The OpenPMU Project: Challenges and Perspectives

David M. Laverty, *Member, IEEE*, Luigi Vanfretti, *Member, IEEE*, Iyad Al Khatib, *Member, IEEE,* Viktor K. Applegreen, Robert J. Best, D. John Morrow, *Member, IEEE*

Abstract— **The OpenPMU project is a platform for the development of Synchrophasor measurement technology, Phasor Measurement Units (PMU), in an open source manner. The project has now been operating for a number of years and has seen increased adoption at Universities and interest from electrical utilities. The OpenPMU device has recently been tested against the IEEE C37.118 standard and shown to operate within the specification.**

This paper discusses the OpenPMU project from the perspective of the past two years of experience and evaluates successes and opportunities for improvements in both the OpenPMU device and the philosophy of the design.

Index Terms—**Phasor Measurement Unit, Open Source, Synchrophasor, Smart Grid**

I. Introduction

The OpenPMU project was originally developed as an outcome from research at Queen's University Belfast, UK, (QUB) into applications of Synchrophasor technology at low voltages for distribution network applications. Such applications have included Synchronous Islanding of diesel generator sets [1] and differential Anti-Islanding detection [2]. It had been identified that a number of other university projects had developed similar PMU devices, but the source code and design drawings for such devices were not readily available. The OpenPMU project was established so as to share PMU technology developed at QUB with other interested parties with the aspiration that improvements would be made by other institutions and returned to the project.

Following the presentation of the OpenPMU project at the 2010 IEEE PES GM in Minneapolis, a connection was made with the KTH Royal Institute of Technology in Stockholm, Sweden. KTH were looking for an open source PMU and used published design drawings to fabricate their own unit. Subsequently KTH has contributed to the project with middleware that adds an IEEE C37.118.2 [3] communications interface to the OpenPMU device.

The project was the subject of an invited paper to the 2011 IEEE PES GM in Detroit, where it was presented before the panel "Open Source Software: Enabling the Smart Grid" [4]. Updates on the project were presented at the 2012 GM in San Diego [5]. By this time, over fifteen of the OpenPMU units have been deployed in applications by QUB and two units have been fabricated independently by KTH. Other universities and colleges have expressed interest in the project and have fabricated devices or accessed the OpenPMU source code, including Colorado State University and the US Military Academy. National Instruments, whose hardware is used in the OpenPMU device, have published a case study on the project on their website [6]. This paper documents the replication efforts carried out at KTH SmarTS Lab.

Recently, the OpenPMU device has been tested with respect to the IEEE C37.118.1-2011 [7] standard to verify the quality of its measurements. The device passed the relevant tests, which have now been published by the IEEE Instrumentation & Measurement Society [8]. To enhance the phasor estimation process and provide alternatives for synchronized timing, planned hardware upgrades for time synchronization are briefly discussed.

As the project matures and more experience is gained, it is important to address some of the issues and challenges that have arisen. With many units now deployed, the OpenPMU devices have operated cumulatively for many thousands of hours. During this time, many software issues have been identified and resolved, but more intrinsic issues with the design remain. These are discussed in this paper, and a roadmap for the future of the device is identified.

Other issues regarding the structure and organization of the project have yet to be addressed, but one of the fundamental issues is the license under which the project operates. There are several open source license available, but many will be incompatible with existing agreements with funding bodies or industrial sponsors. These issues are discussed within this paper.

II. Background to OpenPMU

The OpenPMU project originated from work at QUB on applications of Synchrophasor technology to distribution network problems. In developing a low cost instrument to acquire GPS synchronized phasor measurement, the authors became aware of a number of similar university projects, but were unable to find a design appropriate to their needs that had source code and design drawings available. Thus it was necessary to work independently and so a solution was developed using a National Instruments data acquisition board

Paper submitted for presentation at the panel session of the IEEE Task Force on Open Source Software, IEEE PES General Meeting, Vancouver, 21-25 July 2013.

D. M. Laverty, R. Best and D. J. Morrow are with the Electrical Power and Energy Research Cluster, Queen's University Belfast, Ashby Building, Belfast, BT9 5AH, Northern Ireland (e-mail: david@laverty.org.uk; r.best@qub.ac.uk, dj.morrow@ee.qub.ac.uk).

L. Vanfretti, I. Al Khatib and V.K. Appelgrenn are with KTH Royal Institute of Technology, Stockholm, Sweden (e-mail: luigiv@kth.se, iyad.alkhatib@gmail.com, viktorap@kth.se).

interfaced to a Garmin GPS Engine using a custom microcontroller solution. Details of this design have been published in the proceedings of the IEE PES GM previously [4].

Realizing that such duplication of effort was not an efficient use of time or resources, the authors determined to share the design of the QUB type phasor measurement unit under an open source license, thus the OpenPMU project was formed. At this stage, the output of the OpenPMU was in a CSV formatted ASCII string. OpenPMU is developed largely using the Labview development environment.

The OpenPMU was identified as a solution which met the needs of projects at KTH Royal Institute of Technology. Here the first OpenPMU outside of QUB was fabricated. KTH was able to provide programming experience to add C37.118 communication to the device.

It was established that the OpenPMU should be modularized so as to separate the functions of GPS disciplined measurements (a largely hardware orientated task) from the functions of phase estimation (mathematics) and telecommunications (programming and computer science skills). The structure of the OpenPMU is shown in Fig. 1.

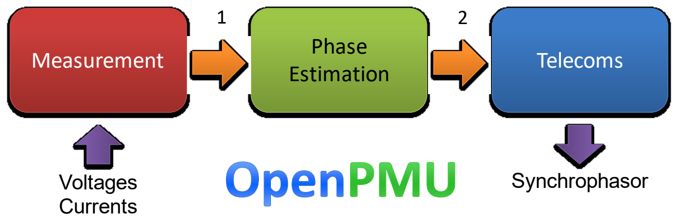


Fig. 1. Subsystems of the OpenPMU platform

The individual functions of the OpenPMU communicate with their neighboring functions using XML over UDP. This methodology was chosen since it reduces dependence on any particular vendor. Hardware can be swapped for other hardware that supports streaming sample data in XML, and phase estimation, for example, can be performed in any programming language provided a UDP interface can be established. The present Telecoms module is written in C#.

III. Measurement Standards Compliance

Compliance with the IEEE C37.118.1-2011 standard for Synchrophasor measurements was tested experimentally and is the subject of a paper published in IEEE Transactions on Instrumentation and Measurement [8]. In summary, a series of test waveforms were prepared in the MATLAB environment according to the scenarios described in [7]. These waveforms were exported in WAVE audio format, yielding compatibility with many test instrumentation platforms. The Omicron CMC156 test set was used to generate three-phase voltages to apply to the OpenPMU to test its compliance with [7] under both nominal frequency and dynamic conditions. Tests included verifying the magnitude and phase angle response of the PMU, and modulation of signal amplitude and phase. A bandwidth test was also performed, and is presented in Fig. 2.

A Total Vector Error (TVE) of 3% is allowed up to a modulation frequency of 1 Hz. Although the OpenPMU falls slightly outside of specification, it is comparable with PMUs from several vendors [9], [10]. An independent study of the performance of multiple vendors PMUs will be pursued in the near future by the authors.

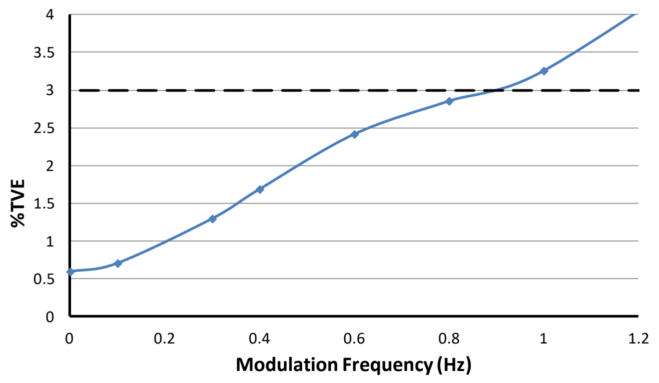


Fig. 2. TVE (%) versus Modulation Frequency of the OpenPMU as tested in accordance with IEEE Std. C37.118.1, Subclause 5.5.6.

IV. Communications Interface

A major task of any PMU is to be able to communicate with other ends and send measured data. However, this brings about a few challenges which can be divided into four categories: design, implementation, performance, and standards. These challenges are discussed below, but before that basic design of the communication module for the OpenPMU is provided.

Fig. 3 shows the OpenPMU Communication Middleware (OCM) and its interaction with the data source (Labview in this example) and any data requester (e.g. a Phasor Data Concentrator (PDC)) [8]. The Labview module (or any other data wrapper) sends the phasor data to a special input interface. This is designed so far to suit a network connection or a data base (DB) interface. Using any of the aforementioned has its advantages and its challenges. The OCM comprises three sub-modules: input interface, communication engineer, and an output interface). The Input interface accepts data transmitted from the Labview module, of which UDP is best to be used since it offers independence of the machine and the HW used. Using TCP, offers the same advantage but can lead to more processing power due to its heavier code and need for acknowledgements. Using a DB has its advantages, but its management, memory requirements, processing speed (that may go against real-time requirements) can heavily hinder delivering acceptable quality of service (QoS).

The communication engine does all the required computation to decide, what to send, how to send it, when to drop data, check for errors, choose the standards, and understand requests from an outside source. However, the sub module that is responsible for communicating with the data requester and carrying the requests to the engineer is the Output interface, which is also responsible for receiving data from the communication engine, wrap it with the right protocol and transmit it out. At the moment, the Output interface supports only IEEE C37.118 as an application layer protocol over the TCP/IP suit.

Having described the basics for the OCM, we can point out the challenges that such a service faces. We list and describe these challenges as follows:

A. *Design*

1) Human factor

One of the most significant challenges while working on such a multidisciplinary project is to find the right people with the right skills and the right mindset. We tried to solve the human factor problems by having different workers from different background and skills, but the gaps were huge to fill and this issue was time consuming. Hence, building the team to work together with such significant differences was a challenge, and we have now some experience in how to overcome such a challenge in some areas. However this experience consumed time and a lot of effort.

2) Design optimization

The design phase of the OCM can be made in many ways, but the challenge is to know which OCM design is optimal. Every error or unoptimized section of the design will cost nearly 10*x* more to go back to fix. Hence, having an optimal design from the very beginning is a challenge. Such a challenge is still till now dependent on point (a), the human factor. It is very difficult at such a current state to automate the design process of the OCM.

3) Application Characterization

From the OCM's design and implementation in a specific language (here in VC++), and after assessing its performance, it was necessary to determine how this application was functioning with respect to multiple cases of input. This was a challenge since it requires specific ways in characterizing the application. For instance, an assessment of the throughput of the application with respect to multiple inputs is being performed.

4) Real-time issues

An important parameter to respect in the OCM is time. The "level" of real time granularity needs to be defined, and this is challenging. In some cases it is in seconds, in others it is in ms, and now we even look at the microseconds and still may need more. Paralleling the application functions helps, but is not the solution. The challenge is to have a dynamic decision for real-time requirements and limits and relating them to both HW and SW, i.e. HW-SW co-decision making.

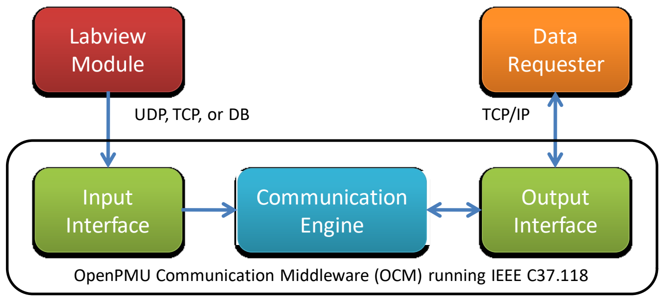


Fig. 3. The OpenPMU Communication Middleware running IEEE C37.118.

B. *Implementation*

1) Human factor

As discussed in point A.1, the human factor remains a large challenge for implementation also. One big problem we faced is that the implementation requires a programmer that knows, very well, how to program in many different environments, Operating Systems, and platforms. In addition, it requires the ability to send/receive data between the different platforms. For instance, in this work, we need to have an experienced Labview programmer experienced in standard C, VC++, as well as in HW issues. This remains a challenge.

2) Choice of Real-time

Real-time parameter choices remain hard since we usually leave them for the programmer, but in this case the programmer may not know very well the application behavior so the challenge is who chooses or decides on real-time parameters, checks, testing, etc. In our case we have two major real time requirements that may be improved if more parameters are added, but then this depends on if the HW part can affords that much calculation in real time.

C. *Performance*

1) Design of experiments

Up till now, no good design of experiments for testing is satisfactory and this remains an open issue.

2) Delay

We differentiate between computation delay and communication delay. We also look at the challenges of reception and transmission speeds. Since time is a sensitive parameter, delays are treated accordingly and the challenge is to find the correlation between computational delays and communication delays based on different standards, and intrinsic-machine differences in speeds of reception and transmission.

3) Scalability

The challenging question is: how scalable can the SW implementation be on a specific HW connected to a specific communication network based on the design made? This is, in itself a study that may take months to estimate and perform.

D. *Standards*

1) Supporting multiple standards

A major challenge is to be able to dynamically support multiple different standards (e.g. IEEE C37.118.2 and IEC 61850-90-5).

2) Concurrent different standards

Even if point D.1 is solved a challenge would be faced when trying to connect the OCM module to different PDCs with different standards at the same time.

3) Scalability

Point C.3 brings up the issue of scalability with a multitude of standards, which remains a significant challenge also.

4) Which HW resources for which standard

A good idea, theoretically would be to assign different HW resources to different standards, but this comes with human factor, OS, and programming language challenges.

5) Standards are not perfect

We noticed a very important problem in standards: they are not perfect and their interpretation is not transparent. Many software glitches occur due to some missed point in the standard that we had to compensate for via our own design and implementation. This remains a challenge for the implementer, who finds the implementation failing sometimes but due to an unstudied or misunderstood point in the standard.

With all the aforementioned challenges we are sure that more will arise as we continue our work in this multidisciplinary project. Therefore, we believe it is very relevant and important to share our experience so that this can speed up the development and progress towards a better OCM and a better PMU community.

V. OpenPMU Replica at KTH SmarTS Lab

As mentioned earlier, there have been different efforts of independent fabrication of the OpenPMU at other universities. One such unit was replicated by the KTH SmarTS Lab and is shown in Figs. 4 and 5. All components described in the OpenPMU website where assembled on PCB fabricated by a small company in Stockholm. Fig. 4 shows the set-up at the laboratory comprised of the Garmin GPS Engine, main board, DC power supply and a laptop with the OpenPMU Labview software, while Fig. 5 shows a close up of the main board and GPS engine. Finally, a capture of the OpenPMU estimated phasor from the laboratory mains is shown in Fig. 6.

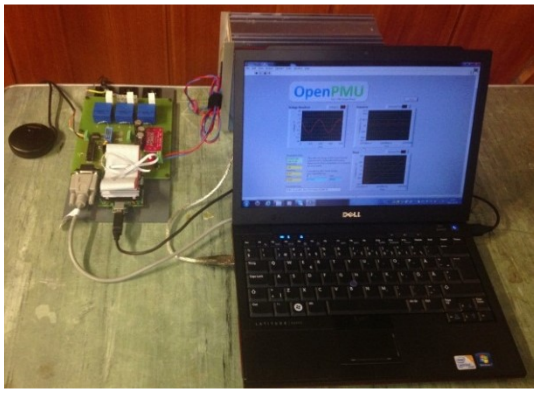


Fig. 4. Replicated OpenPMU set-up at KTH SmarTS Lab.

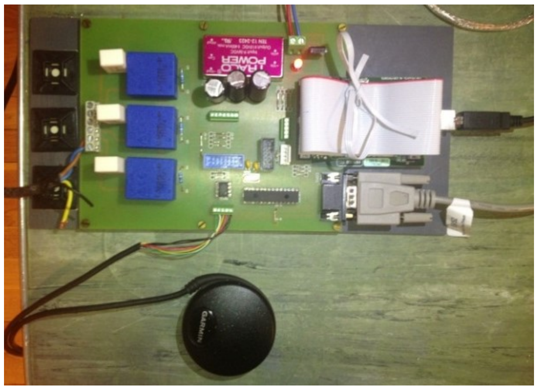
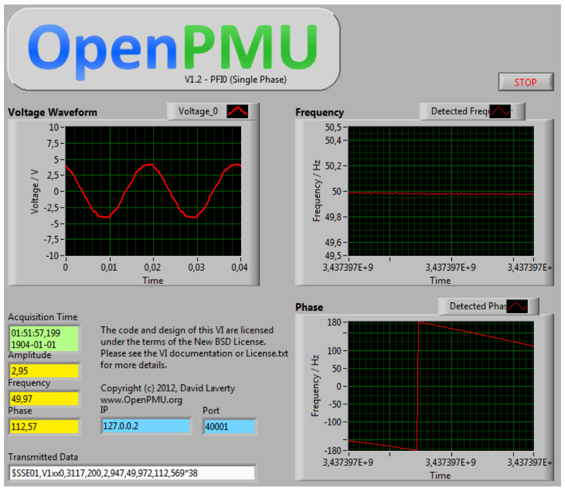


Fig. 5. Main board of the OpenPMU replicated at KTH SmarTS Lab.



Acquisition Time
01:51:57,199
1904-01-01
Amplitude
2,95
Frequency
49,97
Phase
112,57

The code and design of this VI are licensed under the terms of the New BSD License. Please see the VI documentation or License.txt for more details.

Copyright (c) 2012, David Laverty
www.OpenPMU.org

IP
127.0.0.2
Port
40001

Transmitted Data
$SSE01,V1xx0,3117,200,2,947,49,972,112,569*38

Fig. 6. Estimated phasor from KTH' SmarTS Lab mains

VI. Planned Hardware Upgrades for Time Synchronization

Timing information is currently provided to the OpenPMU via a Garmin GPS engine and a custom microcontroller. This is suitable for installations that do not have a stationary GPS antenna installed. However, for installations where stationary GPS antennae and antenna splitters are available, replacing the current GPS engine for a receiver IC might be more suitable.

The GPS-1513R high performance compact GPS receiver in Fig. 7 [11] is a highly configurable device that can be incorporated into OpenPMU's main board so that GPS signals from stationary antennas can be used for time synchronization. To this aim, a modified PCB design will be developed and tested at KTH SmarTS Lab during 2013. Other modifications to reduce the footprint, simplify fabrication and usage.

Another planned improvement is the replacement of the internal clock of the NI DAQ board with an external timing signal. This timing signal will be derived from a Phase-Locked-Loop which will be implemented in the PCB and will be synchronized with the GPS receiver. This modification will help improving the accuracy of the phasor estimation process.

Fig. 7. GPS-1513R high performance compact GPS receiver

Additionally, the cRIO platform from National Instruments is now being considered as a data acquisition module for the OpenPMU project. This new hardware avenue will be developed in parallel to the present data acquisition solution. The National Instruments cRIO platform is rapidly becoming a common feature of laboratory hardware, so the inclusion of this option will lower the barriers to enter into the OpenPMU project and aims to expand the community of developers.

VII. Discussion of Open Source Licenses

The OpenPMU project is operated according to the principles of the GNU General Public License (GPL) [12]. The ambitions of the GNU GPL can be paraphrased to fit with the OpenPMU project as follows. The user should have the freedom to:

1) Run the OpenPMU program,
2) Study how the OpenPMU works,
3) The freedom to distribute copies of the OpenPMU,
4) Distribute copies of the modified OpenPMU to others.

While the authors fully intend to strive to realize these ideals, there are difficulties. The most obvious is the fact that the OpenPMU is a hardware device. While software licenses can be applied to its phase estimation algorithms and hardware timing firmware, hardware cannot be copyrighted in the same way. For this reason, the design drawings of the OpenPMU are protected under copyleft licenses which require redistribution of modified works so as to avoid attempts at closing modifications to the OpenPMU design.

There are concerns that GPL like licenses are 'infectious' and that if included in projects that have commercial software, this essentially requires that the commercial code is released and distributed under GPL. The OpenPMU project wishes to avoid such scenarios, and as such accepts that individual modules in the project might require separate licensing by necessity of existing licensing agreements, non-disclosure agreements and so on.

Ideally, it is desired to reach a point in time when a commercial vendor could use an OpenPMU algorithm in their product, without compromising their intellectual property, but with acknowledgement that OpenPMU technology is in use in their device.

VIII. Community Involvement

To get involved with the OpenPMU project and to communicate with the authors of this paper, please visit www.OpenPMU.org [13]. The project is hosted on the Codeplex system, which provides Wiki style documentation, message boards and news feeds. For less formal contact and announcements, the authors maintain a Facebook page. Simply search for 'OpenPMU' within Facebook.

The authors particularly welcome suggestions for improving the project, and welcome partners interested in contributing to the project. Please do not hesitate to make contact if you have any questions regarding taking part in OpenPMU or building your own OpenPMU.

IX. Conclusions

OpenPMU developed by Queen's University Belfast out of the need for a low cost PMU, with transparent operation, for use in applications on low voltage networks were the expense of current commercial PMUs could not be justified. Functions such as disturbance / transient event recorder are not necessary for many of these applications. To avoid duplication of effort at other academic institutions, the QUB PMU was released under an Open Source license.

Since the initial release of OpenPMU, the project has been enhanced by the contributions of KTH, specifically with the addition of IEEE C37.118.2 telecommunications support. The replication efforts from KTH reported in this article serves to show that the OpenPMU can be fabricated using the design drawings on www.OpenPMU.org.

Future activity will see the continued improvement of the current instrumentation platform and phase estimation software, and the inclusion of a National Instruments cRIO measurement system to allow more rapid development of an OpenPMU system where such components are available in the laboratory.

The authors encourage others with interests in PMU technology to become involved in the OpenPMU project and help the community develop.

X. References

[1] Best, R.J.; Morrow, D.J.; Laverty, D.M.; Crossley, P.A.; , "Synchrophasor Broadcast Over Internet Protocol for Distributed Generator Synchronization," *Power Delivery, IEEE Transactions on* , vol.25, no.4, pp.2835-2841, Oct. 2010

[2] Laverty, D.M.; Morrow, D.J.; Best, R.; Cregan, M.; , "Anti-islanding detection using Synchrophasors and Internet Protocol tele-communications," *Innovative Smart Grid Technologies (ISGT Europe), 2011 2nd IEEE PES Int. Conf. and Exhibition on* , pp.1-5, 5-7 Dec. 2011

[3] "IEEE Standard for Synchrophasor Data Transfer for Power Systems," IEEE Std C37.118.2-2011, 2011, Online: http://standards.ieee.org/findstds/standard/C37.118.2-2011.html [Accessed: April 12 2012]

[4] Laverty, D.M.; Morrow, D.J.; McKinley, A.; Cregan, M.; , "OpenPMU: Open source platform for Synchrophasor applications and research," *Power and Energy Society General Meeting, 2011 IEEE* , vol., no., pp.1-6, 24-29 July 2011

[5] Laverty, David M.; Vanfretti, Luigi; Best, Robert J.; Morrow, D. John; Nordstrom, Lars; Chenine, Moustafa; , "OpenPMU technology platform for Synchrophasor research applications," *Power and Energy Society General Meeting, 2012 IEEE* , vol., no., pp.1-5, 22-26 July 2012

[6] National Instruments, "OpenPMU: The Open-Source Phasor Measurement Unit", *NI Case Studies,* Internet: http://sine.ni.com/cs/app/doc/p/id/cs-14787 [Accessed: Dec 05 2012]

[7] "IEEE Standard for Synchrophasor Measurements for Power Systems," IEEE Std C37.118.1-2011, 2011, Online: http://standards.ieee.org/findstds/standard/C37.118.1-2011.html [Accessed: April 12 2012]

[8] Laverty, D.; Best, R.; Brogan, P.; Khatib, I.; Vanfretti, L,, Morrow, J.; "The OpenPMU Platform for Open Source Phasor Measurements" (2012), *IEEE Transactions on Instrumentation & Measurement*, (in press since October 2012)

[9] R.M. Moraes, Y. Hu, G. Stenbakken, K. Martin, J.E.R. Alves, A.G. Phadke, H.A.R. Volskis, and V. Centeno, "PMU Interoperability, Steady-State and Dynamic Performance Tests," *IEEE Transactions on Smart Grid*, Early Access Article, Sept. 2012. 10.1109/TSG.2012.2208482

[10] G.Y. Yang, K.E. Martin, and J. Østergaard, "Investigation of PMU Performance under TVE Criterion," 5th International Conference on Critical Infrastructure (CRIS), Sept. 2010.

[11] RF Solutions, "Low-Power High-Performance and Low-Cost 65 Channel SMD GPS Module." Data Sheet, Version 1.4, RF Solutions, Jul. 9, 2009.

[12] The GNU Foundation, "GNU General Public License", Internet: http://www.gnu.org/licenses/gpl.html [Accessed: Dec 05 2012]

[13] The OpenPMU Project, Internet: http://www.openpmu.org/ [Accessed: December 7 2012]