

The New Tevatron Beam Position Monitor Front-End Software

Luciano Piccoli, Margaret Votava and Dehong Zhang for the Tevatron BPM Upgrade Project

Abstract— Tevatron is a proton anti-proton accelerator collider operating at the Fermi National Accelerator Laboratory. The machine is currently delivering beam for the CDF and D0 experiments, which expect increasing luminosity until the conclusion of Run II, planned for 2009. The Laboratory defined a plan for achieving higher luminosity, and one of the tasks is the upgrade of the accelerator's beam position monitor (BPM). The Tevatron was built during the early eighties and some of its control systems, including the BPMs, are still the original ones. This paper describes the front-end software, from the requirements to the implementation, and the underlying hardware setup. The front-end software designed is presented, emphasizing its modularity and reusability, allowing it to be applied to other Fermilab machines.

I. INTRODUCTION

The original Tevatron BPM front-end system [1] has been used since the construction of the accelerator in the early 1980's. The system relied on Z80 microprocessors with very limited amount of memory (up to 32KBytes). The front-end software was written in assembly and had to carefully manage the scarce resources.

Despite the hardware limitations, the original system provided beam position data until recently, when improved accuracy and reliability became necessary for current and future Tevatron operations.

Among the requirements for the system upgrade include the improvement in the position measurements in an order of magnitude [2]. The upgraded system is also expected to extend the BPM system capabilities by simultaneously providing position information for both proton (p) and anti-proton (\bar{p}) particles. The original system is unable to extract \bar{p} information, which becomes necessary as the \bar{p} intensity is expected to increase in the near future.

The following section describes the Tevatron BPM upgraded hardware components. Section III lists the front-end software requirements, including the modes of operation for the system. Next the design of a generic front-end system is introduced, followed by the description of the software configuration to meet the Tevatron requirements. The performance of the complete system in the production environment is shown and its use on other Fermilab machines discussed.

II. SYSTEM OVERVIEW

The proton (p) and anti-proton (\bar{p}) beam signals are captured by BPM pickups mounted within the superconducting

quadrupoles around the accelerator ring. These devices are part of the original system and remained the same for the upgrade. Each pickup is a pair of 50 ohm striplines 18cm long, with a circular aperture 6.6cm. A typical BPM pickup is shown in figure 1.

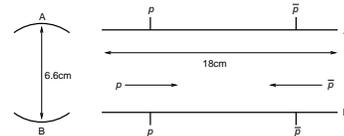


Fig. 1. View of a typical Tevatron BPM pickup

Both striplines (plate A and plate B) have two connectors, which are used to harvest the p and \bar{p} signals. The original monitored only proton positions, therefore requiring only the p signals. The pickups are arranged either horizontally or vertically (as in figure 1). The total number of pickups around the accelerator ring is 240, accessible through 27 service buildings (also referred to as a *house*).

Four cables bring the pickup signals up to the service buildings, where they connect to the upgraded front-end VME crate. Each crate contains the following components: one Motorola MVME2400 processor; one timing board; and up to six pairs of analog filter boards and digitizer boards. The components are described below.

1) *Processor Board*: The processor board chosen for the BPM upgrade is the Motorola MVME2400-0361 with 512 MBytes of memory. It runs the VxWorks Real Time operating system (version 5.5) and is the host for the front-end software described in section IV.

2) *Timing Board*: Custom made board that generates timing signals that control and initiate acquisition of the BPM signals. Timing signals are based on inputs from the accelerator controls clock system, which include the Tevatron RF clock (RFCLK), Tevatron beam sync clock (TVBS) and the Tevatron event clock (TCLK). The board provides synchronization signals for up to eight digitizers.

3) *Filter Board*: Custom made board that receives the beam signals directly from the BPM pickups. The filter board conditions the pickup signals to be used as inputs of the digitizer boards. Each filter board contains eight channels, which can handle two BPM pickups (there are four channels per BPM). The filter board, combined with the timing board, is capable of generating diagnostic signals. These signals can be directed to the digitizers or can be sent to the BPM pickups.

4) *Digitizer Board*: Commercial boards from EchoTek corporation (model ECDR-GC814-FV-A), containing high speed

digitizers that convert the analog inputs (from the filter board) into digital information. The final outputs from digitizer are the down converted signals, defined as an I and Q pair per sample. Each digitizer board also eight input channels, which is capable of handling two BPMs.

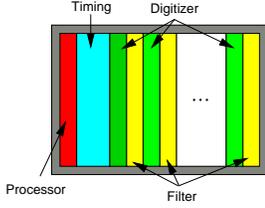


Fig. 2. Distribution of the BPM front-end components within the VME crate

The modules are organized in the front-end crate as depicted in figure 2. The first slot contains the processor board, followed by the timing module. The number of filter and digitizer boards depends on the number of BPMs handled by the front-end. It varies from four to twelve BPMs, which can be handled by six filter board and digitizer pairs. Data are exchanged between the front-end software and the online applications through a 10Mbits Ethernet connection.

The configuration and setup of the hardware described above is responsibility of the front-end software. The different modes of operation and the type of data to be extracted from the system is described in the next section.

III. FRONT-END REQUIREMENTS

The front-end software has to constantly monitor beam position information acquired through the BPM pickups. The beam structure varies according to the mode of operation of the machine. The system has to configure itself based on mode changes and user configuration requests.

The modes of operation defined for the front-end software are: closed orbit, turn-by-turn, injection, safe injection and diagnostics. For all modes, the front-end has to compute the beam position and intensity based on the raw I and Q values provided by the digitizer boards.

The intensity is computed as the sum between the signals from the A and B plates, according to 1.

$$\text{intensity} = |A| + |B| \quad (1)$$

The position is given by 2, where s is the scale factor. In addition, the front-end should allow calibration factors, such as electrical and mechanical offsets, to be added to 1 and 2.

$$\text{position} = s \left(\frac{|A| - |B|}{|A| + |B|} \right) \quad (2)$$

Due to the current 10:1 proton-to-antiproton bunch intensity ratio, the \bar{p} intensity and position calculation requires corrections derived from the p measurements [3]. The front-end must apply proper corrections for every \bar{p} measurements.

A. Closed Orbit

This is the default mode of operation for the front-end system. The closed orbit position for both p and \bar{p} particles is computed at a fixed frequency rate of 500Hz.

The closed orbit is an average measurement of the beam position and intensity during approximately 65 turns (each turn takes 21 milliseconds). The averaging is performed by the digitizer boards and the software is responsible for reading out the measurements and maintaining them in memory.

The system must keep a few buffers associated with closed orbit measurements. They are: closed orbit, fast abort, slow abort, profile and display. The closed orbit buffer is constantly updated with values independently of the machine state. Those values must always be available.

The fast abort and slow abort buffers are halted upon beam loss. Those buffers contain information that allow users to understand the beam motion previous to a beam loss event. The fast abort contains copies of closed orbit buffer elements and the slow abort is updated at 1Hz with an average of the last 100 elements from the fast abort buffer.

The last two closed orbit buffers, the profile and display, are filled with the latest closed orbit measurement at the reception of certain events. The events are broadcast through predefined TCLK signals, which are fired at certain beam conditions. For example, the profile measurements are taken during beam injection at distinct energy levels.

B. Turn-by-Turn

The turn-by-turn mode is enabled via user request and, differently from closed orbit, the position and intensity for every turn must be measured. The system is configured to record information about 8192 consecutive turns.

The data read out from the digitizer must be kept in a separate buffer, which can be read by the user at any time. After the conclusion of the turn-by-turn measurement the system must immediately return to the default closed orbit mode.

C. Injection

The injection is the turn-by-turn mode triggered by the first proton injection in the machine. As in the previous mode, the system is required to gather information about first 8192 turns.

Data from the injection mode must be kept in the injection buffer and must be available throughout the store, until the next injection. The system also must perform an averaging of the first turns, providing a closed orbit measurement for the injection. This calculation must be carried out by the processor instead of the digitizers. After the injection cycle is complete the system switches to the closed orbit mode.

D. Safe Injection

The safe injection mode is used when the front-end cannot get reliable turn timing. At the start of the injection cycle the system is set up to sample the beam continuously at the maximum digitizing rate.

The data read out from the digitizers must be processed by the front-end, which searches for the intensity peaks. The samples with highest peak are the candidates for the actual beam position. From the selected samples the system computes the initial closed orbit and returns to the default mode of operation.

E. Diagnostics

In diagnostics mode the system must provide several configuration options to the user. It must provide in depth raw data visualization and the capability of easily changing the configuration. This mode of operation is required for verifying cable connections, signal quality, find problematic channels and allow other procedures to insure the correct behavior of the system. The diagnostic signal generated by the timing and filter boards can be enabled on this mode.

IV. SOFTWARE DESIGN

The upgraded front-end software was designed to meet the Tevatron requirements keeping in mind its potential use in other BPM systems at Fermilab. As result, the functionality and data structures that can be shared between machines were isolated in a generic module. Object oriented design techniques using UML notation were used throughout the design phase.

The general principle of operation for the BPM front-end system follows a simple data acquisition (DAQ) sequence: arm, trigger and data read out. Every step is started by the occurrence of an event in the system, for example the injection mode is armed when the machine starts an injection sequence.

The front-end software design is derived from that simple DAQ sequence. It can be defined as an event driven environment, where every operation must be preceded by the occurrence of some key event. Additional functionality is added to the model in order to define a complete generic environment (GBPM). An overview of the general front-end software is shown on figure 3.

The active components of the system are represented by three types of tasks: control task, alarm task and data acquisition task. The control task provides general coordination and is responsible for starting up the system by creating data buffers and initializing remaining tasks.

Any entity in the system is capable of generating alarm events, which are internally routed to the alarm task. The alarm task has internal thresholds and can be configured to dispatch alarms to external systems via the controls network.

The actual data acquisition job is carried out by the DAQ tasks. The basic DAQ task remains waiting for an event in its input queue. Upon the reception of an event, the task performs the data read out. The DAQ task reads data from a *data source*, processes it if necessary, and stores it in a *data buffer*.

A data source is an entity capable of generating data, and it implements methods to get a single or multiple elements. The data buffer is the source counterpart, allowing a single or multiple elements to be saved. Additionally a data buffer can also perform as data source, allowing data to be moved between buffers.

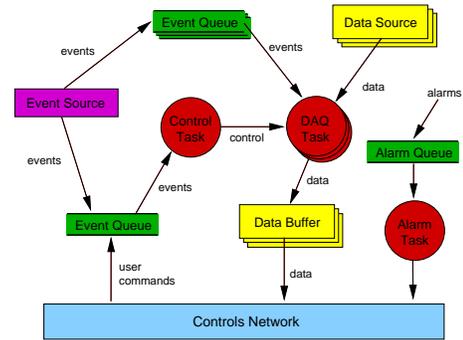


Fig. 3. Generic design for the BPM data acquisition environment

There are two types of data buffers: circular and FIFO. The circular buffer continuously accept new data and when it is full the oldest element added is replaced. The FIFO buffer accepts data until it becomes full, and new data are rejected until the buffer is explicitly cleared.

All tasks in the system have an associated event queue. Events are generic wrappers for any information that needs to be exchanged between tasks or other components of the system. The events defined for the generic system include:

- a) *Interrupt event*: generated upon the occurrence of an interrupt.
- b) *Mode change event*: generated when the system is requested to switch to a new mode of operation.
- c) *Alarm event*: created internally to signal an abnormal condition.
- d) *Time event*: used to trigger periodic tasks.
- e) *State change event*: produced when a *state device* has the state changed.
- f) *TCLK event*: generated upon the decoding of a predefined TCLK event.

These basic building blocks can be arranged in any sequence to compose the Tevatron BPM front-end system. However, the front-end is not complete without adding components that are specific of for the Tevatron system, such as the extension of the data acquisition tasks to perform data read out according to the upgrade requirements.

A. Tailoring System for the Tevatron

This section describes how the basic building blocks defined in the generic library are used to configure the Tevatron front-end system according to the requirements listed in section III.

The modes of operation for the front-end system are reduced to two modes: closed orbit and turn-by-turn. The remaining modes are either extension of those or require a slightly different configuration.

The closed orbit is the default mode of operation. It involves several data acquisition tasks, which are shown in figure 4. In closed orbit mode, only the closed orbit task has access to the digitizer boards. The timing, filter and digitizer boards are configured by the control task when entering this mode of operation. The data acquisition cycle is started by the timing board, which transmits a signal to the digitizers and generates an interrupt to the processor every two milliseconds.

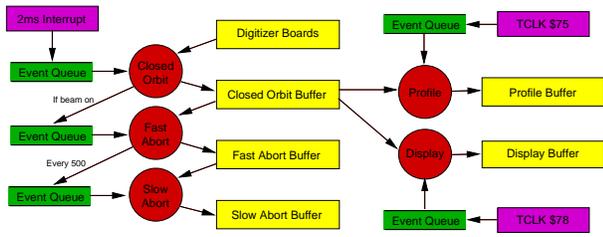


Fig. 4. Tevatron closed orbit data acquisition tasks

When the signal is received by the digitizer, the following 65 turns are averaged. Meanwhile, the interrupt generates an input event that triggers the digitizer read out by the closed orbit task. The data read out starts when the digitizers interrupt the processor signaling that the averaging is complete. The data are processed and stored in the closed orbit buffer. The system keeps reading out closed orbit data even if there is no beam in the machine. Position and intensity information is always available while the system in this mode.

While there is beam in the machine, the closed orbit task generates a read out event to the fast abort task, which transfers the most recent value from the closed orbit buffer to the fast abort buffer. Similarly, the fast abort task generates a read out event to the slow abort task on every 500th event. The slow abort task processes the event by averaging the latest hundred elements from the fast abort buffer and saving the result as a new entry in the slow abort buffer.

The closed orbit mode is completed by the profile and display data acquisition tasks. Both tasks receive input events from the Tevatron clock (TCLK). The TCLK signal is decoded by the timing board, which interrupts the processor. If a profile (clock \$75) or display (clock \$78) TCLK is received an equivalent event is produced and sent to the respective task. When processing the event, the tasks get the latest value from the closed orbit buffer and save it in the profile or display buffers.

Differently from the closed orbit mode, the turn-by-turn mode produces position and intensity information for single turns. The configuration procedure is similar to the previous mode, all timing, filter and digitizer boards are configured when changing the mode.

In turn-by-turn, the timing board sends multiple triggers to the digitizers and a single interrupt to the processor. Every trigger corresponds to a turn, and the digitizer selects one sample per turn. The interrupt generated causes the turn-by-turn task to wait for the digitization of the 8192 turns. Upon completion, the data from the digitizers is read by the processor using DMA operations. The data are saved in the turn-by-turn buffer, which can later be accessed by online applications.

Regular turn-by-turn measurements can be taken at any time while there is circulating beam in the machine. However, there are special cases for turn-by-turn measurements, as shown in figure 5. These are the injection mode and the safe mode.

Both injection and safe modes are used when beam is first injected in the machine. In injection mode, the 8192 turns are taken after the timing system receives a TVBS clock \$7C

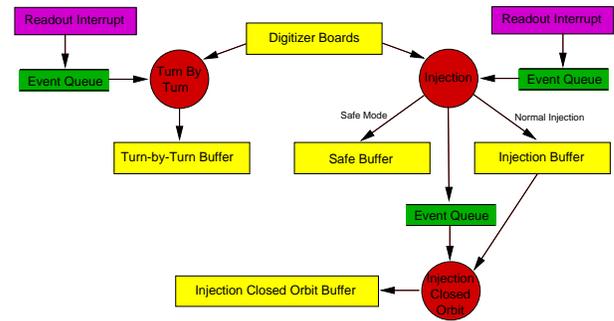


Fig. 5. Tevatron turn-by-turn data acquisition tasks

indicating that beam is coming on the next turn marker. The acquisition procedure is the same as in the regular turn-by-turn, except that data are saved in a special injection buffer and the injection task sends an event to the injection closed orbit task. The latter computes the average position and intensity from the injection, producing an initial closed orbit measurement, also available in a separate buffer.

The safe mode can be used as alternative to the injection turn-by-turn when the TVBS turn marker is not present or is not reliable. In this mode, the timing board is configured to send a single trigger to the digitizer, which is configured to collect 8192 consecutive samples (at its maximum rate). The resulting data are stored in the safe buffer and later processed by a simple beam search algorithm used to find the intensity peaks. The selected samples (about 75 turns) are saved in the injection buffer and used to compute the injection closed orbit.

In addition to the modes described above, the system can be configured to produce diagnostics data. When configured to run in diagnostic mode, the front-end saves all samples taken by the digitizer boards in a special diagnostics buffer. The diagnostic data are read by a diagnostics user application that is capable of displaying several attributes of the data: magnitude, phase, position and intensity. Additionally through the user application it is possible to change triggers, acquisition controls and delays.

B. Controls Network

The 27 BPM front-end crates are individual entities and do not communicate with each other. The read out and analysis of the data acquired by the front-ends is handled by higher level software (referred to as online), which is able to address each crate individually.

The communication between online applications, such as the diagnostics application, and the front-end uses the Accelerator Controls Network — ACNET [4]. It defines a communication protocol over Ethernet, allowing control and monitoring data to be exchanged between front-end computers and online applications. Access to the data is possible through ACNET device readings and settings.

Every front-end defines up to 130 ACNET devices. System properties, such as channel delays, calibration and correction constants can be changed via setting devices. Status and beam position and intensity are read out using ACNET reading devices. The front-ends define a range of beam position

readable devices, providing either the raw data directly from the digitizer or calibrated and corrected measurements.

V. IMPLEMENTATION

The front-end system is implemented in the C++ language. The generic environment is contained in the GBPM library whereas the Tevatron implementation is defined in the TBPM library. Additional software is available on separate libraries, such as the digitizer driver code.

The front-end system provides several command line tools that allow experts to verify the current software status. It is possible to monitor the amount of data collected, current configuration and operation history.

The system uses the TRACE tool [5] for helping debugging and profiling the system. The tool was developed at Fermilab and is available for the Linux and VxWorks operating systems. The front-end libraries include TRACE directives are throughout the code, which can be configured at runtime to produce a stack trace for given code sections.

VI. SYSTEM PERFORMANCE

Profiling measurements using TRACE show that the front-end system can sustain periodic closed orbit measurement at 500Hz for a full loaded crate (six filter-digitizer pairs). The closed orbit measurement cycle takes on average 1.8 ms to complete, leaving a 20% idle time for additional processing used by the slow abort, profile and display data acquisition.

The closed orbit cycle is dominated by the averaging on the digitizing boards, which takes 1.38 ms. During the averaging the processor is available for lower priority processing, such as user data requests. After a measurement is completed, the boards take 235 μ s to be prepared for the next cycle. From the remaining 185 μ s, 110 μ s are required for transferring the data through VME single word reads; 50 μ s are used for calculating the intensity and position from the raw values and applying calibration and correction factors; and it takes 25 μ s for the additional tasks of saving the data and reorganizing internal buffers.

The closed orbit processing cost is minimal when compared with the cost of certain mode transitions. For example between the turn-by-turn and closed orbit mode require the digitizer boards to be reconfigured. The total setup time for a full crate is in the order of 61 ms.

The total time to complete a turn-by-turn cycle and return to the default closed orbit mode takes 325 ms, which includes the 61 ms for loading the turn-by-turn digitizers setup at mode change time. The setup reloading time defines the minimum delay required by the front-end to arm for a turn-by-turn measurement. For the critical injection mode, the system receives a prepare for beam TCLK 2.7 seconds in advance, showing that the reload time is not critical for the operation.

After the turn-by-turn measurement is triggered it takes 170 ms for digitizers to sample information for the following 8192 turns. On completion, the data are read out by the processor through the VME backplane using DMA. The operation takes approximately 5.3 ms per board, resulting in a total time of 32 ms for the full system.

Immediately after the data are transferred the system switches back to closed orbit mode, which requires the remaining 61 ms for restoring the digitizer closed orbit configuration. The time required for configuring the timing board is negligible.

The intensity and position calculation along with the buffer management time for a turn-by-turn measurement is taken from the processor's idle closed orbit cycles, i.e. while the digitizers average beam data and during the 20% spare cycles from every measurement. The low priority turn-by-turn data processing is feasible because user applications do not need the turn-by-turn data immediately after the measurement is taken.

The front-end software takes advantage of the real time aspects of the operating system by controlling the priorities of the data acquisition tasks according to the mode of operation. Turn-by-turn data acquisition tasks have the priority increased when the system receives a turn-by-turn request and lowered as soon as the system gets back to the default closed orbit mode.

The Tevatron front-end software defines thirteen processing tasks, of which nine are dedicated to data acquisition, one is the control task, one is the alarm task, and the additional two are used for logging purposes. The total memory used by the system does not exceed 25% of the 512MB available in the processor board, most of it is reserved by the driver for reading out the digitizer boards.

VII. CONCLUSION

The paper has described the Tevatron upgraded front-end software. The proposed design and implementation met the upgrade requirements, delivering a flexible and reliable software. New modes of operation, data acquisition tasks and data buffers can easily be added to the system.

In the near future, the Tevatron front-end software is expected to be used for the Main Injection BPM upgrade. It is possible to reuse the same structure defined in the GBPM package and have the system configured to meet different requirements. The Main Injector is more complex than the Tevatron and requires additional modes of operation, which implies in more data acquisition tasks and data buffers. Additionally, the generic environment can also be used for projects beyond the BPM scope. It is suitable and can be tailored to other types of data acquisition systems, such as slow controls.

REFERENCES

- [1] A. E. Baumbaugh R. E. Shafer, R. E. Gerig and C. R. Wegner, "The tevatron beam position and beam loss monitoring systems," in *XII International Conference on High Energy Accelerators*, Batavia, IL, August 1983.
- [2] S. Wolbers et al, "Tevatron beam position monitor upgrade," in *Particle Accelerator Conference (PAC05)*, Knoxville, TN, May 2005.
- [3] R. Webber R. K. Kutschke, J. Steimel and S. Wolbers, "Simltaneous position measurements of protons and anti-protons in the tevatron," in *Particle Accelerator Conference (PAC05)*, Knoxville, TN, May 2005.
- [4] J. Patrick, "Evolution of the fermilab control system," in *IX International Conference on Accelerator and Large Experimental Physics Control System*, Gyeongju, Korea, October 2003.
- [5] R. Rechenmacher and D. Holmgren, "Fermi linux TRACE," <http://fermitools.fnal.gov/abstracts/trace/abstract.html>.