SYSTEM LEVEL IMPLEMENTATION OF BEAM POSITION MONITORS WITH LOCAL DATA PROCESSING CAPABILITY FOR THE CORNELL ELECTRON STORAGE RING*

M.A. Palmer†, J.A. Dobbins, B.Y. Rock, C.R. Strohman, Cornell University, Ithaca, NY 14853, USA
J.R. Moffitt, College of Wooster, Wooster, OH 44691, USA

Abstract

We describe the architecture and system-level implementation of a bunch-by-bunch beam position monitoring system for the Cornell Electron Storage Ring (CESR). With this system an individual readout module is supplied for each detector. Each readout module is equipped with its own digital signal processor (DSP) which provides automatic gain and timing control for the acquisition of signals corresponding to particular bunches in the ring. The DSPs also provide local data processing for applications such as multi-turn averaged position information, betatron phase measurements, and bunch-by-bunch tune measurements.

INTRODUCTION

An upgraded beam position monitoring (BPM) system, with bunch-by-bunch readout capability, has recently been installed in one sector of CESR. Electron and positron beams in CESR occupy the same beampipe and are electrostatically separated into different *pretzel* closed orbits so that collisions occur at a single interaction point. Typical operation is with 9 trains of 4 or 5 bunches in each beam and with a bunch spacing of 14 ns within each train. In order to perform the necessary timing and gain control functions to probe individual bunches at each BPM location, each module is equipped with its own DSP. The presence of the DSP also allows us to perform a number of high level monitoring functions in the readout modules.

Figure 1 illustrates how the BPM modules are connected to the CESR Control System [1] and Timing System [2]. A brief description of operation and performance issues pertinent to the BPM system is presented here.

CONTROL SYSTEM INTERFACE

All command and data transfers to or from a BPM module are initiated from programs running on one of the high-level OpenVMS Alpha Workstations. Data is only moved when it is requested; it cannot be independently pushed up from a BPM module. A program constructs a request packet in the MPM (Multi-Port Memory) system which specifies the desired BPM and either the source (when writing) or the destination (when reading) of the data. Up to 256 32-bit words can be transferred per request. The request packet identity is passed off to the appropriate XBUS processor. The XBUS is a proprietary fieldbus, which transfers data in a byte-serial format (*i.e.*, it is a byte-wide parallel bus with multiple bytes per data transfer).

* Work supported by the National Science Foundation
† map36@cornell.edu

The XBUS processors are I/O computers which transfer data between the MPM and the accelerator hardware, in this case a BPM module. The XBUS processor receives the request packet and looks up the address of the BPM module. The address specifies the crate and card slot of an SXIO (SERIAL XBUS I/O) card, the channel on the SXIO card where the BPM module is connected, and a register on that channel. BPM modules are connected to SXIO cards with a 1-bit, half-duplex, serial connection. The SERIAL XBUS protocol requires two 16-bit data transfers, of approximately $25\ \mu\mathrm{sec}$ each, to move each 32-bit data word to or from the DSP. This translates to a 10K word/sec bandwidth limit which is the impetus to place as much data processing capability as possible within the BPM modules.

TIMING SYSTEM INTERFACE

Figure 1 also illustrates how the BPM modules are connected to the CESR precision timing system. An interface card in the timing system distributes a 24 MHz TTL clock signal by way of several coaxial distribution cables around the accelerator. Near each BPM module is a timing cable tap designed to minimize any impedance discontinuity on the coax cable. The circuitry inside each tap converts the TTL clock signal into an LVDS signal which is sent to the BPM module on a twisted-pair cable.

The timing system interface card encodes several pieces of data on the clock signal by means of pulse-width variations. The rising edges of the clocks occur every 42 ns (24 MHz). A data bit of *zero* is encoded by having the falling edge occur 14 ns after the rising edge. A data bit of *one* has the falling edge occurring 28 ns after the rising edge. To keep the DC content of the clock signal constant and to allow for synchronization and error checking, each bit of data is sent normally in one clock period and its complement in the next clock period.

The revolution time of CESR is about $2.56\ \mu\mathrm{sec}$. This is equal to 61 periods of the 24 MHz clock. These 61 periods provide for 30.5 bits of data. The first 2.5 bits are used for synchronization and as a turn marker. These bits are encoded as the *illegal* binary data stream of 11101 in 5 clock cycles. Upon receipt of this stream, a turn marker is generated for use by the internal timing system of the BPM module and the remaining 28 bits of BPM data are latched in a register. The first 18 bits are broken into 9 bits of horizontal and 9 bits of vertical shaker phase data. These phase "words" allow betatron phase measurements to be made with the system (see below). The next 8 bits are used as a system-wide command byte for the BPM system. This byte allows us to synchronize operations of all the BPM modules. Typically a control system program will sequen-

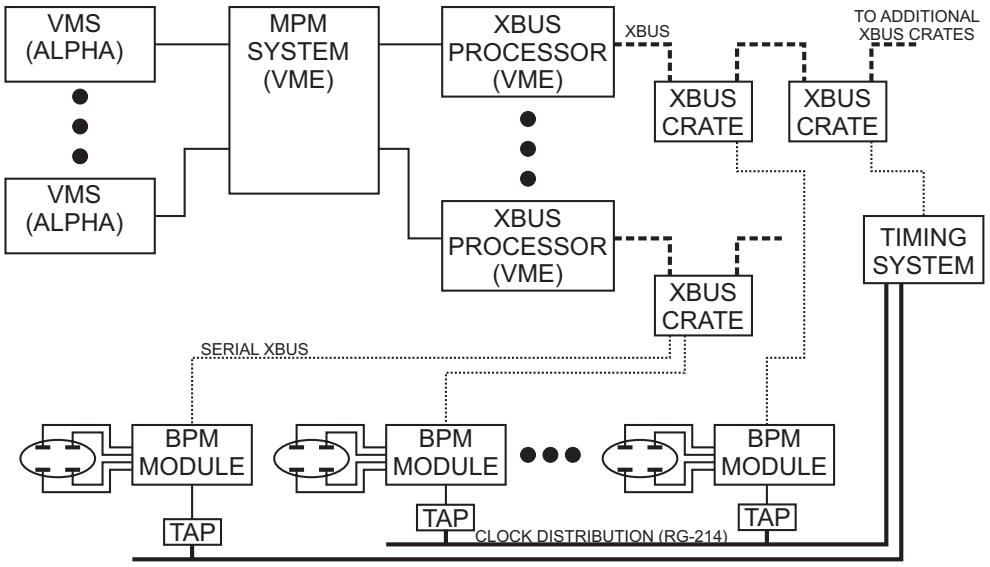

Figure 1: Diagram of the interface between the CESR control system, timing system, and the new BPM system.

tially prepare each BPM module for a measurement, specifying parameters like which bunch to measure and how many turns of data to average. When all of the modules have been programmed, a single command word will be sent on the clock signal. All of the modules will then begin making measurements simultaneously. Due to constraints in the timing system, the command word can be changed at a maximum rate of 60 Hz. The final 2 bits contain hardware trigger information which allows us to synchronize all of the BPM modules to external events. Transitions on the trigger lines are encoded on the clock signal for the next turn so the response to hardware triggers can be very fast. The BPM modules can be configured to take action on individual trigger signals, or on a combination of the pair. To date we have used a 60 Hz line trigger to synchronize measurement with the AC power system and the CESR injection trigger to study transients during injection.

DSP INTERFACE

The front ends of each BPM readout module have been described in detail elsewhere [3]. The digital portion of each module is built around an Analog Devices ADSP-21061 SHARC DSP and an Altera EPF10K30A PLD. The DSP provides 128 KB of memory of which we use 64 KB for code and 64 KB for data storage. A persistent copy of the DSP code is stored in a FLASH memory and is loaded into DSP memory upon reboot. An extensive suite of functions have been programmed for the DSP. These functions fall into four general categories: diagnostic, data acquisition, calibration, and data processing.

The Altera PLD provides the logic for the SERIAL XBUS interface, manages the communications bus on the digital board, decodes the timing system signal, and can reprogram the FLASH memory of the SHARC DSP. An important aspect of having the PLD handle both XBUS communications and the on-board bus is that this allows the control system to probe the DSP memory irrespective of what state the DSP is in.

DSP Function Suite

The core of the DSP functionality is provided by a set of low level data acquisition routines. These routines set the global timing delay for the readout module to look at a particular bunch, set relative delays between button channels so that each channel digitizes at the peak of the button signal, set each digitally controlled channel gain to optimize the signal levels on each channel, and finally acquire the ADC readings from all four buttons upon request. Considerable flexibility is built into the the way in which the ADC data is collected and processed. First, the start of data-taking can be configured to begin at a specified turns delay after any combination of our two hardware trigger bits. A set of data samples, up to our buffer limit of 1024, can then be collected on successive turns or with a specified number of skipped turns between samples. The resulting ADC values for each button can then be stored as is or they can have pedestals subtracted and gain conversions applied before being saved into the data buffer.

In order to obtain the timing information for 90 individual bunches at each BPM location and provide automatic gain control for a large range of signal sizes, an extensive suite of calibration routines is provided in each DSP. Timing calibration begins by putting a single bunch in CESR. Each module then scans its timing offsets until it finds a signal of the correct polarity to within 1 ns. Once the bunch has been located, the known bunch pattern in CESR is used to determine timing offsets for every bunch of that species and these timing offsets are saved to an initialization file. When taking data on specified bunches, a second, more precise timing calibration takes place. A precision calibration algorithm adjusts the overall delay for the module in steps of 17.5 ps. Independent timing delays for each channel, also with 17.5 ps step size, are then scanned and a fit is

carried out to determine the peak of each channel's waveform. Based on this fit each channel's operating delay is then set. The signal waveform from each button is such that timing variations at the 10 ps level lead to a position uncertainty at the few μm level. An additional calibration feature is automatic gain control for each channel on each module. Because bunch currents range from the 100 nA to 10 mA scale in CESR, each module must be able to set its gain level accordingly. In addition, because CESR operates with electrostatically separated beams, the gain control function must be separate for each channel in order to optimize signal-to-noise. The automatic gain control and timing delay calibration functions are tightly coupled because of variations in propagation delay in the variable gain amplifier used in this system.

A number of data-taking options are supported by our DSP code. These include acquiring turn-by-turn data for particular bunches directly, processing turns data in the DSP to provide averaged position and current information for one or more bunches, taking betatron phase measurements, and FFT processing of turns data to provide transverse tune information for specific bunches. In each case a *high-level* DSP routine exists which knows what setup parameters are needed from the control system program and which manages the data acquisition and data processing steps in the DSP.

An example of an application requiring turn-by-turn data is the study of injection transients. The BPM modules start data acquisition a fixed number of turns after a hardware injection trigger. Typically we acquire a full 1024 turn sample for each trigger. Our readout modules are capable of monitoring injected bunches at the 100 nA level.

Averaged position data is used for orbit monitoring and differential trajectory measurements. In this mode, the DSP can cycle through a set of requested bunches, average up to 1024 turns of data for each, and make the resulting mean and RMS information available to the control system. For every turn of data acquired, approximately 2.6 turns are required for on-board processing. This corresponds to a 100 Hz bunch sampling rate for 1024 turn samples.

Betatron phase measurements [4] are made at each detector using the phase information from the horizontal and vertical shakers as encoded on the timing system clock signal. Every BPM module receives new data on every turn. However, due to propagation delays and beam direction, there may be a difference of several turns between when a shaken bunch arrives at a beam detector and when the phase data arrives and is decoded from the clock signal. A calibration procedure was developed which allows us to specify the number of turns of offset between detector data and phase data at each BPM module. Accumulating phase information for 40,000 turns provides a measurement with a few tenths of a degree resolution every 2 seconds.

Good frequency resolution for bunch-by-bunch tune measurements requires a large number of position samples in memory simultaneously. By restricting ourselves to measurements in a single plane and using 16 kwords of data memory, the DSPs can accumulate 8192 turns of data, calculate either the horizontal or vertical position at each sample time, and then perform an FFT on the data using an *in place* algorithm [5]. The resulting frequency resolution for each bunch is better than 50 Hz.

Monitoring of DSP Memory

In order to monitor operation of the large number of independent processors in this system, we have configured our hardware and software to allow an image of key portions of the memory of each DSP to be maintained in the control system program that services the entire system. To do this, equivalent data structures are defined in both the DSP and control system code. These data structures contain DSP configuration and calibration information, working data, and processed output. Transfer of the data structures between the DSP and the control system is facilitated by a pair of lookup tables for use by the Altera PLD. One table contains the starting DSP address of each structure and the other table contains the size of the structure. These tables are initialized by the DSP code when it starts up. Thus for the control system to access a DSP data structure, it simply has to specify the structure's *tag* (*ie.,* its index in the lookup table) and the entire data structure can be passed as a unit via one or more XBUS packet transfers. On the control system side, a key is maintained for each data structure on each DSP. This key contains the DSP tag for that structure, a set of pointers to the control system copy of each data member, a type specification for each data member, a set of pointers to conversion functions for each, as well as various other supporting pieces of information (*eg.*, read-write permissions). The data conversion functions allow proper handling of complex data types as well translation between IEEE and VMS floating point formats.

CONCLUSION

We have described the system-level implementation of a general-purpose BPM readout system. The system is presently in active use in one sector of the CESR ring and we are continuing to explore and develop its capabilities. We would like to thank the members of the CESR Operations Group for their support in its development.

REFERENCES

[1] C.R. Strohman and S.B. Peck, "Architecture and Performance of the New CESR Control System", Proc. of the 1989 IEEE Part. Accel. Conf., p.1687.

[2] R.E. Meller, "Precision Timing Control System", Proc. of the 1997 IEEE Part. Accel. Conf., p.2505.

[3] M. Palmer *et al.,* "An Upgrade for the Beam Position Monitoring System at the Cornell Electron Storage Ring", Proc. of the 2001 IEEE Part. Accel. Conf., p.1360.

[4] D. Sagan *et al.*, Phys. Rev. ST Accel. Beams **3** 092801(2000).

[5] Numerical Recipes, W. H. Press *et al.*, Cambridge University Press, 2nd Ed. (1992)