A Field Monitoring System of the LANSCE DTL Systems

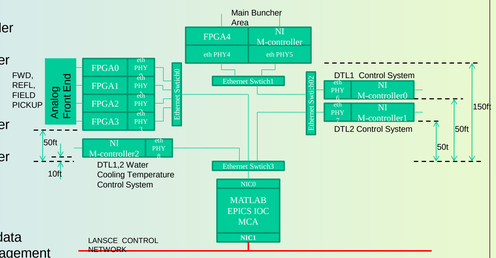
Sungil Kwon, L. Castellano, M. Prokop, and P.Torrez
LANL, Los Alamos, NM 87545, U.S.A.

Summary

In this note, we address the data monitoring system of the LANSCE main buncher and 100MeV DTL sector. The system consists of 5 FPGA digital receiver systems, 4 PXI based data acquisition systems. FPGA based digital receiver systems process the oversampled cavity field IF signals, forward/reflect IF signals, and yield base band signals. Three of the 4 PXI based data acquisition systems obtain video signals of the current analog low level RF systems for DTL1, 2 and main buncher, and the 4th PXI based data acquisition system obtains the temperature control system signals, slug-tuner resonance control systems signals. Since each channel has a few thousand samples per 800 usec RF pulse, channel signals of each system are packed before transmitted. Using Ethernet, data from data acquisition systems are transmitted to a local host computer which is equipped with EPICS softIOC. In the local host computer, a MATLAB application processes data, and upon the request from the user, it sends a frame data for a RF pulse or average data, or sample data at a specific instant of time through LANSCE control system network. By retrieving data, the analysis of the cavities, high power systems, and resonance control systems is performed. In the mentioned system, softIOC is implemented in the local host computer. An effort has been performed to embed the softIOC on the custom hardware. As a initial step, an operating system, RTEMS running on NIOS II softcore processor of Altera was tried. Based on this, softIOC will be ported to RTEMS on NIOS II processor.

System Overview

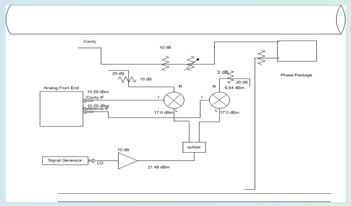
1. Four FPGA chasses that process FWD, REFL, and Cavity Field IFs of DTL Module 1 and 2.
2. One NI PXI Chassis with an embedded controller for DTL1 Video Signal
3. One NI PXI Chassis with an embedded controller for DTL2 Video Signal
4. One FPGA Chassis that processes FWD, REFL and Cavity Field IFs of Main Buncher
5. One NI PXI Chassis with an embedded controller for Main Buncher Video Signal
6. One NI PXI Chassis with an embedded controller for Water Temperature Monitoring and Slugger tuner signal Monitor
7. One Windows PC used for MCA, SOFT IOC

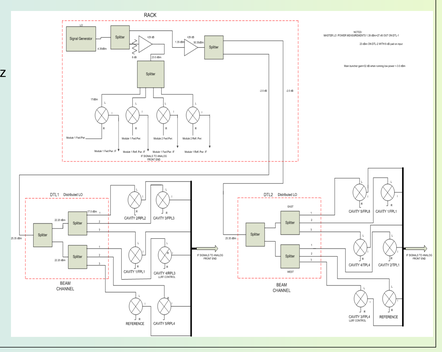


The data obtained from the FPGA boards and NI data acquisition systems is transferred to the data management system in which MATLAB application builds the Ethernet links to the FPGA boards and NI data acquisition board. The data management system is equipped with the EPICS SOFT IOC and MATLAB channel access (MCA). In the MATLAB environment, MCA passes the data received from the FPGA boards and NI data acquisition systems to the SOFT IOC and SOFT IOC broadcasts data outside the system using the network.

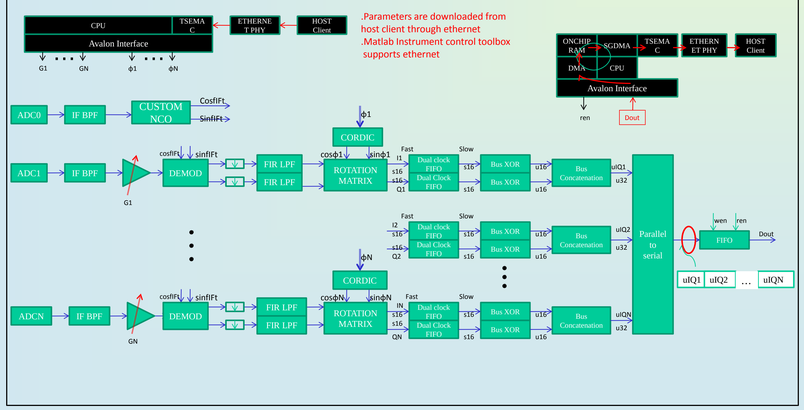
Analog Front End for inputs and outputs of FPGAs

The analog front end receives IF signals from the mixers located inside the beam tunnel. The mixer outputs come into the analog Front end and they are filtered using 25 MHz bandpass filters and Amplified for the further processing in the digital receiver on the FPGA.





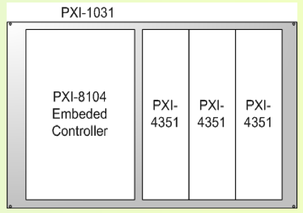
Data Processing in a FPGA



In FPGA, 14bit ADCs are equipped. The ADC clocks are generated from the FPGA using PLL. The ADCs clock is 100.625 MHz which is 4 times of the IF input frequency. This setting of the ADC clock makes the FPGA signal processing easy, because the FPGA clock is 100.626 Hz (which is 4 times of the IF frequency). The first ADC is used to digitize the reference IF. There ADCs are used for cavity Ifs or fwd IF, reflected IF. Since the signal path length is different, it is recommended to downconvert the input IF signals using reference IF signal. The digitized reference is conditioned so that the processing yields the cosine IF sequence (1, 0, -1,0, 1,0, -1,0…) and the sine IF sequence (0,1,0,-1,0,1,0,-1,…). Using these cosine and sine IF sequences, the base band signals whose phases are obtained with respect to the reference IF. Downconverted signals are processed through the low pass FIR filters. The cutoff frequency of the FIR filters is determined so that they cut off the harmonics of the IF frequency. The FPGA clock is 100.625 MHz and so for a 1 milli-second RF pulse example case, about 100 K data samples are obtained. This high rate data samples are not necessary for the control or diagnostic purpose. The high rate data samples are processed in the dual clock FIFO where the input write clock is the slow, programmed PLL clock obtained from the FPGA clock but the output read clock rate is that of the FPGA clock. The slow clock rate is programmable and currently it is 100. With this rate, the 100 K data samples are reduced to 1K data samples. The dual clock FIFO output is read at the FPGA clock rate and the time domain compression is 100, which means that the data samples corresponding to 1 milli-second RF pulse is downsampled to 1 K data samples of 10 microsecond duration in the domain of the FPGA clock. This process is applied to all 3 pairs of I and Q signals. The I and Q signals are eventually transferred to the external host computer using ethernet. The Ethernet Media Access Controller (MAC) implemented in the FPGA is triple speed Ethernet media access controller(TSE MAC). THE TSE MAC core includes the 32 bit FIFO. In order to utilize this FIFO, the 16 bit signed integer I and Q signals are merged as the unsigned 32 bit data. First 16 bit signed integer I and 16 bit signed integer Q are converted to 16 it unsigned integers by bit-XOR (BUS XOR) and then the two 16 bit unsigned integers are merged to 32 it unsigned integer by BUS builder. Note that for the convenient future data reconstruction process, This process is applied to three pairs of I and Q signals. The three parallel 32 bit unsigned integer signals are merged into a long sequence of 32 bit data samples. This is performed by enabling the second dual clock FIFO reading by the first dual clock FIFO writing done signal and the third dual clock FIFO reading by the second dual clock FIFO writing done. The XOR of three read enables of the FIFO yield a long logic signal, which is used for the single clock FIFO write enable signal. The addition of the three Dual Clock FIFO output data is the merged 32 bit data samples and this data is stored into a single clock FIFO. Altera DMA supports the data transfers from peripheral to memory, from memory to peripheral, and from memory to memory. In the SOPC builder system design, the Avalon slave read interface to the single clock FIFO output of the DSP builder design is treated as a peripheral. So, the memory mapped register address of that Avalon slave read interface becomes the source of the DMA data transfer. The destination of the DMA data transfer is either on-chip memory or the external memory of the SOPC builder system design. This defines the write address of the DMA controller write address register. Here, a part of the on-chip memory, named packet_memory in the SOPC builder system design is the DMA data transfer destination.

As soon as the DMA data transfer is completed, the data in the memory is ready to be sent via Ethernet using udp or tcp/ip. In this project, udp protocol is selected. For the Ethernet based data transmit, SGDMA (The Scatter-Gather Direct Memory Access) is used, where the destination of the SGDMA is the Ethernet MAC.

National Instrument PXI BasedData Acquisition System



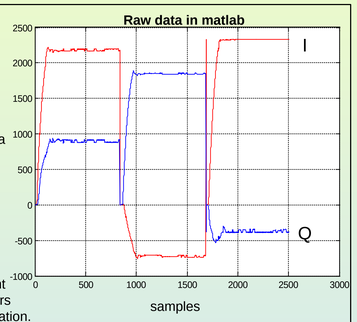
The data acqusition system consists of [1]
1. PXI-1031 : 4 slot 3U XI Chassis with Universal AC
2. PXI-8104 Embedded Controller : 1.8 GHz Celeron M440 Embedded Controller. 10/100/100BASE –TX(Gigabit) Ethernet is supported and Windows OS and drivers are installed
3. PXI-4351 : High-precision temperature and voltage logger. 16 differential Voltage inputs are available. 24bit ADC resolution is provided.

There are fast data acqusition system and slow data acqusition system. The fast data acquisition system digitizes the current LANSCE low level RF system signals of two DTL s which are outputs of the RF detect diodes and phase detector. It also acquires the controller inputs and outputs signals. In the embedded controller, a Labview model receives the channel data from voltage logger and formats the channels data in series to fit the data format for gigabit ethernet. The data is unicast to the local Host Computer to serve the data request.

The slow data acqusition system digitizes the data of the cavity cooling water control system and the resonance frequency control system (slugger tuner system). In the embedded controller, a Labview model receives the channel data from voltage logger and formats the channels data in series to fit the data format for gigabit ethernet. The data is unicast to the local Host Computer to serve the data request.

The sampling rate of PXI-4351 is programmable (10, 50 60) and 16 inputs are simultaneously sampled and so the channel speed is sample rate/number of channels acquired.

MATLAB Channel Access (MCA)



Since data of different feature and different sources are processed, our field Monitoring System centralizes the data in the dedicated local Host Computer and run the (computational intensive) data process and cast the results. MATLAB application in the dedicated local Host Computer builds the network connections to FPGA boards and PXI data acqusition systems. Since each data source sends a few data samples and only one PHY is assigned for data receiving, a polling scan applied. Each Data samples contains its source ID and appropriate data processing in the MATLAB application is determined based on the ID.

In order to interface the MATLAB application to EPICS, MATLAB channel access (MCA) is used. Channel Access (CA) is one of the core components of an EPICS system. It is the software component that allows a CA client application to access control system data which may be located on different hosts throughout a network. It is based on a client-server model and the client makes requests and sends them to the server. Often, there are several Servers and running on different IOCs and the client application is running on a workstation. A client application uses function calls from the client library and using these calls, a client builds a virtual circuit (channel) between itself and a process variable (PV). The server which is located on the same host as the PV returns a channel ID. Base on the channel ID, the client can request further services such as returning PV's value, change PV, etc to the server and the server fulfills the tasks [2].

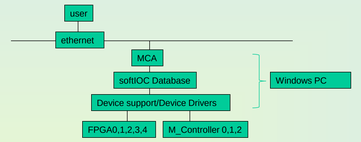
The MATLAB Channel Access(MCA) implements most of the functionality of CA client library of EPICS in a small number of MATLAB functions [3]. These functions can be called from the MATLAB command line or used in a program written in MATLAB language. Values to be read from or to be written to process variables (PV's) are variables in the MATLAB workspace. They are immediately available for use in calculations or plotting in MATLAB in the local Host Computer for development. Or they are accessed by the end users via installed soft IOC (Input Output Controller) running on hosts plugged into the LANSCE control system(LCS) network.

LANSCE CONTROL SYSTEM (LCS) Network

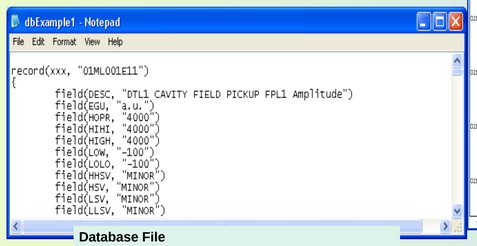
This is the communication network which allows the IOCs and OPIs to communicate. Hardware for LAN Ethernet and the communication protocol is TCP/IP via sockets. UDP is also supported.

EPICS softIOC(Host based IOC)

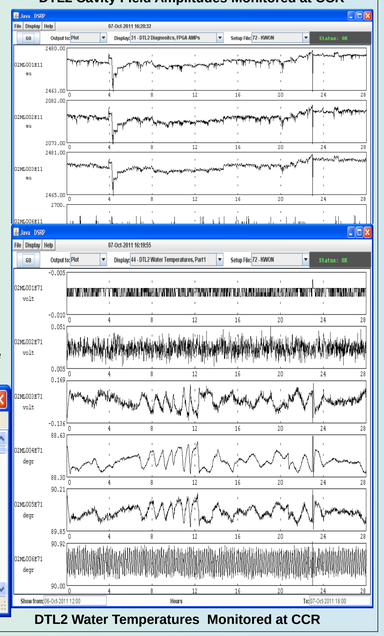
An IOC(Input Output Controller) performs input/output operations to attached hardware devices, associates the values of EPICS PVs (process variables) with the results of these input/output operations, also can perform sequencing operations, closed-loop control and other computations [4]. The soft-IOC is different from a traditional IOC, in that it is implemented in software while conventional IOCs are embedded in the VME CPU or other CPUs. In our case, softIOC is installed in a rack mounted windows PC.



The heart of an IOC is a memory resident database together with various memory resident structures describing the contents of the database. EPICS supports a large and extensible set of record types, e.g., ai (analog input), ao(analog output), etc. Each record type has a fixed set of fields. Some fields are common to all record types and others are specific to particular record types. Every record has a record name and every field has a field name. The first field of every database record holds the record name, which must be unique across all IOCs that are attached to the same TCP/IP subnet. Figure shown below shows the database file for our system. In the figure below, 'xxx' means softIOC, and '01ML001E11' is the record name which should not be over-assigned. The right-side figure shows the plot of the contents of the database accessed via CA at the user workstation.



Database File



RTEMS on NIOS II SOFTCORE PROCESSOR

Current system implements the softIOC on the windows PC. If the IOC is integrated to our Altera FPGA board, it gives us great flexibility and operation capability. For this, it is necessary to choose a proper operating system running on the NIOS II softcore Processor. RTEMS [5] is the on the top of the operating system but the support from NIOS II processor is very limited. The current status is that the user application can be running on the RTEMS on NIOS II on the Altera FPGA stratix III 3SL150. The following shows the raw ethernet application between the FPGA board and a linux machine.

RTEMS on NIOS II

```
[root@pn1215337 TSEMAC]# nios2-download -g o-optimize/*.exe |nios2-terminal
nios2-terminal: connected to hardware target using JTAG UART on cable
nios2-terminal: "USB-Blaster [USB 6-1.2]", device 1, instance 0
nios2-terminal: (Use the IDE stop button or Ctrl-C to terminate)
Input data 0Xf, 15
Auto negotiation is completed (Phy Status Register): 0x796d
Rec. Data 0: 0x 0
Rec. Data 1: 0x 1
Rec. Data 2: 0x 2
Rec. Data 3: 0x 3
Rec. Data 4: 0x 4
Rec. Data 5: 0x 5
Rec. Data 6: 0x 6
Rec. Data 7: 0x 7
Rec. Data 8: 0x 8
Rec. Data 9: 0x 9

Rec. Data 0: 0x 1
Rec. Data 1: 0x 2
```

Linux Machine

```
[root@pn1215337 TSEMAC]# ./test_send
Socket creation success.
458617552
data 0x0
data 0x1
data 0x2
data 0x3
data 0x4
data 0x5
data 0x6
data 0x7
data 0x8
data 0x9

frame 0x0
Send success (526).
data 0x1
data 0x2
data 0x3
```

```
[root@pn1215337 TSEMAC]# nios2-download -g o-optimize/*.exe |nios2-terminal
nios2-terminal: connected to hardware target using JTAG UART on cable
nios2-terminal: "USB-Blaster [USB 6-1.2]", device 1, instance 0
nios2-terminal: (Use the IDE stop button or Ctrl-C to terminate)
Input data 0Xf, 15
Auto negotiation is completed (Phy Status Register): 0x796d
MAC post-initialization: CMD_CONFIGURE=0x400020b
The Number of Valid unicast frames transmitted to 00-07-ED-FF-49-00: 0X00000000
The Number of Valid unicast frames transmitted to 00-07-ED-FF-49-00: 0X00000001
The Number of Valid unicast frames transmitted to 00-07-ED-FF-49-00: 0X00000002
The Number of Valid unicast frames transmitted to 00-07-ED-FF-49-00: 0X00000003
The Number of Valid unicast frames transmitted to 00-07-ED-FF-49-00: 0X00000004
The Number of Valid unicast frames transmitted to 00-07-ED-FF-49-00: 0X00000005
The Number of Valid unicast frames transmitted to 00-07-ED-FF-49-00: 0X00000006
```

```
[root@pn1215337 TSEMAC]# tcpdump -x -i eth0
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
09:02:35.526003 arp-#2570 for proto #772 (6) hardware #258 (5)
0x0000: 0102 0304 0506 0a0a 0a0a 0a0a 0a0a 0a0a
0x0010: 0a0a 0a0a 0a0a 0a0a 0a0a 0a0a 0a0a 0a0a
0x0020: 0a0a 0a0a 0a0a 0a0a 0a0a 0a0a 0a0a 0a0a
0x0030: 0a0a 0a0a 0a0a 0a0a 0a0a 0a0a 0a0a 0a0a
0x0040: 0a0a 0a0a 0a0a 0a0a 0a0a 0a0a 0a0a 0a0a
0x0050: 0a0a
09:02:35.733708 arp-#2827 for proto #772 (6) hardware #258 (5)
0x0000: 0102 0304 0506 0b0b 0b0b 0b0b 0b0b 0b0b
0x0010: 0b0b 0b0b 0b0b 0b0b 0b0b 0b0b 0b0b 0b0b
```

Conclusion and Future Works

In this note, we build the Data monitoring system of LANSCE Front End LINAC Systems. We implement EPICS softIOC and cast data to LANSCE control system network.
In addition, an effort has been done for running a real time operating system RTEMS on the NIOS II softcore of Altera.
A future work is to embed EPICS IOC on RTEMS running on NIOS II processor

References
[1] http://www.ni.com/
[2] J. Hill, EPICS R3.14 Channel Access Reference Manual.
[3] Andrei Terebilo, "CHANNEL ACCESS CLIENT TOOLBOX FOR MATLAB"
[4] M.R. Kraimer, J. Anderson, A. Johnson, E. Norum, J. Hill, R. Lange, EPICS: Input/Output Controller Application Developer's Guide
[5] http://www.rtems.org