Reading xrf pdz files with help of Luc

2023-11-09

Birgit asked if I could help with the data analysis of the ink measurements at Ets Haim (2020-014) with the new XRF spectrometer.

2020-07-05

Luc's translated python code

```
from array import *

import struct
class XRFSpectrum:
    datapoints = array('l')
    energyPerChannel = 0
    def __init__(self):
        print("this is the constructor of XRFSpectrum")

def readByte(pdzfile):
    return struct.unpack("B",pdzfile.read(1))[0]

def readShort(pdzfile):
    return struct.unpack("<h",pdzfile.read(2))[0]

def readInt(pdzfile):
    return struct.unpack("<i",pdzfile.read(4))[0]

def read32bitFloat(pdzfile):
    return struct.unpack("<f",pdzfile.read(4))[0]

def readString(pdzfile):
    length = readInt(pdzfile)
    print("read string of length "+str(length))
    return pdzfile.read(length*2).decode("utf16")

def readSpectrumParameters(pdzfile,spectrum):
    print("spectrum parameters:")
    print(readInt(pdzfile))
    print(read32bitFloat(pdzfile))
    print(readInt(pdzfile))
    print(readInt(pdzfile))
    print(read32bitFloat(pdzfile))
    print(read32bitFloat(pdzfile))
    print("real time: "+str(read32bitFloat(pdzfile)))
    print("live time: "+str(read32bitFloat(pdzfile)))
    print(str(read32bitFloat(pdzfile)))
    print(read32bitFloat(pdzfile))
    print(read32bitFloat(pdzfile))
    print("tube voltage: "+str(read32bitFloat(pdzfile)))
    print("tube current: "+str(read32bitFloat(pdzfile)))
    print(read32bitFloat(pdzfile))
    print(read32bitFloat(pdzfile))
    print(read32bitFloat(pdzfile))
    print(read32bitFloat(pdzfile))
    print(read32bitFloat(pdzfile))
    print(readShort(pdzfile))
    print(read32bitFloat(pdzfile))
    spectrum.energyPerChannel = read32bitFloat(pdzfile)
    print("energy per channel (eV): "+str(spectrum.energyPerChannel))
    print(readInt(pdzfile))
    print("abscissa: "+str(read32bitFloat(pdzfile)))
    print(read32bitFloat(pdzfile))
    print(read32bitFloat(pdzfile))
    print(read32bitFloat(pdzfile))
    print(read32bitFloat(pdzfile))
    print(read32bitFloat(pdzfile))
    print(read32bitFloat(pdzfile))
    print("spectrum name: "+readString(pdzfile))
#    depending on measurement mode treat spectrum name differently
    print(readShort(pdzfile))
    print("finished reading spectrum parameters")
```

```
def readSpectrumArray(pdzfile,spectrum):
    for i in range(2048):
        spectrum.datapoints.append(readInt(pdzfile))
    print("finished reading datapoints")

def main(pdzfile_name):
    spectrum = XRFSpectrum()
    spectrum2 = XRFSpectrum()
#    pdzfile = open("/home/luc/Documents/00020-Mudrock Dual.pdz","rb")
    #pdzfile = open("/home/luc/Documents/02959-Mudrock Dual.pdz","rb")
    pdzfile = open(pdzfile_name, "rb")
    version = readShort(pdzfile)
    print("version: ")
    print(version)
    if version != 25:
        print("wrong pdz version")
        exit()
    print(readInt(pdzfile))
    print(pdzfile.read(10).decode("utf16"))
    print(readInt(pdzfile))
    print(readShort(pdzfile))
    print(readInt(pdzfile))
    instrumentID = readString(pdzfile)
    print("instrument id: "+instrumentID)
    print(readString(pdzfile))
    print(pdzfile.read(6))
    print("detector type: "+readString(pdzfile))
    print(readString(pdzfile)+": "+str(readShort(pdzfile)))
    print(readString(pdzfile))
    listLength = readInt(pdzfile)
    for i in range(listLength):
        print(str(readShort(pdzfile))+": "+readString(pdzfile))
    print(readShort(pdzfile))
    print(readInt(pdzfile))
    print(readInt(pdzfile))
    print(readInt(pdzfile))
    print(readInt(pdzfile))
    print(readInt(pdzfile))
    print(readInt(pdzfile))

    print(read32bitFloat(pdzfile))
    print(read32bitFloat(pdzfile))
    print(read32bitFloat(pdzfile))
    print(read32bitFloat(pdzfile))
    print(read32bitFloat(pdzfile))
    print(read32bitFloat(pdzfile))

    measurementMode = readString(pdzfile)
    if measurementMode == "Artax":
        print("Artax version: "+readString(pdzfile))
    else:
        print("measurement mode: "+measurementMode+" "+str(readInt(pdzfile)))
    print("username: "+readString(pdzfile))
    print("some short: "+str(readShort(pdzfile)))
    readSpectrumParameters(pdzfile,spectrum)
#    here starts the array of 2048 32 bit integers (little endian)
    readSpectrumArray(pdzfile,spectrum)
    if measurementMode == "Artax":
        print("ready")
        exit()
    if readShort(pdzfile) == 3:
        readSpectrumParameters(pdzfile,spectrum2)
        readSpectrumArray(pdzfile,spectrum2)
        print(readShort(pdzfile))
        print(readInt(pdzfile))
        print(readInt(pdzfile))
        print(readInt(pdzfile))
        print(readInt(pdzfile))
        print(readShort(pdzfile))
```

```
cd /home/frank/Work/Projecten/DoRe/notebooks/.2020-06-30-reading-xrf-pdz-files-with-help-o
```

```
/mnt/datadisk/Work/Projecten/DoRe/notebooks/.2020-06-30-reading-xrf-pdz-files-with-help-of-lu
```

```
!ls
```

```
'00886-Spectrometer Mode.pdz'           PDZReader_py.txt
'01-lichte huid kin onder lip rechts.pdz'  'XRF measurments'
'02839-Mudrock Dual.pdz'           'XRF measurments.zip'
 PdzReader.java
```

```python
pdzfile_name = '02839-Mudrock Dual.pdz'
```

```python
main(pdzfile_name)
```

```
this is the constructor of XRFSpectrum
this is the constructor of XRFSpectrum
version:
25
14
pdz25
1
1
218
read string of length 8
instrument id: 900F4767
read string of length 8
SK5-4767
b'---A}\x00'
read string of length 3
detector type: SDD
read string of length 4
RxBx: 2056
read string of length 7
Movable
read string of length 10
1: 2.3.48.267
read string of length 5
2: 13.09
read string of length 4
3: 6.03
read string of length 4
4: 3.03
read string of length 4
5: 9.2F
read string of length 4
6: 1.02
read string of length 4
7: 1.11
read string of length 4
8: 1.01
2
```

```
88
1
131256
78756
0
0
0.937000036239624
0.6650000810623169
0.27400001883506775
0.2710000276565552
0.39100003242492676
1.0
read string of length 12
measurement mode: Mudrock Dual 0
read string of length 4
username: User
some short: 3
spectrum parameters:
8338
0.0
131256
78756
0.0
0.0
real time: 1.0
live time: 0.6650000810623169
0.27400001883506775
0.2710000276565552
0.39100003242492676
tube voltage: 40.0
tube current: 22.299999237060547
2.2959182325159954e-39
3.159144711210905e-38
0.0
1.401298464324817e-45
2.7361061089735655e+23
17103
4.105804500471714e-43
energy per channel (eV): 20.02522850036621
-644022270
abscissa: 3.4192701616546965e-34
3.673503924227508e-40
1.1020637889544313e-39
```

```
4.867281296373534e-39
-2.0
3.859912025077604e-34
5.044674471569341e-44
read string of length 15
spectrum name: MudRock Trace40
1
finished reading spectrum parameters
finished reading datapoints
```

```
#        listlength2 = readInt(pdzfile)
#        for i in range(listlength2):
#            print(str(i)+": "+str(readInt(pdzfile))+", "+str(readInt(pdzfile)))
main()
```

2020-06-30

Just biting of the tail

```
import struct
import numpy as np
import os
import matplotlib.pyplot as plt
import glob

plt.ioff()
```

```
%matplotlib notebook
```

```
def unpack(pdz_file, fmt, offset=0):
    '''Bite of some bytes'''

    nbytes = struct.calcsize(fmt)

    with open(pdz_file, 'rb') as fh:

        fh.read(offset)
        buffer = fh.read(nbytes)
    char_list = struct.unpack(fmt, buffer)

    return char_list


def get_spectrum(pdz_file, keV=40):
    '''Skip header and read last 2048 four-byte integers from pdz_file'''

    n_channels = 2048
    file_size = os.path.getsize(pdz_file)
    offset = file_size - 4 * n_channels
    fmt = 'i' * n_channels

    counts = np.array(unpack(pdz_file, fmt, offset=offset))
    channels = np.arange(len(counts))

    # no energies yet
    if keV is not None:
        channels = np.linspace(0, keV, n_channels)

    return channels, counts

def spectral_stackplot(pdz_file_list):

    spectra = [get_spectrum(pdz_file) for pdz_file in pdz_file_list]

    nrows = len(spectra)

    fig = plt.figure(figsize=[8, nrows * 1.5])

    #fig, axs = plt.subplots(nrows=nrows, ncols=1, figsize=[8, nrows * 1.5], squeeze=False
    axs = [fig.add_subplot(nrows, 1, i + 1) for i in range(nrows)]
    #plt.subplots_adjust(hspace=0.1, right=0.4)

    for i, [x, y] in enumerate(spectra):
        axs[i].plot(x, y, label=pdz_file_list[i])
        axs[i].text(0.98, 0.92, pdz_file_list[i], ha='right', va='top', transform=axs[i].t
        axs[i].set_xlim(0, 20)
    plt.tight_layout()

    return fig
```

2020-06-30

Trying the Ets Haim data WN2020-014

```
cd /home/frank/Work/Projecten/DoRe/notebooks/.2020-06-30-reading-xrf-pdz-files-with-help-o
```

```
/mnt/datadisk/Work/Projecten/DoRe/notebooks/.2020-06-30-reading-xrf-pdz-files-with-help-of-lu
```

```
pdz_file_list = glob.glob('*.pdz')
```

```
pdz_file_list
```

```
['2020-014-f265-13 40kv 6 mA original text .pdz',
 '2020-014-f263 verso-18 40kv 6 mA original text .pdz',
 '2020-014-f265-08 15kv 6 mA second scribe .pdz',
 '2020-014-f263 verso-25 40kv 6 mA new written text 2 .pdz',
 '2020-014-f265-07 40kv 6 mA second scribe .pdz',
 '2020-014-f265-09 40kv 6 mA second scribe .pdz',
 '2020-014-f265-10 40kv 6 mA second scribe 1st line .pdz',
 '2020-014-f265-11 40kv 6 mA second scribe 1st line .pdz',
 '2020-014 front blue.pdz',
 '2020-014 front yellow.pdz',
 '2020-014-f263 verso-22 40kv 6 mA blotting .pdz',
 '2020-014-f265-16 40kv 6 mA parchement 2 .pdz',
 '2020-014-f263 verso-21 40kv 6 mA blotting .pdz',
 '2020-014-f265-02 signature.pdz',
 '2020-014-f265-01 parchement.pdz',
 '2020-014 front red.pdz',
 '2020-014-f265-17 40kv 6 mA parchement 3 .pdz',
 '2020-014-f263 verso-27 40kv 6 mA perkament 2 .pdz',
 '2020-014-f263 verso-26 40kv 6 mA perkament .pdz',
 '2020-014-f263 verso-24 40kv 6 mA new written text .pdz',
 '2020-014-f265-12 40kv 6 mA original text .pdz',
 '2020-014-f265-15 40kv 6 mA parchement 1 .pdz',
 '2020-014-f263 verso-23 40kv 6 mA blotting edge paper .pdz',
 '2020-014-f263 verso-20 40kv 6 mA original text title 2 .pdz',
 '2020-014-f265-04 signature.pdz',
 '2020-014-f265-03 signature 15 kv.pdz',
 '2020-014 front blue 2.pdz',
 '2020-014-f265-06 second scribe .pdz',
 '2020-014-f263 verso-19 40kv 6 mA original text title .pdz',
 '2020-014-f265-14 40kv 6 mA original text .pdz',
 '2020-014-f265-05 signature.pdz']
```

```
fig = spectral_stackplot(pdz_file_list)
```

```
<IPython.core.display.Javascript object>
```

```
<IPython.core.display.HTML object>
```

```
fig.savefig('WN2020-014_Ets-Haim_XRF-spectra.pdf')
```

```
!pwd
```

```
/mnt/datadisk/Work/Projecten/DoRe/notebooks/.2020-06-30-reading-xrf-pdz-files-with-help-of-lu
```

2020-06-30

Trials with examples from Luc

Unfortunately, the new Bruker (?) instrument produces .pdz files with a new binary format, as explained by Luc:

> Van: "Megens, Luc" L.Megens@cultureelerfgoed.nl Datum: 29 juni 2020 om 21:08:43 CEST Aan: "Ligterink, Frank" F.Ligterink@cultureelerfgoed.nl Onderwerp: pdz hack
>
> Dag Frank,

> Bij deze de PdzReader en drie voorbeelden. In de Reader moet de functie readPdz25() hebben voor de huidige versie van pdz files.
>
> Java is Big endian, maar de Pdz files little endian, dus er zit wat conversie in die in Python geloof ik niet nodig is.
>
> Float 16 bit floating point Short 16 bit integer
>
> Int 32 bit integer
>
> Double 32 bit floating point
>
> Met vriendelijke groet,
>
> dr. L. (Luc) Megens Senior Onderzoeker

Not trivial. Here is some documentation, but no format specification. http://www.xrf.guru/WorkshopVI/Tracer

Trials

Ehm, can not see a clear pattern in the data files in the hex editor. Will do some trial and error using python. I did something comparable in: http://localhost:8888/notebooks/Work/Projecten/DoRe/noteboo
03-17-vsc-utils-dev.ipynb

See: https://docs.python.org/3/library/struct.html

```
cd .2020-06-30-reading-xrf-pdz-files-with-help-of-luc
```

```
/mnt/datadisk/Work/Projecten/DoRe/notebooks/.2020-06-30-reading-xrf-pdz-files-with-help-of-lu
```

```
!ls -l
```

```
total 60
-rw-rw-r-- 1 frank frank  9130 jun 30 09:41 '00886-Spectrometer Mode.pdz'
-rw-rw-r-- 1 frank frank  8660 jun 30 09:41 '01-lichte huid kin onder lip rechts.pdz'
-rw-rw-r-- 1 frank frank  9068 jun 30 09:41 '02839-Mudrock Dual.pdz'
-rw-rw-r-- 1 frank frank 22789 jun 30 09:41  PdzReader.java
```

```
import struct
import numpy as np
import os
import matplotlib.pyplot as plt
import glob

plt.ioff()
```

```
%matplotlib notebook
```

```
def unpack(pdz_file, fmt, offset=0):
    '''Bite of some bytes'''

    nbytes = struct.calcsize(fmt)

    with open(pdz_file, 'rb') as fh:

        fh.read(offset)
        buffer = fh.read(nbytes)

    char_list = struct.unpack(fmt, buffer)

    return char_list


def get_spectrum(pdz_file):
    '''Skip header and read last 2048 four-byte integers from pdz_file'''

    n_channels = 2048
    file_size = os.path.getsize(pdz_file)
    offset = file_size - 4 * n_channels
    fmt = 'i' * n_channels

    counts = np.array(unpack(pdz_file, fmt, offset=offset))

    # no energies yet
    channels = np.arange(len(counts))

    return channels, counts

def spectral_stackplot(pdz_file_list):

    spectra = [get_spectrum(pdz_file) for pdz_file in pdz_file_list]
    nrows = len(spectra)

    fig, axs = plt.subplots(nrows=nrows, ncols=1, figsize=[10, nrows * 1.2], squeeze=False

    for i, [x, y] in enumerate(spectra):
        axs[i, 0].plot(x, y, label=pdz_file_list[i])
        axs[i, 0].legend()

    return fig
```

```
pdz_file_list = glob.glob('*.pdz')
```

```
!ls -l *.pdz
```

```
-rw-rw-r-- 1 frank frank 9130 jun 30 09:41 '00886-Spectrometer Mode.pdz'
-rw-rw-r-- 1 frank frank 8660 jun 30 09:41 '01-lichte huid kin onder lip rechts.pdz'
-rw-rw-r-- 1 frank frank 9068 jun 30 09:41 '02839-Mudrock Dual.pdz'
```

```
pdz_file = '01-lichte huid kin onder lip rechts.pdz'
```

```
spectral_stackplot(pdz_file_list)
```

```
<IPython.core.display.Javascript object>

<IPython.core.display.HTML object>
```

```
y = np.array(unpack(pdz_file, 'i'*2048, offset=468))
x = np.arange(len(y))
```

```
plt.plot(x, y)
```

```
<IPython.core.display.Javascript object>

<IPython.core.display.HTML object>
```

```
str([x.decode('utf16') for x in unpack(pdz_file, '2s'*800)])
```

```
"['\\x19', '\\x0e', '\\x00', 'p', 'd', 'z', '2', '5', '\\x01', '\\x00', '\\x01', 'Þ', '\\x00
```

```
unpack(pdz_file, 'H'*800)
```

```
(25,
 14,
 0,
 112,
 100,
 122,
 50,
 53,
 1,
 0,
 1,
 222,
 0,
 8,
 0,
 57,
 48,
 48,
 70,
 52,
 55,
 54,
 55,
 8,
 0,
 83,
 75,
 53,
 45,
 52,
 55,
 54,
 55,
 11565,
 16685,
 125,
 3,
 0,
 83,
 68,
 68,
 4,
 0,
 82,
 120,
 66,
 120,
 0,
 7,
 0,
 77,
 111,
 118,
 97,
 98,
 108,
 101,
 8,
 0,
 1,
 9,
 0,
 56,
 46,
 48,
 46,
 48,
 46,
 52,
 55,
 54,
 2,
 6,
 0,
 50,
 48,
 48,
 46,
 51,
 57,
 3,
 4,
 0,
 54,
 46,
 48,
 51,
 4,
 4,
 0,
 51,
 46,
 48,
 51,
 5,
 4,
 0,
 57,
 46,
 50,
 70,
 6,
 6,
 0,
 50,
 48,
 48,
 46,
 51,
 57,
 7,
 4,
 0,
 49,
 46,
 49,
 49,
 8,
 4,
 0,
 49,
 46,
 48,
 49,
 2,
 92,
 0,
 1,
 0,
 3398,
 149,
 721,
 107,
 0,
 0,
 0,
 0,
 4461,
 17065,
 62273,
 17037,
 33552,
 16808,
 11530,
 16726,
 42208,
 16967,
 0,
 17076,
 5,
 0,
 65,
 114,
 116,
 97,
 120,
 9,
 0,
 56,
 46,
 48,
 46,
 48,
 46,
 52,
 55,
 54,
 4,
 0,
 116,
 101,
 115,
 116,
 3,
 8308,
 0,
 0,
 0,
 3398,
 149,
 721,
 107,
 0,
 0,
 0,
 0,
 0,
 17076,
 62273,
 17037,
 33552,
 16808,
 11530,
 16726,
 42208,
 16967,
 0,
 16928,
 0,
 16576,
 0,
 0,
 0,
 0,
 0,
 0,
 2,
 0,
 49624,
 0,
 17079,
 293,
 0,
 6436,
 16800,
 2,
 25973,
 16292,
 2019,
 2,
 4,
 21,
 10,
 37,
 11,
 525,
 24576,
 17536,
 2048,
 30,
 0,
 0,
 0,
 1,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 1,
 0,
 23,
 0,
 127,
 0,
 857,
 0,
 3373,
 0,
 8098,
 0,
 12677,
 0,
 12265,
 0,
 7370,
 0,
 2907,
 0,
 700,
 0,
 123,
 0,
 22,
 0,
 7,
 0,
 4,
 0,
 5,
 0,
 4,
 0,
 0,
 0,
 1,
 0,
 4,
 0,
 5,
 0,
 1119,
 0,
 1079,
 0,
 852,
 0,
 702,
 0,
 525,
 0,
 435,
 0,
 425,
 0,
 410,
 0,
 408,
 0,
 437,
 0,
 443,
 0,
 440,
 0,
 485,
 0,
 486,
 0,
 459,
 0,
 431,
 0,
 384,
 0,
 319,
 0,
 347,
 0,
 374,
 0,
 363,
 0,
 358,
 0,
 310,
 0,
 337,
 0,
 418,
 0,
 395,
 0,
 416,
 0,
 418,
 0,
 383,
 0,
 347,
 0,
 312,
 0,
 282,
 0,
 318,
 0,
 301,
 0,
 314,
 0,
 338,
 0,
 362,
 0,
 438,
 0,
 601,
 0,
 743,
 0,
 819,
 0,
 732,
 0,
 648,
 0,
 400,
 0,
 387,
 0,
 326,
 0,
 312,
 0,
 305,
 0,
 328,
 0,
 394,
 0,
 571,
 0,
 754,
 0,
 1038,
 0,
 1135,
 0,
 1356,
 0,
 1583,
 0,
 1774,
 0,
 1991,
 0,
 1868,
 0,
 1572,
 0,
 1109,
 0,
 913,
 0,
 817,
 0,
 882,
 0,
 1074,
 0,
 1180,
 0,
 1331,
 0,
 1271,
 0,
 1114,
 0,
 943,
 0,
 872,
 0,
 961,
 0,
 1107,
 0,
 1482,
 0,
 1936,
 0,
 2282,
 0,
 2813,
 0,
 3895,
 0,
 6778,
 0,
 12826,
 0,
 22913,
 0,
 35225,
 0,
 46549,
 0,
 53072,
 0,
 52709,
 0,
 48993,
 0,
 46516,
 0,
 44125,
 0,
 41332,
 0,
 34354,
 0,
 24991,
 0,
 15637,
 0,
 8684,
 0,
 4693,
 0,
 3125,
 0,
 3158,
 0,
 4036,
 0,
 5455,
 0,
 7023,
 0,
 8691,
 0,
 9169,
 0,
 8718,
 0,
 7093,
 0,
 5479,
 0,
 4302,
 0,
 3876,
 0,
 4109,
 0,
 4294,
 0,
 4693,
 0,
 4941,
 0,
 5133,
 0,
 5953,
 0,
 7182,
 0,
 7986,
 0,
 8086,
 0,
 7135,
 0,
 5345,
 0,
 3858,
 0,
 2681,
 0,
 2097,
 0,
 1840,
 0,
 1836,
 0,
 1869,
 0,
 1867,
 0,
 1903,
 0,
 1791,
 0,
 1754,
 0,
 1731,
 0,
 1944,
 0,
 2325,
 0,
 2734,
 0,
 3024,
 0,
 3119,
 0,
 2835,
 0,
 2286,
 0,
 1733,
 0,
 1382,
 0,
 1248,
 0,
 1145,
 0,
 1032,
 0,
 1008,
 0,
 964,
 0,
 954,
 0,
 1038,
 0,
 1265,
 0,
 1773,
 0,
 2536,
 0,
 3780,
 0,
 5414,
 0,
 6623,
 0,
 7441,
 0,
 7059,
 0,
 5856,
 0,
 4285,
 0,
 2801,
 0,
 1790,
 0,
 1201,
 0,
 797,
 0,
 712,
 0,
 697,
 0,
 682,
 0,
 756,
 0,
 844,
 0,
 1064,
 0,
 1143,
 0,
 1315,
 0,
 1481,
 0,
 1331,
 0,
 1327,
 0,
 1055,
 0,
 945,
 0,
 818,
 0,
 690,
 0,
 675,
 0,
 668,
 0,
 700,
 0,
 630,
 0,
 608,
 0,
 644,
 0,
 663,
 0,
 644,
 0,
 643,
 0,
 648,
 0,
 706,
 0,
 700,
 0,
 748,
 0,
 768,
 0,
 806,
 0,
 934,
 0,
 1047,
 0,
 1094,
 0,
 1117,
 0,
 1117,
 0,
 952,
 0,
 925,
 0,
 889,
 0,
 866,
 0,
 805,
 0,
 849,
 0,
 825,
 0,
 807,
 0,
 822,
 0,
 779,
 0,
 802,
 0,
 816,
 0,
 775,
 0,
 773,
 0,
 817,
 0,
 770,
 0,
 803,
 0,
 786,
 0,
 854,
 0,
 870,
 0,
 835,
 0,
 832,
 0,
 838,
 0,
 815,
 0,
 848,
 0,
 824,
 0,
 852,
 0,
 815,
 0,
 855,
 0,
 827,
 0,
 862,
 0,
 828,
 0,
 837,
 0,
 829,
 0,
 899,
 0,
 926,
 0,
 949,
 0,
 908,
 0,
 886,
 0,
 974,
 0,
 1032,
 0,
 992,
 0,
 990,
 0,
 949,
 0,
 1019,
 0,
 979,
 0,
 992,
 0,
 996,
 0,
 997,
 0,
 1020,
 0,
 1024,
 0,
 987,
 0,
 1071,
 0,
 1018,
 0,
 1076,
 0,
 1006,
 0)
```

```
fmt = 'hhh' # formt string is 3x short (little endian 2-byte integer)
hsize = struct.calcsize(fmt) # 6 bytes
hsize
```

```
6
```

```
with open(pdz_file, 'rb') as fh:

    hbytes = fh.read(hsize)

start, stop, step  = struct.unpack(fmt, hbytes)
start, stop, step
```

```
(25, 14, 0)
```