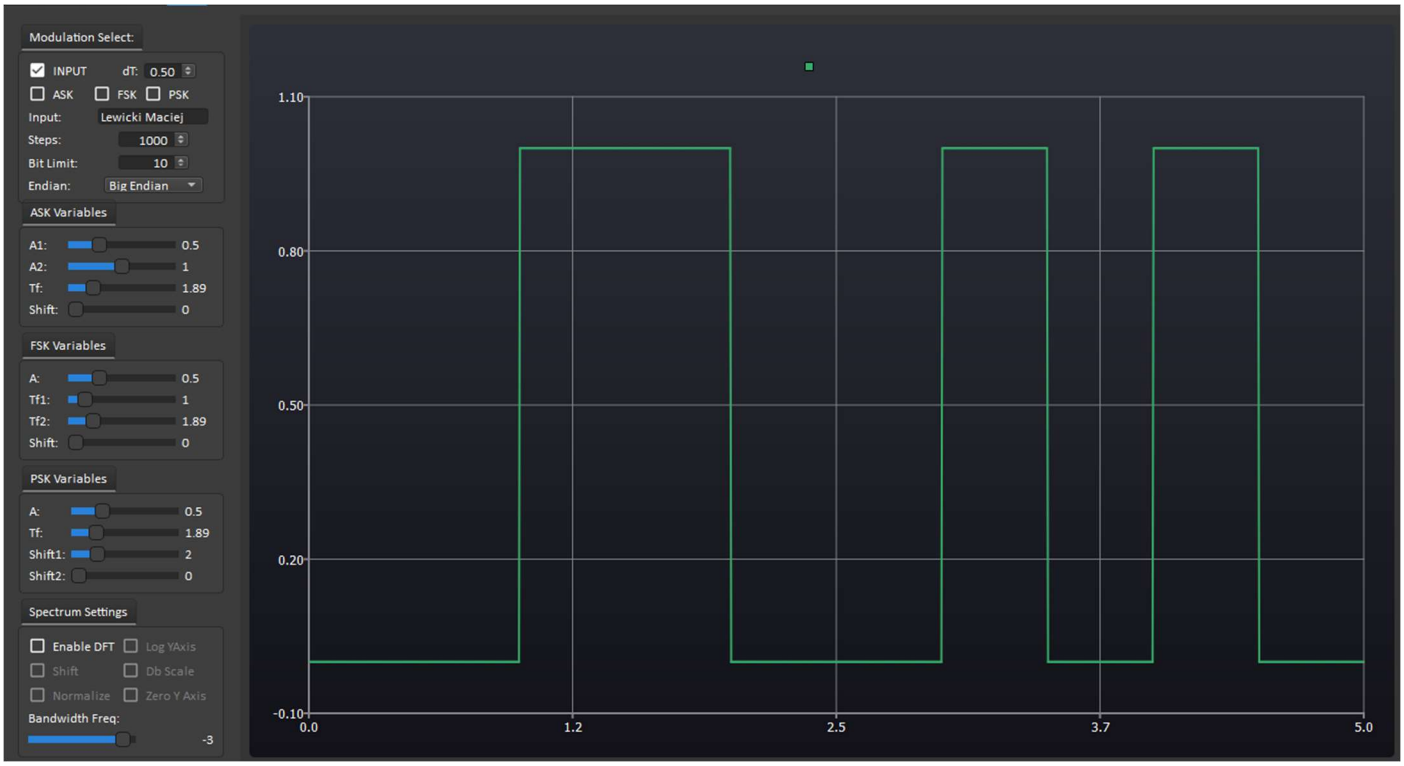
Laboratorium nr 5

1) Napisz funkcję, która na wejściu przyjmuje zadany ciąg znaków ASCII, a na wyjściu zwraca strumień binarny i wypisuje go na konsolę w kolejności little endian lub big endian. Przykład pseudokodu:

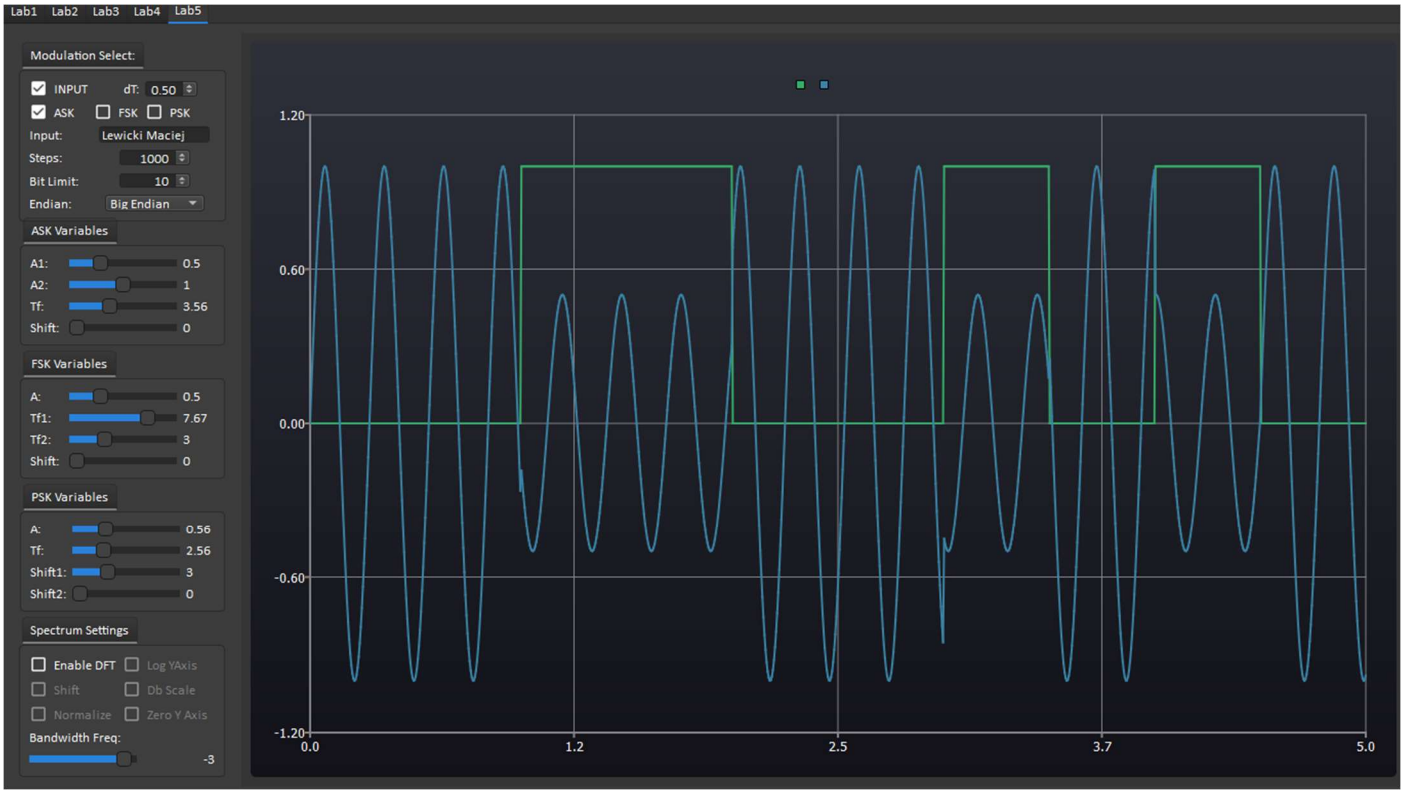
```
QBitArray Lab5::bitsFromString(QString s, Endian e)
{
    QByteArray ba = s.toLocal8Bit();
    const char* c = ba.data();
    QBitArray bitArr = QBitArray::fromBits(c,ba.size()*8);
    if(e == LittleEndian)
        reverseBitArray(bitArr);
    return bitArr;
}
void Lab5::reverseBitArray(QBitArray &arr)
{
    for(int i=0; i<arr.count()/2;i++)
    {
        bool bit = arr.at(i);
        arr.setBit(i,arr.at(arr.count()-1-i));
        arr.setBit(arr.count()-1-i, bit);
    }
}
```

Wykresy

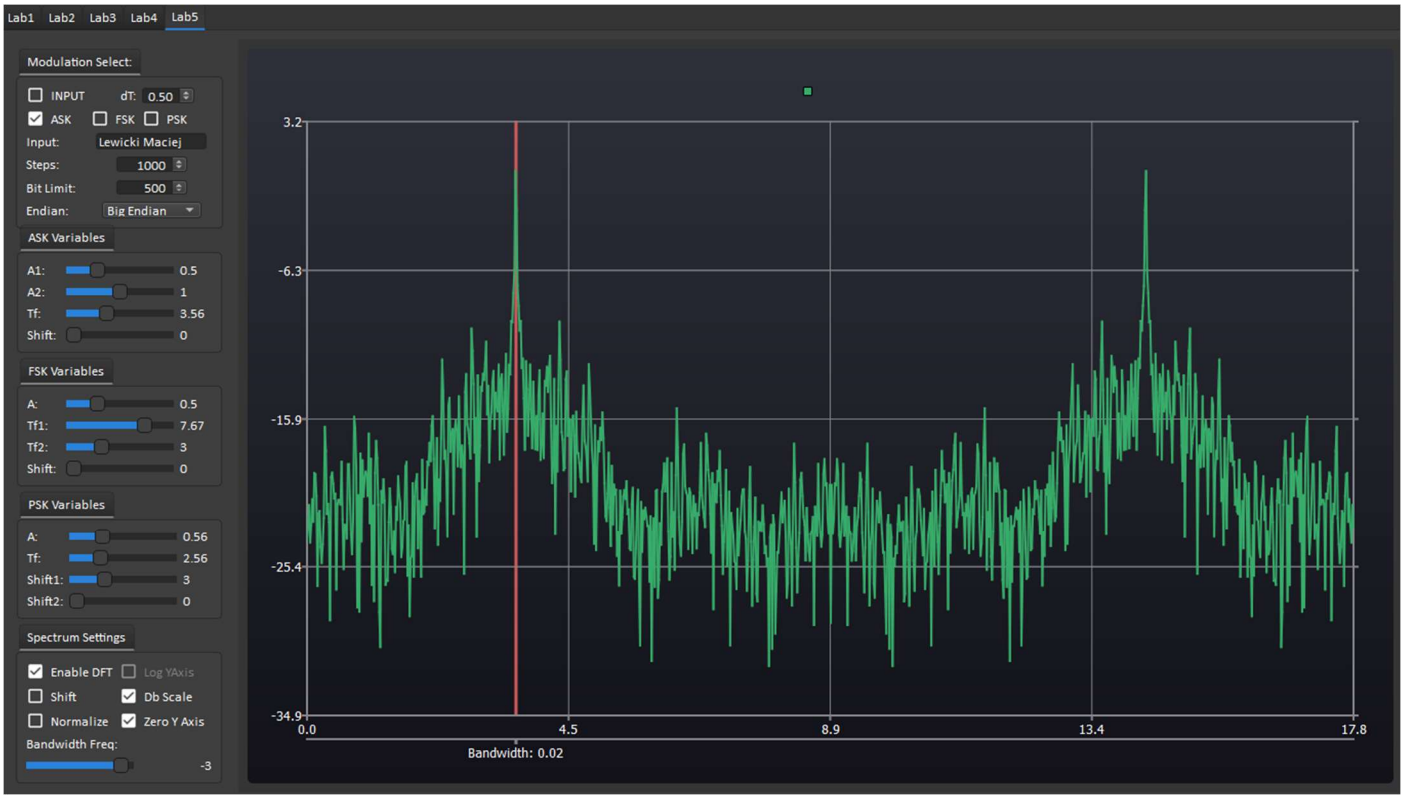
Sygnał wejściowy:



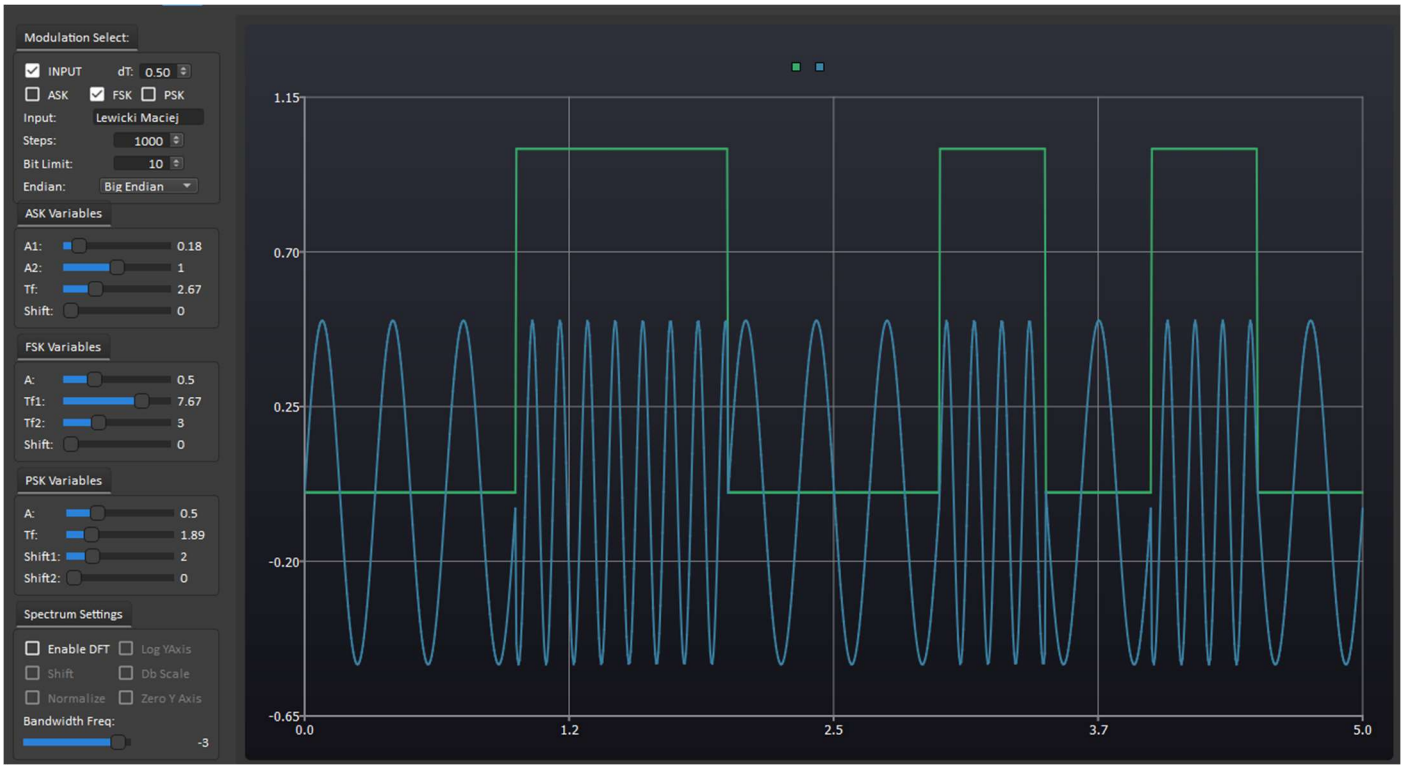
ZA(t):



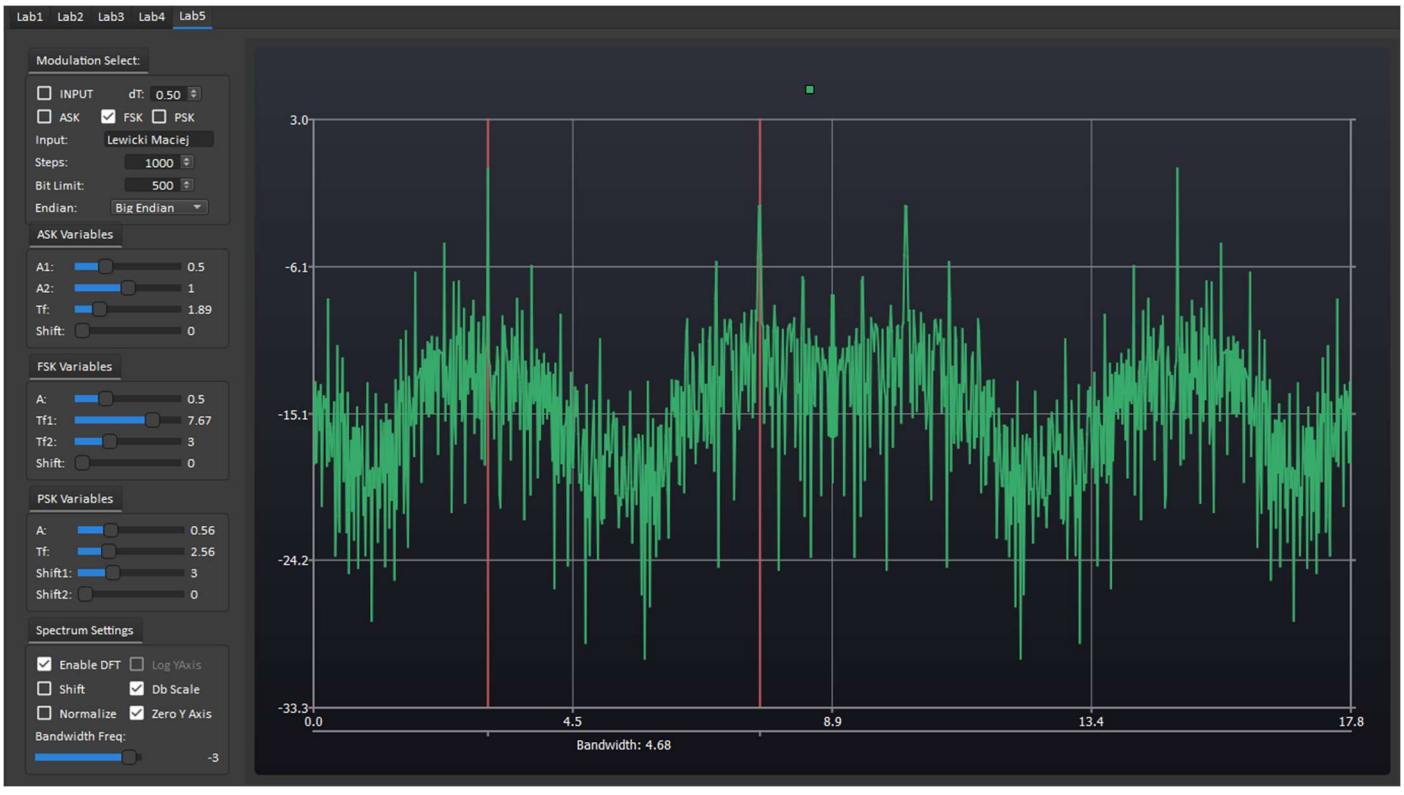
Widmo + szerokość pasma = 0.02[Hz]:



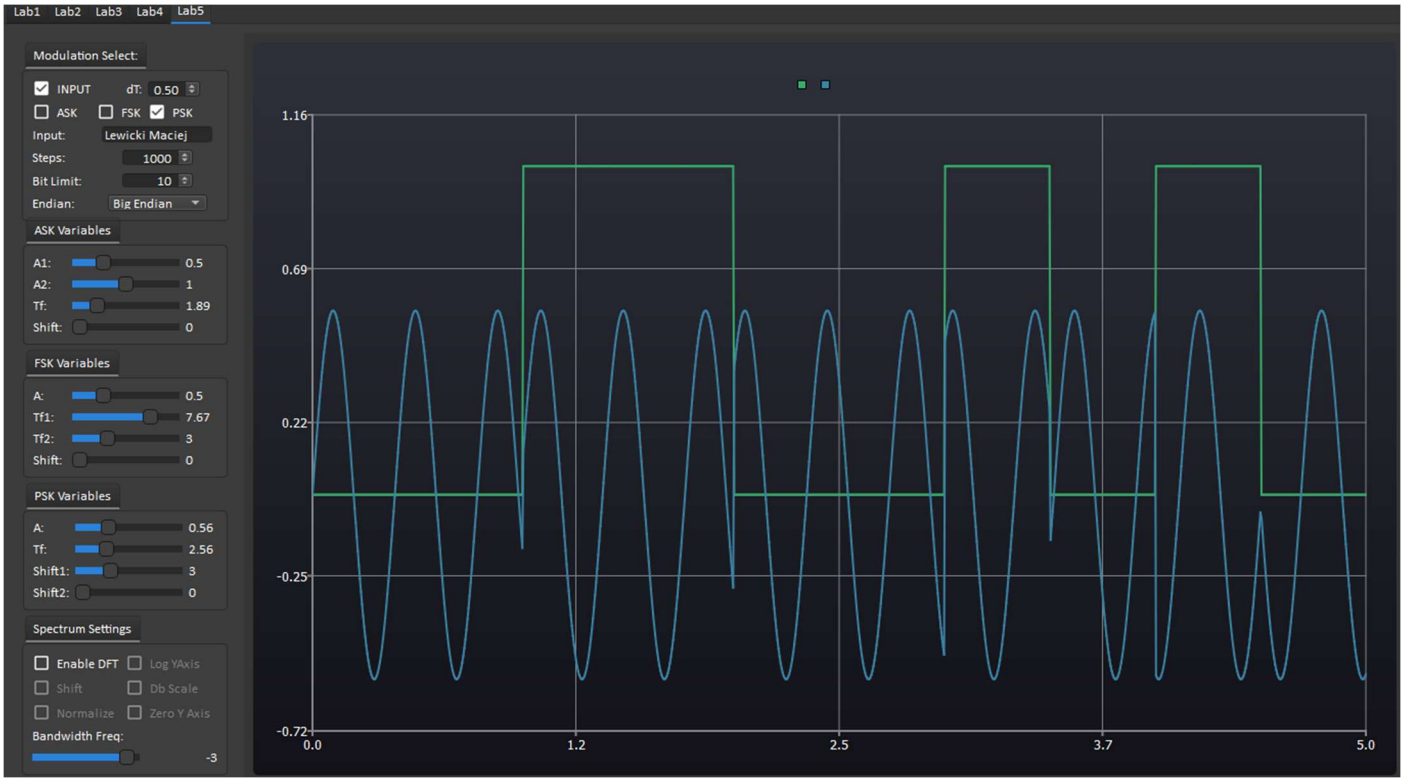
ZF(t):



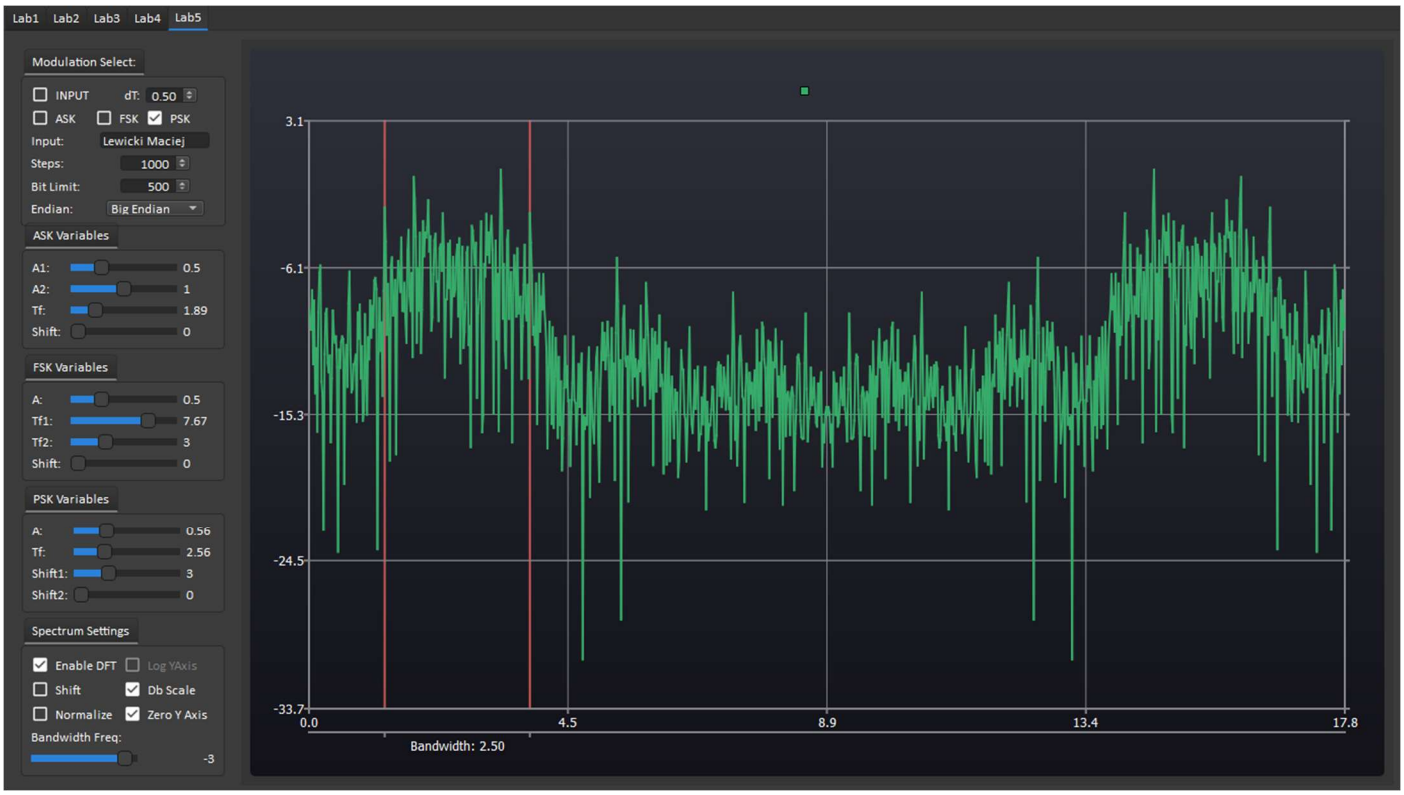
Widmo + szerokość pasma = 4.68[Hz]:



ZP(t):



Widmo + szerokość pasma = 2.50[Hz]:



Fragment kodu odpowiedzialny za modulacje ASK, FSK, PSK:

```
if(ask->checkState()==2)
{
    double y;
    QLineSeries* series = new QLineSeries(this);
    for(int i=0; i<stepsVal; i++)
    {
        if(vecY.at(i))
            y = (askAmp1_l)*(sin(2*M_PI*askTargetFreq_l*vecX.at(i)+askShift_l));
        else
            y = (askAmp2_l)*(sin(2*M_PI*askTargetFreq_l*vecX.at(i)+askShift_l));
        series->append(vecX.at(i), y);
    }
    chartView->chart()->addSeries(series);
}
if(fsk->checkState()==2)
{
    double y;
    QLineSeries* series = new QLineSeries(this);
    for(int i=0; i<stepsVal; i++)
    {
        if(vecY.at(i))
            y = (fskAmp_l)*(sin(2*M_PI*fskTargetFreq1_l*vecX.at(i)+fskShift_l));
        else
            y = (fskAmp_l)*(sin(2*M_PI*fskTargetFreq2_l*vecX.at(i)+fskShift_l));
        series->append(vecX.at(i), y);
    }
    chartView->chart()->addSeries(series);
}
if(psk->checkState()==2)
{
    double y;
    QLineSeries* series = new QLineSeries(this);
    for(int i=0; i<stepsVal; i++)
    {
        if(vecY.at(i))
            y = (pskAmp_l)*(sin(2*M_PI*pskTargetFreq_l*vecX.at(i)+pskShift1_l));
        else
            y = (pskAmp_l)*(sin(2*M_PI*pskTargetFreq_l*vecX.at(i)+pskShift2_l));
        series->append(vecX.at(i), y);
    }
    chartView->chart()->addSeries(series);
}
```