Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Szélessávú Hírközlés és Villamosságtan Tanszék

Rádióátviteli mérések laboratórium 2

9. mérés
Digitális KF

Szilágyi Gábor NOMK01

Budapest, 2023. március 23.

1. A feladat

Tehát a megfejtés szinkronizáló minta utáni része a következő: `0x46D277306`

Hivatkozások

A. adsb.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

// ADS-B signal length
#define PCKT_LEN  240

// Low Pass FIR Filter length
#define FIR_LEN  8

// input data buffer length
#define BUF_SIZE  8192

int main(int argc,char **argv)
{
    // general index variables
    int i,j,k;

    // input data buffer
    unsigned char buffer[BUF_SIZE];
    int read_len, bix;

    // ADS-B preamble pattern
    unsigned char adsb_preamble[16]={1,0,1,0,0,0,0,1,0,1,0,0,0,0,0,0};

    // store previous samples for FIR filtering
    unsigned int fifo[FIR_LEN] = {0};
    unsigned int fptr = 0;

    // Absolute value Look-up Table: I-Q --> ABS(.)
    // input data: unsigned char I and unsigned char Q
    unsigned int iq_to_abs[256][256];

    // "iq_to_abs" array initialization
    for(i=0;i<256;i++) {
        for(j=0;j<256;j++) {
            iq_to_abs[i][j] = (unsigned int)sqrt( (i-128)*(i-128) + (j-128)*(j-128) );
        }
    }

    // variables
    unsigned int abs_val;
    int accumulator = 0;
    unsigned char bit;
    unsigned int stm;
    unsigned char hex;

    i = 0;
    j = 0;

    // main loop
    do {
        read_len = fread(buffer, 1, BUF_SIZE, stdin);   // read data to input buffer
        stm = 0;
        hex = 0;
        for(bix=0; bix<read_len-1; bix+=2) {
            // convert I-Q to magnitude
            abs_val = iq_to_abs[buffer[bix]][buffer[bix+1]];

            // FIR filtering
            accumulator = accumulator-fifo[fptr]+abs_val;
            fifo[fptr] = abs_val;
            fptr = (fptr+1)%FIR_LEN;

            // Decoding
            if(fifo[(fptr-FIR_LEN/2)%FIR_LEN] > accumulator/FIR_LEN) {
                bit = 1;
            }
            else {
                bit = 0;
```

```
            }

            // ADS-B packet search and print
            // State machine
            if(stm < 16) {
                if(adsb_preamble[stm] == bit) {
                    stm++;
                }
                else {
                    stm = 0;
                }
            }
            else {
                if(stm < 240) {
                    if(stm == 16) {
                        printf("\n*");
                    }
                    // Manchaster-decode
                    if(stm>15 && stm%16==15) {
                        printf("%02x", hex);
                    }
                    if(stm%2==0) {
                        hex = hex << 1;
                        if(bit==1) {
                            hex = hex | 1;
                        }
                    }
                    stm++;
                }
                else {
                    stm = 0;
                }
            }
            //printf( "%d\t%d\t%d\t%d\n", buffer[bix], buffer[bix+1], abs_val,
    accumulator/FIR_LEN );
        }

        // uncomment if not testing
        //break;
    } while(read_len>0);
    printf("\n");
    return 0;
}
```

B. build

```
#!/bin/bash
gcc adsb.c -lm -o demod
```

C. run

```
#!/bin/sh
cat NOMK01___6186.dat | ./demod > test.txt
#cat NOMK01___6186.dat | ./demod
#gnuplot plotter.gp
```