05ex_Pandas

March 14, 2020

0.1 Pandas analysis

In the following a series of exercises is proposed on a dataset containg timing information from a series of Time-to-Digital-Converters (TDC) implemented in a couple of FPGA's. Each measurement (i.e. each raw) consists of the address of the TDC providing the signal, 'FPGA' and 'TDC_Channel, and the timing information itself, 'ORBIT_CNT', 'BX_COUNTER' and 'TDC_MEAS'. Each TDC count correspond 25/30 ns, whereas the BX_COUNTER feauters gets updated every 25 ns and the ORBIT_CNT every 'x' BX_COUNTER. You can see these way of storing the time as similar to hours, minutes and seconds.

```
In [1]: import pandas as pd
        import numpy as np
```

1. Create a Pandas DataFrame by read N raws of the 'data_000637.txt' dataset. Choose N to be smaller than or equal to the maximum number of raws and larger that 10k.

```
In [2]: data = pd.DataFrame(pd.read_csv( 'data_000637.txt', sep=",", nrows=16000))
        data
```

```
Out[2]:         HEAD  FPGA  TDC_CHANNEL   ORBIT_CNT  BX_COUNTER  TDC_MEAS
        0          1     0          123  3869200167        2374        26
        1          1     0          124  3869200167        2374        27
        2          1     0           63  3869200167        2553        28
        3          1     0           64  3869200167        2558        19
        4          1     0           64  3869200167        2760        25
        5          1     0           63  3869200167        2762         4
        6          1     0           61  3869200167        2772        14
        7          1     0          139  3869200167        2776         0
        8          1     0           62  3869200167        2774        21
        9          1     0           60  3869200167        2788         7
        10         1     1            7  3869200167        2785         4
        11         1     0           64  3869200167        2786        19
        12         1     1            6  3869200167        2792        18
        13         1     0           36  3869200167        2791        23
        14         1     0           56  3869200167        2789         3
        15         1     1          139  3869200167        2797         0
        16         1     1            8  3869200167        2787        14
        17         1     0           63  3869200167        2790        10
        18         1     1            5  3869200167        2795         4
```

```
19        1    0          53  3869200167        2796        26
20        1    1          10  3869200167        2789        14
21        1    0          57  3869200167        2789        10
22        1    0          61  3869200167        2790        23
23        1    0          38  3869200167        2799        15
24        1    0          58  3869200167        2795        19
25        1    0          62  3869200167        2797        14
26        1    0          59  3869200167        2799        14
27        1    0          59  3869200167        3081        21
28        1    0          61  3869200167        3081         1
29        1    0          60  3869200167        3083        20
...     ...  ...         ...         ...         ...       ...
15970     1    0          64  3869200326         872        16
15971     1    0          58  3869200326         879        11
15972     1    1         128  3869200326         897         6
15973     1    1          62  3869200326         896        17
15974     1    0          63  3869200326         873         4
15975     1    1         128  3869200326         904        12
15976     1    1          64  3869200326         906        25
15977     1    0          40  3869200326         885         7
15978     1    1         139  3869200326         908         0
15979     1    1          59  3869200326         908         2
15980     1    1          61  3869200326         905        16
15981     1    1          38  3869200326        1117         1
15982     1    0          24  3869200326        1118         4
15983     1    0          59  3869200326        1119         4
15984     1    0          63  3869200326        1116        28
15985     1    0          60  3869200326        1124        29
15986     1    1          40  3869200326        1128        14
15987     1    0          61  3869200326        1119        27
15988     1    0          64  3869200326        1121        12
15989     1    1          61  3869200326        1263         1
15990     1    1          38  3869200326        1264        18
15991     1    1           4  3869200326        1271         8
15992     1    0          15  3869200326        1268         1
15993     1    0          16  3869200326        1274        13
15994     1    0          58  3869200326        1312         6
15995     1    1           6  3869200326        1311        27
15996     1    0         123  3869200326        1315        21
15997     1    0          63  3869200326        1313        17
15998     1    1           7  3869200326        1315        25
15999     1    0          54  3869200326        3101        11

[16000 rows x 6 columns]
```

2. Find out the value of ‘x’

```
In [3]: print( data.sort_values(by='BX_COUNTER') )
```

```
       HEAD  FPGA  TDC_CHANNEL   ORBIT_CNT  BX_COUNTER  TDC_MEAS
2894      1     0           46  3869200196           0         1
14400     1     0           62  3869200311           0        26
8000      1     0           63  3869200247           0        19
14404     1     1           13  3869200311           0        27
13797     1     1           16  3869200305           0        14
14398     1     1            6  3869200311           0        17
14397     1     0           59  3869200311           0        14
15515     1     0          122  3869200322           0         0
128       1     0           74  3869200169           1        18
2891      1     0          103  3869200196           1        24
4748      1     1            2  3869200215           1         2
13793     1     1          139  3869200305           2         0
14395     1     1          139  3869200311           2         0
14399     1     0          139  3869200311           2         0
14408     1     1           10  3869200311           3        19
4744      1     1          139  3869200215           3         0
10369     1     0           64  3869200271           4         2
4464      1     0          124  3869200212           4        26
5018      1     0           63  3869200218           4         7
4466      1     0          120  3869200212           5         8
14402     1     0          124  3869200311           5         1
10367     1     0           25  3869200271           5        26
4749      1     1           26  3869200215           5        26
4750      1     1            5  3869200215           6         4
10368     1     0           28  3869200271           6         7
2892      1     0           16  3869200196           6        25
12792     1     0           23  3869200295           7        26
44        1     0           64  3869200168           7        16
43        1     1            1  3869200168           7        18
12795     1     0           56  3869200295           7        14
...     ...   ...          ...         ...         ...       ...
15508     1     1           12  3869200321        3558        14
5912      1     1            1  3869200226        3558        22
14396     1     1            8  3869200310        3558        20
14609     1     1          139  3869200312        3558         0
2889      1     0          139  3869200195        3559         0
13799     1     0           62  3869200304        3559         6
13791     1     0          139  3869200304        3560         0
4072      1     1           80  3869200207        3561        17
13795     1     1           15  3869200304        3562        17
4746      1     1            5  3869200214        3562        18
14523     1     1           65  3869200311        3562         6
2503      1     0           47  3869200192        3562        18
14406     1     1            7  3869200310        3562        26
15529     1     0           59  3869200321        3562         1
4070      1     1           16  3869200207        3562         9
8001      1     0           64  3869200246        3563         1
```

```
4073     1   0         122  3869200207        3563          22
14394    1   0          61  3869200310        3563           2
15530    1   0          60  3869200321        3563           4
15528    1   0          57  3869200321        3563           7
14401    1   1          11  3869200310        3563          25
2893     1   0          45  3869200195        3563          29
4461     1   0         124  3869200211        3563           6
15517    1   1           7  3869200321        3563          22
15522    1   1           2  3869200321        3563          18
15518    1   0         118  3869200321        3563          15
4745     1   1          27  3869200214        3563           3
14403    1   0          60  3869200310        3563           9
4747     1   1          28  3869200214        3563          10
15511    1   1          16  3869200321        3563          23

[16000 rows x 6 columns]
```

```
In [4]: x=data['BX_COUNTER'].max()
        print(x)
```

```
3563
```

3. Find out how much the data taking lasted. You can either make an estimate on the baseis of the fraction of the measurements (raws) you read, or perform this check precisely by reading out the whole dataset

```
In [6]: full_data=pd.read_csv('data_000637.txt', sep=",")
        print((full_data['ORBIT_CNT'].max()-full_data['ORBIT_CNT'].min())*x*25/1e9, 's')
        #the time is the max-min of orbit, each one updated every x BX_counter which is update
```

```
0.9801813 s
```

4. Create a new column with the actual time in ns (as a combination of the other three columns with timing information)

```
In [7]: data['time']=data['ORBIT_CNT']*data['BX_COUNTER']*25+data['TDC_MEAS']*25/30
        #to start count from 0 ns
        #data['time']=(data['ORBIT_CNT']-data['ORBIT_CNT'].min())*data['BX_COUNTER']*25+data['
        data
```

```
Out[7]:        HEAD  FPGA  TDC_CHANNEL   ORBIT_CNT  BX_COUNTER  TDC_MEAS          time
        0         1     0          123  3869200167        2374        26  2.296370e+14
        1         1     0          124  3869200167        2374        27  2.296370e+14
        2         1     0           63  3869200167        2553        28  2.469517e+14
        3         1     0           64  3869200167        2558        19  2.474354e+14
        4         1     0           64  3869200167        2760        25  2.669748e+14
```

5	1	0	63	3869200167	2762	4	2.671683e+14
6	1	0	61	3869200167	2772	14	2.681356e+14
7	1	0	139	3869200167	2776	0	2.685225e+14
8	1	0	62	3869200167	2774	21	2.683290e+14
9	1	0	60	3869200167	2788	7	2.696833e+14
10	1	1	7	3869200167	2785	4	2.693931e+14
11	1	0	64	3869200167	2786	19	2.694898e+14
12	1	1	6	3869200167	2792	18	2.700702e+14
13	1	0	36	3869200167	2791	23	2.699734e+14
14	1	0	56	3869200167	2789	3	2.697800e+14
15	1	1	139	3869200167	2797	0	2.705538e+14
16	1	1	8	3869200167	2787	14	2.695865e+14
17	1	0	63	3869200167	2790	10	2.698767e+14
18	1	1	5	3869200167	2795	4	2.703604e+14
19	1	0	53	3869200167	2796	26	2.704571e+14
20	1	1	10	3869200167	2789	14	2.697800e+14
21	1	0	57	3869200167	2789	10	2.697800e+14
22	1	0	61	3869200167	2790	23	2.698767e+14
23	1	0	38	3869200167	2799	15	2.707473e+14
24	1	0	58	3869200167	2795	19	2.703604e+14
25	1	0	62	3869200167	2797	14	2.705538e+14
26	1	0	59	3869200167	2799	14	2.707473e+14
27	1	0	59	3869200167	3081	21	2.980251e+14
28	1	0	61	3869200167	3081	1	2.980251e+14
29	1	0	60	3869200167	3083	20	2.982186e+14
...	...	...	...	...	...	...	...
15970	1	0	64	3869200326	872	16	8.434857e+13
15971	1	0	58	3869200326	879	11	8.502568e+13
15972	1	1	128	3869200326	897	6	8.676682e+13
15973	1	1	62	3869200326	896	17	8.667009e+13
15974	1	0	63	3869200326	873	4	8.444530e+13
15975	1	1	128	3869200326	904	12	8.744393e+13
15976	1	1	64	3869200326	906	25	8.763739e+13
15977	1	0	40	3869200326	885	7	8.560606e+13
15978	1	1	139	3869200326	908	0	8.783085e+13
15979	1	1	59	3869200326	908	2	8.783085e+13
15980	1	1	61	3869200326	905	16	8.754066e+13
15981	1	1	38	3869200326	1117	1	1.080474e+14
15982	1	0	24	3869200326	1118	4	1.081441e+14
15983	1	0	59	3869200326	1119	4	1.082409e+14
15984	1	0	63	3869200326	1116	28	1.079507e+14
15985	1	0	60	3869200326	1124	29	1.087245e+14
15986	1	1	40	3869200326	1128	14	1.091114e+14
15987	1	0	61	3869200326	1119	27	1.082409e+14
15988	1	0	64	3869200326	1121	12	1.084343e+14
15989	1	1	61	3869200326	1263	1	1.221700e+14
15990	1	1	38	3869200326	1264	18	1.222667e+14
15991	1	1	4	3869200326	1271	8	1.229438e+14

```
15992     1     0           15  3869200326        1268         1  1.226537e+14
15993     1     0           16  3869200326        1274        13  1.232340e+14
15994     1     0           58  3869200326        1312         6  1.269098e+14
15995     1     1            6  3869200326        1311        27  1.268130e+14
15996     1     0          123  3869200326        1315        21  1.272000e+14
15997     1     0           63  3869200326        1313        17  1.270065e+14
15998     1     1            7  3869200326        1315        25  1.272000e+14
15999     1     0           54  3869200326        3101        11  2.999598e+14

[16000 rows x 7 columns]
```

5. Replace the values (all 1) of the HEAD column randomly with 0 or 1

```
In [8]: data['HEAD']=np.random.randint(2, size=data['HEAD'].count())
        data
```

```
Out[8]:        HEAD  FPGA  TDC_CHANNEL   ORBIT_CNT  BX_COUNTER  TDC_MEAS          time
        0         0     0          123  3869200167        2374        26  2.296370e+14
        1         0     0          124  3869200167        2374        27  2.296370e+14
        2         0     0           63  3869200167        2553        28  2.469517e+14
        3         0     0           64  3869200167        2558        19  2.474354e+14
        4         1     0           64  3869200167        2760        25  2.669748e+14
        5         0     0           63  3869200167        2762         4  2.671683e+14
        6         0     0           61  3869200167        2772        14  2.681356e+14
        7         0     0          139  3869200167        2776         0  2.685225e+14
        8         0     0           62  3869200167        2774        21  2.683290e+14
        9         0     0           60  3869200167        2788         7  2.696833e+14
        10        1     1            7  3869200167        2785         4  2.693931e+14
        11        1     0           64  3869200167        2786        19  2.694898e+14
        12        0     1            6  3869200167        2792        18  2.700702e+14
        13        1     0           36  3869200167        2791        23  2.699734e+14
        14        1     0           56  3869200167        2789         3  2.697800e+14
        15        1     1          139  3869200167        2797         0  2.705538e+14
        16        0     1            8  3869200167        2787        14  2.695865e+14
        17        0     0           63  3869200167        2790        10  2.698767e+14
        18        1     1            5  3869200167        2795         4  2.703604e+14
        19        1     0           53  3869200167        2796        26  2.704571e+14
        20        0     1           10  3869200167        2789        14  2.697800e+14
        21        1     0           57  3869200167        2789        10  2.697800e+14
        22        1     0           61  3869200167        2790        23  2.698767e+14
        23        1     0           38  3869200167        2799        15  2.707473e+14
        24        1     0           58  3869200167        2795        19  2.703604e+14
        25        1     0           62  3869200167        2797        14  2.705538e+14
        26        1     0           59  3869200167        2799        14  2.707473e+14
        27        1     0           59  3869200167        3081        21  2.980251e+14
        28        1     0           61  3869200167        3081         1  2.980251e+14
        29        0     0           60  3869200167        3083        20  2.982186e+14
        ...     ...   ...          ...         ...         ...       ...           ...
```

```
15970     0     0           64  3869200326         872        16  8.434857e+13
15971     0     0           58  3869200326         879        11  8.502568e+13
15972     0     1          128  3869200326         897         6  8.676682e+13
15973     1     1           62  3869200326         896        17  8.667009e+13
15974     0     0           63  3869200326         873         4  8.444530e+13
15975     0     1          128  3869200326         904        12  8.744393e+13
15976     0     1           64  3869200326         906        25  8.763739e+13
15977     1     0           40  3869200326         885         7  8.560606e+13
15978     0     1          139  3869200326         908         0  8.783085e+13
15979     1     1           59  3869200326         908         2  8.783085e+13
15980     1     1           61  3869200326         905        16  8.754066e+13
15981     1     1           38  3869200326        1117         1  1.080474e+14
15982     1     0           24  3869200326        1118         4  1.081441e+14
15983     1     0           59  3869200326        1119         4  1.082409e+14
15984     0     0           63  3869200326        1116        28  1.079507e+14
15985     1     0           60  3869200326        1124        29  1.087245e+14
15986     0     1           40  3869200326        1128        14  1.091114e+14
15987     0     0           61  3869200326        1119        27  1.082409e+14
15988     1     0           64  3869200326        1121        12  1.084343e+14
15989     1     1           61  3869200326        1263         1  1.221700e+14
15990     1     1           38  3869200326        1264        18  1.222667e+14
15991     0     1            4  3869200326        1271         8  1.229438e+14
15992     0     0           15  3869200326        1268         1  1.226537e+14
15993     0     0           16  3869200326        1274        13  1.232340e+14
15994     0     0           58  3869200326        1312         6  1.269098e+14
15995     0     1            6  3869200326        1311        27  1.268130e+14
15996     0     0          123  3869200326        1315        21  1.272000e+14
15997     0     0           63  3869200326        1313        17  1.270065e+14
15998     0     1            7  3869200326        1315        25  1.272000e+14
15999     0     0           54  3869200326        3101        11  2.999598e+14

[16000 rows x 7 columns]
```

6. Create a new DataFrame with only the raws with HEAD=1

```
In [9]: head1=data[data['HEAD']==1].copy()
        head1
```

```
Out[9]:        HEAD  FPGA  TDC_CHANNEL   ORBIT_CNT  BX_COUNTER  TDC_MEAS          time
        4         1     0           64  3869200167        2760        25  2.669748e+14
        10        1     1            7  3869200167        2785         4  2.693931e+14
        11        1     0           64  3869200167        2786        19  2.694898e+14
        13        1     0           36  3869200167        2791        23  2.699734e+14
        14        1     0           56  3869200167        2789         3  2.697800e+14
        15        1     1          139  3869200167        2797         0  2.705538e+14
        18        1     1            5  3869200167        2795         4  2.703604e+14
        19        1     0           53  3869200167        2796        26  2.704571e+14
        21        1     0           57  3869200167        2789        10  2.697800e+14
```

22	1	0	61	3869200167	2790	23	2.698767e+14
23	1	0	38	3869200167	2799	15	2.707473e+14
24	1	0	58	3869200167	2795	19	2.703604e+14
25	1	0	62	3869200167	2797	14	2.705538e+14
26	1	0	59	3869200167	2799	14	2.707473e+14
27	1	0	59	3869200167	3081	21	2.980251e+14
28	1	0	61	3869200167	3081	1	2.980251e+14
30	1	0	139	3869200167	3085	0	2.984121e+14
31	1	0	62	3869200167	3079	4	2.978317e+14
33	1	0	54	3869200167	3134	4	3.031518e+14
34	1	0	53	3869200167	3136	25	3.033453e+14
36	1	0	59	3869200167	3176	8	3.072145e+14
37	1	0	53	3869200167	3181	8	3.076981e+14
38	1	0	60	3869200167	3175	17	3.071178e+14
46	1	0	63	3869200168	8	28	7.738400e+11
48	1	0	139	3869200168	23	0	2.224790e+12
49	1	0	61	3869200168	18	17	1.741140e+12
55	1	1	59	3869200168	270	21	2.611710e+13
56	1	1	139	3869200168	272	0	2.631056e+13
61	1	1	7	3869200168	277	4	2.679421e+13
63	1	1	8	3869200168	282	17	2.727786e+13
...	...	...	...	...	...	...	...
15930	1	0	45	3869200325	3120	7	3.017976e+14
15931	1	0	139	3869200325	3130	0	3.027649e+14
15932	1	0	46	3869200325	3128	29	3.025715e+14
15937	1	0	62	3869200326	234	17	2.263482e+13
15939	1	0	63	3869200326	427	4	4.130371e+13
15942	1	0	61	3869200326	436	27	4.217428e+13
15943	1	0	62	3869200326	438	17	4.236774e+13
15945	1	1	21	3869200326	452	2	4.372196e+13
15946	1	0	57	3869200326	457	4	4.420561e+13
15947	1	1	1	3869200326	454	10	4.391542e+13
15948	1	0	58	3869200326	457	27	4.420561e+13
15951	1	1	139	3869200326	467	0	4.517291e+13
15953	1	0	60	3869200326	458	26	4.430234e+13
15954	1	1	4	3869200326	465	17	4.497945e+13
15959	1	0	35	3869200326	866	10	8.376819e+13
15962	1	0	54	3869200326	866	14	8.376819e+13
15963	1	0	56	3869200326	870	13	8.415511e+13
15964	1	0	33	3869200326	866	21	8.376819e+13
15969	1	0	35	3869200326	878	25	8.492895e+13
15973	1	1	62	3869200326	896	17	8.667009e+13
15977	1	0	40	3869200326	885	7	8.560606e+13
15979	1	1	59	3869200326	908	2	8.783085e+13
15980	1	1	61	3869200326	905	16	8.754066e+13
15981	1	1	38	3869200326	1117	1	1.080474e+14
15982	1	0	24	3869200326	1118	4	1.081441e+14
15983	1	0	59	3869200326	1119	4	1.082409e+14

```
15985     1     0           60  3869200326        1124          29  1.087245e+14
15988     1     0           64  3869200326        1121          12  1.084343e+14
15989     1     1           61  3869200326        1263           1  1.221700e+14
15990     1     1           38  3869200326        1264          18  1.222667e+14

[8084 rows x 7 columns]
```

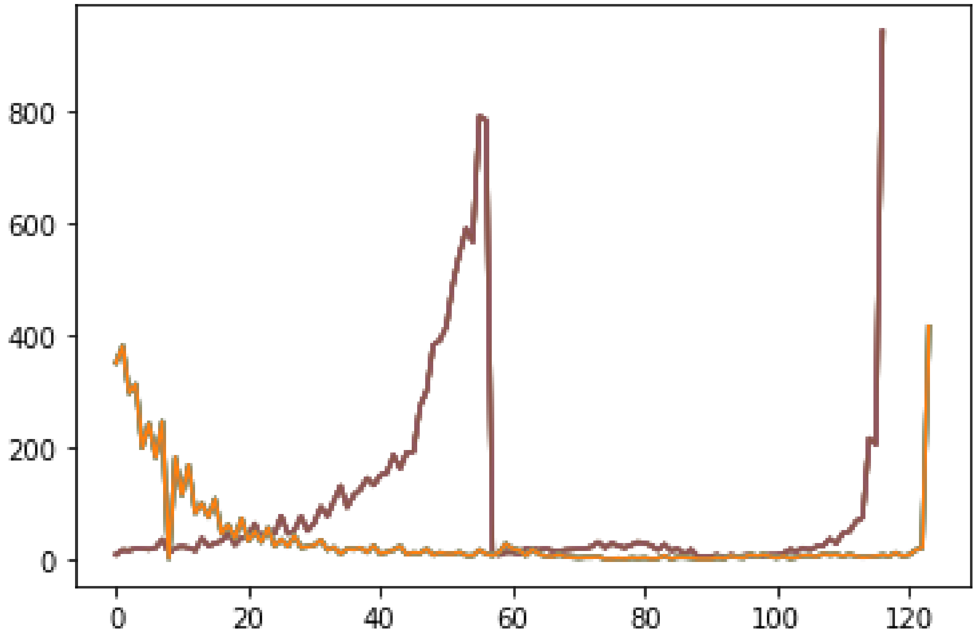
7. Make two occupancy plots (one per FPGA), i.e. plot the number of counts per TDC channel

```
In [10]: #data[data['TDC_CHANNEL']==1].groupby('TDC_CHANNEL').count()
         fpga0=np.array(data[data['FPGA']==0].groupby('TDC_CHANNEL').count())
         fpga1=np.array(data[data['FPGA']==1].groupby('TDC_CHANNEL').count())

         %matplotlib inline
         import matplotlib.pyplot as plt

         plt.plot(np.arange(len(fpga0)), fpga0)
         plt.plot(np.arange(len(fpga1)), fpga1)
         #mettere assi
```

```
Out[10]: [<matplotlib.lines.Line2D at 0x7efbbb957278>,
          <matplotlib.lines.Line2D at 0x7efba8089358>,
          <matplotlib.lines.Line2D at 0x7efba80894a8>,
          <matplotlib.lines.Line2D at 0x7efba80895f8>,
          <matplotlib.lines.Line2D at 0x7efba80b2630>,
          <matplotlib.lines.Line2D at 0x7efba8089860>]
```



8. Use the groupby method to find out the noisy channels, i.e. the TDC channels with most counts (say the top 3)

```
In [11]: count= data.groupby(['TDC_CHANNEL']).count()
         count= count.sort_values(by=['HEAD'], ascending=False).iloc[:3,0]
         print(count)
```

```
TDC_CHANNEL
139    1355
64      800
63      796
Name: HEAD, dtype: int64
```

9. Count the number of unique orbits. Count the number of unique orbits with at least one measurement from TDC_CHANNEL=139

```
In [12]: print(len(np.array(full_data.groupby('ORBIT_CNT').count())))
         print(len(np.array(full_data[full_data['TDC_CHANNEL']==139].groupby('ORBIT_CNT').coun
```

```
11001
10976
```

```
In [ ]:
```

```
In [ ]:
```