TECHNICAL UNIVERSITY OF CRETE
ELECTRICAL AND COMPUTER ENGINEERING DEPARTMENT
TELECOMMUNICATIONS DIVISION

Experimental study of satellite signal processing with software defined radios

by

Roza Chatzigeorgiou

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DIPLOMA OF

ELECTRICAL AND COMPUTER ENGINEERING

July 2020

THESIS COMMITTEE

Professor Aggelos Bletsas, *Thesis Supervisor*
Professor George N. Karystinos
Professor Michalis Zervakis

Abstract

This work studies the decoder of a low Earth orbit (LEO) micro-satellite (CubeSat) that transmits a GFSK telemetry in the UHF frequency band and specifically at 437.225 MHz. The first stage of the decoding process is FM demodulation, which converts the GFSK signal to a PAM waveform. The PAM symbols are time-synchronized by the Mueller and Mller algorithm, which exploits the structure of a Phase Locked Loop and estimates the time delay of the receiver sampling clock and the optimal sampling time. Next, the symbol samples are convolutionally decoded and frame detection is performed. Finally, descrambling and Reed-Solomon decoding follows. Frame synchronization and bit detection of this decoder are sub-optimal, so a different decoding scheme is proposed. We suggest performing preamble-based frame synchronization first, using correlators. The bits of the detected frame are decoded using a non-coherent FSK detector, a convolutional decoder, a descrambler and a Reed-Solomon decoder. The simulation results showed that for uncoded and unscrambled signals, the suggested decoder achieved approximately a 2 dB gain. Finally, the process of capturing and decoding the satellites GFSK signal is thoroughly explained. The signal is captured using a low-cost software-defined radio (SDR) receiver and a Yagi antenna. The decrypted telemetry bits in the GFSK signal give indications about the status of the satellites electronics, such as the current and voltage of the its transmitter and receiver, the PLL status and the temperature of the CubeSat microcontroller.

Thesis Supervisor: Professor Aggelos Bletsas

Acknowledgements

First of all, I would like to express my gratitude and appreciation for my supervisor A. Bletsas whose guidance, support and encouragement has been invaluable throughout this study.

I would also like to thank G.Vougioukas and D. Estevez whose assistance was a milestone in the completion of this thesis.

I cannot forget to thank my friends G. Apostolakis, E. Pagkalos, I. Vardakis, G. Karvounakis, M. Strataki, M. Manolikaki, N. Geramanis and G. Danalis for all the memories we shared these five years.

Finally, my deep and sincere gratitude to my family for their continuous and unparalleled love, help and support.

Table of Contents

List of Figures

Chapter 1

Introduction

A satellite is an object in space that orbits or circles around a bigger object. There are two kinds of satellites: natural (such as the moon orbiting the Earth) or artificial (such as the International Space Station orbiting the Earth). Artificial satellites, however, did not become a reality until the mid-20^{th} century. The first artificial satellite was Sputnik, a Russian beach-ball-size space probe that lifted off on Oct. 4, 1957. It was not until 2003 with the launch of the first CubeSat, that satellite launching became affordable. CubeSats are miniature satellites that are commonly used in low Earth orbit, have been used for educational purposes, and recently for applications such as remote sensing, scientific experimentation, or communications. As engineers become more familiar with the technology, CubeSats are also being considered for flights outside Earth's orbit: to locations such as the Moon, Mars, or Jupiter.

Considering the importance of CubeSats in today's world, we study the GFSK (Gaussian Frequency Shift Keying) decoder of the Chinese CubeSat, LilacSat-2. The decoder is part of the gnu-radio application gr-satellites [1], developed by Daniel Estevez. The decoder of the specific CubeSat exploits FM radios to detect the transmitted GFSK signal. In this thesis, it is explained why an FM receiver is not optimal for GFSK/FSK detection and an alternative bit detection scheme is proposed. Furthermore, suggestions for a more efficient frame synchronization are made. Considering these, a new decoding scheme is presented. Finally, the process of capturing and decoding a satellite signal is thoroughly explained.

A brief overview of the thesis follows. In Chapter 2, we will discuss about the CubeSat LilacSat-2, focusing on it's GFSK telemetry. In Chapter 3, the concept of symbol timing synchronization is studied and the Muller and Müeller algorithm is thoroughly explained. In Chapter 4, we analyze the process of convolutional encoding and decoding. In Chapter 5, we explain why the bit and frame detection scheme of the decoder presented in Chapter 2 is sub-optimal and a new decoding scheme is suggested. In Chapter 6, experimental results about capturing and decoding the LilacSat-2 GFSK are shown. Finally, Chapter 7 contains conclusions and future work.

Figure 1.1: Three small CubeSats float above the Earth after deployment from the International Space Station. Astronaut Rick Mastracchio tweeted the photo from the station on the 19^{th} of November, 2013. (Image credit: Rick Mastracchio (via Twitter as @AstroRM)).

Chapter 2

LilacSat-2 Cubesat

LilacSat-2 is CubeSat for education, amateur radio communication and technology demonstration, built by a team of 15 students at Harbin Institute of Technology, China. It is a cube-shaped $20cm \times 20cm \times 20cm$ satellite with a weight of $11kg$ that was launched along with eight other amateur radio satellites on September 19, 2015. The satellite transmits a CW beacon on 144.39MHz, a 4800 baud rate GFSK telemetry on 437.225MHz and a 9600 baud rate BPSK telemetry on 437.200MHz [2].

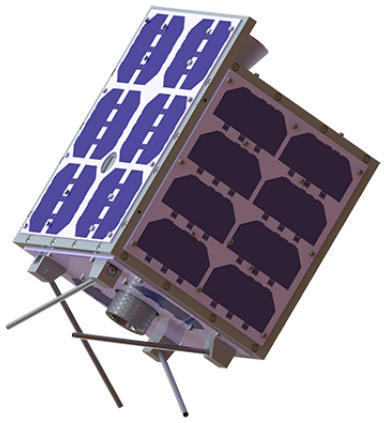


Figure 2.1: Illustration of LilacSat-2 satellite.

2.1 Orbital Elements

Kepler's first law states that the path followed by a satellite around its primary (earth) will be an ellipse. This ellipse has two focal points (foci) F1 and F2 as shown in figure 2.2. Center of mass of the Earth (heaviest object) will always present at one of the two foci of the ellipse.

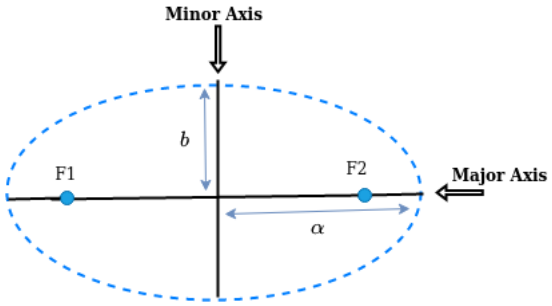


Figure 2.2: Kepler's First Law

The satellite's orbital elements are six and are shown in table 2.1.

LilacSat-2 Orbital elements	
Eccentricity	0.0014329
Inclination	97.4913°
Argument of Perigee	249.2150°
Longitude of the ascending node	205.9813°
Mean anomaly	178.6614°
Semi-major axis	6901 km

Table 2.1: LilacSat-2 orbital elements table

The length of Semi-major axis defines the size of satellites orbit. It is half of the major axis. Eccentricity indicates the deviation of the orbit's shape from a circle. If the lengths of semi major axis and semi minor axis of an elliptical orbit are α and b, then the mathematical expression for eccentricity (e) will be:

$$e = \frac{\sqrt{a^2 - b^2}}{a} \tag{2.1}$$

For a satellite, the point which is closest from the Earth is known as Perigee. Mean anomaly (M) gives the average value of the angular position of the satellite with reference to perigee. Inclination defines the orientation of the orbit by considering the equator of earth as reference.

Satellite orbit cuts the equatorial plane at two points. The first point is called as descending node, where the satellite passes from the northern hemisphere to the southern hemisphere. The second point is called ascending node and is the point where the satellite passes from the southern hemisphere to the northern hemisphere. Argument of perigee (ω) is the angle between ascending node and perigee. Longitude of ascending node (Ω) is the angle between line of Aries and ascending node towards east direction in equatorial plane.

2.2 LilacSat-2 GFSK Telemetry

The satellite carries the linear transponder ADF7021-N. The transmit power is 400mW and its bandwidth is 40kHz[3]. For GFSK (Gaussian Frequency Shift Keying) transmission, the modulated bits are given by:

$$x(t) = A(t)\cos\left(2\pi f_c t + 2\pi h \int_{-\infty}^{t} m(t)dt\right), \tag{2.2}$$

where $A(t)$ is the signal amplitude, f_c is the carrier frequency, $h = 2Tf_d$ is the modulation index, and $m(t)$ is the Gaussian-shaped information bits, given by

$$m(t) = \sum_{n=0}^{N} a_n g_s(t - nT), \tag{2.3}$$

where a_n are the NRZ (non-return to zero) information bits and $g_s(t)$ is the Gaussian shaping pulse. The Gaussian pulse $g_s(t)$ of the transponder ADF7021-N has a normalized 3-dB bandwidth equal to 0.5. Also, the frequency seperation is $F_{sep} = 2f_d = h/T = 9.6kHz$.

2.2.1 Packet structure and decoder

Packet structure

As shown in figure 2.5, the telemetry information bits of LilacSat2 are in a CSP (CubeSat Protocol) packet of N bytes[2]. CSP is a Network layer protocol, specially designed for communication with cubesats and follows the TCP/IP (Transmission Control Protocol/Internet Protocol) model. The protocol is using a 32-bit header which contains transport and network layer information. After this header, follow the telemetry bits. The packet also ends with a CRC (Cyclic Redundancy Check), used by earth stations as a final check that the entire frame and all packets were received correctly. Before the CSP packet, there is a byte for the packet length. The KISS (Keep It Simple Stupid) protocol is applied on the resulting PDU (Protocol Data Unit). The KISS protocol is used for packet delimiting. The resulting KISS stream is Reed-Solomon (R-S) encoded. Reed-Solomon is a class of powerful burst error correcting codes. Then, a scrambler (or randomizer) randomizes the R-S encoded KISS stream. A scrambler randomizes bits in a known manner and the ASM (Attached Sync Marker) is a frame synchronization bit sequence that is placed before the scrambled bits. Finally, the resulting non-return to zero bit sequence is convolution-

ally encoded. The convolutional code used is the CCSDS NASA GFSC (Consultative Committee for Space Data Systems NASA Goddard Space Flight Center).

Decoder

The gnuradio application gr-satellites, developed by Daniel Estevez [1], includes many CubeSat decoders, including one for LilacSat-2. The decoder is displayed in figure 2.3 and a simplified block diagram for the whole decoding process is displayed in 2.4. The input to the decoder is the down-converted and FM-demodulated received signal. For a GFSK signal, FM demodulation gives an Pulse Amplitude Modulated (PAM) signal:

$$x_{FM}(t) = \frac{d\angle x_{bb}(t)}{dt} = 2\pi h m(t), \tag{2.4}$$

where $\angle x_{bb}(t)$ is the phase of baseband equivalent of $x(t)$ from equation (2.2). The decoder employs a clock synchronization algorithm to obtain the correct symbol samples, since the receiver is not synchronized with the PAM signal. Clock synchronization or symbol timing synchronization will be further discussed in Chapter 2.

Then, the symbol samples need to be convolutionally decoded. To find the bit corresponding to each codeword, the Viterbi algorithm is employed. In the specific decoder, two Viterbi decoders are needed, because the specific convolutional encoder produces a two bit codeword for each information bit and the input signal to the decoder is not frame-synchronized. The convolutional decoding process is studied in Chapter 4.

Frame synchronization is made by detecting a 32-bit preamble (sync-word or ASM) from the convolutional decoded bit sequences. After that, descrambling and Reed-Solomon decoding is performed. Descrambling is the process of recovering the scrambler's input bits. From the KISS stream, we can easily obtain the telemetry data bits in each CSP packet, using the block 'KISS to PDU'.

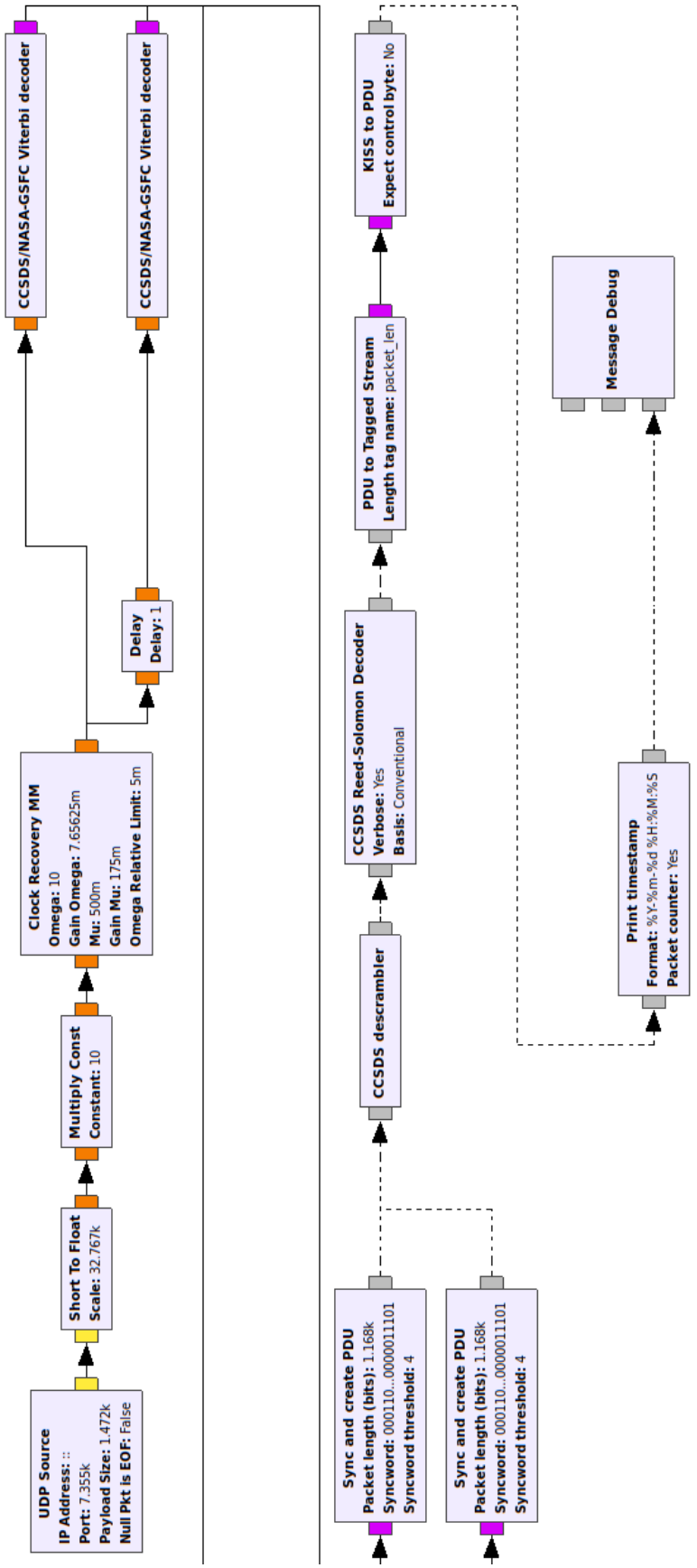


Figure 2.3: gr-satellites decoder for LilacSat-2 GFSK telemetry.

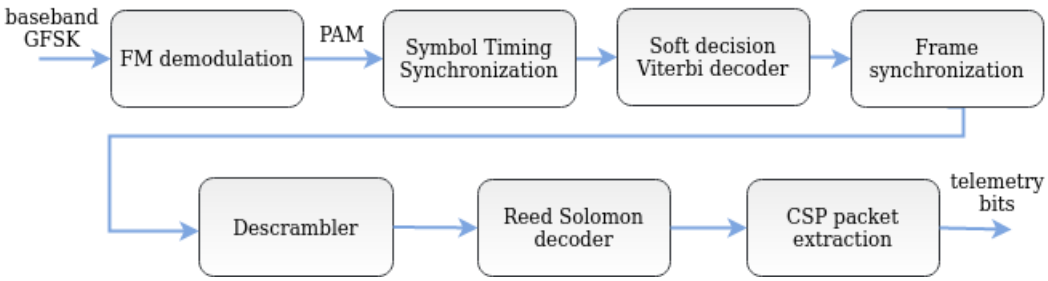


Figure 2.4: Simplified block diagram illustrating the decoding process.

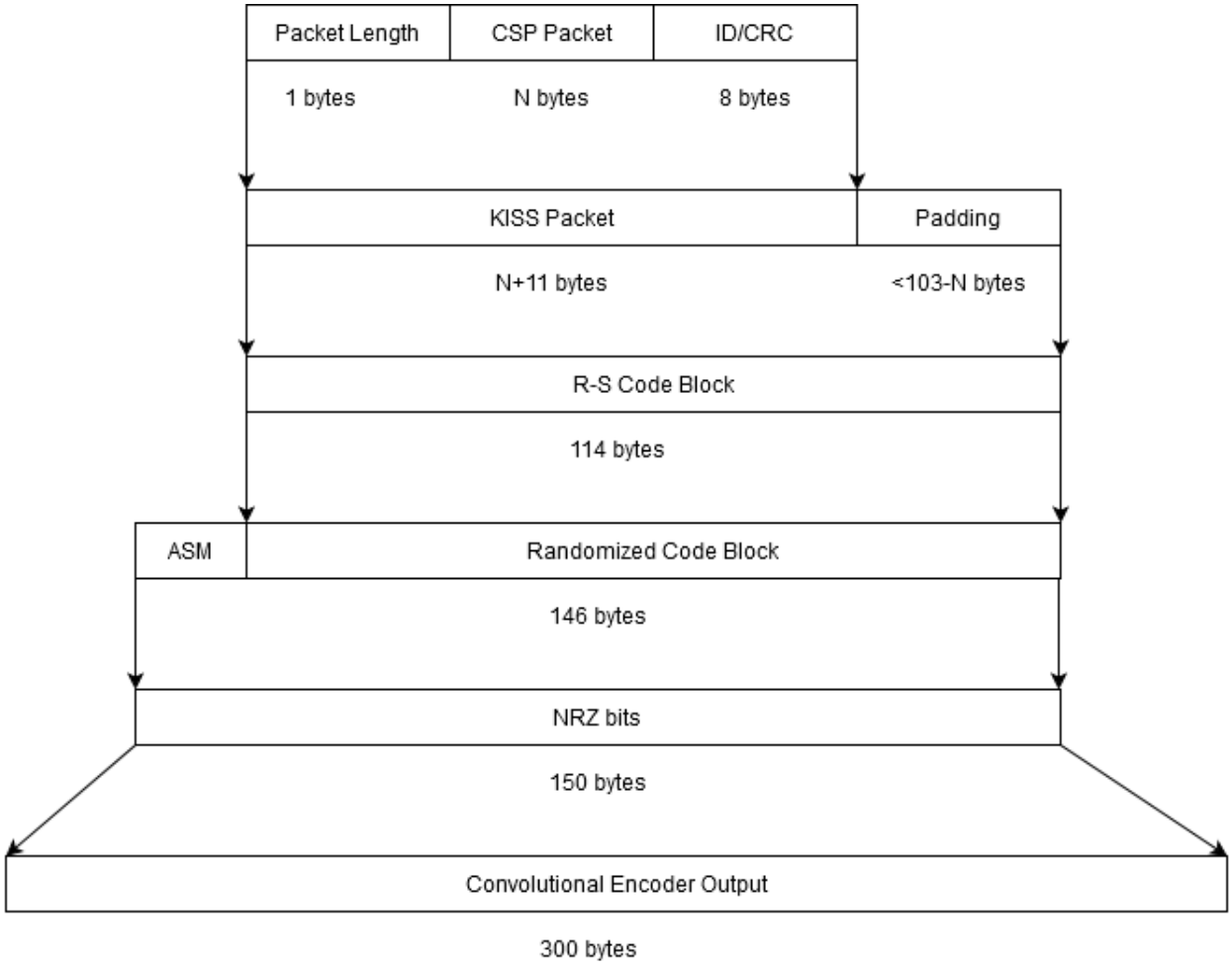


Figure 2.5: Packet structure of LilacSat-2 GFSK.

Chapter 3

Symbol timing synchronization

In wireless communication systems, a receiver must know the exact symbol timing instants in order to correctly demodulate the transmitted waveform. Assuming pulse amplitude modulated symbols, the matched filter output is given by:

$$y(t) = \sum_k \alpha_k g(t - kT - \tau) + n(t), \tag{3.1}$$

where α_k are the transmitted symbols, $g(t) = g_T(t) \circledast g_T(T-t)$ is the shaping pulse at the matched filter output, T is the symbol period, τ is an unknown timing delay and $n(t) \sim \mathcal{CN}(0, N_o)$ is the circularly symmetric additive white Gaussian noise. Ideally, (3.1) should be sampled at $t = kT + \tau$. Since the timing delay τ is unknown to the receiver, it must be estimated.

To sample at the correct symbol timing, the decoder of LilacSat-2 employs the Mueller and Müller algorithm, which exploits the structure of a Phase Locked Loop.

3.1 Phase Locked Loops

A phase locked loop (PLL) is a simple device that constantly adjusts the phase of an oscillator to the phase of an input signal. We assume that the input signal is given by:

$$y_{in}(t) = \cos(2\pi f t + \phi_{in}(t)). \tag{3.2}$$

A PLL has three key elements: an oscillator, a phase detector and a loop filter. The oscillator produces a periodic signal $y_{osc}(t)$ with a specific frequency and phase.

$$y_{osc}(t) = \cos(2\pi f t + \phi_{osc}(t)). \tag{3.3}$$

In an analog PLL, the oscillator is called VCO (Voltage Controlled Oscillator), while in a digital PLL, it is called DDS (Direct Digital Synthesizer) . For a noisless input signal, the phase detector calculates a function of the phase offset $\theta_e(t)$ between the

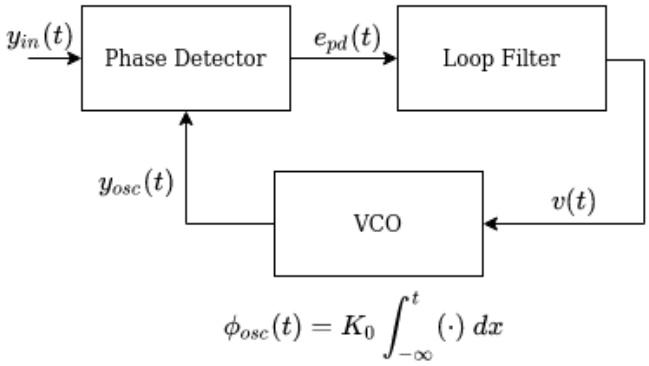


Figure 3.1: Block diagram of an analog PLL.

input and the oscillator signal:

$$e_{pd}(t) = g(\theta_e(t)) = g(\phi_{in}(t) - \phi_{osc}(t)). \tag{3.4}$$

To linearize the loop, we assume that for small phase errors $e_{pd}(t) \approx K_{pd}\theta_e(t)$, where K_{pd} is the phase detector gain. The phase detector output is filtered by the loop filter, which establishes the dynamic performance of the loop. In addition, noise and high-frequency signal components often are suppressed by the loop filter. The VCO adjusts its phase according to the control voltage $v(t)$, produced by the loop filter:

$$\phi_{osc}(t) = K_0 \int_{-\infty}^{t} v(t)dt, \tag{3.5}$$

where K_0 is the oscillator gain. Equations (3.2) to (3.4) also apply to a digital PLL, by setting $t = nT_s$. The DDS adjusts its phase according to

$$\phi_{osc}(nT_s) = K_0 \sum_{k=-\infty}^{n-1} v(kT_s), \tag{3.6}$$

where T_s is the sampling period. A PLL is locked when the phase error becomes zero or has small flactuations (noisy case).

3.1.1 Proportional-plus-integral Loop Filter

If the Loop Filter is a proportional-plus-integral (PI) filter, the PLL tracks frequency and phase offsets. Considering an analog PLL, the output of the PI filter for the

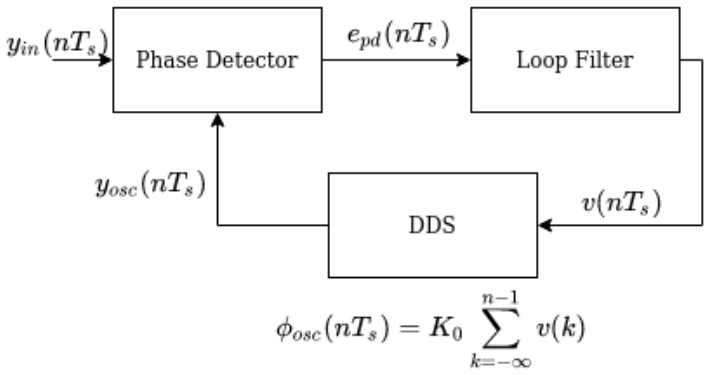


Figure 3.2: Block diagram of a digital PLL.

phase detector output $e_{pd}(t)$, is given by:

$$e_{pi}(t) = e_p(t) + e_i(t) = K_p e_{pd}(t) + K_i \int_{-\infty}^{t} e_{pd}(t)dt, \tag{3.7}$$

where K_p and K_i is the proportional and integral gain, respectively. The proportional filter $e_p(t)$ tracks constant phase shifts, while the integral filter $e_i(t)$ tracks frequency offsets. For a digital PLL, the PI filter is given by:

$$\begin{aligned} e_{pi}(nT_s) &= e_p(nT_s) + e_i(nT_s) \\ &= K_p e_{pd}(nT_s) + e_i((n-1)T_s) + K_i e_{pd}(nT_s). \end{aligned} \tag{3.8}$$

To understand how a PI filter works, it is important to define the terms of instantaneous phase and instantaneous frequency. Consider a cosine wave

$$x(t) = \cos(2\pi f_o t) = \cos\phi(t), \tag{3.9}$$

where the argument of the cosine is called instantaneous phase. The instantaneous frequency of $x(t)$ is given by:

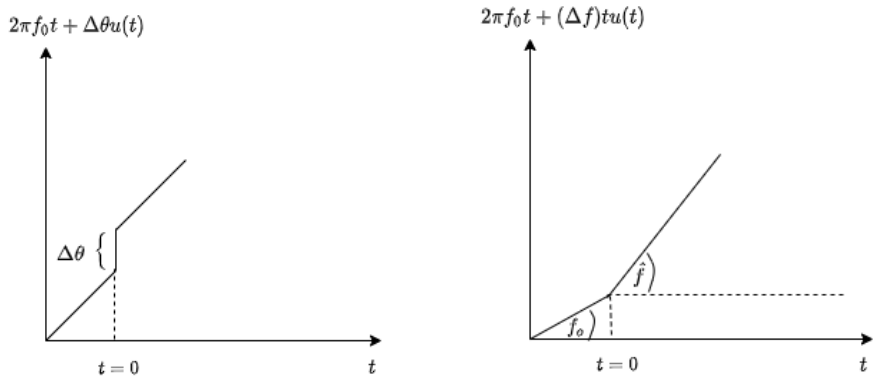
$$f = \frac{1}{2\pi}\frac{d\phi(t)}{dt}. \tag{3.10}$$

A phase shift in $x(t)$ occurs if the instantaneous phase is incremented by a constant factor $\Delta\theta$:

$$\phi(t) = 2\pi f_o t + \Delta\theta. \tag{3.11}$$

This of course does not change the instantaneous frequency of the signal:

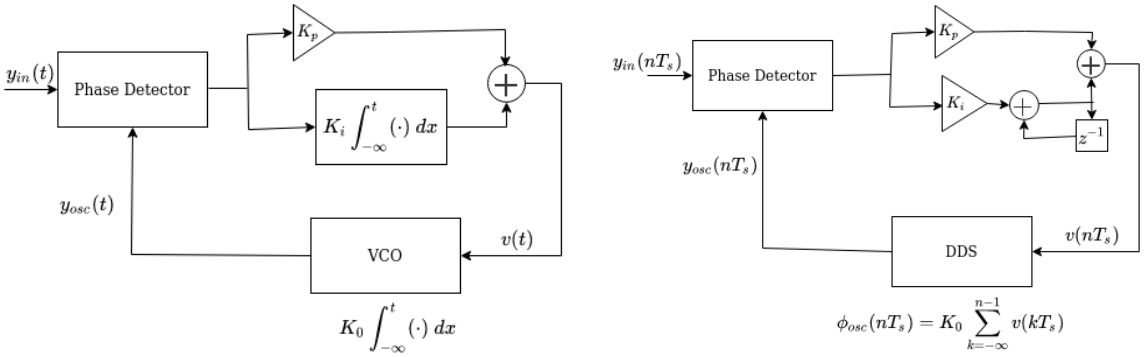
$$f = \frac{1}{2\pi}\frac{d\phi(t)}{dt} = \frac{1}{2\pi}\frac{d[2\pi f_o t + \Delta\theta]}{dt} = f_o. \tag{3.12}$$



(a) Instantaneous phase $\phi(t)$ due to a constant phase shift $\Delta\theta$ at $t = 0$.

(b) Instantaneous phase due to a constant frequency shift $\Delta\omega$ at $t = 0$.

Figure 3.3: The instantaneous phase due to a step (a) and ramp (b) input at t=0. The unit step function is denoted as $u(t)$.



(a) Analog PLL with PI filter

(b) Digital PLL with PI filter

Figure 3.4: Phase Locked Loops employed with a PI filter.

Assuming a frequency shift Δf, the change in the instantaneous phase is modeled as a phase ramp:

$$\phi(t) = 2\pi f_o t + \Delta f t = 2\pi f_o t + \theta(t). \tag{3.13}$$

A phase ramp results to a shift in the instantaneous frequency of the signal:

$$f = \frac{1}{2\pi}\frac{d\phi(t)}{dt} = f_o + \Delta f = \hat{f}. \tag{3.14}$$

Hence, the integrator filter integrates the phase errors to generate a phase ramp error, which corresponds to a constant frequency error.

3.1.2 Phase lock and Loop Filter gains

Considering a PLL with PI filter, the time to achieve phase lock depends on the noise bandwidth B_n and the damping factor ζ_n of the loop. From [4], the phase lock time is approximated by:

$$T_{LOCK} \approx T_{FL} + T_{PL}, \tag{3.15}$$

where T_{FL} denotes the time for phase lock and T_{PL} denotes the time to achieve frequency lock. Their values can be approximated by:

$$T_{FL} \approx 4\frac{(\Delta f)^2}{B_n^3} \text{ and } T_{PL} \approx \frac{1.3}{B_n}, \tag{3.16}$$

where Δf is the frequency offset. It is possible for the frequency offset Δf to be so large that the loop can never acquire lock. The range of frequency offsets for which the loop can acquire lock is called pull-in range Δf and is well approximated by:

$$(\Delta f)_{pull-in} \approx \left(2\pi\sqrt{2}\zeta\right) B_n \tag{3.17}$$

The damping factor and the noise bandwidth is determined by the designer of the loop. Assuming that K_0 and K_{pd} are known, the gains of the PI filter can be calculated from the transfer function of the loop. It occurs that the gains of the PI filter are calculated by [4]:

$$K_p = \frac{4\zeta B_n}{\left(\zeta + \frac{1}{4\zeta}\right)} \frac{1}{K_0 K_{pd}} \text{ and } K_i = \frac{4B_n^2}{\left(\zeta + \frac{1}{4\zeta}\right)^2} \frac{1}{K_0 K_{pd}}, \tag{3.18}$$

for an analog PLL and

$$K_p = \frac{4\zeta \left(\frac{B_n T_s}{\zeta + \frac{1}{4\zeta}}\right)}{1 + 2\zeta \left(\frac{B_n T_s}{\zeta + \frac{1}{4\zeta}}\right) + \left(\frac{B_n T_s}{\zeta + \frac{1}{4\zeta}}\right)^2} \text{ and } K_i = \frac{4\left(\frac{B_n T_s}{\zeta + \frac{1}{4\zeta}}\right)^2}{1 + 2\zeta \left(\frac{B_n T_s}{\zeta + \frac{1}{4\zeta}}\right) + \left(\frac{B_n T_s}{\zeta + \frac{1}{4\zeta}}\right)^2}, \tag{3.19}$$

for a digital PLL.

3.2 Mueller and Müller

Mueller and Müller (M&M) is a clock synchronization algorithm that applies to binary or multi-level PAM signals. The input to the M&M algorithm is the baseband signal

$y(t)$ of Equation (3.1). The output of the algorithm is the samples/decisions at the correct symbol timing instances. The algorithm block diagram is shown in figure 3.5.

3.2.1 The Mueller and Müller PLL

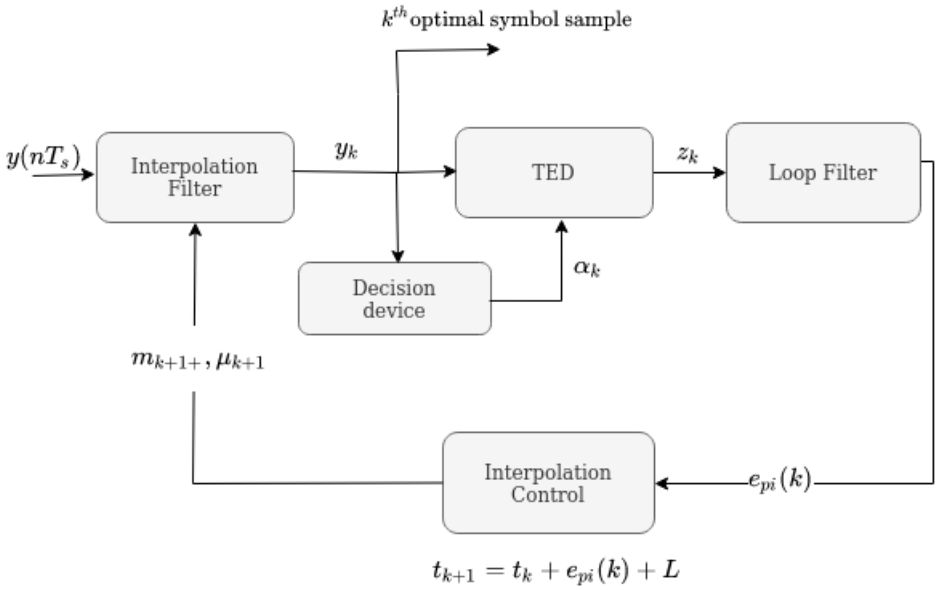


Figure 3.5: Block diagram of Mueller and Müller Clock Recovery.

Timing Error Detector and Decision device

The Timing Error Detector (TED) estimates the timing (phase) error of the receiver clock from sampling optimally, using the samples of the matched filter output. The optimal timing instants are approximately located at the peaks of the input signal. The TED function is given by:

$$z_k = \frac{1}{2}\frac{y_k a_{k-1} - y_{k-1} a_k}{E[a_k^2]}, \tag{3.20}$$

where α_k and α_{k-1} are obtained by performing hard decisions on the current and previous symbol sample, y_k and y_{k-1}, respectively. The expectation of the timing error is equal to the timing error function $f(\tau)$ [5], which computes the symmetry error of a symbol's impulse response:

$$f(\tau) = E[z(k)] = \frac{1}{2}\left(g\left(\tau - T\right) - g(\tau + T)\right) \triangleq \frac{1}{2}\left(g_{-1} - g_1\right). \tag{3.21}$$

The symmetry error $f(\tau)$ becomes zero when $g_1 = g_{-1}$, or equivalently, when the sample taken is at the peak of the symbol impulse response. If the symmetry error is not zero, the sampling time for the next symbol should be corrected. The samples g_{-1} and g_1 are located T timing instants before and after a symbol sample g_0, respectively. The M&M timing error function is an approximation of $f(\tau)$, because the samples g_1 and g_{-1} are not directly available in the matched filter output. This is due to the fact that transmitted signal $r(t)$ is the superposition of the delayed symbol impulse responses.

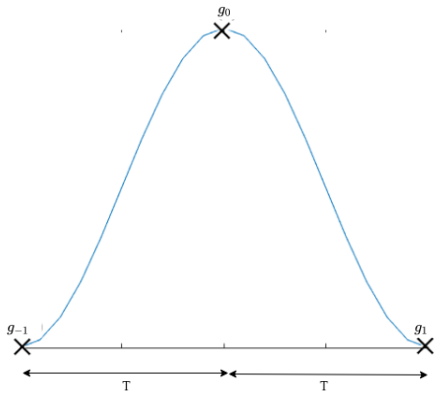


Figure 3.6: Impulse response of a symbol with $g(\tau - T) = g(\tau + T)$ or $f(\tau) = 0$, which implies that the sampling time for a symbol is correct.

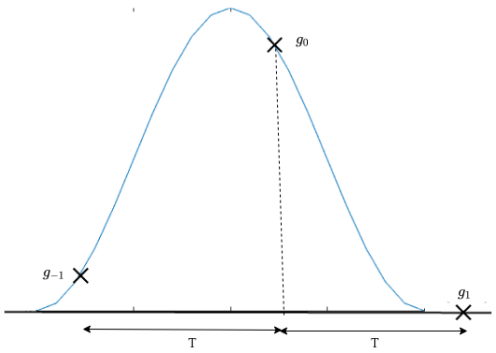


Figure 3.7: Impulse response of a symbol with $f(\tau) > 0$, which implies late symbol timing.

Loop Filter

Assuming a PI filter, its output is a timing phase to compensate a step and a ramp timing phase error, in T_s units:

$$e_{pi}(k) = e_p(k) + e_i(k) = K_p z(k) + e_i(k-1) + K_i z(k) \tag{3.22}$$

Interpolation Control

The Interpolation Control uses the timing phase error from the Loop Filter to compute the optimal sampling time for the next symbol:

$$t_{k+1} = t_k + e_{pi}(k) + L, \tag{3.23}$$

where $L = T/T_s$ is the nominal period of receiver clock in units of T_s. Using (3.23), we compute the following values:

$$m_{k+1} = \lfloor t_{k+1} \rfloor \text{ and } \mu_{k+1} = t_{k+1} - \lfloor t_{k+1} \rfloor, \tag{3.24}$$

where m_{k+1} is called the basepoint index and μ_{k+1} the fractional interval. The basepoint index is an exact multiple of the sampling period, while the fractional interval is a fraction of the sampling period. To compute the sample at the optimal sampling instant, we need to obtain from the sampled matched filter the sample $(m_{k+1} + \mu_{k+1})$. The sample at the basepoint index is already available. The sample at the basepoint index, delayed by μ_{k+1}, is calculated by an interpolation filter.

Ideal interpolation filter

In time domain, the reconstruction of a continuous signal from samples can be considered as an interpolation process of filling the gaps between neighboring samples. From the Sampling Theorem, we know that for perfect signal reconstruction, all frequencies exceeding the Nyquist frequency should be discarded. Thus, the ideal interpolation filter $H_I(\omega)$ is an ideal low-pass filter that discards all frequencies exceeding the Nyquist frequency.

$$H_I(\omega) = \begin{cases} T_s, & \text{if } |\frac{\omega}{2\pi}| < \frac{1}{2T_s} \\ 0, & \text{otherwise} \end{cases} \tag{3.25}$$

where T_s is the sampling period. In the time domain, the ideal interpolation function is given by:

$$h_I(t) \triangleq \text{sinc}(t). \tag{3.26}$$

Hence, to ideally reconstruct a signal $y(t)$ from its samples:

$$y(t) = \sum_{n=-\infty}^{\infty} y(nT_s)h_I(t - nT_s). \tag{3.27}$$

Let $T_I = (m_k + \mu_k)T_s$, where m_k is the basepoint index (integer) and μ_k is the fractional interval (fractional number). The sample $x(T_I)$ is calculated by resampling (3.27) at $t = T_I$:

$$y(T_I) = y((m_k + \mu_k)T_s) = \sum_{n=-\infty}^{\infty} y(nT_s)h_I(T_I - nT_s). \tag{3.28}$$

FIR interpolation filter

The ideal interpolation filter is IIR (infinite impulse response), thus a FIR (finite impulse response) estimation would reduce the large computational cost of using such filter. Assuming that $h_I(t)$ takes significant values in the interval $-I_1T_s < t < I_2T_s$:

$$y(T_I) = \sum_{n=m_k-I_2}^{m_k+I_1} y(nT_s)h_I((m_k - n)T_s + \mu_k T_s). \tag{3.29}$$

Changing the summation index of the above equation into $i = m_k - n$, gives:

$$y(T_I) = \sum_{i=-I_1}^{I_2} y((m_k - i)T_s)h_I(iT_s + \mu_k T_s). \tag{3.30}$$

To compute (3.30), a Minimum Mean Square Error (MMSE) interpolation filter is employed. An MMSE Interpolation filter calculates a finite number of filter taps $h_n(\mu_k)$ by minimizing the quadratic error between the frequency response of the IIR ideal interpolator and its FIR approximation[6],[7]:

$$E(\mu) = \int_{-2\pi B}^{2\pi B} \left| H_I\left(e^{j\omega T_s}, \mu\right) - \sum_{n=-N}^{N-1} h_n(\mu)e^{-j\omega T_s n} \right|^2 d\omega, \tag{3.31}$$

where N is the filter order, $B = \frac{1}{NT_s}$ its one-sided bandwidth. The frequency response of the ideal interpolation filter for a fractional interval μ is given by[6] [7]:

$$H_I\left(e^{j\omega T_s}, \mu\right) = \frac{1}{T_s} \sum_{n=-\infty}^{\infty} H_I\left(\omega - \frac{2\pi}{T_s} n, \mu\right), \tag{3.32}$$

where $H_I(\omega, \mu)$ is the Fourier transform of the ideal interpolation filter, delayed by μT_s:

$$H_I(\omega, \mu) = \begin{cases} T_s \exp(j\omega\mu), & \text{if } |\frac{\omega}{2\pi}| < \frac{1}{2T_s} \\ 0, & \text{otherwise} \end{cases} \tag{3.33}$$

The MMSE filter taps $h_n(\mu)$ are calculated by minimizing the error function $E(\mu)$, so we set the gradient of (3.31) to zero[8]:

$$\frac{\partial E(\mu)}{\partial h_l(\mu)} = 0, \tag{3.34}$$

for $-N \leq l \leq N-1$. After the calculation of the filter taps for a specific fractional index μ, the optimal symbol sample for the M&M algorithm is obtained from Equation (3.30).

M&M clock siynchronization for binary PAM.

$e_I(0) \leftarrow 0$ ▷ initialize integrator
$k \leftarrow 1$ ▷ initialize symbol counter
$\mu_1 \leftarrow 0.5$ ▷ initiliaze fractional interval
$L \leftarrow T/T_s$ ▷ receiver clock period in T_s
while $m_k \leq \text{length}(y)$ **do**
  $y_k \leftarrow \sum_{n=-N}^{N-1} y(m_k - n) h_n(\mu_k)$ ▷ interpolation filtering
  if $y_k > 0$ **then** ▷ hard decision
    $a_k \leftarrow 1$
  else
    $a_k \leftarrow -1$
  $z_k \leftarrow (\alpha_{k-1} y_k - \alpha_k y_{k-1})/2E[\alpha_k^2]$ ▷ M&M TED function
  $e_P(k) \leftarrow K_p z(k)$ ▷ proportional filter
  $e_I(k) \leftarrow e_I(k-1) + K_I z(k)$ ▷ integator filter
  $e_{PI}(k) \leftarrow e_I(k) + e_P(k)$ ▷ PI filter
  $t_{k+1} \leftarrow t_k + e_{PI}(k) + L$ ▷ optimal sampling time
  $m_{k+1} \leftarrow \lfloor t_{k+1} \rfloor$ ▷ update basepoint-index
  $\mu_{k+1} \leftarrow t_{k+1} - \lfloor t_{k+1} \rfloor$ ▷ update fractional interval
  $k \leftarrow k + 1$ ▷ increment symbol counter

3.3 Symbol timing synchronization for LilacSat-2

The FM demodulated LilacSat2 GFSK signal, which has become a binary PAM, is given as input to the M&M algorithm, to find the samples at the correct timing instants. The proportional gain of the PI Loop Filter was selected $K_1 = 0.175$ and the integral gain is a function of the proportional gain, $K_2 = 0.25K_1^2$. The integral gain was set very small compared to the proportional gain, to avoid large frequency shifts that may drive the PLL out of lock. The frequency range to achieve lock $(\Delta f)_{pull-in}$ is set to $0.5\%L$ and the interpolation filter order is $N = 4$.

3.3.1 Simulation results

Assuming that the receiver sampling is late by τ seconds, we show the symbol samples of an FM demodulated LilacSat-2 GFSK signal, before and after M&M clock synchronization. The receiver, without M&M, fails to sample at the optimal timing instants (peaks of the binary PAM signal, located at $\{-1, +1\}$), due to the unknown timing delay τ. After performing the M&M algorithm on the sampled signal, the symbol samples are brought closer to the peaks of the PAM signal. The SNR for the GFSK signal is defined as in equation (5.12) of Chapter 5.

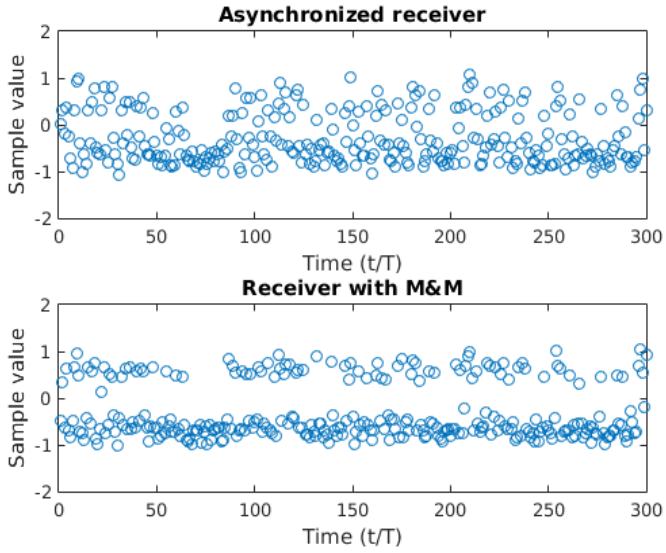


Figure 3.8: SNR= $16dB$, $\tau = 0.3$

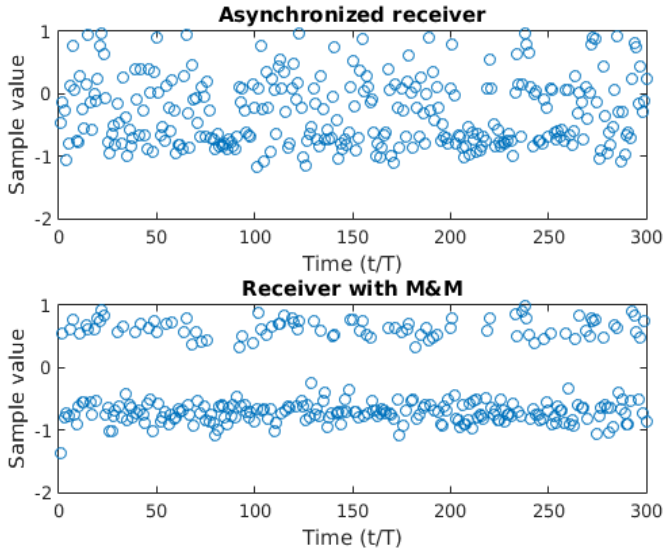


Figure 3.9: SNR= $16dB$, $\tau = 0.4$

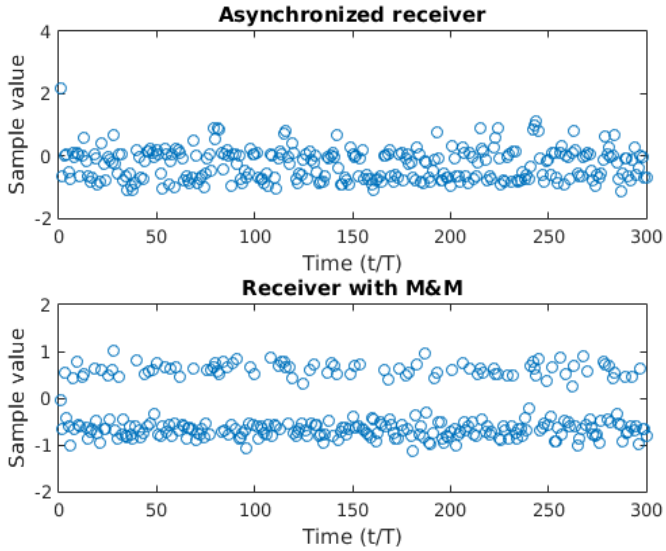


Figure 3.10: SNR= $16dB$, $\tau = 0.5$

Chapter 4

Viterbi convolutional decoding

4.1 Convolutional encoder

The rate of a convolutional code is given by

$$r = \frac{n}{k},$$

where n is the number of input bits to the encoder and k is the codeword length. A convolutional encoder consists of a shift register with M stages and k adders. Each adder generates a codeword bit by adding the input bits and the contents of the memory elements that it is connected to (modulo-2 addition, or equivalently, an exclusive-or operation). The constraint length of a convolutional code, denoted as K, is the number of codewords a single information bit can affect. A convolutional encoder may be represented by a set of k generator polynomials. Each polynomial $g_i(s)$ is of degree $K-1$ or less and describes the connection of the shift register to the i^{th} adder, where the coefficient of each term is either a 1 or a 0, depending on whether a connection exists or does not exist between the shift register and the adder. For the simple encoder illustrated in Figure 4.1, the generator polynomials are given by:

$$g_1(s) = s^2 + s + 1 \text{ and } g_2(s) = s^2 + 1$$

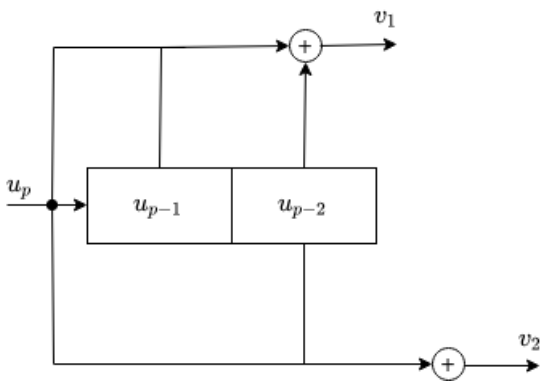


Figure 4.1: Encoder of a $(r, K) = (0.5, 3)$ convolutional code

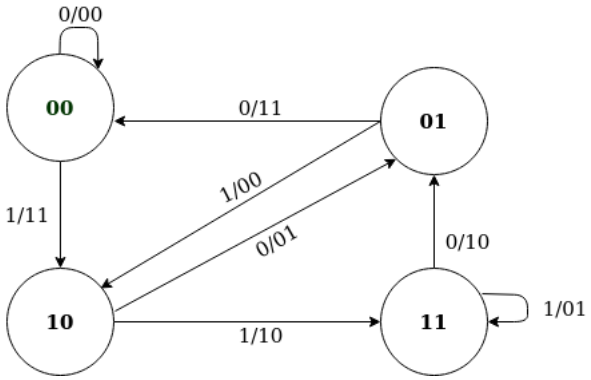


Figure 4.2: State diagram of the encoder illustrated in Figure (4.1). For each state transition the notation u_p/v_1v_2 is used, where u_p is the input bit to the encoder and v_1v_2 the corresponding encoded message.

Each time n input bits are fed into the M-stage shift register, k output bits are generated and the contents of the shift register are shifted to the right. A convolutional encoder is a discrete linear time-invariant system and can be viewed as a finite-state machine (FSM) with 2^M states. The state of the encoder at each timing instant is described by the contents of the M-bit shift register after the right-shift operation. At $t = 0$ the shift register resets to zero. A new state transition is caused by a new input to the encoder.

4.2 Decoding of Convolutional Codes

The Viterbi algorithm is the most widely used method for the maximum-likelihood (ML) decoding of convolutional codes with low constraint length (typically $K \leq 8$). Viterbi finds the most likely transmitted sequence $\mathbf{x}$ sent for a received sequence $\mathbf{y}$. This leads to the maximization of the joint probability:

$$\hat{\mathbf{x}} = \arg\max_{\mathbf{x}} P(\mathbf{x}, \mathbf{y}). \tag{4.1}$$

Assuming equiprobable data, it is equivalent to maximizing the a-posteriori probability:

$$\begin{aligned} \arg\max_{\mathbf{x}} P(\mathbf{x}|\mathbf{y})P(\mathbf{y}) &= \arg\max_{\mathbf{x}} P(\mathbf{x}|\mathbf{y}) \\ &= \arg\max_{\mathbf{x}} P(\mathbf{y}|\mathbf{x})\frac{P(\mathbf{x})}{P(\mathbf{y})} \\ &= \arg\max_{\mathbf{x}} P(\mathbf{y}|\mathbf{x}). \end{aligned} \tag{4.2}$$

We define the branch metric (BM) and path metric (PM) as the log-likelihood function of the corresponding conditional probability functions as follows:

$$BM(\mathbf{x}_i, \mathbf{y}_i) = \ln P(\mathbf{y}_i|\mathbf{x}_i), \tag{4.3}$$

$$PM(\mathbf{x}, \mathbf{y}) = \ln P(\mathbf{y}|\mathbf{x}). \tag{4.4}$$

Then we can write the path metric as the sum of the branch metrics

$$PM(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{N} \ln P(\mathbf{y}_i|\mathbf{x}_i). \tag{4.5}$$

From (4.1) to (4.5), we get that Viterbi maximizes the path matric or equivalently, the sum of branch metrics.

4.2.1 Soft decision decoding

Assuming that the modulation used to transmit the encoded message is binary PSK, the j^{th} bit of a codeword $\mathbf{c_i} = \begin{bmatrix} c_{i1} & \cdots & c_{ij-1} & c_{ij} & \cdots & c_{ik} \end{bmatrix}$ may be expressed as:

$$x_{ij}(t) = \begin{cases} x(t) & \text{if } c_{ij} = 1 \\ -x(t) & \text{if } c_{ij} = 0 \end{cases} \tag{4.6}$$

where $x(t)$ is a signal that is zero outside the time interval T and has energy equal to $\mathcal{E}$. To perform soft decision decoding, the channel must experience Additive White Gaussian Noise. The input to the decoder is the sampled matched filter output. Considering this, the i^{th} received codeword $\mathbf{y}_i$ may be expressed as:

$$\mathbf{y}_i = h\mathbf{x}_i + \mathbf{n_i} \tag{4.7}$$

where $\mathbf{n}_i$ is a vector of independent complex Gaussian random variables $n_{ij} \sim \mathcal{CN}(0, N_o)$ and $h \sim \mathcal{CN}(0, 1)$. Since the channel is AWGN, the likelihood probability is given by the Gaussian probability density function:

$$P(y_{ij}|x_{ij}) = e^{\frac{-(x_{ij}-y_{ij})^2}{2\sigma^2}} \cdot \frac{1}{\sqrt{2\pi\sigma^2}}. \tag{4.8}$$

The log-likelihood of this probability is given by:

$$\ln P(y_{ij}|x_{ij}) = -\frac{(x_{ij}-y_{ij})^2}{2\sigma^2} - \ln\sqrt{2\pi\sigma^2}, \tag{4.9}$$

For a k-bit codeword, we get that the log-likelihood is

$$\ln P(\mathbf{y}_i|\mathbf{x}_i) = \sum_{j=1}^{k} \ln P(y_{ij}|x_{ij}) = \sum_{j=1}^{k} \frac{-(x_{ij}-y_{ij})^2}{2\sigma^2} - \ln\sqrt{2\pi\sigma^2}, \tag{4.10}$$

Maximizing the above quantity leads to:

$$\begin{aligned}
\arg\max_{\mathbf{x}_i} \ln P(\mathbf{y}_i|\mathbf{x}_i) &= \arg\max_{\mathbf{x}_i} \left\{ \sum_{j=1}^{k} \frac{-(x_{ij}-y_{ij})^2}{2\sigma^2} - \ln\sqrt{2\pi\sigma^2} \right\} \\
&= \arg\min_{\mathbf{x}_i} \sum_{j=1}^{k} \frac{(x_{ij}-y_{ij})^2}{2\sigma^2} \\
&= \arg\min_{\mathbf{x}_i} \sum_{j=1}^{k} (x_{ij}-y_{ij})^2.
\end{aligned} \tag{4.11}$$

According to (4.11), the branch metric for soft decision decoding is given by

$$BM_{soft}[\mathbf{x}_i, \mathbf{y}_i] = \sum_{j=1}^{k} (x_{ij}-y_{ij})^2. \tag{4.12}$$

The path that maximizes the a-posteriori probability is the one that minimizes the sum of Branch Metrics:

$$\begin{aligned}
\arg\max_{\mathbf{x}} \ln(P(\mathbf{y}|\mathbf{x})) &= \arg\max_{\mathbf{x}} \sum_{i=1}^{N} \ln P(\mathbf{y}_i|\mathbf{x}_i) \\
&= \arg\max_{\mathbf{x}} \left\{ \sum_{i=1}^{N}\sum_{j=1}^{k} \frac{-(x_{ij}-y_{ij})^2}{2\sigma^2} - \ln\sqrt{2\pi\sigma^2} \right\} \\
&= \arg\max_{\mathbf{x}} -\sum_{i=1}^{N}\sum_{j=1}^{k} (x_{ij}-y_{ij})^2 \frac{1}{2\sigma^2} \\
&= \arg\min_{\mathbf{x}} \sum_{i=1}^{N} BM_{soft}(\mathbf{x}_i, \mathbf{y_i}) \frac{1}{2\sigma^2} \\
&= \arg\min_{\mathbf{x}} \sum_{i=1}^{N} BM_{soft}(\mathbf{x}_i, \mathbf{y}_i),
\end{aligned} \tag{4.13}$$

where N is the number of transmitted codewords.

4.2.2 Hard decision decoding

In this section, Viterbi decoding in the case of a Binary Symmetric Channel (BSC) will be discussed. We assume that the BSC alters a bit of a transmitted codeword with probability $q < 0.5$. Hence, the input to the decoder is hard symbols (bits). Due to the BSC channel, the j^{th} bit of a codeword $\mathbf{c_i}$ at the receiver may be expressed as:

$$y_{ij} = \begin{cases} 1 - c_{ij}, \text{ with } pr = q \\ c_{ij}, \text{ with } pr = 1 - q \end{cases} \tag{4.14}$$

If a received codeword $\mathbf{y}_i$ differs d bits from the transmitted codeword $\mathbf{x}_i = \mathbf{c_i}$, the probability of this event is given by the Bernoulli probability density function:

$$P(\mathbf{y}_i|\mathbf{x}_i) = q^d(1-q)^{k-d} \tag{4.15}$$

where k is the codeword length and $d_H(\mathbf{x}_i, \mathbf{y}_i) = d$ is the Hamming distance between the transmitted and received codeword. Maximizing the log-likelihood of this event gives:

$$\begin{aligned} \arg\max_{\mathbf{x}_i \epsilon C} \ln P(\mathbf{y}_i|\mathbf{x}_i) &= \arg\max_{\mathbf{x}_i \epsilon C} \left\{ d_H(\mathbf{x}_i, \mathbf{y}_i) \ln q + (k - d_H(\mathbf{x}_i, \mathbf{y}_i)) \ln (1-q) \right\} \\ &= \arg\max_{\mathbf{x}_i \epsilon C} \left\{ d_H(\mathbf{x}_i, \mathbf{y}_i) \ln \left(\frac{q}{1-q} \right) + k \ln (1-q) \right\} \\ &= \arg\max_{\mathbf{x}_i \epsilon C} \left\{ d_H(\mathbf{x}_i, \mathbf{y}_i) \ln \left(\frac{q}{1-q} \right) \right\}, \end{aligned} \tag{4.16}$$

Since $q < 0.5$ we get that:

$$q < 0.5 \rightarrow \frac{q}{1-q} < \frac{0.5}{1-q} \rightarrow \ln \left(\frac{q}{1-q} \right) < \ln \left(\frac{0.5}{1-q} \right). \tag{4.17}$$

Since $1 - q > 0.5$ we get that:

$$\ln \left(\frac{0.5}{1-q} \right) < \ln 1 = 0 \tag{4.18}$$

Considering this, (4.16) transforms to:

$$\begin{aligned} \arg\max_{\mathbf{x}_i \epsilon C} \ln P(\mathbf{y}_i|\mathbf{x}_i) &= \arg\min_{\mathbf{x}_i \epsilon C} d_H(\mathbf{x}_i, \mathbf{y}_i) \ln\left(\frac{q}{1-q}\right) \\ &= \arg\min_{\mathbf{x}_i \epsilon C} d_H(\mathbf{x}_i, \mathbf{y}_i). \end{aligned} \tag{4.19}$$

Thus, the BM for the hard decision decoding is given by:

$$BM_{hard}(\mathbf{x}_i, \mathbf{y}_i) = d_H(\mathbf{x}_i, \mathbf{y}_i). \tag{4.20}$$

The path that maximizes the overall log-likelihood is the one that minimizes the sum of Branch Metrics:

$$\arg\max_{\mathbf{x}} \ln P(\mathbf{y}|\mathbf{x}) = \arg\max_{\mathbf{x_i} \epsilon \mathbf{C}} \sum_{i=1}^{N} \ln P(\mathbf{y}_i|\mathbf{x}_i) = \arg\min_{\mathbf{x_i} \epsilon \mathbf{C}} \sum_{i=1}^{N} d_H(\mathbf{x}_i, \mathbf{y}_i). \tag{4.21}$$

A hard decision Viterbi decoder can also be employed when the communication channel experiences AWGN. In this case, the sampled matched filter output must be quantized.

4.3 LilacSat-2 convolutional code

4.3.1 Encoder

In LilacSat-2, a convolutional code with $(r, K) = (0.5, 7)$ is employed. The generator polynomials are the following:

$$g_1(s) = 1 + s + s^2 + s^3 + s^6 \text{ and } g_2(s) = 1 + s^2 + s^3 + s^5 + s^6$$

This convolutional encoding is called CCSDS (Consultative Committee for Space Data Systems) GSFC (Goddard Space Flight Center). Since the shift register consists of $M = 6$ stages, the finite state diagram of the encoder has $2^6 = 64$ states.

4.3.2 Decoder

The decoder of CCSDS NASA GSFC may be the Viterbi algorithm with soft or hard decision decoding. We assume that the channel is AWGN. Viterbi with hard decision decoding requires to decide for each symbol before decoding. Due to noise addition,

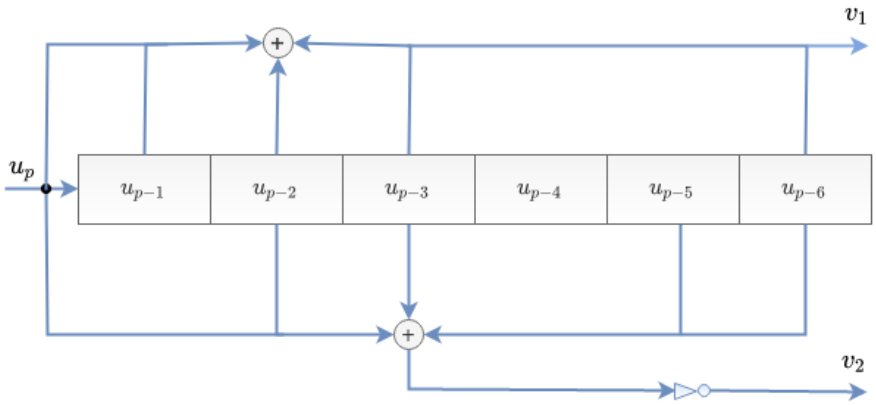


Figure 4.3: Encoder of CCSDS GFSFC convolutional code.

some symbol decisions might be wrong, which will deteriorate the performance of the hard decision decoder. On the other hand, a soft decision decoder uses the sampled matched filter output, avoiding erroneous decisions.

Simulation Results

Assuming an AWGN channel modeled as $n(t) \sim \mathcal{CN}(0, N_o)$, the signal to noise ratio is defined as:

$$SNR = r \log_2(M) \cdot \frac{E_b}{N_o}, \tag{4.22}$$

where r is the rate of the code, M the modulation order and E_b is the average energy per bit. The average energy of noise is N_o. For transmitting a CCSDS GSFC encoded binary PAM over an AWGN channel, we get that:

$$SNR = \frac{1}{2}\frac{E_b}{N_o}, \tag{4.23}$$

where $E_b = 1$. As expected, a soft decision Viterbi decoder achieves a lower bit error rate than a hard decision Viterbi decoder. In gr-satellites, the Viterbi convolutional decoding is made using the GNU Radio FECAPI Viterbi decoder, which performs soft decision[9].

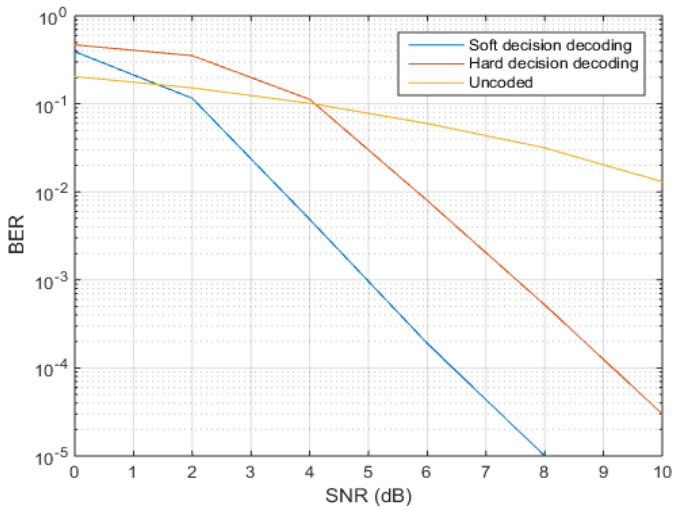


Figure 4.4: Hard and Soft decision decoding of CCSDS GSFC NASA for an AWGN channel.

Chapter 5

Satellite FSK signal detection

Most satellite decoders employ FM radios to extract information from an FSK signal. While FM demodulation is a very simple technique, it is not efficient for decoding an FSK signal. Considering this, we suggest an alternative FSK demodulation method. Additionally, we show how frame synchronization can be made in a more efficient manner. The signal studied is the GFSK telemetry of LilacSat-2.

5.1 Receiver signal model

The spinning of the satellite and its antennas in free space might cause polarization misaligments with the receive antenna. These polarization misalignments introduce spin-fading. Assuming that spin-fading is modeled by a quasi-static Rayleigh flat fading channel, the received GFSK signal is given by:

$$\begin{aligned} r(t) &= h(t)A(t)\cos\left(2\pi f_c t + \phi(t)\right) + n(t) \\ &= h(t)A(t)\cos\left(2\pi f_c t + 2\pi h \int_{-\infty}^{t} m(\tau)d\tau\right) + n(t) \\ &= h(t)s_m(t) + n(t) \end{aligned} \tag{5.1}$$

where $h(t) \sim \mathcal{CN}(0,1)$ models a quasi-static Rayleigh flat fading channel, $n(t) \sim \mathcal{CN}(0, W_o)$ is the circular symmetric additive white Gaussian noise, $\phi(t)$ is the scaled integral of the Gaussian-shaped information message $m(t)$, $h = 2f_dT$ is the modulation index and f_c is the carrier frequency.

5.2 Bit detection

In this section, we discuss about two different bit detection schemes for an FSK signal. The first scheme is an FM demodulator and the second is a non-coherent FSK detector.

5.2.1 FM demodulator

As seen in equation (2.4) of Chapter 2, to perform FM demodulation on a signal, we need to extract the phase of its baseband equivalent. However, this phase does not only contain the phase of the transmitted signal, but also a phase due to AWGN, fading and time asynchronism between the receiver and transmitter clocks. The phase due to AWGN, fading and symbol timing asynchronization, adds unwanted noise in the demodulator output. Considering a time delay τ between the transmitter and receiver clocks, the baseband equivalent of (5.1) is given by:

$$r_l(t) = r(t)e^{-j2\pi f_c t} = h(t)s_{ml}(t-\tau)e^{-j2\pi f_c \tau} + n_l(t) \text{ with } 1 \leq m \leq M. \tag{5.2}$$

where $s_{ml}(t)$ is the lowpass equivalent of the transmitted GFSK message $s_m(t)$, M is the modulation order, $n_l(t) \sim \mathcal{CN}(0, 2N_o)$ is the circular symmetric AWGN in baseband. Extracting the phase of (5.2), gives:

$$\theta(t) = \angle r_l(t) = \phi(t) + \phi_n(t) - 2\pi f_c \tau, \tag{5.3}$$

where $\phi_n(t)$ is the phase due to AWGN and spin-fading.

Another problem with FM demodulation is the threshold effect [10]. Threshold effect is the phenomenon that occurs when the SNR at the detector input decreases below a critical level. Below this level, the resulting output signal gets heavily distorted by noise. Experimentally, it has been verified that the threshold level of FM for the input SNR is roughly 10dB. The noise performance of FM can be improved if the input to the FM demodulator is low-pass filtered. Selecting a filter bandwidth slightly smaller than the bandwidth of the signal, assures that a large amount of noise is rejected, while only a small amount of message power is lost. An upper bound for the bandwidth of a frequency modulated signal is given by the Carson Rule:

$$BW = 2(f_d + 1/T). \tag{5.4}$$

According to the Carson rule, the bandwidth of the LilacSat-2 GFSK is:

$$BW = 2(4.8 + 4.8)kHz = 19.2kHz. \tag{5.5}$$

For the simulation results of this chapter, the MATLAB function fir1() was used to filter the downconverted FM signal. The specific function creates a lowpass filter with cutoff frequency W_n. In the MATLAB documentation, W_n is defined as the

frequency at which the normalized gain of the filter is 6 dB. It was found that the cutoff frequency W_n that gained the best BER performance for the FM demodulation of LilacSat-2 GFSK was 7kHz.

5.2.2 Non-coherent FSK detector

The input to the non-coherent FSK detector is the low-pass equivalent of the received signal. Assuming that the effect of a time shift τ on $s_{ml}(t)$ is negligible, (5.2) becomes:

$$r_l(t) = h(t)e^{-j2\pi f_c \tau} s_{ml}(t) + n_l(t). \tag{5.6}$$

If the fading channel adds a phase ϕ_{spin} to the signal, an unknown phase $\theta(t) = -2\pi f_c \tau + \phi_{spin}$ is introduced between the transmitter and receiver. Since τ and ϕ_{spin} are random, we suppose that $\theta(t)$ is a random variable uniformly distributed between 0 and 2π. Under these assumptions, the detection of such signal is called *non-coherent*. A non-coherent detector for FSK correlates (5.6) with the baseband tones for each modulating frequency:

$$\langle \mathbf{r}_l, \mathbf{s}_{ml} \rangle = \int_{-\infty}^{\infty} r_l(t) s_{ml}^*(t) dt. \tag{5.7}$$

For binary FSK, the tones for each modulating frequency f_m are given by:

$$s_{1l}(t) = e^{j2\pi f_1 t} \text{ and } s_{2l}(t) = e^{j2\pi f_2 t}, \tag{5.8}$$

where f_1 and f_2 are the frequencies for transmitting zero and one, respectively. These frequencies are given by:

$$f_1 = f_c - f_d \text{ and } f_2 = f_c + f_d, \tag{5.9}$$

where $f_d = h/2T$ is the frequency deviation. The detector decides for the tone with the maximum absolute value. Hence, the decision rule for a non-coherent binary FSK detector is given by:

$$\hat{m} = \underset{1 \leq m \leq M}{\arg\max} \left| \int_{-\infty}^{\infty} r_l(t) s_{ml}^*(t) dt \right|. \tag{5.10}$$

The bit error probability of a non-coherent FSK detector under an AWGN channel is given by:

$$p_b = \frac{1}{2} e^{\frac{-SNR}{2}}, \tag{5.11}$$

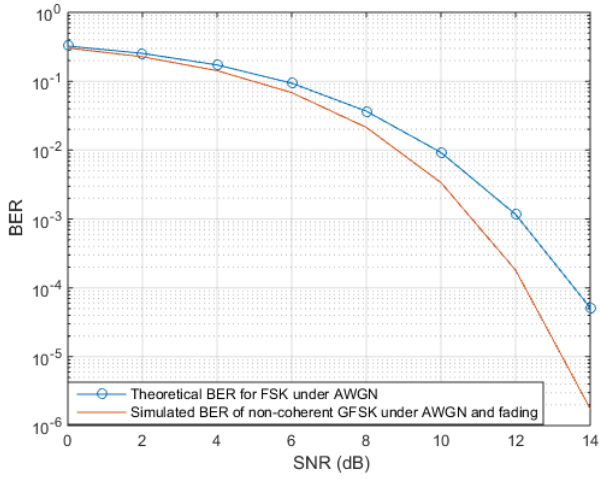


Figure 5.1: Theoretical BER for non-coherent FSK detection under AWGN channel versus experimental BER for non coherent detection of GFSK with quasi-static Rayleigh flat fading channel. Synchronized transmitter and receiver clocks were considered.

where SNR is the signal to noise ratio which is defined as:

$$SNR = \frac{P_s}{P_n} = \frac{P_s}{2N_oB}, \tag{5.12}$$

where $2N_o$ is the power spectral density of noise and B the signal bandwidth. The latter can be computed by the Carson Rule. The average power ber symbol is denoted as P_s in (5.12) and is given by:

$$P_s = \frac{L}{M}\sum_{n=1}^{M}|s_{ml}(n)|^2, \tag{5.13}$$

where L is the oversampling factor and M the number of samples in the signal. Non-coherently detecting the GFSK signal of equation (5.6) results to a higher bit error probability than p_b. This is due to the symbol interference introduced by the Gaussian filter and spin-fading.

5.3 Frame synchronization

In the gr-satellites decoder, frame synchronization is performed on the output of the Viterbi decoders. This means that the GFSK signal has already been demodulated and convolutional decoded. Frame synchronization after bit detection should be

avoided, because any occurring errors on the preamble bits will reduce the probability of synchronizing to a frame. To avoid this, we suggest performing frame synchronization prior to bit detection. Also, if the start of the frame is found correctly, the symbol timing asynchronism is no longer a problem.

5.3.1 Frame synchronization with correlation

In this section, we show how frame synchronization prior to bit detection can be made, using correlators. To implement the correlators, the MATLAB function xcorr() was used. This function computes the true correlation between two signals. To perform frame synchronization, the receiver creates a low-pass GSK signal $p(t)$, which carries the convolutionally encoded preamble bits. Assuming that $r_l(t)$ is the baseband equivalent of the received signal, the receiver correlates it with $p(t)$. The true correlation of $r_l(t)$ and $p(t)$ is given by:

$$R_{x,p}(m) = E\left\{r_l(t+m)p^*(t)\right\} = \frac{1}{N}\sum_{n=0}^{N-m-1} r_l(t+m)p^*(t). \tag{5.14}$$

The xcorr() function computes the non-normalized version of (5.14):

$$\hat{R}_{x,p}(m) = \sum_{n=0}^{N-m-1} r_l(t+m)p^*(t). \tag{5.15}$$

The m that maximizes (5.15) is called the lag index and denotes the delay in samples to find the most probable starting point of a packet in the received signal.

5.4 Suggested decoder model

In this section, we suggest a new decoding scheme for the LilacSat-2 GFSK signal. A preamble-based frame synchronization is performed first, using correlators. Then, non coherent FSK detection is made on the captured frame. Convolutional decoding is performed using a hard decision Viterbi decoder, since the non coherent FSK detector output gives bits and not samples. Descrambling, Reed-Solomon decoding and telemetry bit extraction from the resulting protocol data unit can be made using the corresponding blocks from gr-satellites.

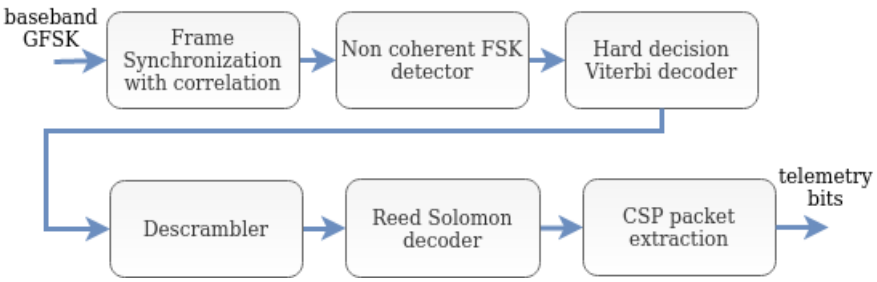


Figure 5.2: Suggested decoder for LilacSat-2 GFSK Telemetry.

5.4.1 Simulation results

In this section, we compare the performance of the gr-satellites decoder and the suggested decoding scheme, which is illustrated in figure 5.2.

To sample at the correct symbol timing instants, the gr-satellites decoder employs a clock synchronization algorithm. The decoder we suggest does not need a clock synchronization algorithm, since the receiver locks to a frame before symbol detection. For the simulation results of this section, the gr-satellite decoder locks to the frame with the minimum error of preamble bits, while the suggested decoder locks at the most probable starting point of a frame equation (5.15) gives. The simulation results in figure (5.3) show that the suggested decoder achieves approximately a 2dB gain.

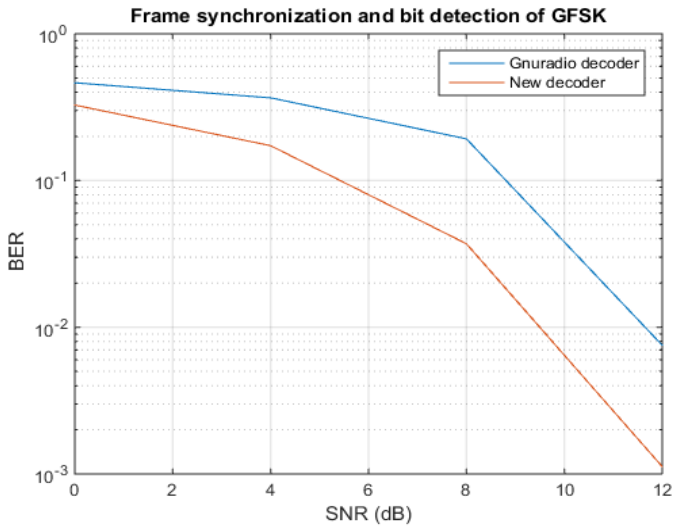


Figure 5.3: Frame synchronization and bit detection of uncoded LilacSat2 GFSK for quasi-static Rayleigh flat fading channel.

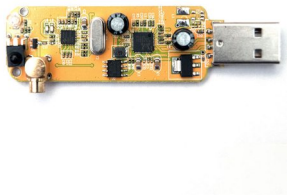
Chapter 6

Experimental Results

6.1 Description

In this chapter, we demonstrate the process of capturing and decoding a satellite signal. The satellite signals were captured using an 8-dollar RTL-SDR receiver connected to a hand-held Yagi antenna at the UHF band. The captured signal was decoded using free-to-use Windows and Linux programs.

(a) The Yagi antenna.



(b) The RTL-SDR receiver.

6.2 Capturing and decoding LilacSat-2 GFSK

To identify the position of the satellite flying overhead, the azimuth and elevation of the satellite were measured, using the application Gpredict [11].

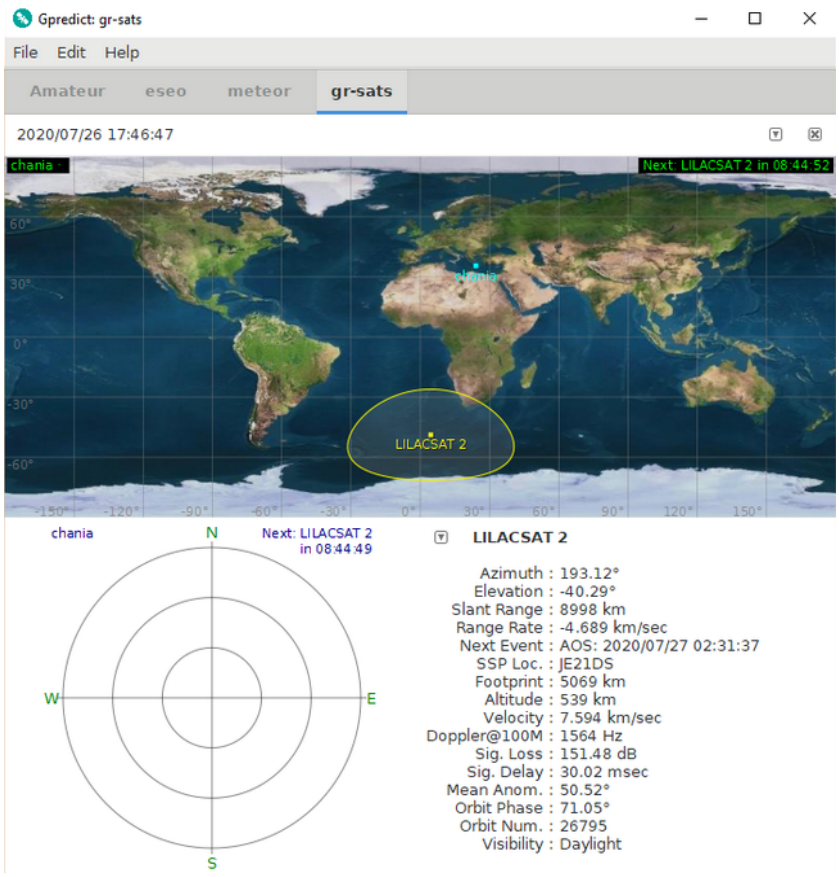


Figure 6.2: Caption from the Gpredict application.

The program SDR Sharp [12] is used to record the satellite signal. The user connects the Gpredict application to SDR Sharp for the Doppler shift correction. Doppler shift is the actual change in frequency, due to relative motion of source (satellite) and observer (ground station).

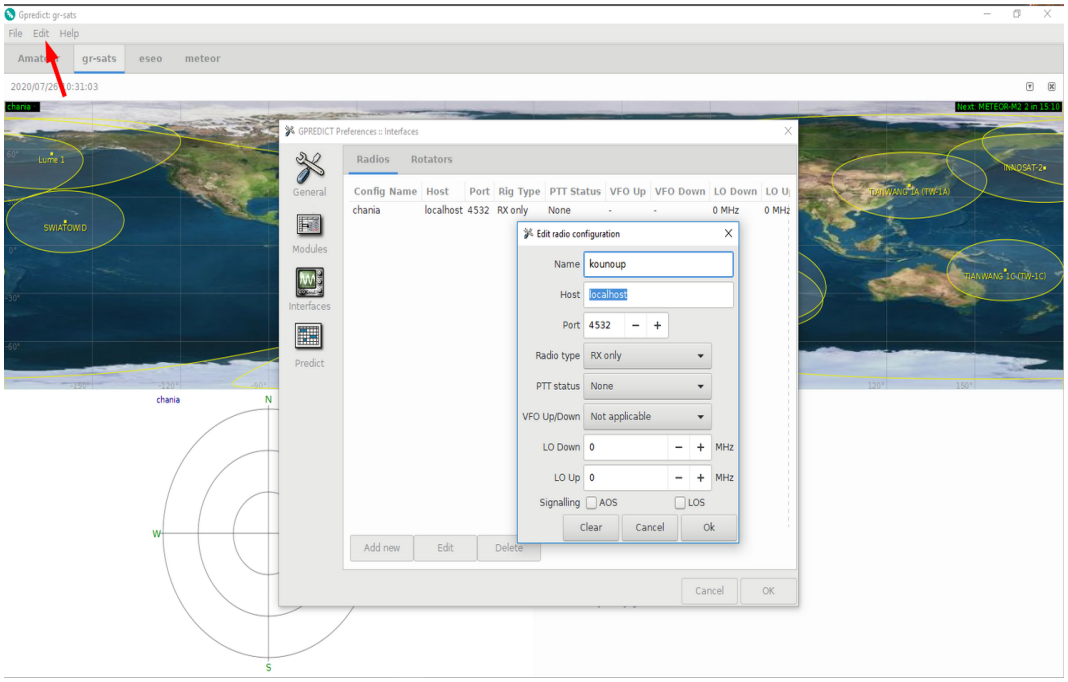


Figure 6.3: Caption from the Gpredict application. Selection of the port that Gpredict will send the Doppler shift to the SDR Sharp application.

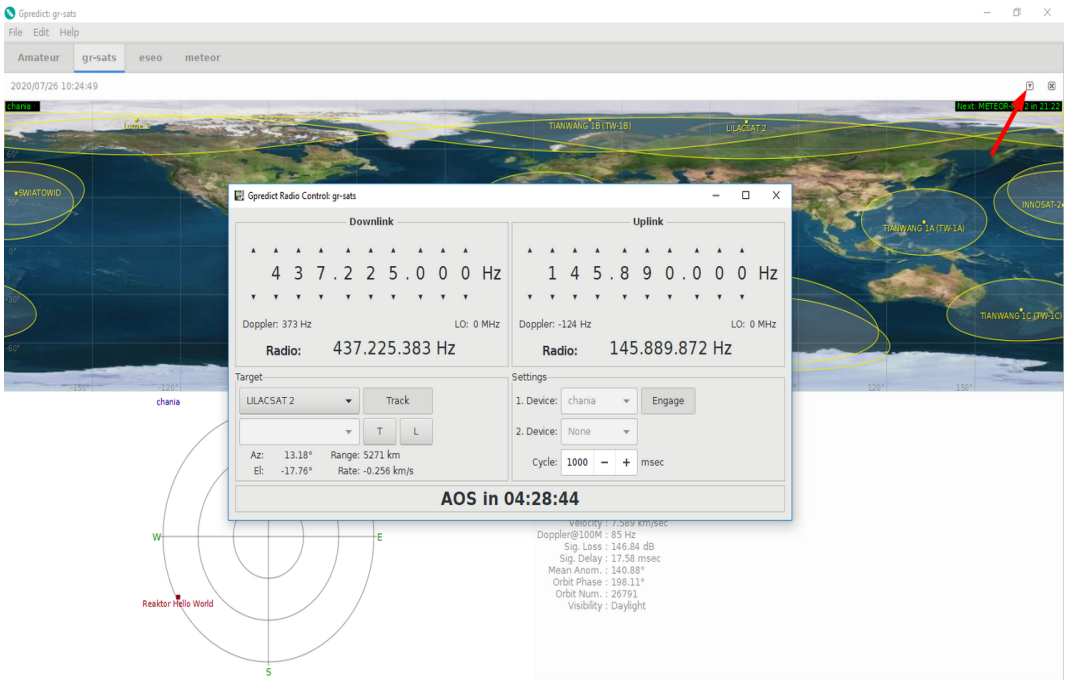


Figure 6.4: Caption from the Gpredict application. The user selects the satellite from Target and presses Track and Engage. Now Gpredict sends to SDR Sharp the Doppler shift.

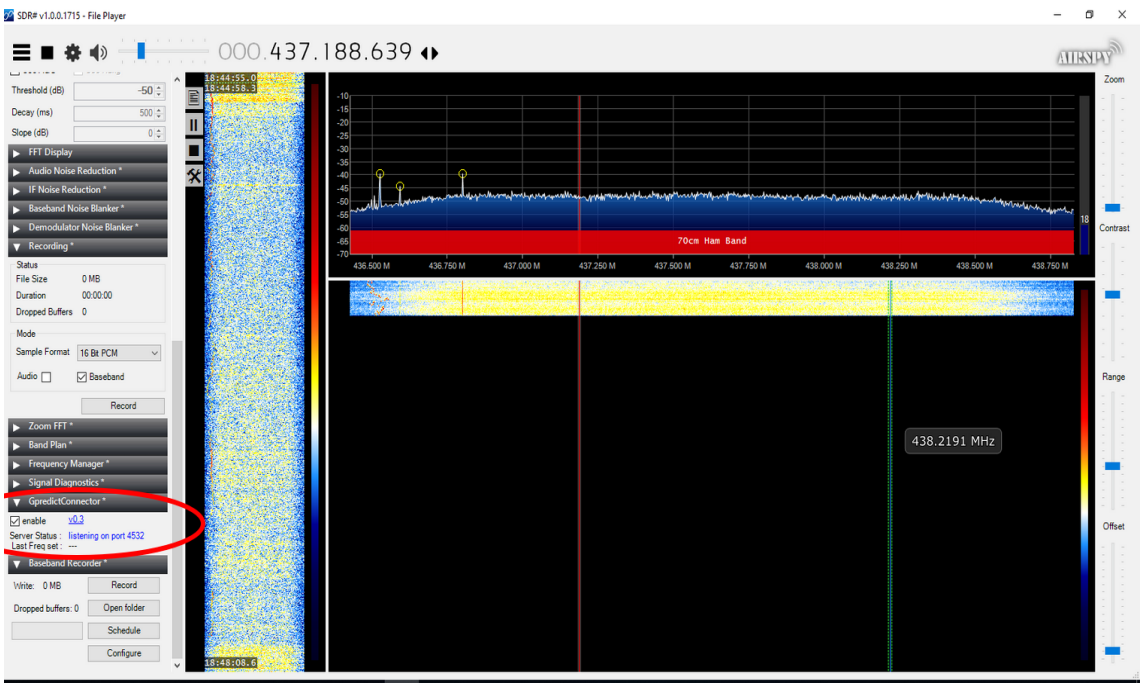


Figure 6.5: Caption from the SDR Sharp application. SDR Sharp is connected to Gpredict via the specified port.

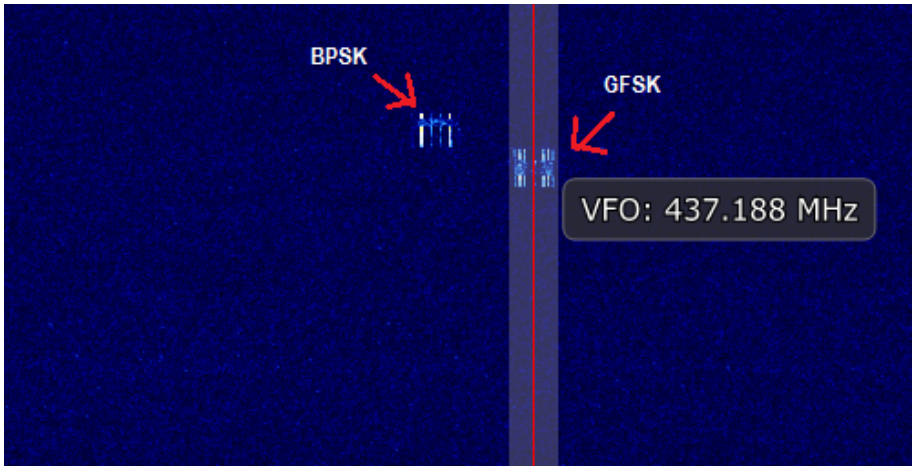


Figure 6.6: Caption from the application SDR sharp, displaying the waterfall of a signal. At 437.188MHz, there is a GFSK packet of LilacSat-2 telemetry. Next to it, at 437.165MHz, there is a BPSK packet of LilacSat-2 telemetry.

The satellite signal is also FM demodulated by the application SDR sharp. Additionally, a bandpass filter of adjustable bandwidth is used to remove the noise. In SDR sharp, the measuring unit of the Fourier transform is dBFS (Decibels relative

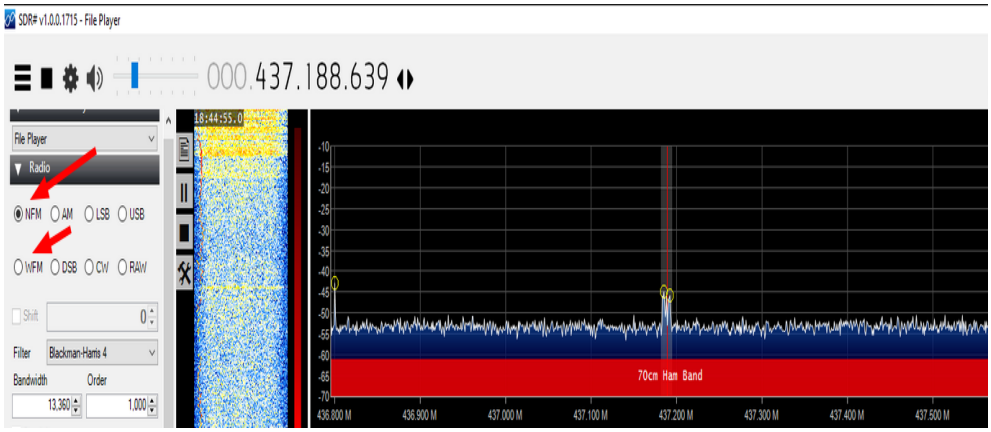


Figure 6.7: Caption from the application SDR sharp, displaying the Fourier of a signal, measured in dBFS. At 437.188MHz, there is a GFSK packet of LilacSat-2 telemetry. The shaded region is the bandwidth of the bandpass filter. The red arrows show the option for NFM (Narrow FM) and WFM (Wideband FM) demodulation.

to full scale), which is calculated by:

$$|X_{FS}(F)| = 20\log_{10}\left(\frac{|X(F)|)}{FS}\right), \tag{6.1}$$

where $|X(F)|$ is the absolute value of the signal's Fourier transform and FS is the full-scale factor. The value 0 dBFS corresponds to 100% full scale (FS) and is the maximum allowed value the Fourier transform can get.

The FM demodulated satellite signal from SDR Sharp is then recorded by Audacity [13] into a 48kHz .wav file. The generated .wav file is sent to the gr-satellites decoder, using the gnuradio application gr-frontends. The gr-satellites decoder is displayed in figure 2.3 of Chapter 1.

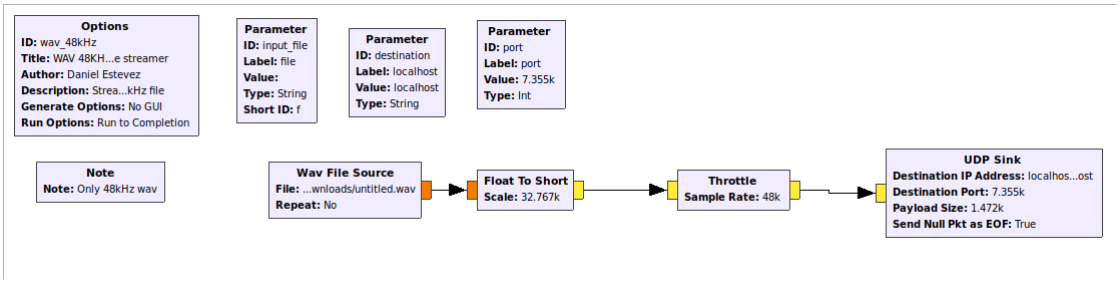


Figure 6.8: The gr-frontends gnuradio application.

The output of the gr-satellites application is a hexdump file (figure 6.9), that contains the CSP packet.

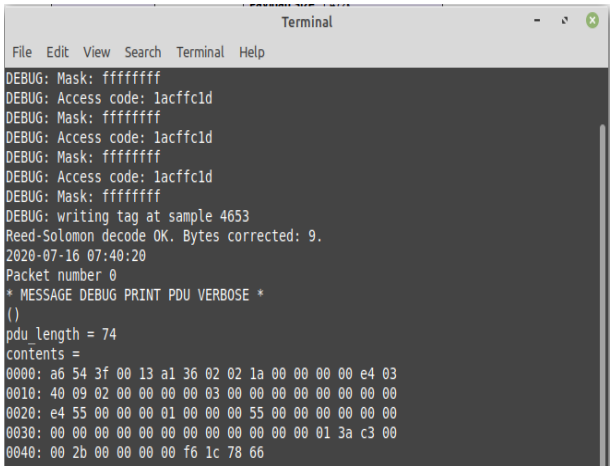


Figure 6.9: Decoded LilacSat-2 GFSK telemetry in hexadecimal form using gr-satellites.

To show the telemetry information graphically, first the CSP packet should be converted to a KISS packet (figure 6.10).

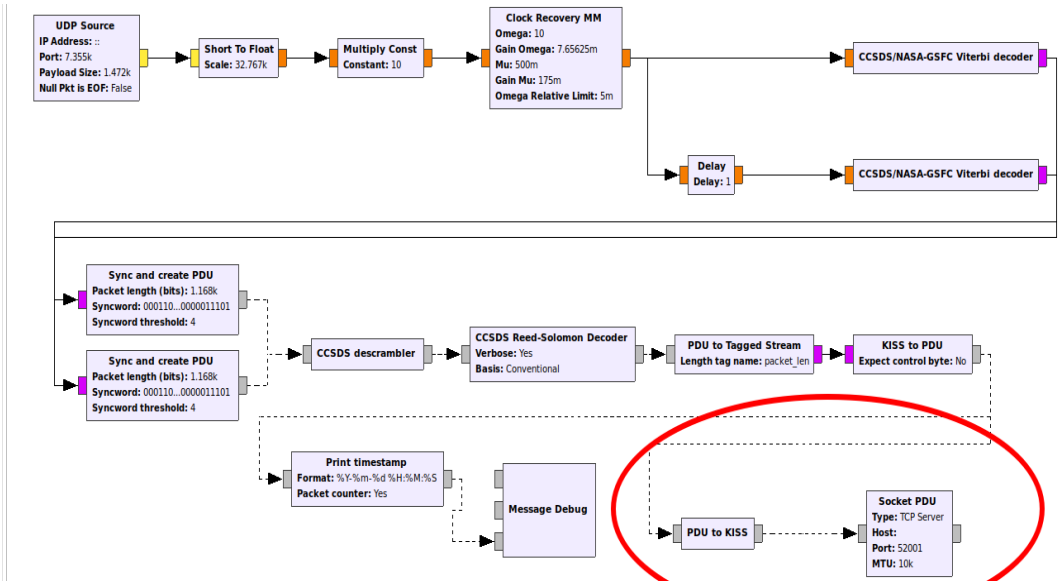


Figure 6.10: The gr-satellites decoder. The CSP packet is converted to a KISS packet (PDU to KISS). The Linux machine that runs the gr-satellites decoder sends to a Windows machine the KISS file via a TCP socket.

The KISS packet is sent via a TCP socket to the client of the Windows program GetKiss[14].

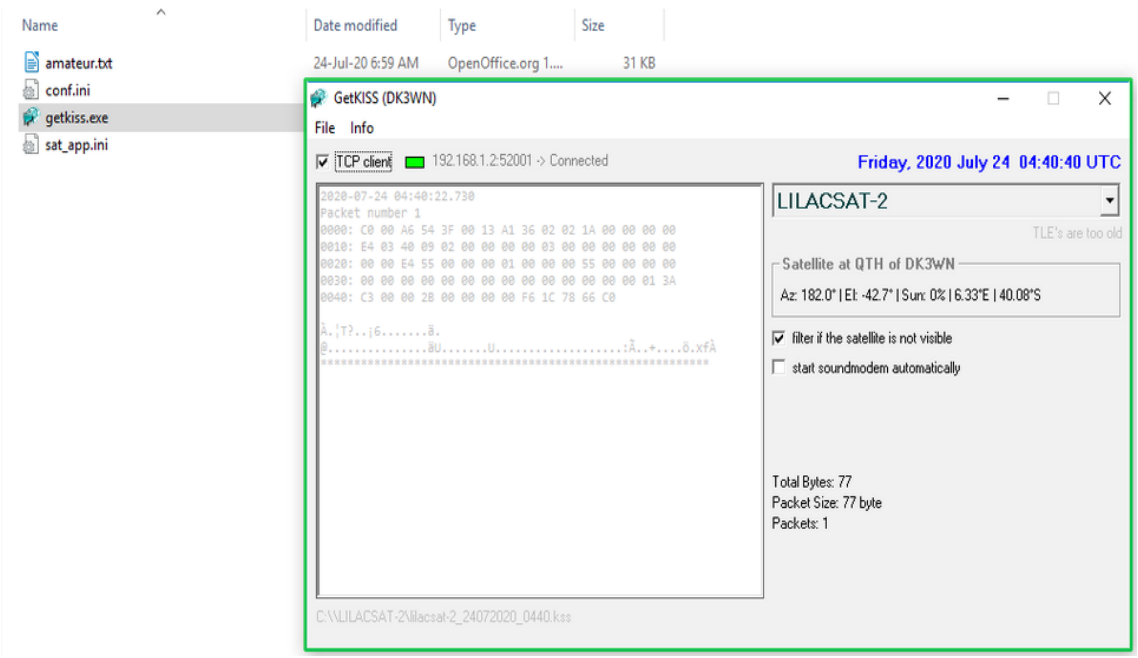


Figure 6.11: The application GetKiss in the Windows machine receives the KISS packet (which is printed in the terminal of the application).

The GetKiss program automatically creates a .kss file that contains the KISS stream obtained from gr-satellites. The .kss file is given as input to the DK3WN Telemetry Decoder for LilacSat-2[14], which is a graphical tool that can be used to display the telemetry of LilacSat-2 graphically from the KISS file.

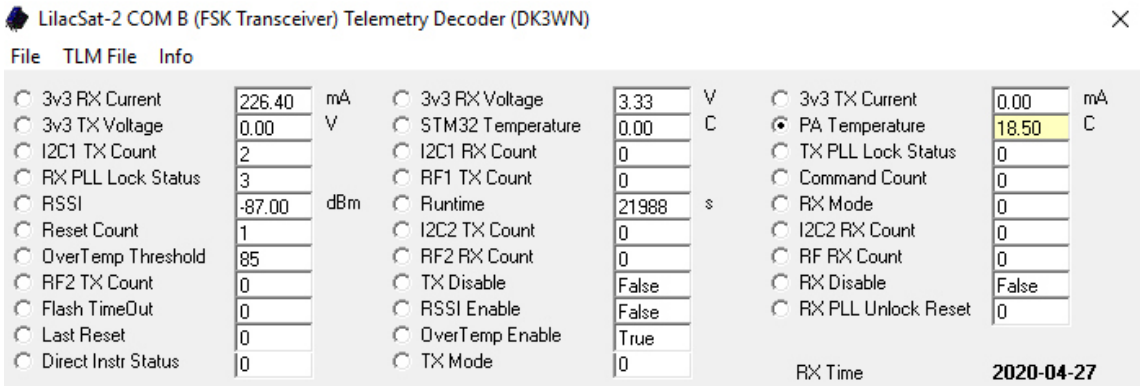


Figure 6.12: Decoded LilacSat-2 GFSK telemetry from the DK3WN Telemetry Decoder program.

The satellite transmits to earth information regarding its receiver and transmitter (current,voltage etc), the temperature of its microcontroller (STM32 Temperature) and power amplifier (PA temperature), the RSSI (Received Signal Strength Indicator) etc.

Chapter 7

Conclusions

This work gave insight to important elements of a satellite decoder. The concepts of symbol timing synchronization and Viterbi convolutional decoding were thoroughly explained. Also, a new decoder for the CubeSat LilacSat-2 GFSK telemetry was suggested. This decoder employed an FSK non-coherent bit detector and used a preamble-based correlation frame synchronization scheme. The simulation results showed that for uncoded LilacSat-2 GFSK signals, the suggested decoder achieves approximately a 2dB gain. Also, the process of capturing and decoding a satellite signal was explained in detail.

As future work, the decoders should be tested for encoded satellite signals. Additionally, a gnuradio implementation of the suggested decoder could be made. Finally, the same bit and frame synchronization scheme could be used for other CubeSat decoders.

Chapter 8

Appendix

The Gaussian shaping pulse $g_s(t)$ is given by[15]:

$$g(t) = h(t) \circledast \chi(t) \tag{8.1}$$

where $h(t)$ is a rectangular pulse of amplitude 1/T on the interval [-T/2,T/2] and $\chi(t)$ is the impulse response of a Gaussian filter with passband at -3dB equal to B:

$$\chi(t) = \sqrt{\frac{2\pi}{\ln 2}} B \exp\left\{-2\pi^2 B^2 t^2\right\} / \ln 2 \tag{8.2}$$

After resolution of the convolution product, the function $g_s(t)$ can be written in the form:

$$g_s(t) = \frac{1}{2T}\left[\text{erf}\left(\pi B\sqrt{\frac{2}{ln2}}\left(t+\frac{T}{2}\right)\right) - \text{erf}\left(\pi B\sqrt{\frac{2}{ln2}}\left(t-\frac{T}{2}\right)\right)\right] \tag{8.3}$$

where T is the symbol period and $\text{erf}(x)$ represents the error function, defined by:

$$\text{erf}(x) = \frac{2}{\sqrt{\pi}}\int_0^x \exp\{-u^2\}du \tag{8.4}$$

The normalized bandwidth BT allows the time spreading of function $g_s(t)$ to be fixed.

Bibliography

[1] D. Estevez, "Gnuradio application for decoding satellite signals." [Online]. Available: https://github.com/daniestevez/gr-satellites/tree/maint-3.8-v2

[2] "PEOSAT website, lilacSat-2." [Online]. Available: https://www.pe0sat.vgnet.nl/satellite/chinese-satellites/lilacsat-2/

[3] R. Proesc, *Technical Handbook for Satellite Monitoring.* Books on Demand, 2019.

[4] M. Rice, *Digital Communications A Discrete-Time Approach.* Pearson, 2008.

[5] K. H. Mueller and M. Müller, "Timing recovery in digital synchronous data receivers," *IEEE Trans. Commun.*, vol. 24, pp. 516–532, May 1976.

[6] S. A. Heinrich Meyr, Mark Moeneclaey, *Digital Communication Receivers.* Wiley Interscience Publications, 1997.

[7] R. W. Schafer and L. R. Rabiner, "A digital signal processing approach to interpolation," *Proceedings of the IEEE*, vol. 61, no. 6, pp. 692–702, 1973.

[8] Y. Kao and C. Hsu, "The empirical formula for designing MMSE interpolation filter," in *2006 International Conference on Communication Technology*, 2006, pp. 1–4.

[9] "FEC API." [Online]. Available: https://www.gnuradio.org/doc/doxygen/page_fec.html/

[10] Thomas-Chandrasekhar, *Analog Communication.* Tata McGraw-Hill Education, 2006.

[11] "Gpredict." [Online]. Available: http://gpredict.oz9aec.net/

[12] "SDR Sharp." [Online]. Available: http://gpredict.oz9aec.net/

[13] "Audacity." [Online]. Available: https://www.audacityteam.org/

[14] "Software DK3WN." [Online]. Available: http://www.dk3wn.info/p/?page_id=75524

[15] Huaping Liu, V. Venkatesan, C. Nilsen, R. Kyker, and M. E. Magana, "Performance of frequency hopped noncoherent GFSK in correlated Rayleigh fading channels," in *IEEE International Conference on Communications, 2003. ICC '03.*, vol. 4, 2003, pp. 2779–2783 vol.4.