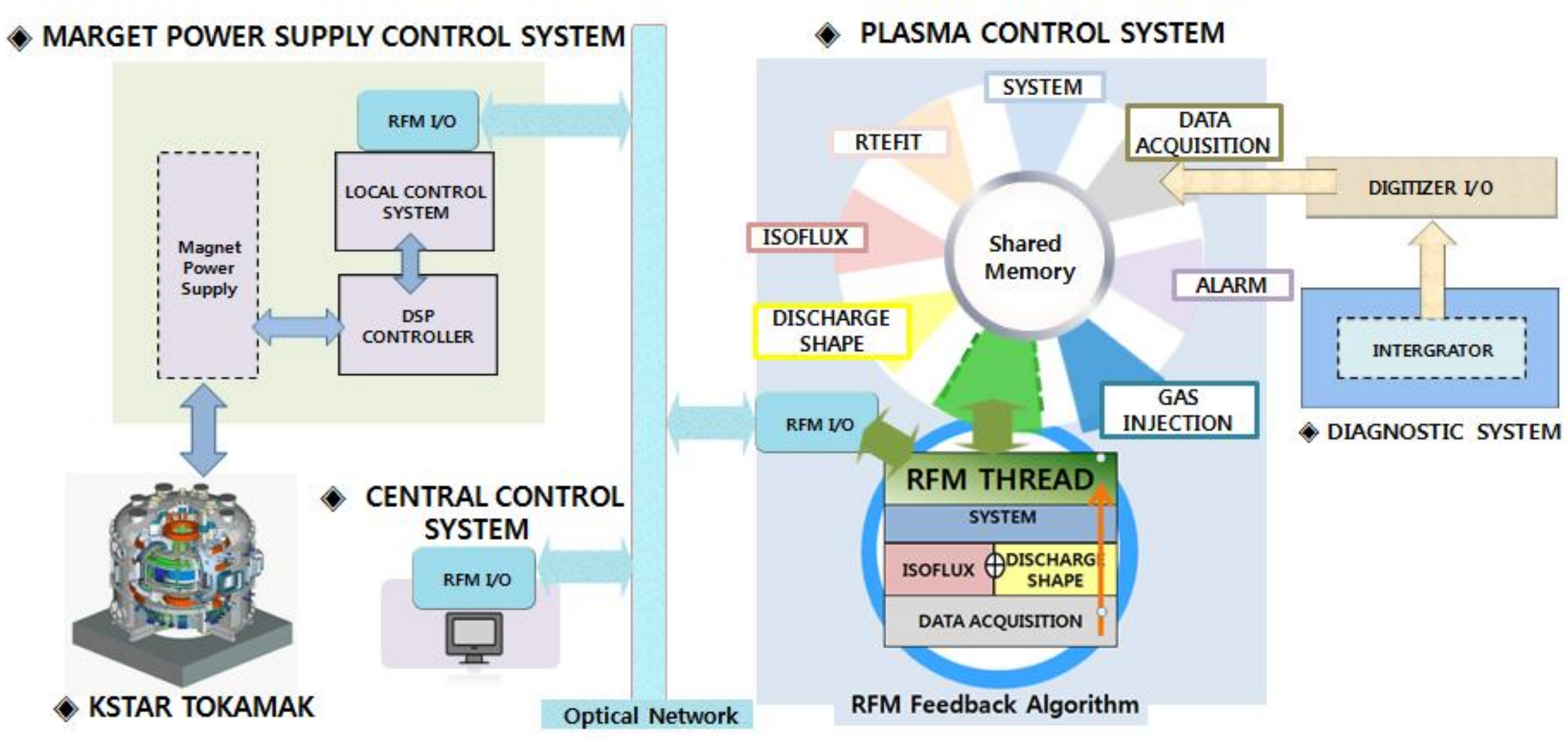
High-Speed data handling using reflective memory thread for tokamak plasma control

Song-Yeol Park[a], B.G. Penaflor[b], R.D. Johnson[b], D.A. Piglowski[b], M.L Walker[b], S.H. Hahn[a], and W.C. Kim[a]
[a] National Fusion Research Institute (NFRI), Daejeon, Republic of Korea
[b] General Atomics, P.O. Box 85608, San Diego, CA 92186-5608, USA
* Author's email address: sypark@nfri.re.kr

Abstract

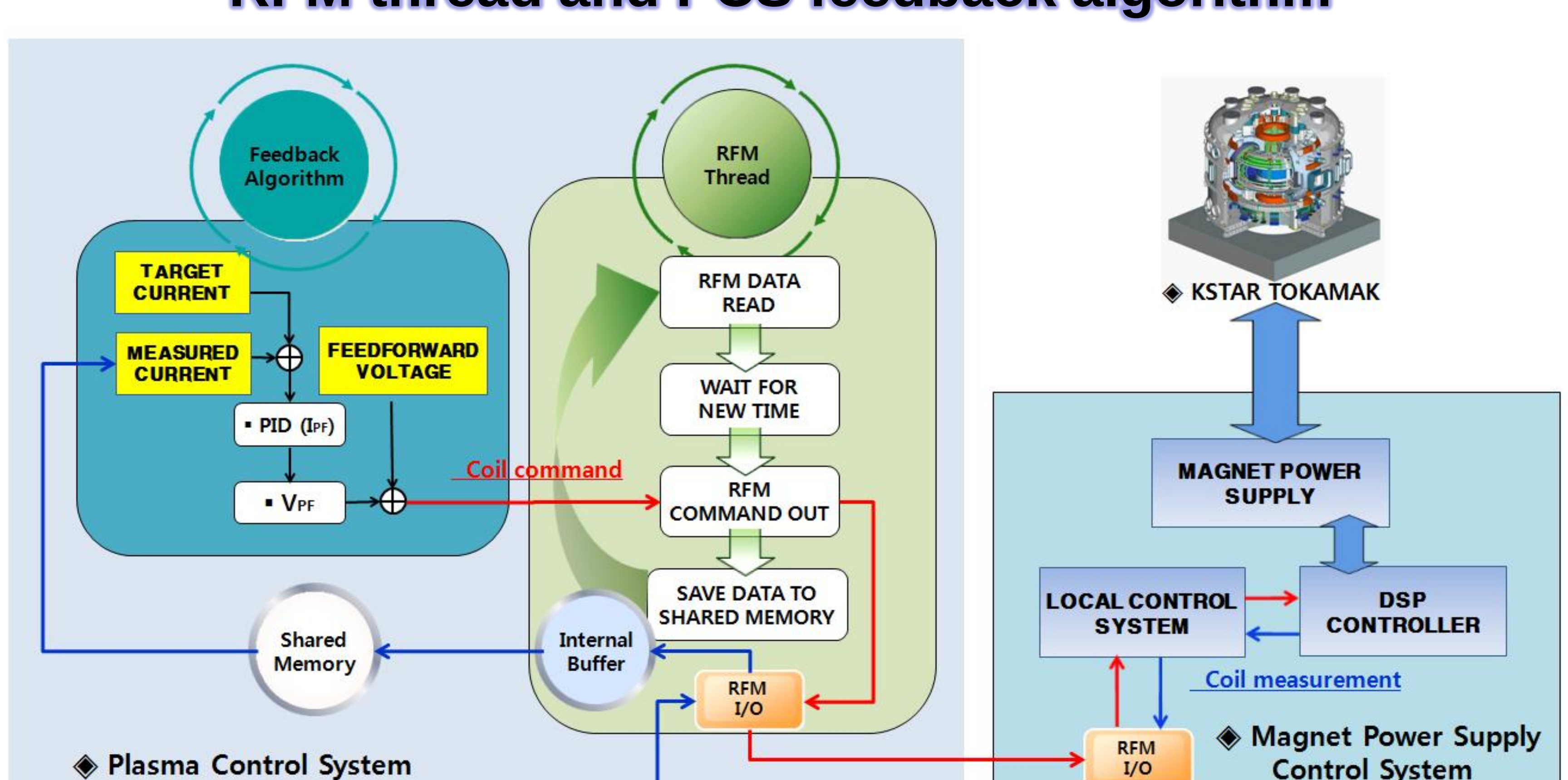
The Korea Superconducting Tokamak Advanced Research (KSTAR) plasma control system (PCS) is defined as a system consisting of electronic devices and control software, which identifies and diagnoses various plasma parameters and calculates appropriate control signals to keep the plasma sustained in the KSTAR operation regime. The KSTAR PCS consists of a linux system with 8 processors and both analog and digital data acquisition methods are adapted for fast real-time data acquisition. The digital interface uses a reflective memory (RFM) technology to enable PCS to interface with the actuators in real-time and to share data among various subsystems of KSTAR and to do interprocessor communications inside the cluster. In the KSTAR PCS, the communication using the RFM is implemented as a thread to handle the fast control of the RFM data transfer

Introduction to the plasma control system



- Composed of real-time computers & diagnostic system & communication interface
- Acquires plasma data
- Performs a feedback control algorithm
- Sends commands to the MPS and receives the measurements from the MPS
- Has an optical network interface consisting of Reflective Memory (RFM)
  - Manufacturer : General Electric company
  - Low latency and wide throughput
  - Onboard circuitry automatically transfer performs the transfer to all nodes
  - Transfer rate : 174 MB/s [factory], 96 MB/s [interrupt-free mode]

RFM thread and PCS feedback algorithm



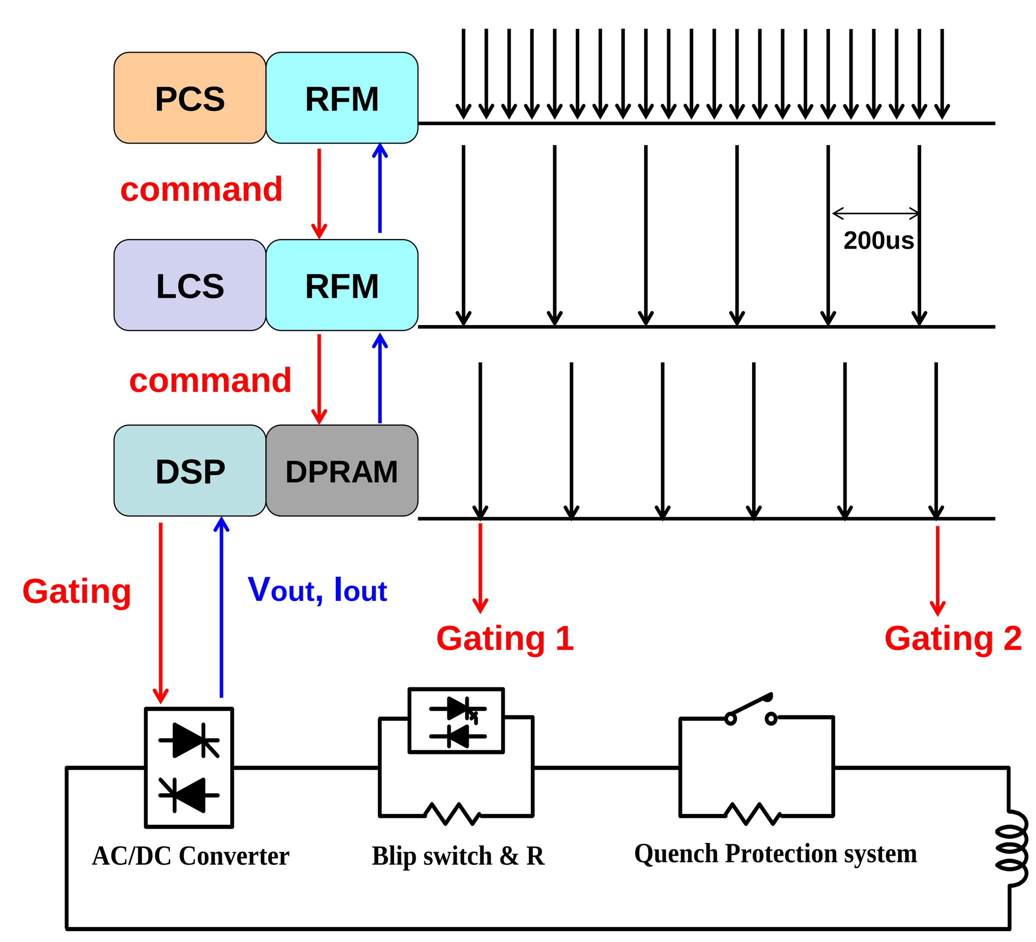
The RFM thread

- Child process of one of real-time processes
- Synchronizes itself with real-time control cycle
- Copies the RFM data to a shared memory area at every cycle
- Reads PF coil currents and voltages from a fixed RFM area
- Waits for "new time"
- Writes the PF commands to the fixed RFM area using DMA transfer

The PCS feedback algorithm

- Calculates feedback error
- Decides the next PF commands
- Send the commands to the RFM thread
- Considering the delay of the MPS interface system

Function of the Interface controller

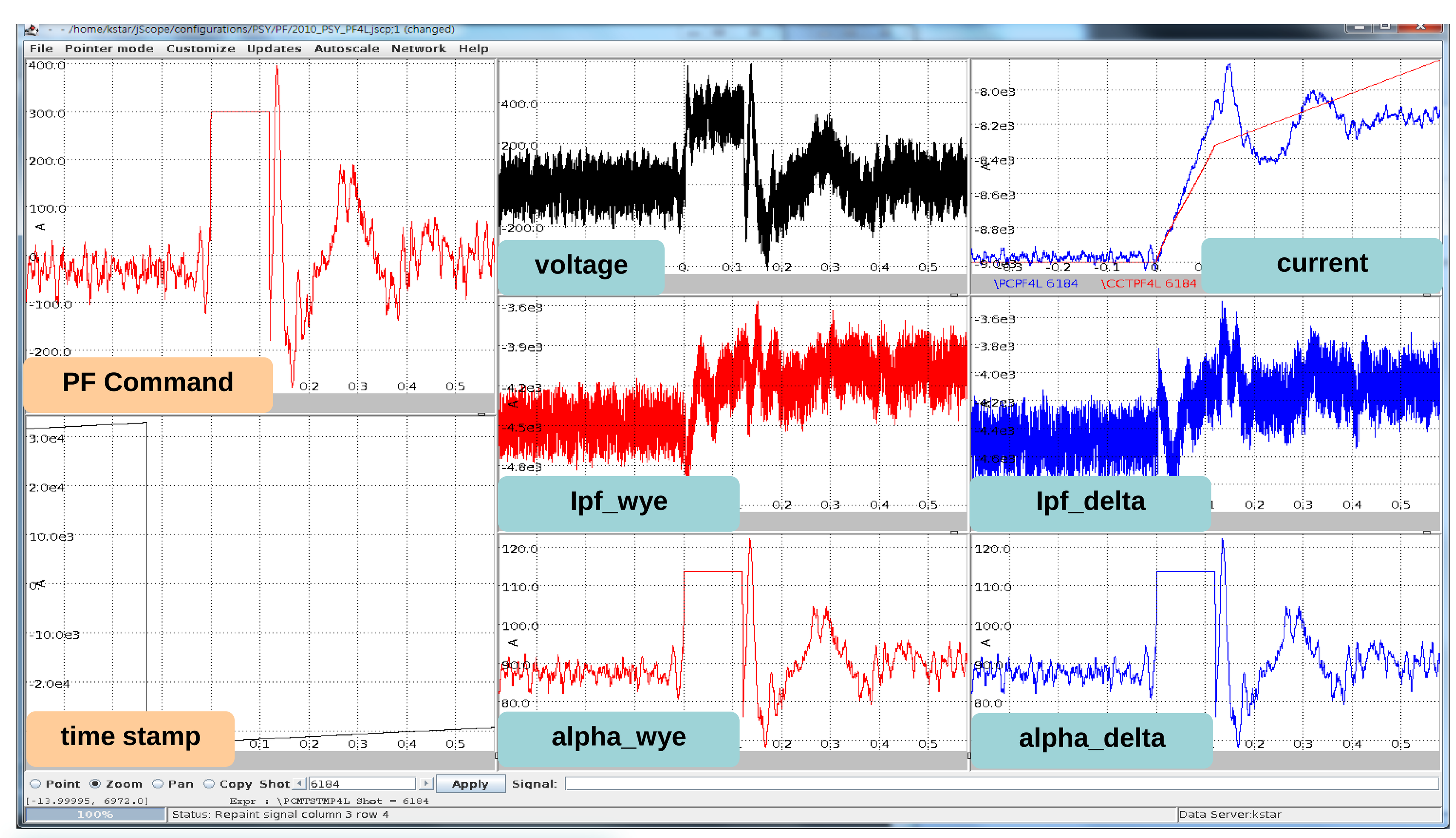


- PCS
  - control cycle : 50us
  - writes commands
  - reads measurements
- LCS
  - control cycle : 200us
  - path between PCS and DSP
- DSP
  - control cycle : 200us
  - reads commands
  - writes measurements

◈ Minimum Feedback cycle : 500us

◈ Gating cycle : 0.69ms ~ 2.78ms

Actual MPS control result



◈ 12 pulse thyristor converter

Command & Data Structure

◈ Command Structure

Var.name	Description
control_method	command type
current_direction	charging direction
timestamp	used as a software watchdog
rt_mode	real-time mode
PF_command	PCS command
current_traj	current trajectory

◈ Data Structure

Var.name	Description
voltage	total output voltage
current	total output current
Ipf_wye	Y-converter output current
Ipf_delta	D-converter output current
alpha_wye	Y-converter alpha degree
alpha_delta	D-converter alpha degree

Issues & Future plan

Issues

- The time counter used in the RFM operation is acquired from the digitizer, hence the clock source of the digitizer should be accurate
- Although the data processing speed of the RFM thread is fast, if the MPS interface system doesn't operate as we expected, there is a possibility that the RFM thread receives the same data during a couple of cycles
- The RFM data written by two different devices is not exactly synchronized in time Nevertheless, the assumption is true in most cases that the read is acceptable as the most recent one if the read cycle is faster than each writing cycle by the actuators.

Future plan

- Increase the amount of data through the RFM network
- Upgrade the RFM memory card to the PCI-express bus format
- We have plan to improve the algorithm of the RFM thread to safely handle other possible unintended situations