In [2]:
```
import sys
import imp
# i put the binfilepy in the same folder as the jupyter notebook for now
sys.path.append('binfilepy-master/binfilepy/')
import binfile
import numpy as np
import time
import pandas as pd
```

In [3]:
```
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
%matplotlib notebook
```

In [4]:
```
imp.reload(binfile)

start_time = time.time()
filename = 'output/2016-02-19-0004a8af.adibin'
with binfile.BinFile(filename, "r") as f:
    # You must read header first before you can read channel data
    f.readHeader()
    # I want to read the entire file
    data = f.readChannelData(offset=0, length=0, useSecForOffset=False,
useSecForLength=False)
elapsed_time = time.time() - start_time
print(elapsed_time)
```

```
34.89329433441162
```

In [65]:
```
# print('Dataformat: ', f.header.DataFormat)
# print('Year: ', f.header.Year)
# print('Month: ', f.header.Month)
# print('Day: ', f.header.Day)
# print('Hour: ', f.header.Hour)
# print('Minute: ', f.header.Minute)
# print('Second: ', f.header.Second)
# print('Seconds per tick: ', f.header.secsPerTick) #sampling rate 300 H
z
```

In [5]:
```
starttime = pd.to_datetime(str(f.header.Month)+'/'+str(f.header.Day)+'/'
+str(f.header.Year)+' '+str(f.header.Hour)+':'+str(f.header.Minute)+':'+
str(f.header.Second))
sf = 300 #this is the max sampling rate for UCI that all the files are u
psampled to
print(starttime)
for fii in np.arange(len(f.channels)):
    print(f.channels[fii].Title, f.channels[fii].Units, 'Scale: ', f.cha
nnels[fii].scale, 'Offset:', f.channels[fii].offset)
    if fii == 0:
        wave = pd.DataFrame(data[fii], columns=[f.channels[fii].Title])
    else:
        wave[f.channels[fii].Title] = data[fii]
    if f.channels[fii].Title in ['GE_ART', 'INVP1']:
        wave.loc[wave[f.channels[fii].Title] < 0, f.channels[fii].Title]
= 0.
    if f.channels[fii].Title in ['GE_ECG', 'ECG', 'PLETH']:
        wave.loc[wave[f.channels[fii].Title] == -1.700000e+308, f.channe
ls[fii].Title] = 0.

wave['timestamp'] = starttime + pd.to_timedelta(wave.index*f.header.secs
PerTick, unit='s') - pd.Timedelta('8H')
wave = wave.set_index('timestamp')

wave.head()
```

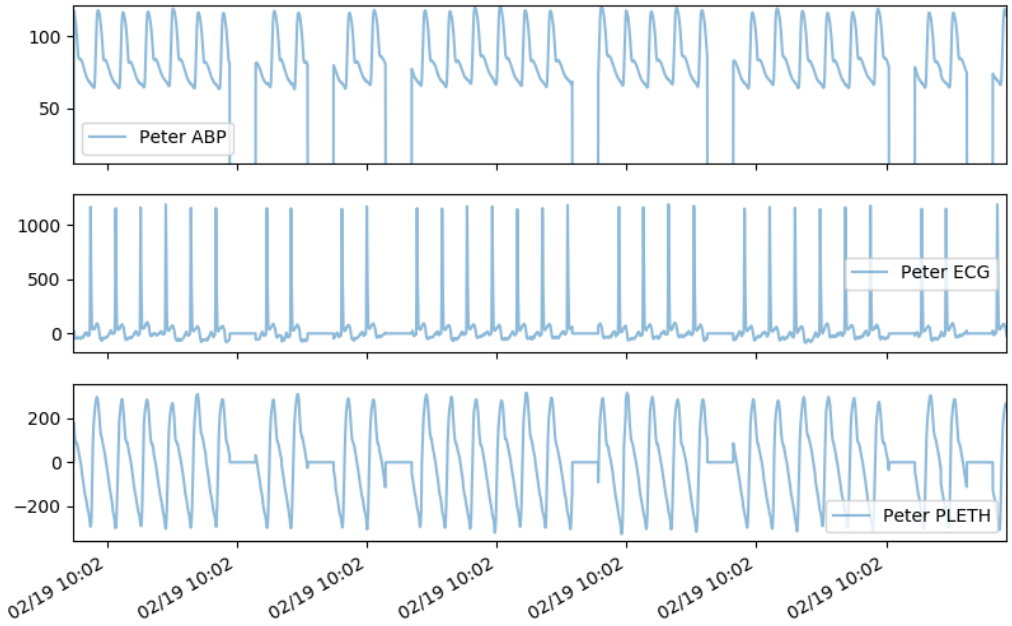
```
2016-02-19 14:30:06
ECG mV Scale:  1.0 Offset: 0.0
INVP1 mmHg Scale:  0.01 Offset: 0.0
PLETH mV Scale:  1.0 Offset: 0.0
CO2  Scale:  1.0 Offset: 0.0
AWP  Scale:  1.0 Offset: 0.0
FLOW  Scale:  1.0 Offset: 0.0
RESP  Scale:  1.0 Offset: 0.0
```

Out[5]:

	ECG	INVP1	PLETH	CO2	AWP	FLO
timestamp						
2016-02-19 06:30:06.000000000	0.0	0.0	-2.0	-1.700000e+308	-1.700000e+308	-1.700000e+30
2016-02-19 06:30:06.003333333	0.0	0.0	-2.0	-1.700000e+308	-1.700000e+308	-1.700000e+30
2016-02-19 06:30:06.006666667	0.0	0.0	-2.0	-1.700000e+308	-1.700000e+308	-1.700000e+30
2016-02-19 06:30:06.010000000	0.0	0.0	-2.0	-1.700000e+308	-1.700000e+308	-1.700000e+30
2016-02-19 06:30:06.013333333	0.0	0.0	-2.0	-1.700000e+308	-1.700000e+308	-1.700000e+30

In [7]:
```
fig, axes = plt.subplots(3, 1, sharex=True)
waveplot = wave.loc[(wave.index >= pd.to_datetime('2016-02-19 10:00')) &
(wave.index <= pd.to_datetime('2016-02-19 10:05'))]

if 'GE_ART' in waveplot.columns:
    axes[0].plot(waveplot['GE_ART'], label='Peter ABP', alpha=0.5)
    axes[1].plot(waveplot['GE_ECG'], label='Peter ECG', alpha=0.5)
else:
    axes[0].plot(waveplot['INVP1'], label='Peter ABP', alpha=0.5)
    axes[1].plot(waveplot['ECG'], label='Peter ECG', alpha=0.5)
if 'PLETH' in waveplot.columns:
    axes[2].plot(waveplot['PLETH'], label='Peter PLETH', alpha=0.5)
fmt = mdates.DateFormatter('%m/%d %H:%M')
axes[1].xaxis.set_major_formatter(fmt)
fig.autofmt_xdate()
axes[0].legend()
axes[1].legend()
axes[2].legend()
```



Out[7]: <matplotlib.legend.Legend at 0x7f3502225400>

In [87]:
```
waveplot['GE_ECG'].min()
```

Out[87]: -1.7e+308