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

In [2]:
```
import matplotlib.pyplot as plt
```

In [3]:
```
import seaborn as sns
```

In [4]:
```
#ES_INDEX_DATE="2018-01-28"
ES_INDEX_DATE="2018-01-30"
ES_INDEX_DATE="2018-02-03"
DATA_FILE="../../rbcdata/%s/spv" % ES_INDEX_DATE
```

In [5]:
```
#%matplotlib inline
#import mpld3
#mpld3.enable_notebook()
```

In [6]:
```
%pylab inline
pylab.rcParams['figure.figsize'] = (16, 7)
```

```
Populating the interactive namespace from numpy and matplotlib
```

In [7]:
```
data=pd.read_csv(DATA_FILE)
```

In [8]:
```
section=75*60*60*48
```

In [9]:
```
temp=data.head(section)
#temp.head()
temp["ts"]=pd.to_datetime(temp["timestamp"])
temp=temp.set_index("ts")
temp=temp.tz_localize('UTC').tz_convert('Asia/Kolkata')
```

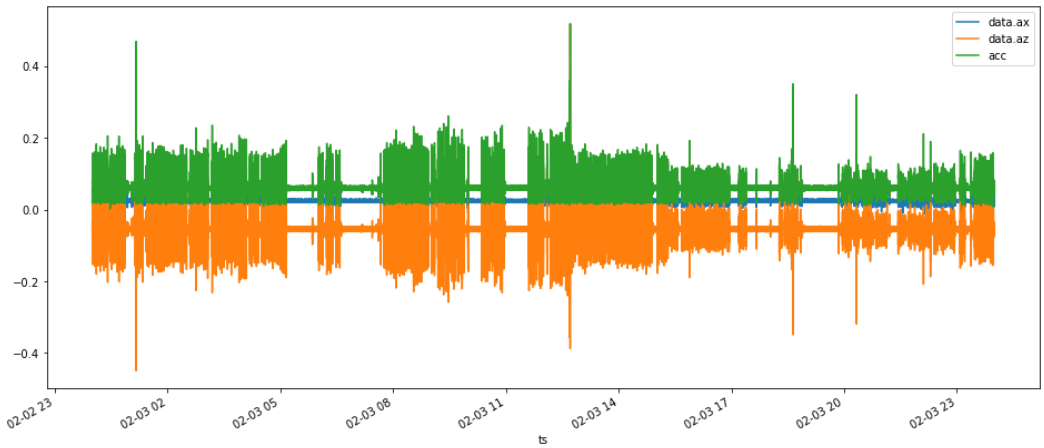
```
/home/sampad/Desktop/RBCCPS/lib/python2.7/site-packages/ipykernel_launche
r.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-doc
s/stable/indexing.html#indexing-view-versus-copy
  This is separate from the ipykernel package so we can avoid doing impor
ts until
```

In [10]:
```
temp.loc[:,"acc"]= np.sqrt(temp["data.ax"]**2 + temp["data.az"]**2)
```

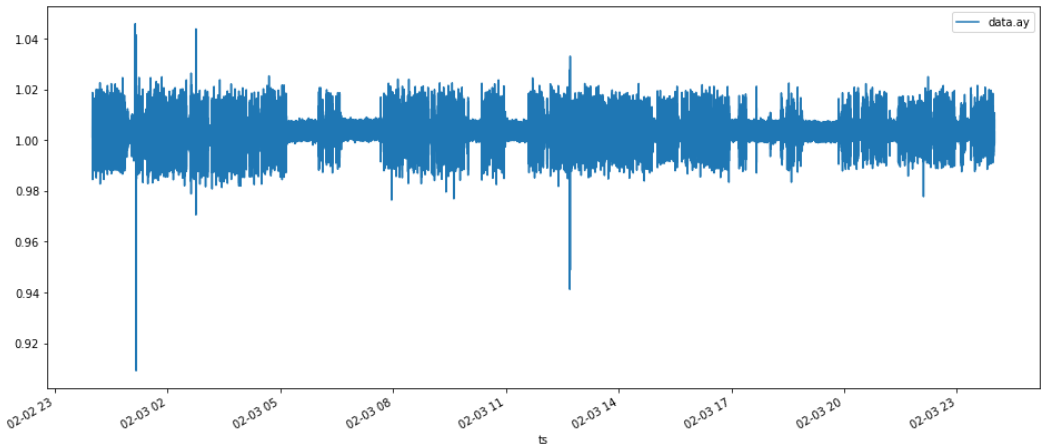
In [11]:
```
temp[["data.ax","data.az","acc"]].plot()
```

Out[11]: <matplotlib.axes._subplots.AxesSubplot at 0x7fc435f1add0>



In [12]:
```
temp[["data.ay"]].plot()
```

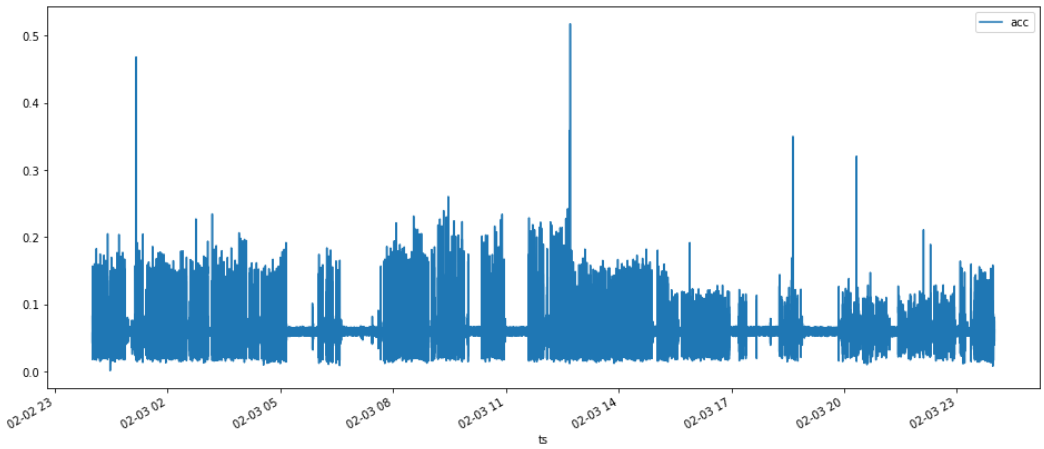
Out[12]: <matplotlib.axes._subplots.AxesSubplot at 0x7fc446bebfd0>



In [13]:
```
try:
    temp=temp.drop(columns=["timestamp","data.gx","data.gy","data.gz","d
ata.ax","data.ay","data.az"])
except:
    temp=temp.drop(columns=["timestamp","data.ax","data.ay","data.az","d
ata.A1","data.A2","data.A3"])
```

In [14]:
```
#temp=temp.head(75*60*60)
temp.plot()
```

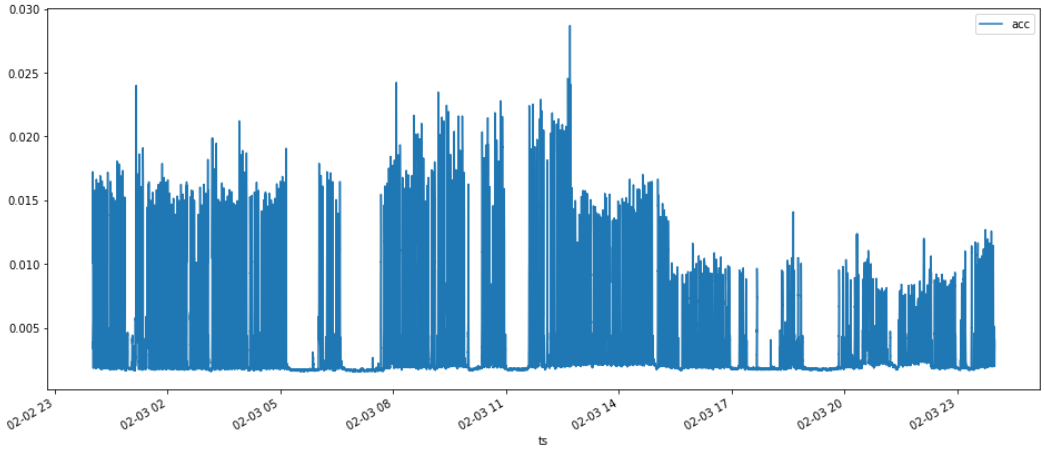
Out[14]: <matplotlib.axes._subplots.AxesSubplot at 0x7fc402ef7850>



In [15]:
```
x=temp.rolling(75*15).std()
```

In [16]:
```
x.plot()
```

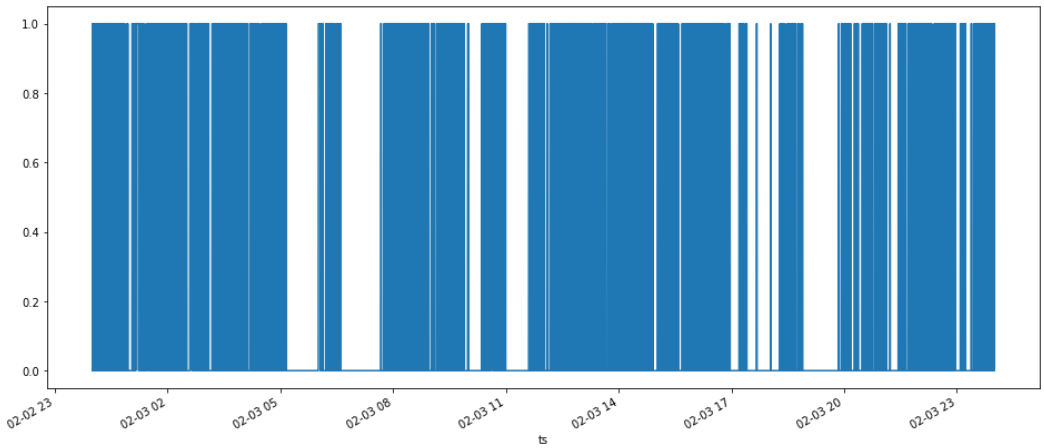
Out[16]: <matplotlib.axes._subplots.AxesSubplot at 0x7fc3db7e8a50>



In [17]:
```
if ES_INDEX_DATE=="2018-02-02":
    thresholdVal=0.003
elif ES_INDEX_DATE=="2018-02-03":
    thresholdVal=0.0035
else:
    thresholdVal=0.0035
x['acc_clipped'] = np.where(x['acc']>=thresholdVal, 1,0)
```

In [18]:
```
x["acc_clipped"].plot()
```

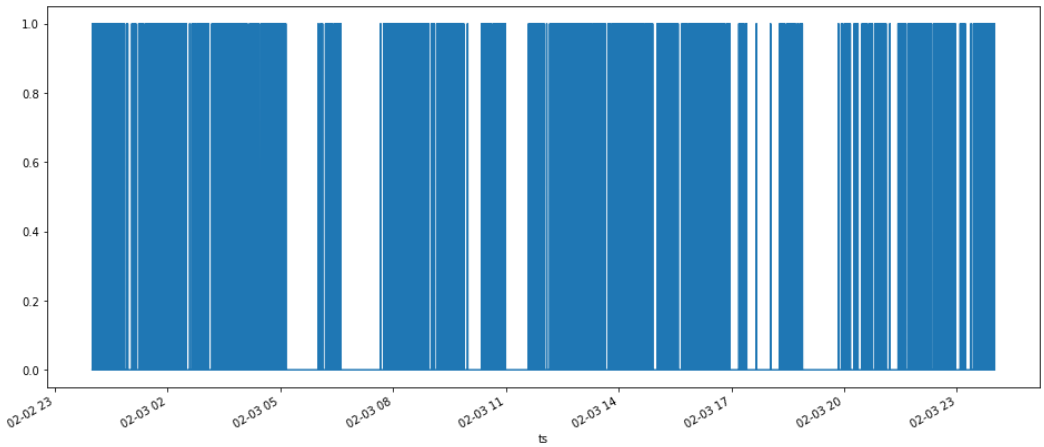
Out[18]: <matplotlib.axes._subplots.AxesSubplot at 0x7fc3c9814190>



In [19]:
```
x['change'] = x['acc_clipped'] - x['acc_clipped'].shift(1)
x['change'] = np.where(x['change']>0, 1,0)
```

In [20]:
```
x["change"].plot()
```

Out[20]: <matplotlib.axes._subplots.AxesSubplot at 0x7fc3ae9801d0>

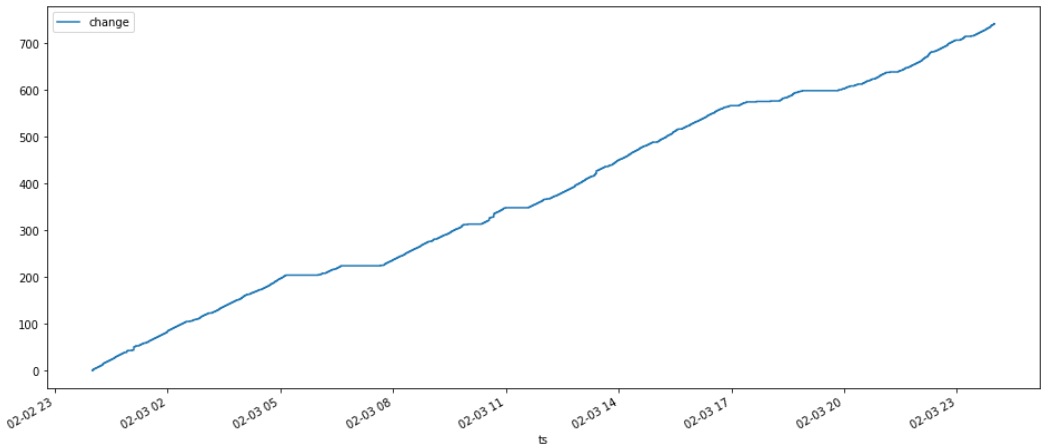


In [21]:
```
y=x[["change"]]
```

In [22]:
```
z=y.cumsum()
```

In [23]:
```
z.plot()
```

Out[23]: <matplotlib.axes._subplots.AxesSubplot at 0x7fc39578df10>



In [24]:
```
y.change.sum()
```

Out[24]: 741

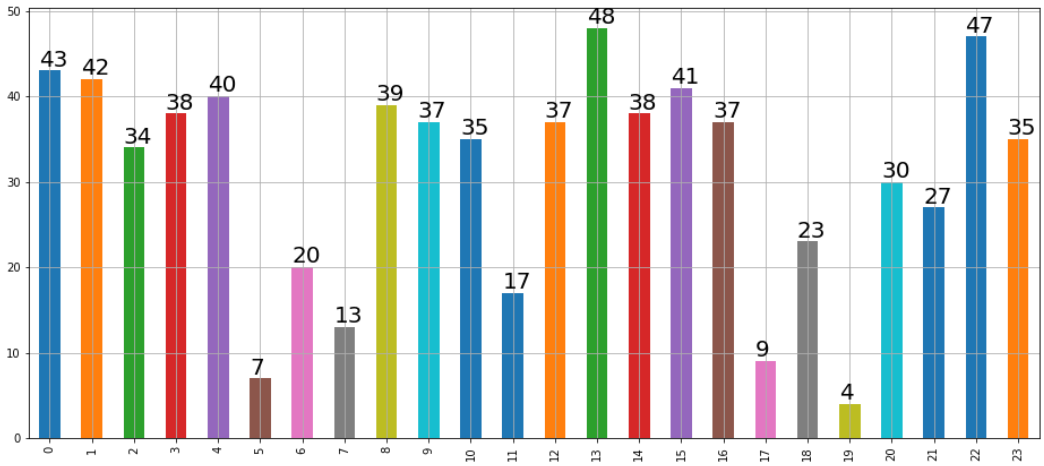
In [25]:
```
dformat= ES_INDEX_DATE + " %s:%s"
npcb=[]
for hr in range(0,24):
    start=dformat%(hr,"00:00")
    end=dformat%(hr,"59:59")
    zz=y.loc[start:end]
    ww=zz.change.sum()
    #print ww
    npcb.append(ww)
    print [hr,ww],

print "\n"
print "TOTAL--->",sum(npcb)
disection=y.loc[dformat%("00","00:00"):dformat%("23","59:59")]
print disection.change.sum()
```

```
[0, 43] [1, 42] [2, 34] [3, 38] [4, 40] [5, 7] [6, 20] [7, 13] [8, 39] [9
, 37] [10, 35] [11, 17] [12, 37] [13, 48] [14, 38] [15, 41] [16, 37] [17,
9] [18, 23] [19, 4] [20, 30] [21, 27] [22, 47] [23, 35]

TOTAL---> 741
741
```

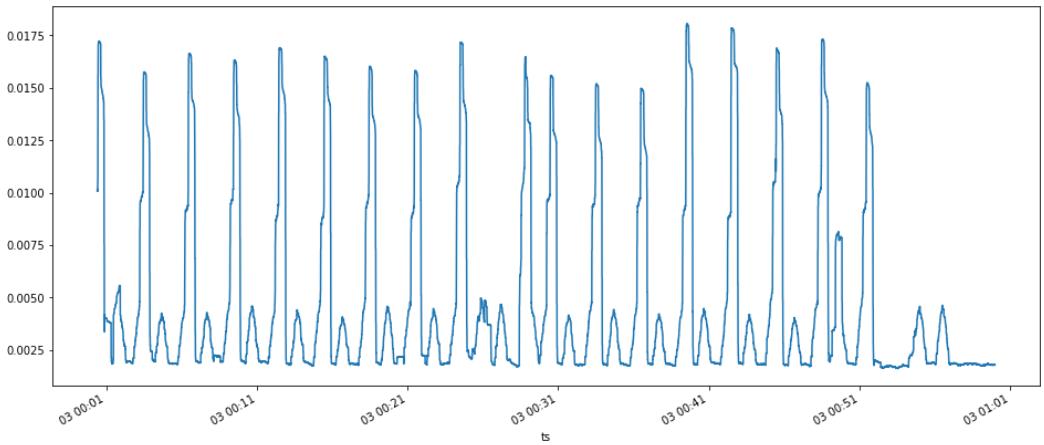
In [26]:
```
ax=pd.Series(npcb).plot.bar(grid=True)
for i in ax.patches:
    # get_x pulls left or right; get_height pushes up or down
    if i.get_height():
      ax.text(i.get_x(), i.get_height()+0.5, i.get_height() ,fontsize=20
,color='black')
```



In [27]:
```
_hour=0
_startmins = "00:00"
_endmins = "59:59"
```

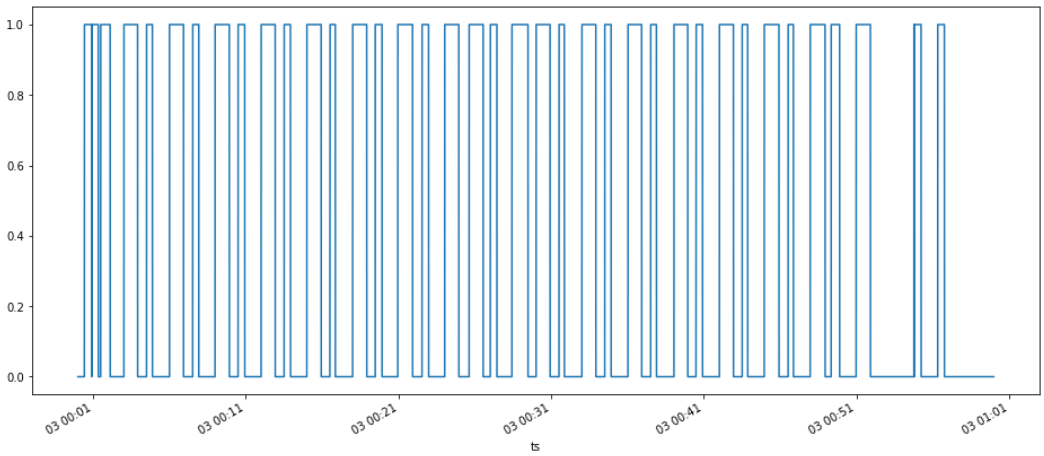
In [28]:
```
x.loc[dformat%(_hour,_startmins):dformat%(_hour,_endmins)].acc.plot()
```

Out[28]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fc366be0790>
```



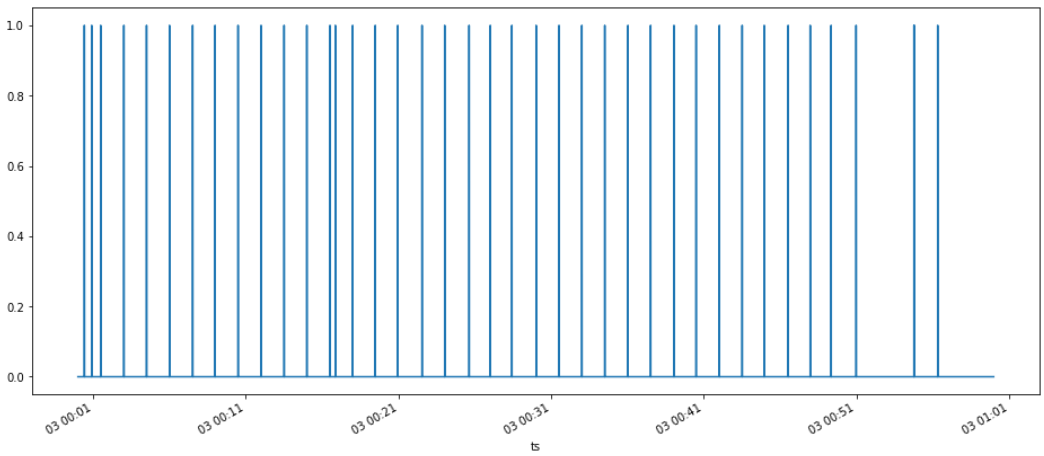
In [36]:
```
x.acc_clipped.loc[dformat%(_hour,_startmins):dformat%(_hour,_endmins)].p
lot()
```

Out[36]: <matplotlib.axes._subplots.AxesSubplot at 0x7fc41cfd6990>



In [30]:
```
x.change.loc[dformat%(_hour,_startmins):dformat%(_hour,_endmins)].plot()
```

Out[30]: <matplotlib.axes._subplots.AxesSubplot at 0x7fc366391710>



In [31]:
```
#Time PER PCB
```

In [84]:
```
pcbTime = x.loc[x.change==1]
```

In [85]:
```
pcbTime["timestamp"]=pcbTime.index
```

```
/home/sampad/Desktop/RBCCPS/lib/python2.7/site-packages/ipykernel_launche
r.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-doc
s/stable/indexing.html#indexing-view-versus-copy
  """Entry point for launching an IPython kernel.
```

In [86]:
```
len(pcbTime)
```

Out[86]: 741

In [91]:
```
pcbTime.timestamp.diff().mean()
```

Out[91]:
```
Timedelta('0 days 00:01:56.714842')
```

In [95]:
```
pcbTime.timestamp.diff().std()
```

Out[95]:
```
Timedelta('0 days 00:04:11.818123')
```