# AN IN-LINE EMITTANCE SCANNER BASED ON A LABVIEW STYLE STATE MACHINE WITH SEQUENCER

W. Blokland[1] and C. Long[2]
*[1]ORNL, SNS*, Oak Ridge, USA, [2]Innovative Design, Knoxville, USA*

## ABSTRACT

The Spallation Neutron Source (SNS) accelerator systems will deliver a 1.0 GeV, 1.4 MW proton beam to a liquid mercury target for neutron scattering research. The SNS diagnostics platform is PC-based running Windows XP Embedded and LabVIEW. The software to implement instruments is based on templates. The main template is a LabVIEW style state machine. For the emittance scanner, which has two motion axes per plane, a variation of the standard template was created to include a sequencer that generates a sequence of commands based on the task to be performed by the state machine. The goal of this template is to handle the additional complexity of dealing with multiple motion axes and keep the system responsive even during long motions without significantly complicating the overall software structure. The software for the emittance scanner, including control, remote display, and analysis was completed and successfully commissioned in about three man-months. This paper describes the template and software tools used to implement the emittance scanner and the scanner itself.

## INTRODUCTION

The Diagnostics Group uses rack-mounted PCs running LabVIEW to acquire and process the data, and EPICS IOC to communicate with the SNS control system for many instrument types, such as Laser Wire Scanners and Beam Current Monitors (BCM), see [1].

### *LabVIEW Style State Machine Template*

A standard template based on the generic LabVIEW style state machine is used to start each project. The advantages of the state-machine based template are as follows:

- built-in support for the Shared Memory Interface to EPICs IOC (see [2]), a default configuration file, and the local and remote user interface;
- strict organization of program code to guide development by showing where what code should go;
- simple structure to speed development and debugging;
- applicable to various instrument types, e.g. beam position monitor and beam current monitor;
- supports jumping to other states to handle exceptions.

A typical state sequence is as follows: the poll state to wait for data (“Poll”) to be acquired or a user interaction to occur (remote or local), get the data (“Get Data”), analyze the data (“Analyze”), update local displays (“Update local”), update remote data using Channel Access (“Update Remote”), setup the acquisition for the next round (“Setup”), and back to wait (“Poll”), Figure 1. An example of the block diagram code is given in Figure 2. The pull-down menu in the figure shows the implemented states. Because the state machine is generally applicable to many instruments, most instruments’ code started by copying and renaming the code from an already existing instrument. This reuse of code gives the programmer a significant head start on implementing the overall program.

* SNS is managed by UT-Battelle, LLC, under contract DE-AC05-00OR22725 for the U.S. Department of Energy. SNS is a partnership of six national laboratories: Argonne, Brookhaven, Jefferson, Lawrence Berkeley, Los Alamos and Oak Ridge.

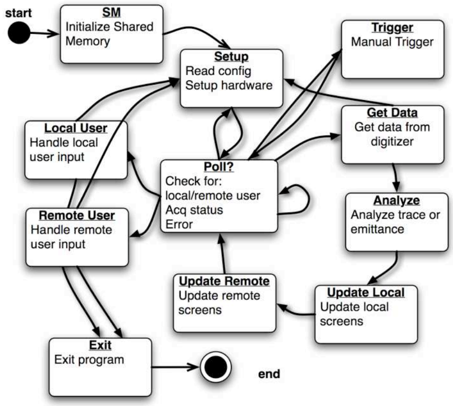


Figure 1. State machine diagram.

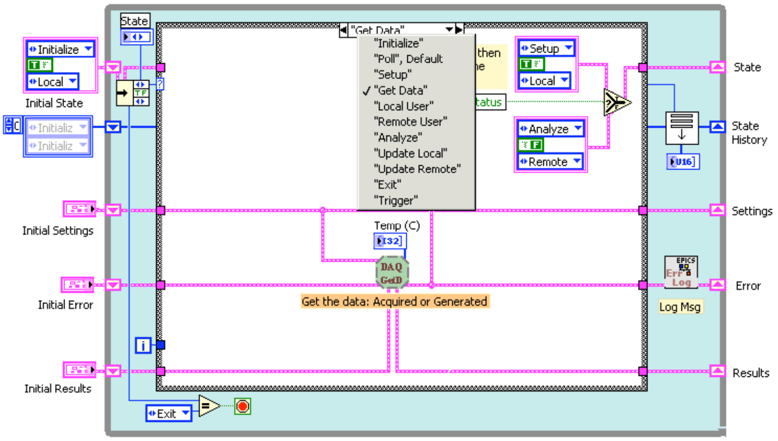
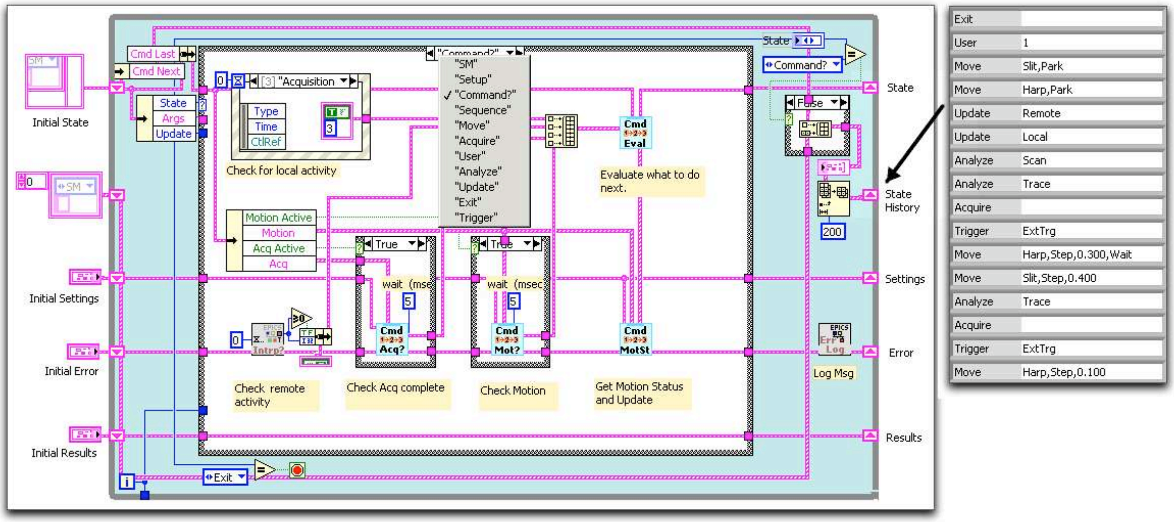


Figure 2. The BCM state machine block diagram.

While the state machine template works well for many instruments, others that require a more complex logical structure, such as those with motion, a different structure would fit better. While a new state can be added to handle motion, the state could be busy for a long time to complete the move. The “Poll” state would not be visited during this time meaning that the instrument would not respond to local or remote user input. Adding a check for user input within the move state would alleviate this, but now two states have code looking for user input. That would mean that now it is no longer clear where what functionality is located and/or that the same functionality has to be implemented twice. Another option is to start multiple tasks, for example one to handle the motion and one to handle the rest. This was done to implement the software for a Faraday Cup. However, the added complexity needed to handle communication and synchronization between the tasks led to an increase in development time.

### *LabVIEW Style State Machine Template with Sequencer*

The emittance scanner has two moving parts, a slit and a harp. Each must be moved at different times and a scan can be described as a sequence of moves for the harp and the slit, acquisitions of data, and a series of analysis of traces, including one final emittance calculation. The template’s state machine already implements states such as analysis and acquisition. By adding a state which can generate a sequence of states given a set of parameters, the state machine can now execute a complete scan and, through continuously visiting the “Poll” state, be responsive to user input as long as the time spent in each state is short enough. To make sure that each state has a relatively short duration, the “Poll” state checks the progress of the motion and handles the progression of the sequence. Figure 3 shows the block diagram code with an added sequence state. Compared to the previous state machine, states have also been renamed or merged (e.g. local and remote have been merged). A state parameter is passed to each state to indicate why the type of update or analysis is needed or how far of a move must be made. An example of the sequence is shown on the right of the figure with the most recent state on the top.



| | |
|---|---|
| Exit | |
| User | 1 |
| Move | Slit,Park |
| Move | Harp,Park |
| Update | Remote |
| Update | Local |
| Analyze | Scan |
| Analyze | Trace |
| Acquire | |
| Trigger | ExtTrg |
| Move | Harp,Step,0.300,Wait |
| Move | Slit,Step,0.400 |
| Analyze | Trace |
| Acquire | |
| Trigger | ExtTrg |
| Move | Harp,Step,0.100 |

Figure 3. The LabVIEW State machine with sequencer.

Figure 4 shows the state diagram of the sequencer program. It shows that the heart of the state machine is the “Command?” state. It is called after each state (except “Exit”) to determine what will be the next state. The sequence generation and handling allows any of the states to be called in any order and, because the “Command?” state is called after each state, the user can interrupt a sequence and/or view the status of the acquisition and the progress of the motion. This new template now inherits all the advantages and similarity of the simple state machine and can handle motion with a small amount of additional complexity.

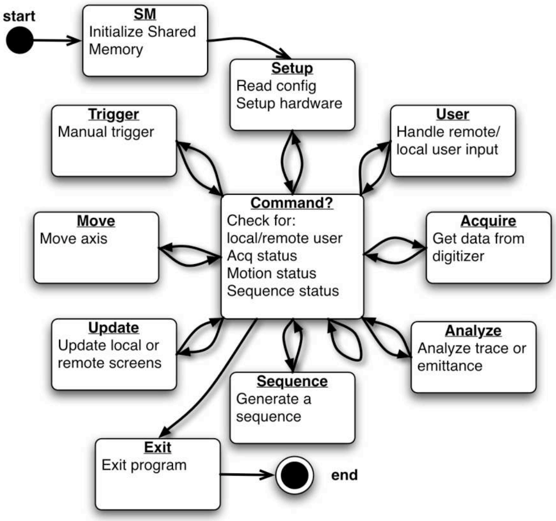


Figure 4. Sequencer State Machine diagram.

## EMITTANCE SCANNER IMPLEMENTATION

The emittance scanner vacuum hardware is described in [3]. The electronic amplifiers were designed and implemented at ORNL, see [4]. The data acquisition system is based on a 4U rackmount PC with an ICS-645 digitizer with 16 channels of a 5 MHz sample rate per channel. Also installed was a National Instruments 7334 motion control card to control position of the harp and slit in the beam pipe. There are two scanners, one for the horizontal plane and one for the vertical plane. Each scanner controls one slit and one harp. A collision avoidance scheme was implemented to avoid the horizontal and vertical slits being inserted at the same time, as they would collide. The collision avoidance disables the forward motion of the other slit or harp by activating the other's forward limit switch. The slit and harp are pictured in Figure 5, courtesy of T. Roseberry.

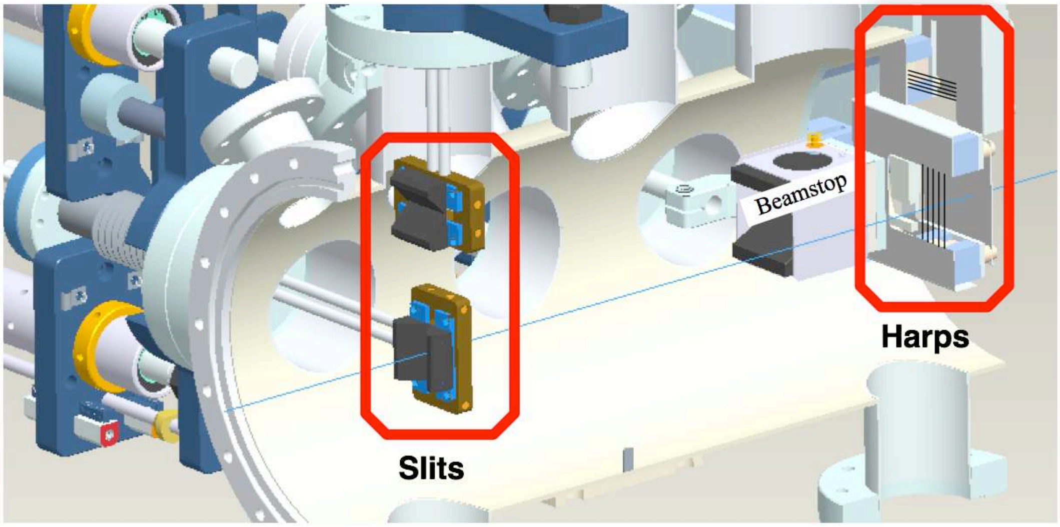
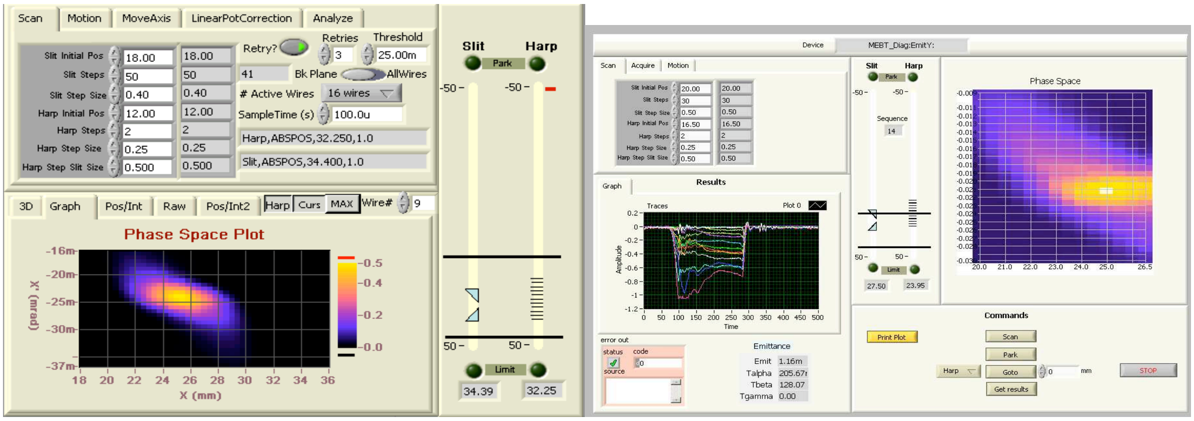


Figure 5. A CAD drawing of the slit and harp.

Figure 6 shows the local application, while the right side shows the remote panel. The remote panel is implemented with LabVIEW and the Channel Access Client, see [5], and runs on the Linux based operator consoles. The panels show the position of the harp and slit. You can view the traces for each harp position and view the intensity plot updating as the system is scanning. The emittance program generates a results page that can be pasted into the electronic log, see Figure 7. The data traces are also available to other applications.



| | | |
|---|---|---|
| Slit Initial Pos | 18.00 | 18.00 |
| Slit Steps | 50 | 50 |
| Slit Step Size | 0.40 | 0.40 |
| Harp Initial Pos | 12.00 | 12.00 |
| Harp Steps | 2 | 2 |
| Harp Step Size | 0.25 | 0.25 |
| Harp Step Slit Size | 0.500 | 0.500 |

Retry? ●
Retries 3
Threshold 25.00m
41
Bk Plane ○ AllWires
# Active Wires 16 wires
SampleTime (s) 100.0u
Harp,ABSPOS,32.250,1.0
Slit,ABSPOS,34.400,1.0
Wire# 9

34.39 32.25

Device MEBT_Diag:EmitY:

| | | |
|---|---|---|
| Slit Initial Pos | 20.00 | 20.00 |
| Slit Steps | 30 | 30 |
| Slit Step Size | 0.50 | 0.50 |
| Harp Initial Pos | 16.50 | 16.50 |
| Harp Steps | 2 | 2 |
| Harp Step Size | 0.25 | 0.25 |
| Harp Step Slit Size | 0.50 | 0.50 |

Sequence 14

27.50 23.95

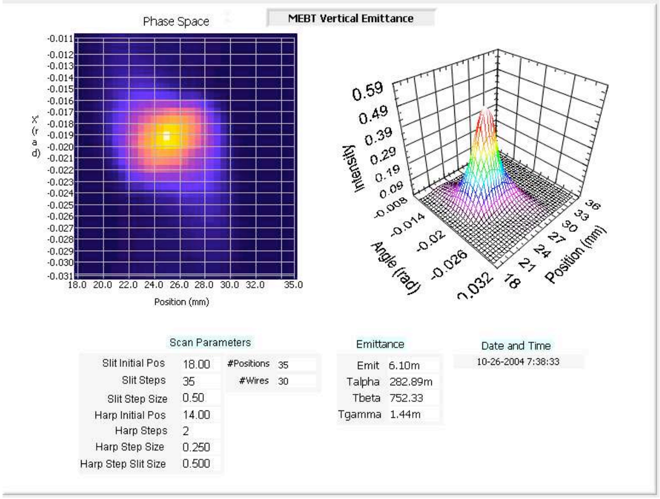
error out
status ✓ code 0
source

Emittance

| | |
|---|---|
| Emit | 1.16m |
| Talpha | 205.67 |
| Tbeta | 128.07 |
| Tgamma | 0.00 |

Harp Goto 0 mm

Figure 6. The local control panel on the left and the remote panel on the right.



Scan Parameters

| | | | |
|---|---|---|---|
| Slit Initial Pos | 18.00 | #Positions | 35 |
| Slit Steps | 35 | #Wires | 30 |
| Slit Step Size | 0.50 | | |
| Harp Initial Pos | 14.00 | | |
| Harp Steps | 2 | | |
| Harp Step Size | 0.250 | | |
| Harp Step Slit Size | 0.500 | | |

Emittance

| | |
|---|---|
| Emit | 6.10m |
| Talpha | 282.89m |
| Tbeta | 752.33 |
| Tgamma | 1.44m |

Date and Time

10-26-2004 7:38:33

Figure 7. The results image of the emittance scanner.

### *Analysis*

The analysis is implemented by averaging part of the harp wire signal after a baseline is subtracted from the trace for each movement of the harp. The integrated value is stored together with the angle and position. At the end of the scan, the emittance is calculated using the standard emittance calculations shown in equation 1. During the scans, the integrated values are plotted so that the operator can see the plot as it develops and, if necessary, abort the scan and retry with new scan parameters.

$$\tilde{\varepsilon} = \sqrt{\overline{x}^2 \overline{x}'^2 - (\overline{xx}')^2} \quad \text{with } \overline{x} = \frac{\sum_{x,x'} I(x,x') \bullet x}{\sum_{x,x'} I(x,x')},\ \overline{x}' = \frac{\sum_{x,x'} I(x,x') \bullet x'}{\sum_{x,x'} I(x,x')}$$
$$\overline{x}^2 = \frac{\sum_{x,x'} I(x,x') \bullet (x - \overline{x})^2}{\sum_{x,x'} I(x,x')},\ \overline{x}'^2 = \frac{\sum_{x,x'} I(x,x') \bullet (x' - \overline{x}')^2}{\sum_{x,x'} I(x,x')} \text{ and } \overline{xx}' = \frac{\sum_{x,x'} I(x,x') \bullet (x' - \overline{x}')(x - \overline{x})}{\sum_{x,x'} I(x,x')} \tag{1}$$

## RESULTS

A significant factor in accelerating the program development is that LabVIEW's programming environment gives you at all times a graphical or numerical display of the internal values of each subroutine. During run-time you can open each routine and see the in- and out-puts. The visual debugger shows you the exact flow of data. For a system with the complexity of the emittance scanner, this means that you are not flying blind or have to invest a lot of time getting data displayed. The state machine with sequencer template helped speed the rapid development of the emittance scanner beyond even what is normal for LabVIEW. The structure helped to easily define the sequence of events that comprise a scan, i.e. axis movement, data collection, data analysis, and data storage, as well as keeping up with user interaction with a control flow that is easy to follow and debug. The initial development of the emittance scanner was on the order of six weeks of pure development time. This includes time that was spent developing a support library for the motion and testing the operation of the digitizer. This was followed by another six weeks of intermittent testing and refinement (we were limited by beam time to run actual tests).

Once completed and tested, the emittance scanner system was able to provide valuable data about the cross section of the beam to the end users in a timely and useful manner.

## SUMMARY

An emittance scanner system based on a slit and a harp was implemented in LabVIEW using the sequencer state machine template in about six weeks with an additional six weeks of on and off commissioning. The system has successfully been used to determine the characteristics of the MEBT beam.

## ACKNOWLEDGEMENTS

This paper describes only the software; many people have worked on the scanner design, vacuum hardware, electronics, installation, etc. Saeed Assadi did the proof of feasibility and initial design. T. Roseberry made the CAD drawing of the Slits and Harps.

## REFERENCES

[1] W. Blokland, T. Shea, and M. Stettler, “Network Attached Devices,” pp 151-153, DIPAC2003, Mainz, Germany, May 5-7, 2003.

[2] D. Thompson and W. Blokland, “A Shared Memory Interface between LabVIEW and EPICS”, ICALEPCS 2003, pp275-277. Gyeongju, Korea, Oct 13-17 2003.

[3] T. Roseberry, S. Assadi, G. Murdoch, “Development Of A New Beam Diagnostics Platform”, PAC 2005, Knoxville, TN, USA, May 16-20 2005.

[4] V. Gaidash and J. Pogge, “Circuit design for Emittance scanner amplifiers”, internal document SNS, ORNL, 2004.

[5] A. Liyu et al., “LabVIEW Library to EPICS Channel Access,” PAC05 conference, Knoxville, TN, USA, May 16-20, 2005.