>
<tr>
<td>Square Wave</td>
<td>1.75  Vpp</td>
<td>1.754 Vpp</td>
<td>1.753 Vpp</td>
<td>1.752 Vpp</td>
<td>1.755 Vpp</td>
<td>1.756 Vpp</td>
<td>Goes out of Lock the amplitude decreased to 5V</td>
</tr>
</tbody>
</table>

### 4.6 INFERENCE AND DISCUSSION ON PLL DESIGN

In this work, a gain-boosting CP that has good current matching characteristics and stability in the PLL is designed. By using a simple gain-boosting circuit to increase its output resistance, a CP with good current matching characteristics is achieved without the need for stacking more cascade devices; operational amplifier with current mirror is provided to reduce the current mismatch. This output impedance is similar to triple-cascade output impedance but with less voltage over head. That is, if the drain voltage increases due to any variations in the input, its control over the current extends further towards the source S. the un-inverted region expands towards the source, shorting the length of the channel region. This effect called channel-length modulation effect. Since the resistance is proportional to the channel-length, shorting the channel decreases its output resistance.
4.7 PLL BASED CARRIER SYNCHRONIZATION

In this thesis, the PLL design is to envisage switching points between the frequency ramps during “locked” status. It also ensures that the switching points be in accord with rising and falling edges of the PLL. A positive (negative) frequency slope corresponds to a high (low) PLL output. The chosen PLL design specifications are given in Table 4.5. The flow chart in Figure 4.19 shows the estimation process.

Table 4.5 PLL design specifications

<table>
<thead>
<tr>
<th>S. No</th>
<th>Operation description</th>
<th>Expected Outcome</th>
</tr>
</thead>
</table>
| 1     | Signal lock           | (i) To obtain phase and frequency  
|       |                       | (ii) Prediction of switching points |
| 2     | Advancement of clock by 90° | (i) To estimate frequency  
|       |                       | (ii) Responsive to frequency ramp polarity change |

4.8 PROPOSED SYSTEM

The proposed system for the classification of signals in cognitive radio mainly consists of two different phases namely the training and classification phase. The two phases are explained in the following sub sections and test schemes includes DPSK, PAM, PSK (π/4), PSK (π/8) and 64QAM
4.8.1 FEATURE EXTRACTION

Feature extraction is a fundamental pre-processing step for classification and all machine learning problems. In the proposed method, 2\textsuperscript{nd} order cumulants of real and imaginary part of the complex envelope are used as features for the classification of signals. The block diagram of feature extraction phase is shown in Figure 4.20.
The generated signal is first modulated by using different modulation scheme such as PAM, 32QAM and 64QAM. These modulated signals are passed through an AWGN channel with a predefined SNR level. The second-order statistical features extracted from the received signals and stored in the database for the classification purpose.

![Block diagram of feature extraction phase](image)

**Figure 4.20 Block diagram of feature extraction phase**

The SLBC and Non SLBC classifiers are trained by using the database generated in the training phase. The algorithm is as follows:

- **Input** Generated signals
- **Output** The feature vector of all modulated signal with noise as database

1. Modulate the signal by using selected modulation scheme.
2. Pass the modulated signal through an AWGN channel with predefined SNR level
3. Calculate the $2^{nd}$ order cumulants
4. Step 3 is repeated for real and imaginary part of the complex envelope.
5. Insert this feature vector and the known class into the database.
6. Repeat the above steps for other modulation schemes.
4.8.2 CLASSIFICATION PHASE

In the classification phase, the unknown signal is classified as any one of the three modulation types. The second order statistical features are extracted from the unknown signal and this feature vector is processed with the features in the database by using the SLBC and Non SLBC classifier. The algorithm is as follows:

Algorithm II: Identification Algorithm

[Input] unknown signal and the database
[Output] the class of the signal to which this unknown signal is assigned
1) Calculate the 2nd order cumulants
2) Step 1 is repeated for real and imaginary part of the unknown signal
3) Test with the trained SLBC and Non SLBC classifier and find the class of the unknown signal.

4.9 BLIND SIGNAL IDENTIFICATION

The automatic recognition of the modulation format of a detected signal, the intermediate step between signal detection and demodulation, is a major task of an intelligent receiver. Obviously, with no knowledge of the transmitted data and many unknown parameters at the receiver, such as the signal power, carrier frequency and phase offsets, timing information, etc., blind identification of the modulation is a difficult task. This becomes even more challenging in real-world scenarios with multipath fading, frequency-selective and time-varying channels. In general, AMC is a challenging task, especially in a non-cooperative environment, where in addition to multipath propagation, frequency-selectivity and time-varying nature of the channel, no prior knowledge of the
incoming signal is available. A classifier is supposed to correctly choose the modulation 
format of the incoming signal from a pool of N mod candidate modulations. The work 
focus on algorithms for ASK, PSK, QAM, and FSK classification by considering the 
most intuitive way to identify the modulation class of the incoming signal by using the 
information extracted from its instantaneous amplitude, phase and frequency. FSK 
signals are characterized by constant instantaneous amplitude, whereas ASK signals have 
amplitude fluctuations, and PSK signals have information in the phase. The maximum of 
the Discrete Fourier Transform (DFT) of centered (average or first order moment) 
normalized instantaneous amplitude was used as a feature to distinguish between FSK 
and ASK/PSK classes.

In this work, Automatic modulation detection extracts seven parameters (Features 
based on amplitude, frequency and phase) for identification of different modulation 
techniques namely: ASK2, ASK4, FSK2, FSK4, PSK2, PSK4, QAM16 and QAM64. 
The thresholds of different parameters have been calculated for the classification during 
training in real time situation. The parameters are carefully chosen based on signal 
statistics. The parameters selected are

1. AbsEnv

\[
\text{absEnv} = \frac{1}{N} \sum_{i=1}^{N} |A_{en}[i]| \tag{4.5}
\]

2. AbsPhase

\[
\text{absPhase} = \frac{1}{C} \sum_{A_{en} \in C} |\phi_{c}[i]| \tag{4.6}
\]

\[
\phi_{c}[i] = \phi[i] - \frac{1}{N} \sum_{j=1}^{N} \phi[j] \tag{4.7}
\]
3. rEnv

\[ rEnv = \frac{1}{N} \sum_{i=1}^{N} \left| A[i] - m_a \right| / m_a \]  \hspace{1cm} (4.8)

4. absEnv2

\[ rEnv2 = \frac{1}{N} \sum_{i=1}^{N} \left| B_{a_n}[i] - m_b \right| \]  \hspace{1cm} (4.9)

\[ B_{a_n}[i] = |A_{a_n}[i]| \]  \hspace{1cm} (4.10)

\[ m_b = \frac{1}{N} \sum_{i=1}^{N} B_{a_n}[i] \]  \hspace{1cm} (4.11)

5. absFreq

\[ absFreq = \frac{1}{C} \sum_{A_{\mu[-\mu]}} \left| f[i] - f_a \right| / F_{sym} \]  \hspace{1cm} (4.12)

\[ f_a = \frac{1}{C} \sum_{A_{\mu[-\mu]}} f[i] \]  \hspace{1cm} (4.13)

6. absFreq2

\[ absFreq2 = \frac{1}{C} \sum_{A_{\mu[-\mu]}} \left| f_2[i] - \frac{1}{C} \sum_{A_{\mu[-\mu]}} f_2[j] \right| \]  \hspace{1cm} (4.14)

\[ f_2[i] = \left| f[i] - f_a \right| / F_{sym} \]  \hspace{1cm} (4.15)

7. absPhase2

\[ absPhase2 = \frac{1}{C} \sum_{A_{\mu[-\mu]}} \left| \phi_2[i] - \frac{1}{C} \sum_{A_{\mu[-\mu]}} \phi_2[j] \right| \]  \hspace{1cm} (4.16)

\[ \phi_2[i] = |\phi_2[i]| \]  \hspace{1cm} (4.17)
The method of classification described in Figure 4.21 uses the threshold value (variable) and is calculated a priori for various modulation techniques. Based on the flow chart shown in Figure 4.21 the type of modulation transmitted to the receiver is identified.

**4.10 HARDWARE IMPLEMENTATION**

The implementation uses two hardware nodes (ARM 9 and above based architecture with “gnu radio” LINUX drivers installed)

1. Transmitter board consisting of all the modulation blocks (refer Figure 4.22)
2. Receiver board consisting of all the demodulation blocks(refer Figure 4.23)
4.10.1 TRANSMITTER BLOCK

The transmitter board consists of carrier generator, various modulation blocks, modulation selection switch, noise generator, DAC converter as shown in Figure 4.22. The User selects one of the modulations using the “modulation selection switch”.

![Diagram of FL2440 Software Blocks]

Based on the modulation selected, the appropriate software block is activated. The data is modulated using the selected software block. The Carrier is generated using the carrier generation software block. The noise variance is controllable and is generated using the noise generation software block. The modulated wave is transmitted which is converted to in a analog signal using on board DAC and the signal is transmitted using antenna.

**Figure 4.22 Transmitter board consisting of all the modulation blocks**
4.10.2 RECEIVER BLOCK

The receiver board consists of carrier generator, various demodulation blocks, noise generator, ADC converter as shown in Figure 4.23. The modulated signal is sampled through Analog to Digital convertor at a high sampling rate. The digitized signal is then noise filtered by the noise filtering software block. The modulation scheme is detected by applying algorithms on the sampled data. Based on the detected modulation scheme, appropriate demodulation software block is activated and data is demodulated.

![FL2440 SOFTWARE BLOCKS](image)

**Figure 4.23 Receiver board consisting of all the demodulation blocks**

The fundamental block for modulation selector and modulation tracker is shown in Figure 4.24. On concerning the transmission of data in the transmitter, the modulation selector selects the different modulation techniques such as ASK, PSK, FSK
and QAM. Using the modulation selector, the carrier signals are enveloped with the original data by choosing any one of the specialized modulation techniques identified as suitable for transmission. On the other hand, the receiver diagnoses the modulated data. The modulation tracker recognizes, tracks and retrieve the original data from the different modulation techniques.

4.11 IMPLEMENTATION OF SOFTWARE BLOCKS ON FL2440

4.11.1 BLOCKS OF GNU RADIO

<table>
<thead>
<tr>
<th>Package</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gnuradio.gr</td>
<td></td>
</tr>
<tr>
<td>gnuradio.digital</td>
<td>This is the gr-digital package.</td>
</tr>
<tr>
<td>gnuradio.blks2</td>
<td></td>
</tr>
<tr>
<td>gnuradio.audio</td>
<td>This is the gr-audio package.</td>
</tr>
<tr>
<td>gnuradio.trellis</td>
<td></td>
</tr>
<tr>
<td>gnuradio.wavelet</td>
<td>This is the gr-wavelet package.</td>
</tr>
</tbody>
</table>
4.11.2 SIGNAL PROCESSING BLOCKS

<table>
<thead>
<tr>
<th>Module</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gnuradio.digital.gmskmod_bc</td>
<td>GMSK modulator.</td>
</tr>
<tr>
<td>gnuradio.digital.probe_mpsk_snr_est_c</td>
<td>A probe for computing SNR of a signal.</td>
</tr>
<tr>
<td>gnuradio.digital.bpsk.dbpsk_demod</td>
<td>Bpsk and dpsk demodulation</td>
</tr>
<tr>
<td>gnuradio.digital.bpsk.dbpsk_mod</td>
<td>Bpsk and dpsk modulation</td>
</tr>
<tr>
<td>gnuradio.digital.qpsk.dqpsk_demod</td>
<td>Qpsk demodulation</td>
</tr>
<tr>
<td>gnuradio.digital.qpsk.dqpsk_mod</td>
<td>Qpsk modulation</td>
</tr>
<tr>
<td>gnuradio.digital.gmsk.gmsk_demod</td>
<td>Qmsk demodulation</td>
</tr>
<tr>
<td>gnuradio.digital.gmsk.gmsk_mod</td>
<td>Qmsk modulation</td>
</tr>
<tr>
<td>gnuradio.digital.psk.psk_demod</td>
<td>Phase shift demodulation</td>
</tr>
<tr>
<td>gnuradio.digital.psk.psk_mod</td>
<td>Phase shift modulation</td>
</tr>
<tr>
<td>gnuradio.digital.qam.qam_demod</td>
<td>QAM demodulation</td>
</tr>
<tr>
<td>gnuradio.digital.qam.qam_mod</td>
<td>QAM modulation</td>
</tr>
<tr>
<td>gnuradio.digital.qpsk.qpsk_demod</td>
<td>QPSK Demodulation</td>
</tr>
<tr>
<td>gnuradio.digital.qpsk.qpsk_mod</td>
<td>QPSK Modulation</td>
</tr>
</tbody>
</table>
CASE (I) 16 QAM IMPLEMENTATION DETAILS

Figure 4.25 16 QAM implementation block

1. Import GNU radio blocks
   - import gr, gru, modulation, utils
   - import qam,
   - import pi, sqrt
   - import cmath
   - import pprint

2. Select default values
   - Samples/Symbol: 2
   - Excess Bandwidth: 0.35
   - Gray code: True
   - Verbose: False
   - Log: False
   - Costas alpha: None
   - Gain: 0.03
   - Omega relative limit: 0.005
Step 1: perform RRC-Filtered QPSK modulation

Step 2: Turn bytes into K-bit vectors to obtain constellation with parity

Step 3: Implementation Pulse shaping filter

Step 4: Add QAM modulation specific options to the standard parser

Perform backward iterations of Step 3.

**CASE (II) PSK IMPLEMENTATION**

1. Import GNU radio blocks

2. Generate gray coded constellation for different M-PSK values

3. Code Conversion

4. PSK Demodulation

Figure 4.26 PSK implementation block

1. import \( \pi \), sqrt, log 10

2. import math, cmath

3. generate gray code constellation for M-PSK for M=[2,4,8]

4. Convert gray code to binary code for different M values like M=[2,4,8]

4. Perform backward iterations of step 3.
CASE (III) GMSK IMPLEMENTATION

Figure 4.27 GMSK implementation block

1. import gr
   import modulation_utils
   import pi, numpy
   import inspect
   import pprint

2. 
   | Samples/Symbol | 2 |
   | Bandwidth      | 0.35 |
   | Gray code      | True |
   | Verbose        | False |
   | Log            | False |
   | Gain_mu        | None  |
   | Frequency error| 0.0   |
   | Omega relative limit | 0.005 |

3. Step 1: perform Gaussian Filter for GMSK modulation
   Step 2: connect initialize base class
Step 3: Modulation logging tuned on

Step 4: Adds GMSK modulation-specific options to the standard parser.

Step 5: command line options, create dictionary suitable for passing to _init

Perform backward iteration of step 3.

CASE (IV) OFDM IMPLEMENTATION

1. import math,
   import gr, OFDM_packets_utils
   import gr_threading as threading
   import psk, qam

2. Step 1: based on the options fft_length, occupied_tones and cp_length,
   Step 2: this block create OFDM symbols using a specified modulation options.

3. i) Packets to be sent are enqueued by calling send packets

Figure 4.28 OFDM implementation block
ii) Use frequency domain to get doubled – up known symbol for correction in Time domain.

iii) Send the payload


5. Add OFDM-specific options to the options parser.

4.12 CHAPTER CONCLUSION

In this chapter schemes for frequency and phase estimation using PLL is discussed. This improved design can help to achieve low bit error rate at a given SNR. The intrinsic presence of phase compensators and phase error estimators in the modulation tracker is also proposed. Based on the explained concepts two nodes can communicate in the wireless medium without any packet loss at 2.4GHz frequency. (Zigbee interface). The hardware implementation part consists of transmitter and receiver block to implement software radio cores using GNU radio. The details of GNU radio blocks suited for implementation on FL2440 hardware are described.