Laboratorium nr 9

1) Wybierając użyteczne fragmenty kodu pisane podczas wcześniejszych zajęć laboratoryjnych zaimplementuj prosty pseudo tor transmisyjny:

```
LabSeries clkSignal = genCLK(clkVal, rangeF, rangeT, stepsVal);
QBitArray bits = bitsFromString(inputVal, endianVal);

if(hamming->checkState()==2)
    bits = encodeHamming_4bit(bits, errors->value());

bits.resize(bitLimitVal);
decodeOutput->clear();

if(clk->checkState()==2)
    labSeries.append(clkSignal);
if(ttl->checkState()==2)
{
    auto mTTL = modTTL(clkSignal, bits);
    labSeries.append(mTTL);
    if(decode->checkState()==2)
    {
        if(hamming->checkState()==2)
            decodeOutput->appendPlainText("TTL :  "+stringFromBits(decHamming_4bit(decTTL(clkVal, mTTL)), endianVal)+"\n");
        else
            decodeOutput->appendPlainText("TTL :  "+stringFromBits(decTTL(clkVal, mTTL), endianVal)+"\n");

    }
}
if(man->checkState()==2)
{
    auto mMan = modManchester(clkSignal, bits);
    labSeries.append(mMan);
    if(decode->checkState()==2)
    {
        if(hamming->checkState()==2)
            decodeOutput->appendPlainText("Manchester :  "+stringFromBits(decHamming_4bit(decManchester(clkVal, mMan)), endianVal)+"\n");
        else
            decodeOutput->appendPlainText("Manchester :  "+stringFromBits(decManchester(clkVal, mMan), endianVal)+"\n");
    }
}
if(nrzi->checkState()==2)
{
    auto mNRZI = modNRZI(clkSignal, bits);
    labSeries.append(mNRZI);
    if(decode->checkState()==2)
    {
        if(hamming->checkState()==2)
            decodeOutput->appendPlainText("NRZI :  "+stringFromBits(decHamming_4bit(decNRZI(clkVal, mNRZI)), endianVal)+"\n");
        else
            decodeOutput->appendPlainText("NRZI :  "+stringFromBits(decNRZI(clkVal, mNRZI), endianVal)+"\n");
    }
}
if(bami->checkState()==2)
{
    auto mBAMI = modBAMI(clkSignal, bits);
    labSeries.append(mBAMI);
    if(decode->checkState()==2)
    {
        if(hamming->checkState()==2)
            decodeOutput->appendPlainText("BAMI :  "+stringFromBits(decHamming_4bit(decBAMI(clkVal, mBAMI)), endianVal)+"\n");
        else
            decodeOutput->appendPlainText("BAMI :  "+stringFromBits(decBAMI(clkVal, mBAMI), endianVal)+"\n");
    }
}
```

Na etapach kolejno oznaczonych numerami wygeneruj:

1. dane wejściowe
2. dane zabezpieczone kodem kanałowym
3. wykresy trzech modulacji
4. zdemodulowane dane zabezpieczone kodem kanałowym
5. zdekodowany strumień binarny

1. Dane wejściowe:

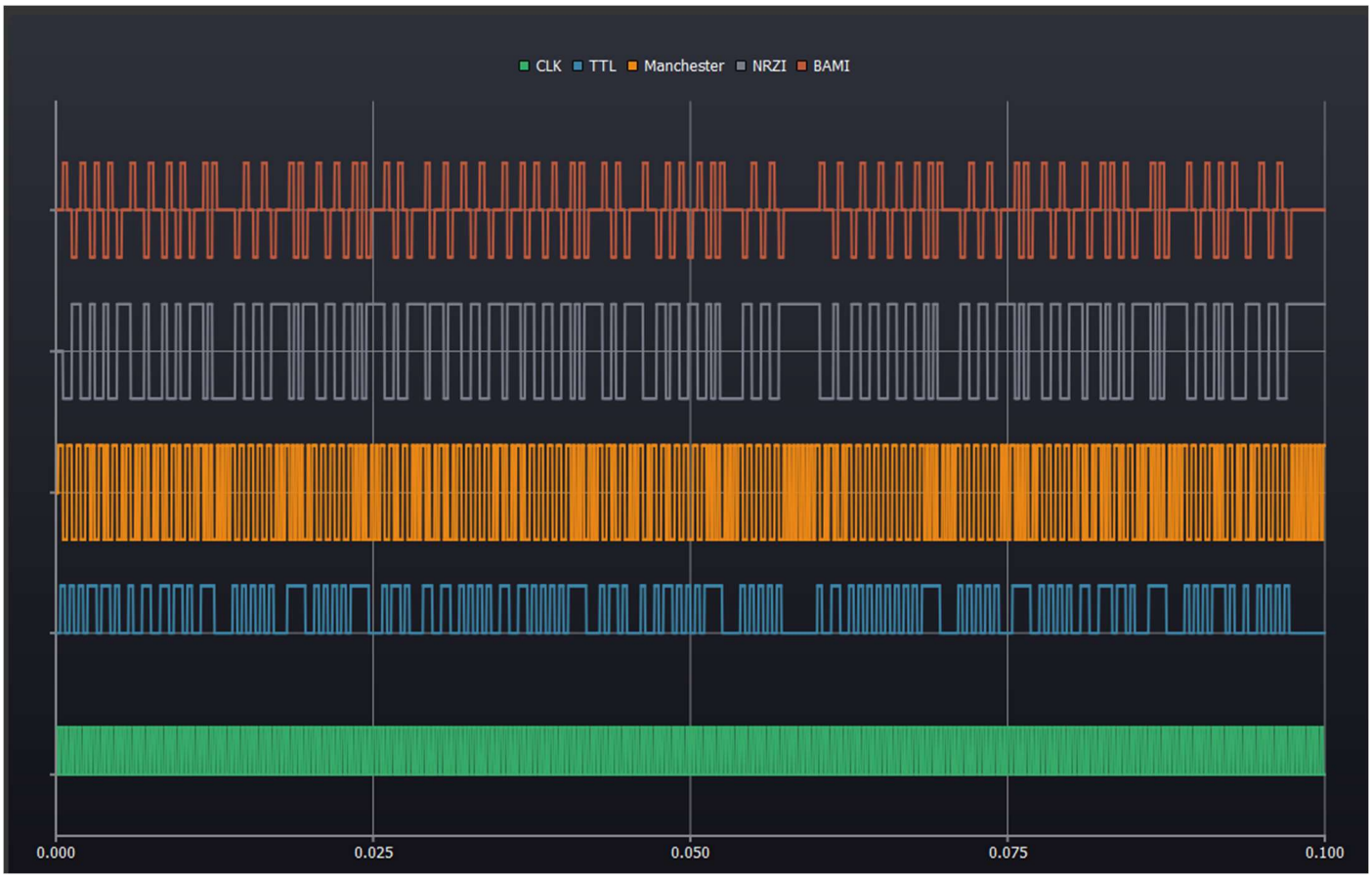
INPUT: „TRANSMISJA DANYCH”

```
INPUT:  QBitArray(0010 1010 0100 1010 1000 0010 0111 0010 1100 1010 1011 0010 1001 0010 1100
1010 0101 0010 1000 0010 0000 0100 0010 0010 1000 0010 0111 0010 1001 1010 1100 0010 0001 0010)
```

2. Dane zabezpieczone kodem kanałowym:

```
Encoded Data with 0 error(s) per byte:  QBitArray(0101 0101 1011 0100 1001 1001 1011 0100 1110
0001 0101 0101 0001 1110 0101 0101 0111 1000 1011 0100 0110 0110 0101 0101 0011 0011 0101 0101
0111 1000 1011 0100 0100 1011 0101 0101 1110 0001 0101 0101 0000 0000 1001 1001 0101 0101 0101
0101 1110 0001 0101 0101 0001 1110 0101 0101 0011 0011 1011 0100 0111 1000 0101 0101 1101 0010
0101 0101)
```

3. Wykresy trzech modulacji:



4. Zdemodulowane dane:

```
Encoded Data:  QBitArray(0101 0101 1011 0100 1001 1001 1011 0100 1110 0001 0101 0101 0001 1110
0101 0101 0111 1000 1011 0100 0110 0110 0101 0101 0011 0011 0101 0101 0111 1000 1011 0100 0100
1011 0101 0101 1110 0001 0101 0101 0000 0000 1001 1001 0101 0101 0101 0101 1110 0001 0101 0101
0001 1110 0101 0101 0011 0011 1011 0100 0111 1000 0101 0101 1101 0010 0101 0101 0000 0000 0000
0000 0000 0000 0000 0000)
```

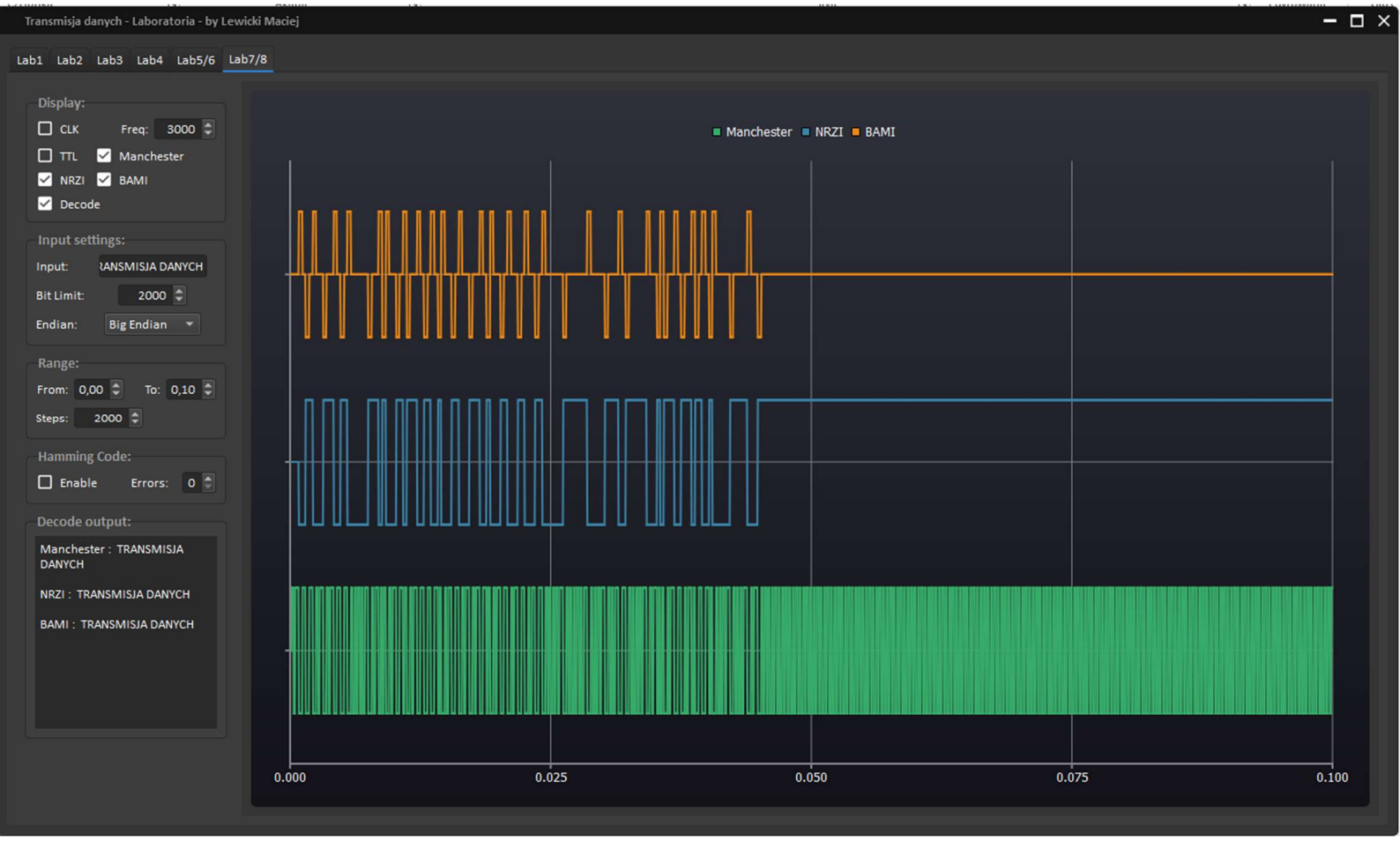
5. Zdekodowany strumień binarny:

```
Decoded Data:  QBitArray(0010 1010 0100 1010 1000 0010 0111 0010 1100 1010 1011 0010 1001 0010
1100 1010 0101 0010 1000 0010 0000 0100 0010 0010 1000 0010 0111 0010 1001 1010 1100 0010 0001
0010 0000 0000 0000 00)
```

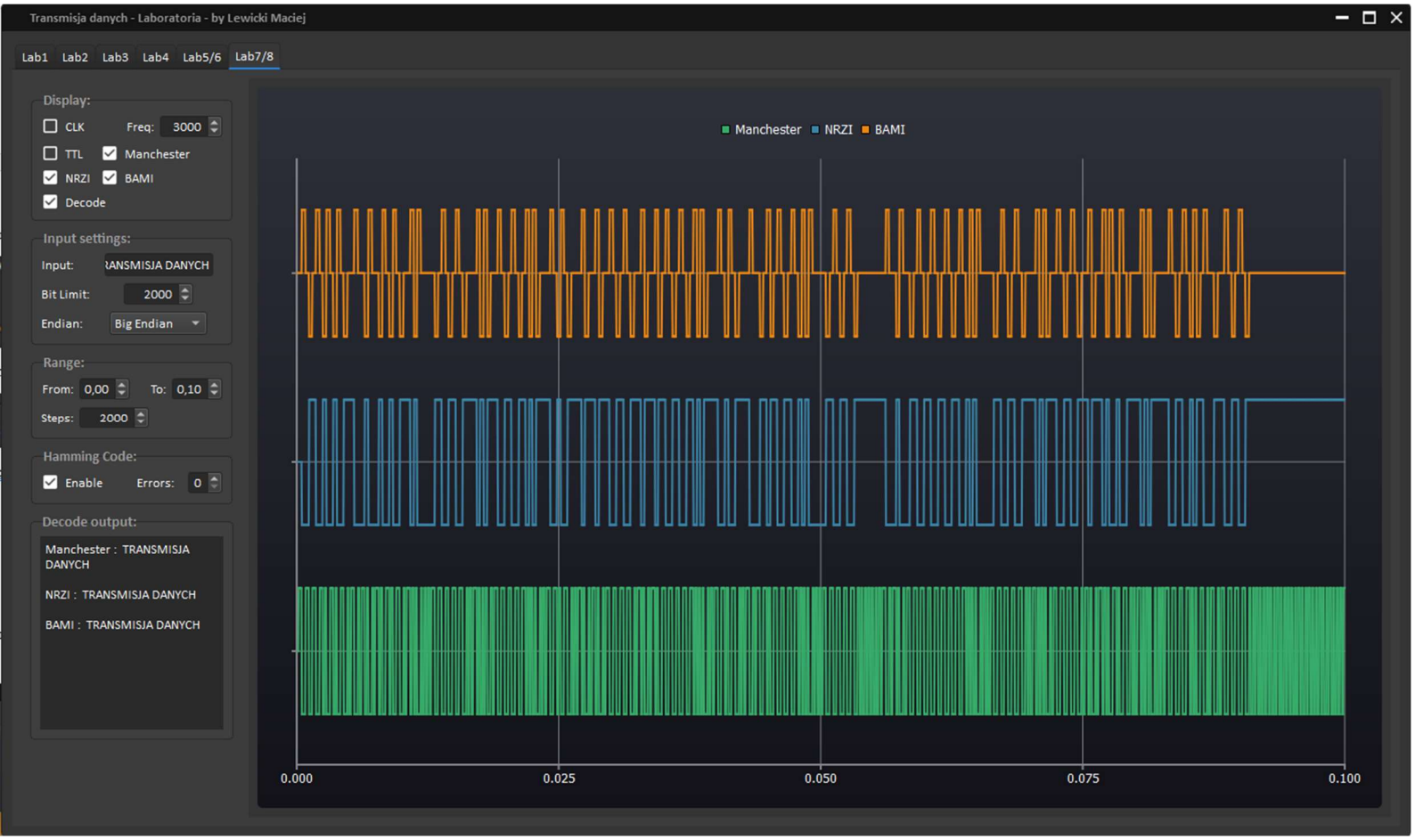
(dodatkowe zera na końcu bo sygnał jest dłuższy niż ilość bitów)

Dodatkowe SS działania programu:

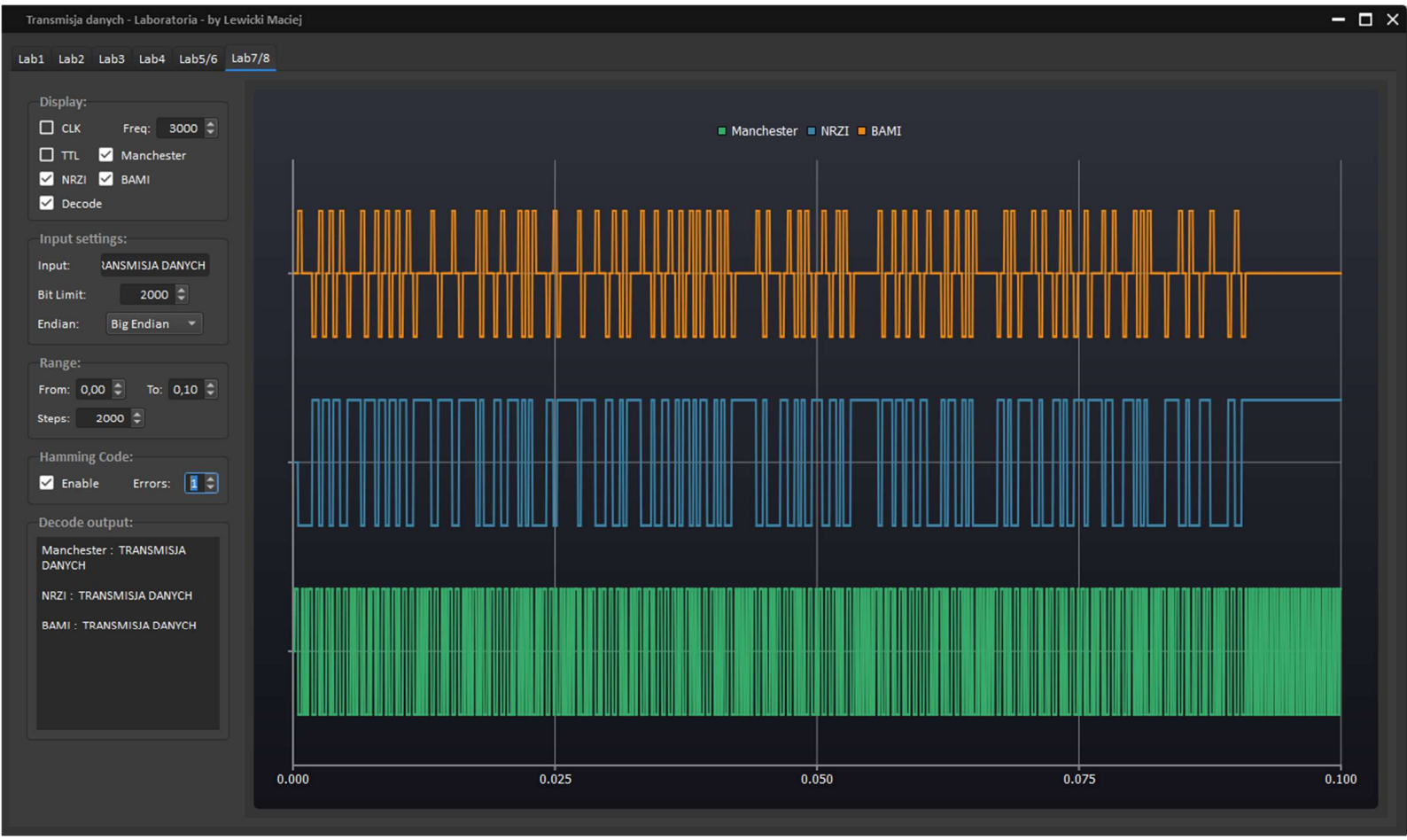
-Modulacja bez kodowania Hamminga



-Modulacja z kodowaniem Hamminga:



-Kodowanie z jednym błędem:



-Kodowanie z 2 błędami (output jest pusty bo każdy bajt został odrzucony przez dekoder):