Architecture Development and Implementation of a Synchrophasor-based Real-time oscillation damping Control System

Eldrich Rebello
Department of Electrical Engineering and Automation
Aalto University
Espoo, Finland
Email: eldrich.rebello@aalto.fi

Homer Simpson
Kungliga Tekniska högskolan
Stockholm, Sweden
Email: luigiv@kth.se
tre

Md. Shoaib Almas
Kungliga Tekniska högskolan
Stockholm, Sweden
Email: msalmas@kth.se

***Abstract*—The Phasor Power Oscillation Damping (Phasor POD) algorithm has been demonstrated to be effective at damping inter-area power system oscillations. Present implementations of the algorithm have been either in offline or real-time SIMULINK simulations. Previous research has demonstrated that wide-area signals are effective at achieving enhanced damping of inter-area modes. This work documents the design proposal and revision process for a real-time hardware prototype of the Phasor-POD algorithm on a Compact Reconfigurable Input Output (cRIO) controller from National Instruments. The real-world applicability of this design and the challenges faced together with the solutions implemented are also investigated.**

I. INTRODUCTION

The goal of this paper is to document the architecture development process and challenges faced in the real-time implementation of Ängquist and Gama's [2] Phasor Power Oscillation (Phasor POD) algorithm. For purposes of comparison, the real-time SIMULINK implementation of the Phasor POD algorithm by Almas and Vanfretti [7] is used as a benchmark. The performance of the hardware prototype developed will be tested and compared to the performance of this implementation of the algorithm. The two-area four-machine model developed by Klein, Rogers and Kundur [3] is used as a test-case and is modified slightly to fit the requirements of this work. A Hardware-in-the-loop setup is constructed around the eMEGASIM real-time simulator from OPAL-RT [8] with the two-area model running in real-time and a synchrophasor-based Phasor-POD algorithm running on a cRIO.

A. Previous Experiences

Subsection text here.

B. Paper Outline

This paper is organised as follows.
II presents this and that.
III presents

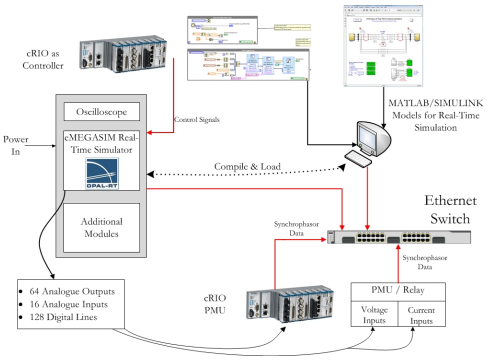

Fig. 1. Simulation Results.

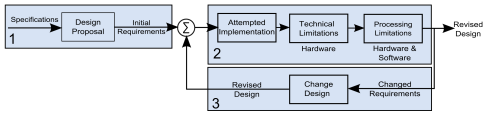

Fig. 2. Iterative Revision Flow Diagram

II. BACKGROUND

III. ARCHTECTURE DEVELOPMENT PROCESS

IV. ARCHITECTURE IMPLEMENTATION AND REFINEMENT

Figure 2 shows a three stage design process with design proposal, implementation and revision. To begin with, only the controller specifications were available. These were used to draft a design proposal. An implementation attempt was made using this draft proposal. When the additional limits imposed by the software and hardware platforms were included, some goals of the original code would have to revised. With these limitations, the original design was modified to generate a revised design. An attempt was then made to implement this revised design. If further limitations were encountered, the design was further modified. This iterative process was repeated till a working implementation was reached.

1) Initial Design: Initially, the goal was to keep the controller (the cRIO) independent of any other devices. Such a controller would be able to take synchrophasor data (in the C37.118 format) directly from the network and extract measurement data. The architecture block diagram is shown in Figure **??**. This controller was completely autonomous, with automated signal selection and processing. This design would be able to compute observability indices for the different input signals and then automatically select the one with the highest observability of a particular mode. This design also incorporated two control functions, a wide area control function and a local control function. The wide area control function selected for implementation was an inter-area oscillation damping control algorithm. The local control function would use locally available data and implement the control function in a manner similar to a Power System Stabiliser (PSS). Automatic switching between the two functions was also considered. If the signal to noise ratio in the wide area signal deteriorated or if the signal delay became prohibitively high, the controller would switch to the local signal or a back-up wide area signal automatically. The principal consideration in this design was the limited resources available on the FPGA and the complexity of the algorithms to be implemented. This required implementation of all algorithms and computation sections on the RT section of the cRIO. In addition, the RT controller has tremendous flexibility as algorithms are implemented using software, rather than hardware as on the FPGA. The RT controller also has more storage space available to implement algorithms. The trade-off was computation speed.

This design was faced with a problem of differing loop rates. The real-time simulator runs at a 50μs time step. The RT controller on the cRIO is not capable of matching this speed. The fastest that it could run was 1ms. To address this issue, this architecture envisaged using the FPGA to perform interpolation between successive data points. The FPGA would run at a loop rate of 50 microseconds, to match the real-time simulator. The region between successive data points was to be interpolated using a suitable interpolation algorithm. Data generated by the FPGA was to be sent over the communication network back to the real-time simulator for use in the simulation.

This design was changed due to limitations of different components. One, no software was available to read a synchrophasor stream on the RT controller and extract measurement data from the stream. This process had to be performed on a desktop computer running a real-time data mediator and LabVIEW.

Two, data generated by the FPGA could not be sent to the real-time simulator directly over the communication network. Though possible, further work is required.

Three, the process of data interpolation on the FPGA is complex. To achieve results better than with a simple linear interpolation, a history of past data points is required. This process is in itself complex and also introduces further delay. However, it is simpler than implementing the damping control algorithm on the FPGA. At this stage, FPGA limitations were the principal factor behind implementing the oscillation damping algorithm on the RT controller.

2) Development and Revision: With the limitations of the initial architecture in mind, it was revised. In this architecture, the process of extracting data from the synchrophasor stream was shifted to a workstation computer. The process of signal selection and signal processing was also moved to this computer. Once data was available, it would be sent to the RT controller, over a communication network. The damping control was kept on the RT controller with the FPGA performing the interpolation required to match the read-interval of the real-time simulator. Besides the damping control algorithm, a local control function was maintained on the RT controller. This revision is shown in Figure **??**.

Here, the complexity of implementing an interpolation algorithm on the FPGA was examined. It was realised that a simple linear interpolation algorithm would not be sufficient. Further, the FPGA output was limited by the speed of the RT controller as new data would only be available after a minimum of 1ms at the earliest (if the RT controller were to run at its maximum speed).

An implementation of the damping control algorithm on the RT controller was also tested. The fastest execution speed that could be achieved was in excess of 25ms. This was slower than the reporting rate of the PMUs.

Communication between the cRIO and the RT simulator using TCP/IP was also abandoned. Output values of the FPGA were instead hardwired to the real-time simulator's analogue inputs.

3) Final Implementation: At this stage, the damping control algorithm was moved to the FPGA with the RT controller being used only for network communication and data monitoring. The local control function was also discarded as it was found to reduce the response speed of the RT controller. If needed, the local control function can be implemented on the FPGA. The FPGA is suited to tasks that are repetitive in nature. The FPGA also has a fast and predictable response time. With the oscillation damping algorithm implemented on the FPGA, resource utilization stood at 78%. Space is still available if further functions need to be implemented on the FPGA. As with the previous design, the process of extracting data from the synchrophasor stream was done on a workstation computer. Signal selection and processing was also done on this computer. This was the final architecture as implemented.

V. Hardware-in-the-Loop Test

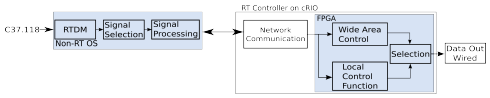

Fig. 3. Architecture Refinement Schematic

VI. Conclusion

The conclusion goes here.

Acknowledgment

The authors would like to thank...

References

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.

[2] L. Ängquist and C. Gama *Damping Algorithm based on Phasor Estimation* in Power Engineering Society Winter Meeting, 2001. IEEE, Volume 3, pp. 1160 - 1165

[3] M. Klein, J. G. Rogers and P. Kundur *A fundamental study of inter-area oscillations in Power Systems* IEEE Trans, PWRS, no. 6, pp. 914-921, 1991.

[4] *eMEGASIM Power Grid Real-Time Digital HArdware in the Loop Simulator* Available Online at : http://www.opal-rt.com/

[5] F. P. Dmello, and. C. Concordia *Concepts of Synchronous Machine Stability as Affected by Excitation Control* IEEE TRANSACTIONS ON POWER APPARATUS AND SYSTEMS, vol. PAS 88, no. 4, 1969.

[6] M. E. Aboul-Ela, A. A. Sallam, J. D. McCalley and A. A. Fouad, *Damping Controller Design for Power System Oscillations Using Global Signals*, IEEE trans. on Power Systems, Vol. 11, No. 2, May 1996, pp. 767-773

[7] M. Shoaib Almas and L. Vanfretti, *Implementation of Conventional PSS and Phasor Based POD for Power Stabilizing Controls for Real-Time Simulation*, IEEE IES IECON14, 29 Oct-1 Nov, 2014, Dallas, USA.

[8] eMEGAsim PowerGrid Real-Time Digital Hardware in the Loop Simulator Opal RT, [Online]. Available: http://www.opal-rt.com/

[9] N.S. Chaudhuri, R. Majumder and B Chaudhuri *Interaction between conventional and adaptive phasor power oscillation damping controllers* in Power and Energy Society General Meeting, 2010 IEEE Minneapolis, MN, 2012 pp. 1-7

[10] L. Vanfretti.; M. Chenine; M.S. Almas; R. Leelaruji; L. Angquist; L. Nordstrom, "*SmarTS Lab A laboratory for developing applications for WAMPAC Systems*," Power and Energy Society General Meeting, 2012 IEEE , vol., no., pp.1,8, 22-26 July 2012

[11] E.V Larsen and E.H Chow, General Electric Company, NY, *Application of Static VAR Systems for System Dynamic Performance*, 1987 IEEE pp. 43-46

[12] L. Vanfretti, Y. Chompoobutrgool, and J.H. Chow, *Chapter 10: Inter-Area Mode Analysis for Large Power Systems using Synchrophasor Data*, Book Chapter, in Coherency and Model Reduction of Large Power Systems, Joe H. Chow (Ed.), Springer, 2013. Available Online at http://www.springer.com/energy/systems%2C+storage+and+harvesting/book/978-1-4614-1802-3

[13] *Operating Instructions and Specifications Compact RIO NI cRIO-9081/9082*, National Instruments, Available Online at http://www.ni.com/pdf/manuals/375714a.pdf

[14] *Operating Instructions and Specifications Compact RIO NI cRIO-9075/9076*, National Instruments, Available Online at http://www.ni.com/pdf/manuals/375650b.pdf

[15] *FPGA Control on Compact RIO Sample Project Documentation*, National Instruments, Available Online at http://www.ni.com/white-paper/14137/en/

[16] Vanfretti, Luigi and Aarstrand, Vemund H. and Almas, M. Shoaib and Perić, Vedran S. and Gjerde, Jan O. , *A Software Development Toolkit for Real-Time Synchrophasor Applications*, IEEE PES Grenoble PowerTech, 2013

[17] N.S Chaudhuri, R Majumder and B Chaudhuri , *Interaction between conventional and adaptive phasor power oscillation damping controllers* in Power and Energy Society General Meeting, 2010 IEEE Minneapolis, MN, 2012 pp. 1-7

[18] I Kamwa *Performance of Three PSS for Interarea Oscillations* SimPowerSystems Examples, MATLAB R2011a

[19] *Generic Power System Stabilizer* Documentation distributed with MATLAB R2014a Available online at http://www.mathworks.se/help/physmod/sps/powersys/ref/genericpowersystemstabilizer.html