在两个频率之间进行切换，f1=1e6/65，f2=1e6/29

帧结构：



Preamble: 5 bytes

20个连续的“10”序列

1010101010101010101010101010101010101010

Sync word：3 bytes

11000001 10010100 11000001

Length byte：1 bytes

数据长度的二进制

例：要发送的数据为[1, 2, 3, 4]，长度为4，4的二进制为00000100

Message：

需要发送的数据的二进制

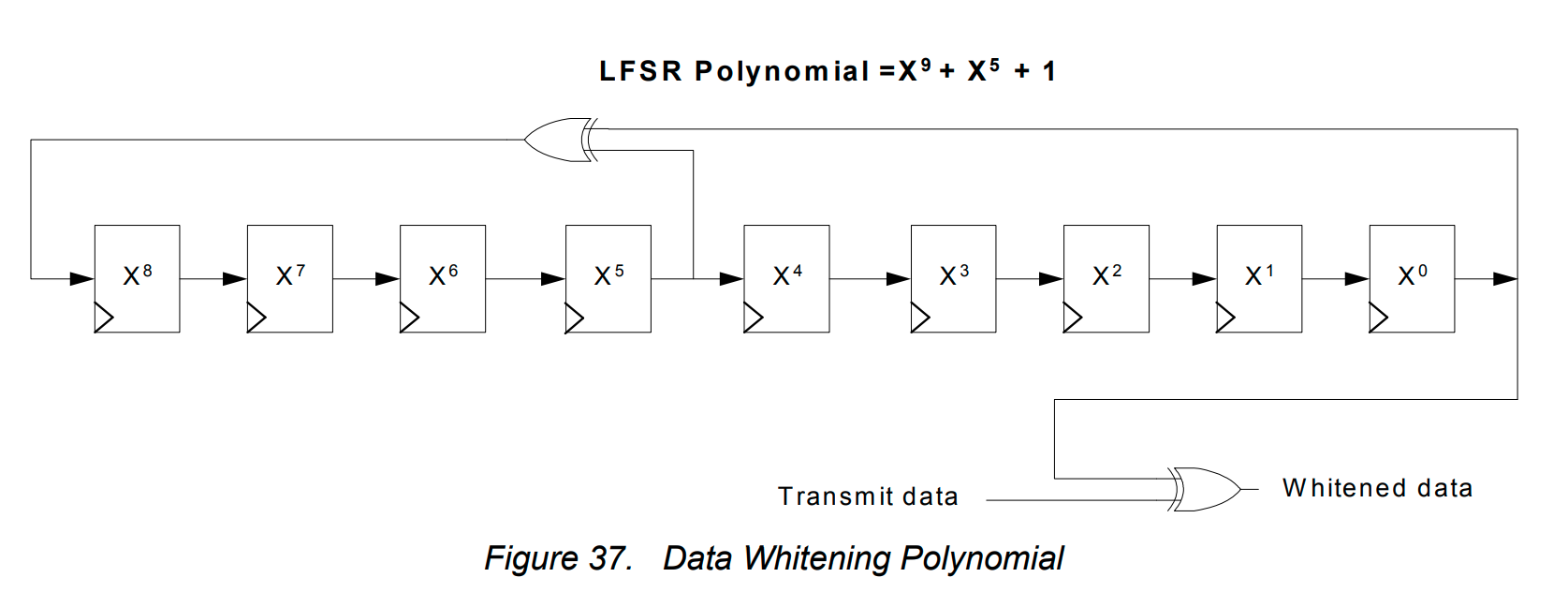
Payload CRC：

对Payload计算CRC，Payload包含Length和Message，

CRC计算规则参考C代码见sx1276文档里附录7.5. Example CRC Calculation，参考fsk.c中CRC\_calc函数

Payload和CRC都需要做whitening，接受与发送时都需要与生成的伪随机序列作异或

Whitening序列生成规则如下：



示例：

9 8 7 6 5 4 3 2 1 0 初始状态值全为1，下一时刻状态值右移，第9位

1 1 1 1 1 1 1 1 1 1 为第5位和第0位的异或，这样不停循环，取出来的

0 1 1 1 1 1 1 1 1 1 第0位即为所求的序列。

0 0 1 1 1 1 1 1 1 1 这里可以直接将提前将序列存下来，参考fsk.c中

0 0 0 1 1 1 1 1 1 1 whitening\_seq数组

0 0 0 0 1 1 1 1 1 1

1 0 0 0 0 1 1 1 1 1

1 1 0 0 0 0 1 1 1 1

1 1 1 0 0 0 0 1 1 1

1 1 1 1 0 0 0 0 1 1

0 1 1 1 1 0 0 0 0 1

0 0 1 1 1 1 0 0 0 0

1 0 1 1 1 1 0 0 0 0

Whitening序列前18\*8位：

11111111 10000111 10111000 01011001 10110111 10100001 11001100 00100100 01010111 01011110 01001011 10011100 00001110 11101001 11101010 01010000 00101010 10111110

例：

以要发送的数据为[1, 2]为例，长度为2

Length数据为00000010，Message数据为 00000001 00000010

对Length和Message即Payload(00000010 00000001 00000010),

计算CRC得到10010011 11100000

再对Length、Message和CRC作Whitening

即 00000010 00000001 00000010 10010011 11100000和上面的Whitening序列前5比特

11111111 10000111 10111000 01011001 10110111相异或得到

11111101 10000101 00101011 10111001 01010111

那么最终一共发送13个bytes

10101010 10101010 10101010 10101010 10101010 （Preamble）

11000001 10010100 11000001 （Sync Word）

11111101 （Length）

10000101 00101011 （Message）

10111001 01010111 （CRC）

参考代码fsk.c

fsk模块的使用：

在/gr-chirp/build目录下运行

sudo make install

sudo ldconfig

即可正确安装block

fsk位置位于gnuradio右侧chirp-→fsk

fsk block需要与Socket PDU连接

通过给localhost 52001端口发送udp数据即可实时传输数据

udp.py实现了每隔两秒用udp发送数据，第5行的data变量为循环发送的数据

example位于/gr-chirp/examples/chirp\_example.grc