Implementation of PLC based software prototype for 45.6 MHz, 100 kW, ICRH DAC using EPICS control system

#Krupa Mehta, Ramesh Joshi[1], S V Kulkarni[1] and Bhavesh Soni[2]
Institute for Plasma Research[1]
#PG student, U V Patel College of Engineering[2]

Abstract— High power Ion Cyclotron Resonance Heating(ICRH) system is an integral part of future fusion reactors and one of the prominent auxillary heating systems utilized for tokamak heating experiment.It is meant for heating the plasma to certain keV of specified pulse length. ICRH-RF generator provides RF power to the plasma during specified pulse lengths.To monitor and control the signals of the the 2kW, 20 kW and 100 kW amplifier stages of RF generator, PLC based ICRH Data Acquisition Control System (DAC) prototype has been designed. As per the requirement of DAC, it needs 32 analog input, 32 digital input, 16 digital output and 16 analog output channels for monitoring and control of the system using Experimental Physics and Industrial control system (EPICS). PLC provides MODBUS interface over Ethernet for communication with computer. MODBUS provides memory maps for read and write coils for digital Input/output (I/O) interaction and 16 bit registers communication to read and write analog registers as required by DAC. EPICS is used as control system synchronization tool with use of channel access layer. Python is used for program logic development and event handling.Control System Studio (CSS) is used for end user interface with channel values which is a user interface framework for EPICS control system.For high speed data acquisition of channel values greater than or equal to 1 kHz, multi-data acquisition card has been interfaced.

Index Terms— EPICS,100kW amplifier stage,ICRH DAC,PLC,Multi- Data Acquisition card,MODBUS,Data monitoring and acquisition,Python.

1 INTRODUCTION

ICRH DAC system software prototype for 100 kW, 45.6 MHz has been designed for RF ICRH experimental activities in tokamak [1] .This system monitors and controls the channel values of 2 kW, 20 kW and 100 kW amplifier stages respectively. ICRH system consists of different power supplies for each amplifier stage. First two stages need single high voltage power supply and 100 kW stage needs four power supplies namely screen grid, filament, plate and control grid. For failsafe operation of the system it needs DAC for control and monitor and synchronization of each RF amplifier stages along with connected power supplies. There are three different requirement of DAC .First one is periodic monitoring of channel values of amplifier stage.Second one is interlock and control of different power supplies connected with each RF amplifier stage and the last requirement is acquisition with the rate of greater than or equal to 1 kHz for RF power generation in pulse mode. PLC based ICRH DAC has been designed using EPICS control system [3]. EPICS is a set of open software tools, libraries and applications developed collaboratively and used worldwide to create distributed soft real-time control systems for scientific instruments and other large scientific experiments[2]. It allows building server applications that interact with the hardware and provide an interface to EPICS clients. MODBUS protocol is used as communication medium between PLC and PC. Python is a script language which is used for logic implementation. Pymodbus library which is an implementation of MODBUS stack in python is used. PyEpics library of python is used for EPICS channel access interaction.

2 High power ICRH system at 45.6MHz, 100 kW

RF generator is in developing phase for ICRH experiments on tokamak in the frequency range of 45.6 MHz. Fig. 1 shows the complete power chain of 100 kW amplifiers which comprises of (A) signal generator, whose output is amplified through (B) low power amplifier and (C) pre-driver 2 kW stage (D) driver

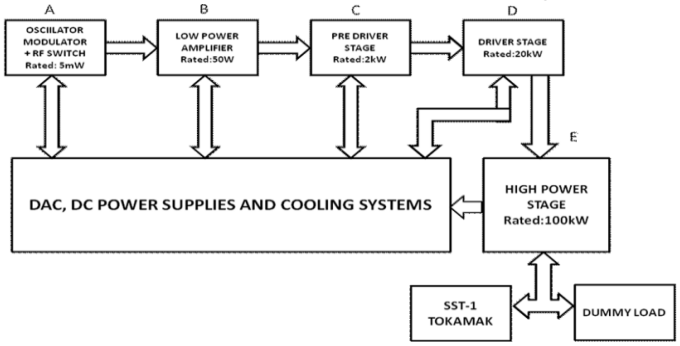

Fig. 1 HIGH Power ICRH 100kW DAC system

20 kW stage and (E) High power 100 kW amplifier stage. It describes connected different DC power supplies rated 4kV-1Amp, 7.5 kV-6Amp & 20KV-25A which feed to amplifier stages C, D & E respectively. Normally in the amplifier stages of 2 kW and above, RF tubes like triodes and tetrodes are used which need many power supplies namely screen grid power supply, control grid power supply, plate power supply and filament power supply. For proper operation of amplifier stages one needs to follow proper switching on, switching off and emergency switching off sequences. The amplifier stages above 2 kW level also needs proper cooling with interlock which should follow certain sequences during operation.

International Journal of Scientific & Engineering Research Volume 6, Issue 5, June-2015
ISSN 2229-5518

PLC based system is used for monitoring and control which has been interfaced with multi data acquisition module for high speed acquisition requirement with two different mechnism one is software interface and another is hardwired trigger.

3 EPICS CONTROL SYSTEM

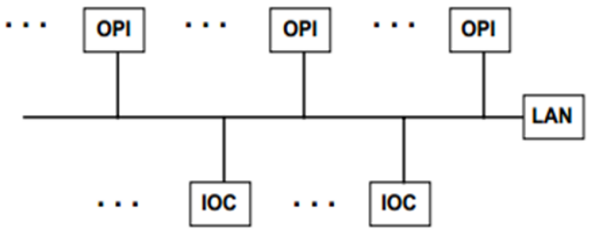

Fig. 2 Physical structure of EPICS control system [5]

As shown in fig. 2 OPI (Operator Interface) is connected to IOC (Input/output Controller) via Ethernet.An OPI is a Windows–based workstation/PC which can run various EPICS clients.IOC is a front-end controller, with various I/O modules for analog and digital signals etc, and for access to field bus such as Ethernet.Channel Access which is an EPICS software, provides network transparent access in between channel access client and server.Each IOC provides a Channel Access server which is willing to establish communication with an arbitrary number of clients.

Best OPI Yet (BOY) of Control System Studio (CSS) as OPI is used which acts as channel access client and IOC is Delta PLC AH500.PLC acts as Channel Access Server. Modbus has been opted as a communication protocol. PLC communicates using MODBUS protocol over TCP which is an application layer messaging protocol.The MODBUS data is simply encapsulated inside a TCP/IP packet. This protocol follows a master and slave architecture, where a master transmits a request which is indicated to a slave and then waits for a response from the slave. Here PLC acts as TCP server (Port 502).There are several MODBUS libraries available to communicate with open source MODBUS protocol with PLCs. Pymodbus library which is a MODBUS implementation in python allows reading of data from PLCs and writing of data to PLC's over Ethernet using specified MODBUS addresses in two way communication [7]. Another library of python PyEpics is used for EPICS channel access interaction [8]. It uses caget and caput to read and write PLC channels.CALab interface contains a softIoc.dbd file which is used to initialize EPICS in the system (Workstation/pc).

4 DAC SYSTEM ARCHITECTURE

As shown in fig. 3 Physical sub-system which includes complete power chain of 100kW amplifier, comprises of signal generator whose output is amplified through amplifier stages 2 kW, 20 kW and 100 kW respectively. After that signals are passed to signal conditioning circuits so that signals meet the requirements for further processing. These signals are provided to PLC and Multi-Data Acquisition Card. Signals are given to PLC for control and monitoring purpose. For acquisition of signals at the rate greater than or equal to 1 KHz signals are passed to DAC card. NI-USB 6343 Multi Data Acquisition card has been procured for the same.

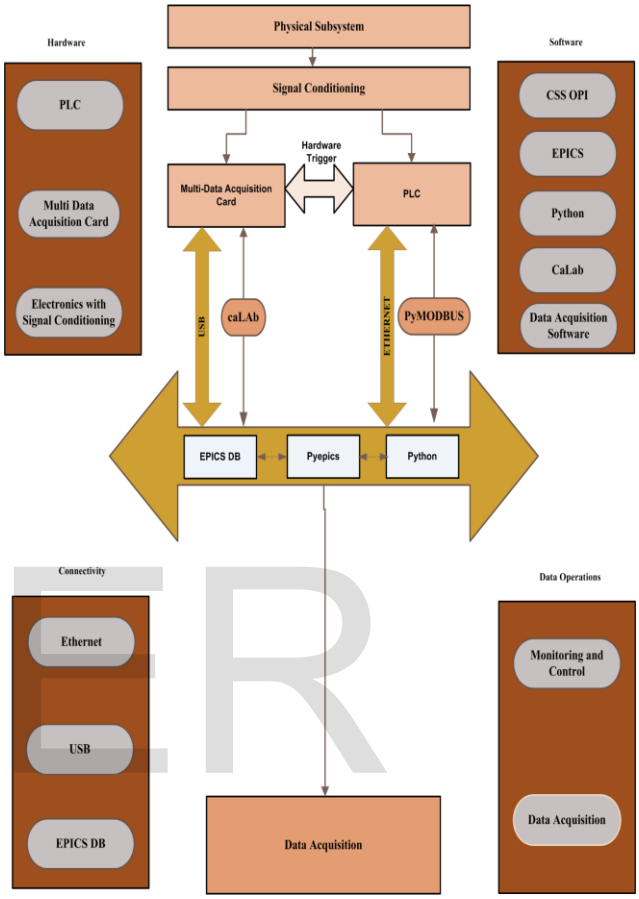

Fig. 3 Architectural details of DAC

PLC communicates with the user interface through Ethernet bus. PLC supports MODBUS protocol over TCP/IP Ethernet to communicate with CSS and Delta ISPSOFT software which resides in computer.CSS is a graphical user interface through which signals can be monitored and controlled. ISPSOFT is software for Delta PLC in which programming is done for feeding pulse to PLC of particular time period for monitoring purpose.PLC and CSS communicate through PYMODBUS library of python. Pymodbus library is implementation of MODBUS stack in python. It provides support to read/write to or from PLC channels.

Multi Data-Acquisition card is connected to computer via USB. It acquires the values of the analog and digital I/O channels. For communicating with EPICS control System CaLAB interface is used[9].CA Lab -library builds an internal PV cache and monitors PVs to improve read and write access and reduce network traffic.To initialize epics base CaLab interface is used. By Hardware trigger PLC channel is connected to Multi Data Acquisition card which acquire PLC channel values

International Journal of Scientific & Engineering Research Volume 6, Issue 5, June-2015
ISSN 2229-5518

which will be further explained in the next section.

CSS comprises of EPICS db, PyEpics and Pymodbus locations so that it can synchronize the system to control and monitor it. EPICS db file consist of process variable names of channel through which CSS controls the channel. Mapping is provided in CSS for PLC's variable and process variable. PLC channel variable are used in CSS to communicate with PLC via PYMODBUS. Process variables are finally available over the network which are termed as shared variables. These shared variables can be accessed by anyone over the network.

5 TEST SETUP

5.1 Test Setup Prerequisites

- PLC and Multi-Data Acquisition Card
- EPICS Version 3.14.12.4
- Python version 2.6
- Python libraries: PYMODBUS 0.5 and PyEpics 3.1.2 Version and numpy and scipy packages
- CSS BOY OPI (3.1.7)
- PC Intel core i5 windows 7 32 bit
- 100 MBPS Ethernet Segment
- USB bus
- Digital Storage Oscilloscope

5.1 Test bed for system prototype

Demo setup is configured for the data acquisition and control system prototype. As per the requirement of DAC,it needs 32 analog input,32 digital input,16 digital output and 16 analog output channels for monitoring and control of the system.PLC has been connected with 230 V which is needed by CPU and 24 V supply to operate PLC connected modules. Using ISP soft software of Delta PLC, Program is designed to configure two output channels namely Y0.2 and Y0.3. One output channel for giving Pulse of specific duration by providing the start and end of the trigger. One output channel for setting the delay, on-off time of the timer. This program is downloaded in the PLC. After that when PLC is connected to Master computer via Ethernet which uses MODBUS/TCP communication protocol and PYMODBUS library. By giving input to the register configured in the program, pulses of specific duration monitor the analog output values of the channels.Analog input values can be set by providing the voltage value in the user interface.For Graphical User Interface Master computer consist of control system studio, where the analog output values are monitored. as well as Digital I/O can be monitored and control in the user Interface created through the BOY OPI of the CSS Package.

Fig 4. shows the configured output channel of Analog Output module of PLC is displayed in Digital Storage Oscilloscope(DSO).As the analog output module works on 24 V,so it is converted to 5 V through this circuit and its phase is inverted.Output channel Y0.2 is configured for trigger pulse at the start and end.Y0.3 channel is configured for number of pulses as per the specified on-time and delay.To set the number of pulses input parameter is provided to registers provided for total time and on-time in the ISP-Soft Program.

For interaction between EPICS and CSS python is used. Fig. 5 shows initialization of EPICS IOC through command prompt which synchronizes PLC with CSS GUI using python. As shown in the GUI according to the python program running in the command prompt, the digital status can be visualize.

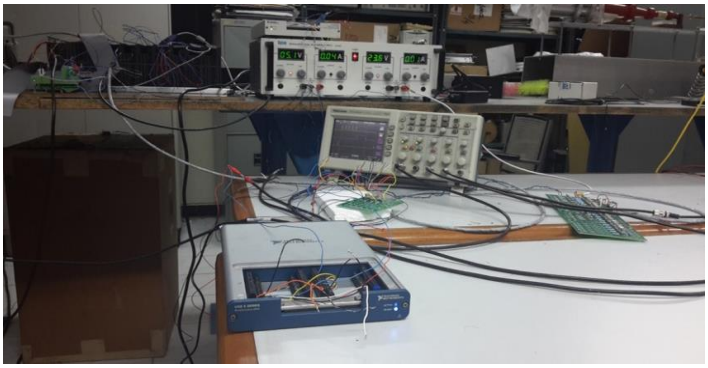

Fig. 4 Test setup

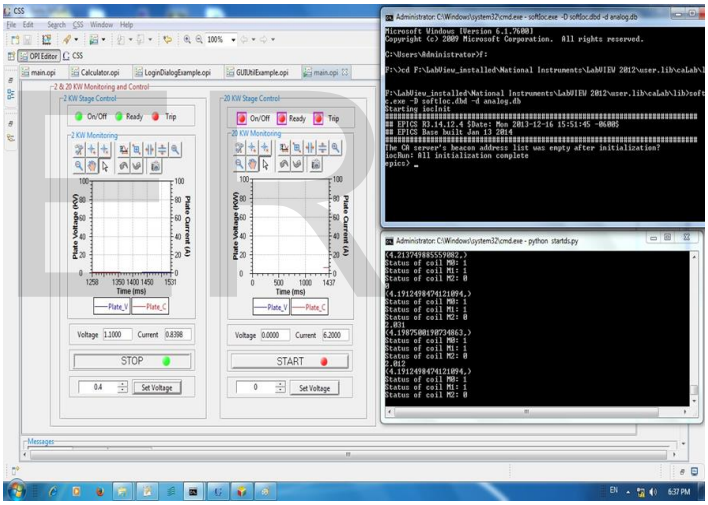

Fig. 5 Interaction between CSS and PLC using EPICS and Python

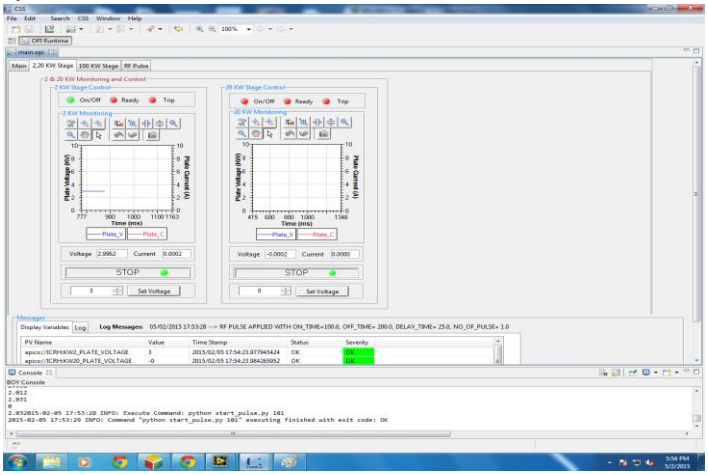

Fig. 6 Control and monitoring of 2 kW and 20 kW amplifier stage

International Journal of Scientific & Engineering Research Volume 6, Issue 5, June-2015
ISSN 2229-5518

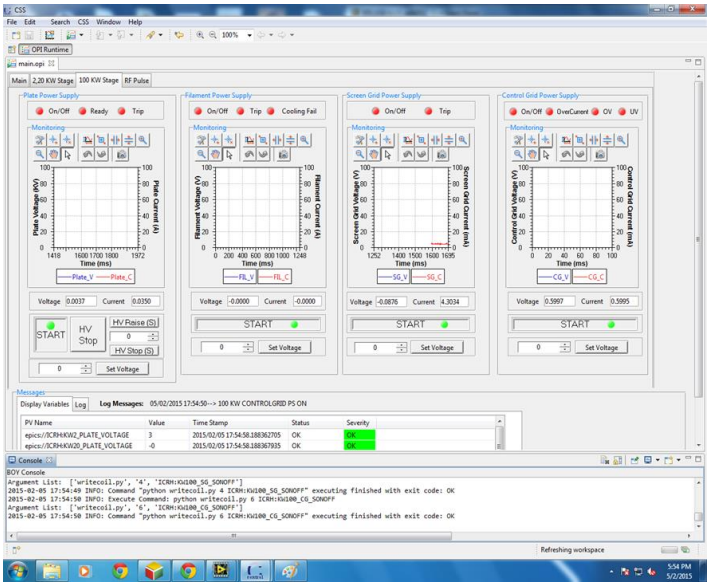

Fig. 7 Control and monitoring of 100kW amplifier stage

Fig 6 and fig. 7 show the digital status of the 2kW and 20kW channels through on/off ready and trip LED'sThese LED's as shown in the fig have been assigned a particular name which resembles PLC's software variables. These variables are declared in the python file, so that PYMODBUS can communicate between CSS and PLC. Pulses of specific duration as configured by the database with CSS program is used to monitor the analog output values of the channels. Analog input values can be set by providing the voltage value in the user interface. For setting the analog output values and for setting the Digital start/stop memory mapping of MODBUS and PLC addresses is required.Digital Input and Analog Input values of PLC are directly taken by PYMODBUS library of python. There is logbook in the bottom which displays the status of the active process variables and its timestamps.

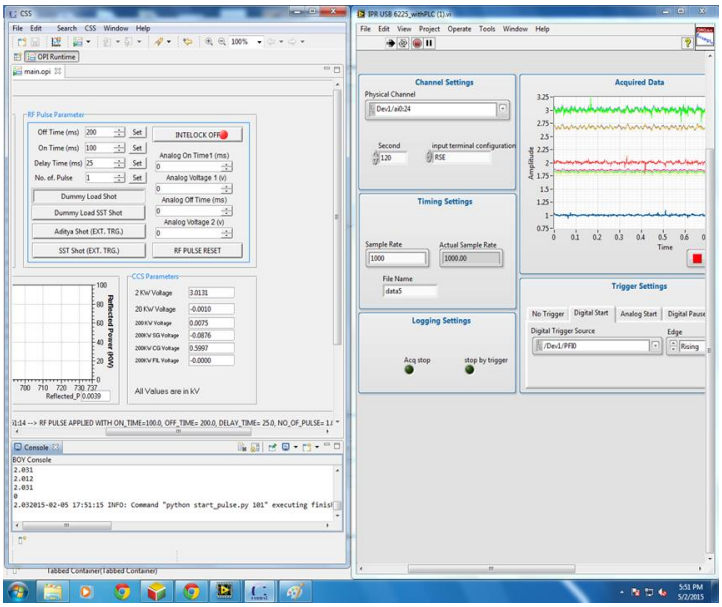

Fig. 8 Acquisition of channel values by trigerring RF pulse

For Acquisition of the channel values at the rate greater than or equal to 1 kHz,RF pulse is trigerred through CSS boy OPI along with required parameters i.e on-time,off-time,delay time and no of pulses. As shown in the fig 8 RF pulse is trigerred which starts acquisition at the rate of 1 kHz in labVIEW with the help of Multi Data Acquisition Card.

4 Conclusion

PLC based software prototype DAC has been designed for high power ICRH system. Demo setup has been successfully deployed with specified hardware as per requirement. PLC had been used for monitoring and control of the channel values of 2 kW, 20 kW and 100kW amplifier stages. CSS BOY OPI had been used as Graphical user interface for monitoring and control due to epics synchronization between CSS and PLC using python. Pymodbus, PyEpics libraries of python and EPICS channel access is used for DAC system operation which provides hardware as well as software independence over network layer. For acquisition at the rate greater than or equal to 1 kHz, Multi-data acquisition card with LabVIEW is interfaced successfully.

Acknowledgment

The authors wish to thank Mr H M Jadav and Mr. Aniruddh Mali for their kind support. This work was carried out under the premisis of prestigious Institute for Plasma Research

References

[1] Engineering design report (EDR) of ICRH-SST1; SST-2.03-050199-RF Group, 2000
[2] http://www.aps.anl.gov/epics/
[3] Ramesh Joshi et al. "Conceptual design of EPICS based implementation for ICRH DAC system", Conference preceding Springer,2013
[4] MODBUS protocol specification http://www.modbus.org/docs/Modbus_Application_Protocol_V1_1b.pdf
[5] Physical structure of EPICS control system: http://www.aps.anl.gov/epics/base/R3-14/12 docs/AppDevGuide/node4.html K. Elissa, "An Overview of Decision Theory," unpublished. (Unplublished manuscript)
[6] Delta AH-500 PLC module reference: http://www.ferret.com.au/ODIN/PDF/Showcases/105517.pdf
[7] PYMODBUS Python Package, https://pypi.python.org/pypi/PYMODBUS
[8] PYEPICS Python Package http://cars.uchicago.edu/software/python/PyEpics3/overview.html
[9] http://www-csr.bessy.de/control/SoftDist/CA_Lab/