

Integration of Lab View with ATR Systems

C. Saltmarsh

December 1994

Collider Accelerator Department
Brookhaven National Laboratory

U.S. Department of Energy

USDOE Office of Science (SC)

Notice: This technical note has been authored by employees of Brookhaven Science Associates, LLC under Contract No. DE-AC02-76CH00016 with the U.S. Department of Energy. The publisher by accepting the technical note for publication acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this technical note, or allow others to do so, for United States Government purposes.

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or any third party's use or the results of such use of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof or its contractors or subcontractors. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

Integration of LabView with ATR Systems

Chris Saltmarsh

December 8, 1994

After the discussions on 12/7/94 regarding the integration of LabView with the (developing) ATR control structure, we drew a data flow diagram to represent the various processes and data involved. It is shown on the next page.

We discuss the processes shown in the figure with an indication of the work that needs to be done.

"Translate Glish \leftrightarrow LabView" Persistent multi-threaded Glish client. Uses SDS BPM memory map definition to translate LabView data read/write commands to Glish events. These events feed to ADOIF client, initially for the memory ADO. Returns data from ADOIF to LabView. We build the communication through a TCP/IP link: thus we can connect to LabView on Mac or Sun, or both at the same time. This program should stay simple: no graphics, no computation, just a translator.

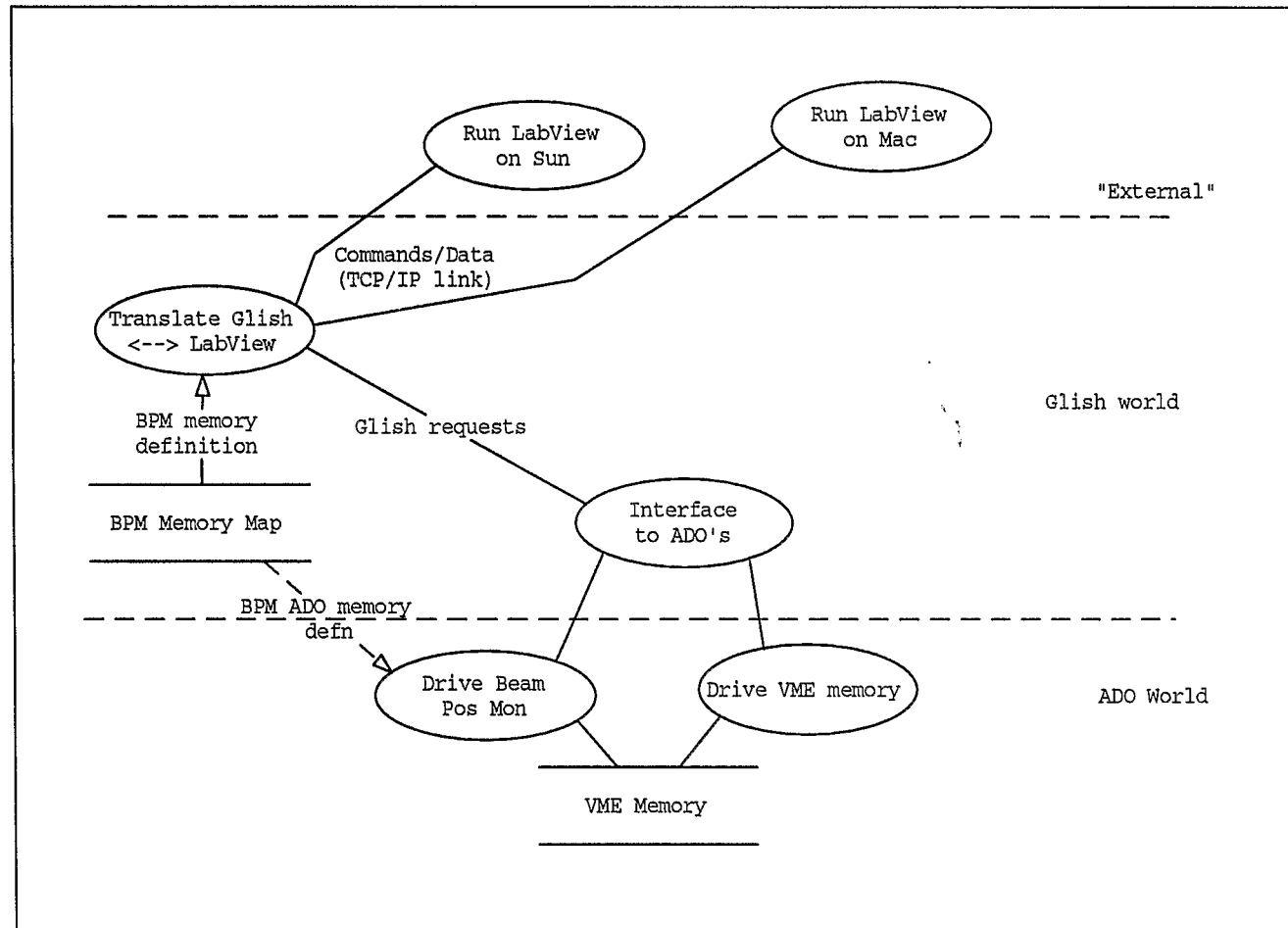
Effort:

- TCP/IP needs some study. Not deadly.
- Non-persistent client is simple. Fine to start.
- Persistence: ask Todd. Being done anyway.
- SDS data map - done. Needs updating (the one detailed at the end of this note is 6 months old) and needs putting in a Holy place; get instrumentation at least used to the idea of putting their best guess in a controlled place.

"LabView"

- Does the cool analysis Tom needs.

Figure 1: Data Flow Diagram for BPM LabView Connections



- Can drive a TCP/IP link.

Effort: TCP/IP has been done before. We need to work on

1. How to rendezvous with the Glish translator
2. What protocol to use talking to it. (KISS)

Make this shot at it simple and basic. See if it's promising. Compare to (e.g.) MatLab. If there is clear benefit, the Glish/LabView connection can be formalised - it is probably not a difficult job (connections to tcl/tk and to perl are already being done, for completely dissimilar reasons). But if it appears worth doing, its worth doing *right*.

"Drive VME memory"

- Memory ADO Exists.
- Using this ADO, we give meaning and context to the data transfers to VME by using data definitions accessed at the Unix level.

"Drive BPM"

- BPM ADO. Doesn't exist yet.
- When this is ready, the same data definition is used to give meaning to transfers - this time the information will be accepted by the ADO at compile or (probably) initialisation time.

The structure of the BPM Memory Map as it existed six months ago is detailed below. The updated version needs to be acquired from Tom.

BeamPosSds created Fri Aug 19 09:44:05 1994

ControlRegisters Structure

DelayCounterRead Long32

EventRead[8] Word
DelayCounterSet Long32
EventSet[8] Word
InitAvOrbitPointer[1](31) Long32 Bitfield
InitTBTPointer[1](30) Long32 Bitfield
ResetRevCounter[1](29) Long32 Bitfield
junkbits[29](0) Long32 Bitfield

OrbitBuffer[512] Structure

Pos1 Word
Pos2 Word
Charge1[14](18) Long32 Bitfield
Record1[2](16) Long32 Bitfield
Charge2[14](2) Long32 Bitfield
Record2[2](0) Long32 Bitfield

TBTBuffer[512] Structure

Pos1 Word
Pos2 Word
Charge1[14](18) Long32 Bitfield
Record1[2](16) Long32 Bitfield
Charge2[14](2) Long32 Bitfield
Record2[2](0) Long32 Bitfield

ConfigRegReadback Structure

RevCounter Long32
AvOrbPointer Long32
TBTPointer Long32
Frontend1Switches Word
Frontend2Switches Word
FixedDelay1 Word

FixedDelay2 Word
junk[3] Byte
BunchNumber Byte
CorrectionCoeff Word

ConfigRegSetting Structure

RevCounter Long32
AvOrbPointer Long32
TBTPointer Long32
Frontend1Switches Word
Frontend2Switches Word
FixedDelay1 Word
FixedDelay2 Word
junk[3] Byte
BunchNumber Byte
CorrectionCoeff Word