E10-2001-198

A.Yu.Isupov

DAQ SYSTEM FOR HIGH ENERGY POLARIMETER AT THE **LHE, JINR**: IMPLEMENTATION BASED ON THE *qdpb* (DATA PROCESSING WITH BRANCHPOINTS) SYSTEM

1 Introduction

Data acquisition system for the High Energy Polarimeter [1] (HEP) now implemented in the framework, provided by a distributed portable data acquisition and processing system *qdpb* [2]. This paper describes implementation of HEP DAQ, specific for such implementation software module – polarization values calculator *azhpol*, and its control utility *polar_cgi*.

Through the following text the file and software package names are highlighted as *italic text*, C language constructions and reproduced "as is" literals – as `typewriter text`. Reference to manual page named "qwerty" in the 9th section printed as ***qwerty(9)***, reference to section in this report – as 4.2.1. Subjects of substitution by actual values are enclosed in the angle brackets: `<polarimeter_name>` (exclude 4.2, where these are part of HTML syntax). All mentioned trademarks are properties of its respective owners.

2 HEP CAMAC hardware subsystem

Online electronics on the High Energy Polarimeter [1] based on a CAMAC hardware currently, so hardware subsystem in terms used by the *qdpb* [2] system is a CAMAC subsystem only.

2.1 HEP CAMAC configuration

Due to a very nonexpansive character of CAMAC hardware used by HEP we concentrate all CAMAC configuration code in the single C header file *azhpol_hardware.h*, which contains definitions of some constants and the following macros:

- `IREG_READ(u_short *ireg)` performs input register reading and fills `ireg` by obtained value.
- `CHECK_INTR(int *res)` checks interrupt validity, and fills `res` by 0 on success, > 0 on failure.
- `POLAR_MODE(u_short ireg, u_short *i)` by supplied `ireg` value recognizes polarization mode, calculates corresponding to it mode numbers and fills `i` by such value. Valid numbers are 0–2.
- `CAMAC_READ()` reads event from CAMAC.
- `CAMAC_INITIALIZE(int *res)` performs CAMAC initialization, enables CAMAC interrupt generation, and fills `res` by 0 on success, > 0 on failure.
- `CAMAC_CLEAN()` cleans CAMAC at errors in interrupt handler.
- `CAMAC_FINISH()` disables CAMAC interrupt generation and performs CAMAC finish operations.

2.2 HEP CAMAC interrupt handler implementation

User handler of CAMAC interrupts for the High Energy Polarimeter is implemented by:

- CAMAC configuration scheme, described in 2.1,
- KK009/KK012 memory access macros (see [4]) for CAMAC handling,
- kernel interface to packet handling (see [2]),
- kernel context interface to branch point (see [2]).

In the current implementation it named `azhpol` and loaded into kernel as follows:
`kldload azhpol_mod.ko`

For actual version of such handler source see *camac_modules/azhpol/azhpol.c* file in the HEP part of the *qdpb* system distribution tree.

`azhpolconf` – configuration utility for `azhpol` CAMAC module of the HEP DAQ system.

`azhpolconf [-v] [-d{<driver>|-}] <module>`
`azhpolconf [-v] -t <module>`

In the first synopsis form the *azhpolconf* utility configures the specified module `<module>` for work with driver "kk0" by default.

In the second synopsis form the *azhpolconf* utility tests configuration of the specified module `<module>` and writes it to the standard error output.

The default behavior of *azhpolconf* may be changed by following options:

`-d<driver>` Configure module for work with driver `<driver>` instead of default "kk0". `-d-` means use compiled–in default for `<driver>`. Default driver name can be changed at *azhpolconf* compilation time.

`-v` Produce verbose output instead of short by default.

The *azhpolconf* utility exits 0 on success, and > 0 on error.

`azhpoloper` – control utility for `azhpol` CAMAC module of the HEP DAQ system.
`azhpoloper [-v] [-b<#>] start|stop|status`

In the such synopsis form the *azhpoloper* performs `oper()` call with sub–function `fun`, defined by first supplied argument, on the CAMAC module (see also [4] for more details) attached to the 0^{th} branch, and writes report about that action to the standard error output. The *azhpoloper* may be used, for example, for implementation of some commands in the ***sv.conf(5)*** file (see 2.3).

The default behavior of the *azhpoloper* may be changed by following options:

`-v` Produce verbose output instead of short by default. With this option *azhpoloper* also uses `oper()` call with `HANDGETSTAT fun` (see also [4] for more details).

`-b<#>` Deal with module, attached to the `#`th branch, instead of 0^{th} by default.

The *azhpoloper* exits 0 on success, and nonzero on error.

2.3 Control over HEP DAQ

Because of a very compact nature of the HEP DAQ system we choose to eliminate supervisor with Graphics User Interface (GUI) – at least for a first stage of system.

For control over HEP DAQ used makefile *azhpolsv.conf*, so users can ***make (1)***'ing it directly without supervisors mediation. Target and variable names are selected to be self–explanatory (for details see an actual *azhpolsv.conf* file).

3 Polarimeter dependent code

To groups together all polarimeter dependent code the *libpol* library was designed. It contains all specific for some polarimeter `<polarimeter_name>`[1] routines (see 3.1) with corresponding low level support (see *bytemacros.h*), and routines for packet (event) types handling (see 3.2).

3.1 Data contents description

```
#define <polarimeter_name>
#include <ePOLAR.h>
```

The *ePOLAR.h* header file contains `#include`'s of all exists *e*`<polarimeter_name>`*.h* header files. Each one contains raw data layout description for the corresponding polarimeter `<polarimeter_name>`, namely:

- `#define NEVTYPES <#>`
  total number of event kinds, represented by packets of `POLAR_PACK` type (see 3.2);
- `#define NMEMB_MAX <#>`
  maximal total number of data fields, contained in a single packet of `POLAR_PACK` type;
- `static int max_memb[NEVTYPES]={ NMEMB_<event_kind>, ... };`[2]
- `static int len[NEVTYPES]={ SIZE_<event_kind>, ... };`
- `static u_short type[NEVTYPES]={ POLAR_<EVENT_KIND>, ... };`[3]
- `NEVTYPES` number of event kind descriptions, consists of following:
  - `#define NMEMB_<event_kind> <#>`
    total number of data fields, contained in single event of kind `<event_kind>`;
  - `#define SIZE_<event_kind> <#>`
    total length of event `<event_kind>` in binary representation (in bytes);

[1] `<polarimeter_name>` anywhere through this report need to be substituted by actual name of polarimeter (for example, "`azhpol`", "`lowpol`", etc.)

[2] `<event_kind>` need to be substituted by name of event kind (for example, "`cyc_end`", etc. – see also 3.2)

[3] `<EVENT_KIND>` need to be substituted by uppercased name of event kind (for example, "`CYC_END`", etc.)

- typedef struct {
  ...
  } <polarimeter_name>_<event_kind>;
  typedef'ed definition of <polarimeter_name>_<event_kind> structure itself, which is user representation of event <event_kind>.

The *fPOLAR.h* header file declares and the *f*<polarimeter_name>*.c* source file implements routines, destined to convert from the binary polarimeter data to corresponding data structures and vise versa.

```
#define <polarimeter_name>
#include <fPOLAR.h>
void b2s_<event_kind>(u_char *buf, char *data)
void s2b_<event_kind>(u_char *buf, char *data)
void (*b2s[NEVTYPES])(u_char *, char *) = { b2s_cyc_end };
void (*s2b[NEVTYPES])(u_char *, char *) = { s2b_cyc_end };
```

`b2s_<event_kind>()` fills already allocated `<polarimeter_name>_<event_kind>` structure, pointed by `*data`, from `*buf`, contains binary representation of event of kind `<event_kind>`. So, `b2s_<event_kind>()` performs conversion from binary to structure polarimeter data representation, after which user can retrieve event fields by its names instead of its offsets.

`s2b_<event_kind>()` fills already allocated buffer `*buf` from `<polarimeter_name>_<event_kind>` structure, pointed by `*data`. So, `s2b_<event_kind>()` performs conversion from structure to binary polarimeter data representation. These functions destined primarily for data makers (hardware subsystem) purposes.

3.2 HEP packet (event) classification

Packet classification mechanisms was originally designed for SPHERE *offline* system. Here we mention only items added to support HEP events. Namely, the *pack_types.h* header file must contains a following definitions:

```
/* Predefined packet types */
#define POLAR_PACK       ...
#define POLAR_RANGE      ...
/* Predefined packet classes in POLAR_PACK type */
#define POLAR_CYC       ...
#define POLAR_CYC_RANGE ...
/* Predefined packet kinds in POLAR_CYC class */
#define AZHPOL_CYC_BEG    ...
#define AZHPOL_CYC_END    ...
...
```

Also *pack_types.h* may contains the following macros, which obtains packet kind (`packet.header.type`, see [2]) as parameter and returns true, if such packet fits into some packet type or class, and false otherwise:

```
#define IS_POLAR_PACK(id)        ...
#define IS_POLAR_CYC(id)         ...
```

4 HEP specific software modules

HEP DAQ system uses at least ***bpget(1)*** service module, ***writer(1)*** work module, and optionally supervisor ***sv(1)*** control module (see also 2.3) from generic software modules assortment provided by the *qdpb* system [2]. Here we discuss only HEP specific software modules.

4.1 Work module: polarization calculator *azhpol*

The polarization values calculator *azhpol* implemented as work module of packet stream terminator type (in terms of the *qdpb* system [2]). Usually *azhpol* handled through HTTP by CGI–style utility *polar_cgi* (see 4.2) instead of launching directly.

```
azhpol [-l] [-t] [-b<bpstat>] [{-d{-|<dir>}|-D{-|<dir>}}]
       [-p{-|<pidfile>}] [-r<N>|-R<N>] [-a<n.nnn>]
       [-n{-|<statusfile>}] [-y<y.yyy>] [-Y<Y.YYY>]
       [-T{-|<targfile>}] -f{-|<ucommfile>} -m<mode>
```

In the such synopsis form the *azhpol* reads polarization packets with type `AZHPOL_CYC_END` (see 3.2) from standard input and user comments from file `<ucommfile>` (see 4.2.1), uses compiled–in target configuration list, performs polarization calculations over each `packet.data`, outputs results in ASCII and HTML representations to a couple of files (see 4.1.1) in the `<html_root>`*/static/azhpol*[4] directory, and issues commands to standard input of the ***gnuplot(1)*** child process.

The default behavior of the *azhpol* may be changed by following options:

`-b<bpstat>` Use branch point (see [2]) as input instead of standard input, and open it at `<bpstat>`[5] state. (Available only for systems, where branch point is implemented).

`-d<dir>` Use `<dir>` as name of "static" directory for output files. (Static means it must already exists and will be persist from run[6] to run. See also 5.) `-d-` means use compiled–in default for `<dir>`. (Mutually exclusive with `-D`).

`-D<dir>` Use `<dir>` as name of "dynamic" directory for output files. (Dynamic means an unique temporary name will be used and removed at run completion. See also 5.) `-D-` means use compiled–in default for `<dir>`. (Mutually exclusive with `-d`).

`-p<pidfile>` At startup write own process identifier (PID) in `<pidfile>`. `-p-` means use compiled–in default for `<pidfile>`.

[4] `<html_root>` anywhere through this report represents ***httpd(8)*** data directory.

[5] `<bpstat>` need to be substituted by `r` for "run", `s` for "stop", and `d` for "discard"

[6] Polarization calculations run (or simply run through this paper) is a one execution of one polarization calculator's instance.

`-r<N>` Read packets from no more than `<N>` bursts, terminate run, and restart itself with the same command string for producing next run. (Mutually exclusive with `-R`).

`-R<N>` Read packets from no more than `<N>` bursts, terminate run, and exit. (Mutually exclusive with `-r`).

`-a<n.nnn>` Terminate run if calculation accuracy less or equal than `<n.nnn>` will be reached. (Exit/restart behaviour controlled by `-R/-r`.)

`-n<statusfile>` Store an ASCII representation of current *azhpol*'s run number in the file `<statusfile>`. `-n-` means use compiled–in default for `<statusfile>`.

`-y<y.yyy>` An `<y.yyy>` value will be used for vector analysing power (A_y), see 4.1.3.

`-Y<Y.YYY>` An `<Y.YYY>` value will be used for tensor analysing power (A_{yy}), see 4.1.3.

`-T<targfile>` Read a targets configuration list for current *azhpol*'s run from the file `<targfile>`. `-T-` means use compiled–in default for `<targfile>`. See 4.1.2 for format description of this file.

`-f<ucommfile>` Read a user comments for current *azhpol*'s run from the file `<ucommfile>`. `-f-` means use compiled–in default for `<ucommfile>`. See 4.2.1 for format description of this file.

`-m<mode>` Polarimeter operation mode. Valid `<mode>` values are: "`vector3m`", "`vector3`", "`vector2`", "`tensor3m`", "`tensor2m`", "`vector3m+tensor3m`", "`vector3+tensor3m`", and "`vector2+tensor2m`".

The *azhpol* exits 0 on success, and > 0 on error. The *azhpol* ignores `SIGHUP`, `SIGINT` and `SIGQUIT` signals. For user termination in accuracy manner `SIGTERM` signal must be used.

4.1.1 Polarization calculator's output files

Here we describe polarization calculator's output files purposes.

current.dat contains current calculation results of the current *azhpol*'s run.

current.png graphical (PNG) version of previous.

current.sta contains current *azhpol*'s run number in ASCII representation.

history.dat contains final calculation results for all complete *azhpol*'s runs.

history.png graphical (PNG) version of previous.

index.html represents on HTML a current calculation results and some related information for the current *azhpol*'s run, or a final results if *azhpol* not runned.

4.1.2 Target configuration file

`<polarization_calculator> -Ttargets.conf ...`[7]

The *targets.conf* is a configuration file for ***azhpol(1)*** polarization calculator of the HEP DAQ system. Comment lines (lines with "`#`" in the first position) and empty lines are ignored. Other lines must contains two fields, separated by the equal sign "`=`".

First field contains target number and has form "`TARG<n>`", where first four characters matched as is (note upper case) and `<n>` is a decimal target number (valid are from 0 to 7).

Second field contains target name in a string representation and ends at end–of–line, so may consists of any characters, exclude newline. Special target name consists of doublequoted space ("`" "`") means unspecified target name.

At least eight valid lines (one per target) must be present to guarantee superseding ***azhpol(1)***'s compiled–in defaults.

4.1.3 Polarization calculations description

Here we describe briefly a polarization values [5] and it's calculations.

Polarized ions source POLARIS [6] can produce beam with two polarization modes (so called "+" and "−"), corresponding to $1 \rightarrow 4$ and $3 \rightarrow 6$ (for vector polarized ions) or $3 \rightarrow 5$ and $2 \rightarrow 6$ (for tensor polarized ions) source's state transitions, as well as nonpolarized beam (so called "0" polarization mode). Each accelerator burst has one of polarization mentioned above, from burst to burst beam polarization changes, and corresponding polarization marks are distributed to polarimeters by POLARIS electronics. POLARIS can provide two polarization burst by burst change regimes – so called three–marks (all possible modes are present) and two–marks (only "+" and "−" modes are present) regimes. Let us introduce some value definitions:

$NL^{+,-,0}$ Polarimeter's left arm scaler (coincidence of some scintillation counters) counts for some accelerator burst with "+", "−", "0" polarization mark, respectively.

$NR^{+,-,0}$ Polarimeter's right arm scaler (coincidence of some scintillation counters) counts for some accelerator burst with "+", "−", "0" polarization mark, respectively.

$NT^{+,-,0}$ Polarimeter's tensor scaler (some triggered system of scintillation counters under zero angle) counts for some accelerator burst with "+", "−", "0" polarization mark, respectively.

$M^{+,-,0}$ Polarimeter's beam monitor scaler (sparkle chamber) counts for some accelerator burst with "+", "−", "0" polarization mark, respectively.

A_y so called vector analysing power (some factor, depends on polarimeter's target nature and beam energy).

[7] `<polarization_calculator>` anywhere through this report need to be substituted by name of polarization calculator program (for example, "*azhpol*", "*lowpol*", etc.).

A_{yy} so called tensor analysing power (some factor, depends on polarimeter's target nature and beam energy).

For more than one bursts with corresponding polarization marks we can calculate average values of mentioned counts as cumulative sums, normalized to burst numbers $n^{+,-,0}$:

$$nx^{+,-,0} := < NX_n^{+,-,0} > = \frac{NX_1^{+,-,0} + \ldots + NX_n^{+,-,0}}{n^{+,-,0}} = \frac{NX_{cum}^{+,-,0}}{n^{+,-,0}} \quad .$$

Or we can use equivalent recurrent expressions for overflows avoiding:

$$nx^{+,-,0} = \frac{< NX_{n-1}^{+,-,0} > + NX_n^{+,-,0}}{n^{+,-,0}} \quad .$$

(Decision is made at *azhpol*'s compilation time by `#define`'ing `RECURR_CALC` variable.) All following values calculated in terms of such average count values $nl^{+,-,0}$, $nr^{+,-,0}$, $nt^{+,-,0}$, $m^{+,-,0}$.

Vector polarization $P_v^{+,-}$ and it's statistic errors $\Delta P_v^{+,-}$ calculated as follows:

Three–marks mode with monitor $M^{+,-,0}$ using (denoted as "`vector3m`" in 4.1):

$$rNL^{+,-} := \frac{NL^{+,-}}{NL^0} \; ; \quad rnl^{+,-} := \frac{nl^{+,-}}{nl^0} \; ;$$
$$rNR^{+,-} := \frac{NR^{+,-}}{NR^0} \; ; \quad rnr^{+,-} := \frac{nr^{+,-}}{nr^0} \; ;$$
$$rM^{+,-} := \frac{M^0}{M^{+,-}} \; ; \quad rm^{+,-} := \frac{m^0}{m^{+,-}} \; ;$$
$$P_v^{+,-} = \frac{(rNL^{+,-} - rNR^{+,-}) \times rM^{+,-}}{3 \times A_y}$$
$$= \frac{(rnl^{+,-} - rnr^{+,-}) \times rm^{+,-}}{3 \times A_y} \; ;$$
$$\Delta P_v^{+,-} = \sqrt{\frac{rNL^{+,-}}{NL^0} + \frac{rNR^{+,-}}{NR^0}} \times \frac{rM^{+,-}}{3 \times A_y}$$
$$= \sqrt{\frac{rnl^{+,-}}{nl^0} + \frac{rnr^{+,-}}{nr^0}} \times \frac{rm^{+,-}}{3 \times A_y \times \sqrt{n^{+,-}}} \quad .$$

Three–marks mode without monitor $M^{+,-,0}$ using (denoted as "`vector3`" in 4.1):

$$rNL^{+,-} := \frac{NL^{+,-}}{NL^0} \; ; \quad rnl^{+,-} := \frac{nl^{+,-}}{nl^0} \; ;$$
$$rNR^{+,-} := \frac{NR^{+,-}}{NR^0} \; ; \quad rnr^{+,-} := \frac{nr^{+,-}}{nr^0} \; ;$$

$$P_v^{+,-} = \frac{rNL^{+,-} - rNR^{+,-}}{(rNL^{+,-} + rNR^{+,-}) \times A_y}$$
$$= \frac{rnl^{+,-} - rnr^{+,-}}{(rnl^{+,-} + rnr^{+,-}) \times A_y} \quad ;$$
$$\Delta\epsilon^{+,-,0} := \frac{2\sqrt{NL^{+,-,0} \times NR^{+,-,0}}}{(NL^{+,-,0} + NR^{+,-,0}) \times \sqrt{NL^{+,-,0} + NR^{+,-,0}}}$$
$$= \frac{2\sqrt{nl^{+,-,0} \times nr^{+,-,0}}}{(nl^{+,-,0} + nr^{+,-,0}) \times \sqrt{nl^{+,-,0} + nr^{+,-,0}} \times \sqrt{n^{+,-,0}}} \quad ;$$
$$\Delta P_v^{+,-} = \frac{\sqrt{(\Delta\epsilon^{+,-})^2 + (\Delta\epsilon^0)^2}}{A_y} \quad .$$

Two–marks mode without monitor $M^{+,-,0}$ using (denoted as "vector2" in 4.1):

$$P_v^{+,-} = \frac{NL^{+,-} - NR^{+,-}}{(NL^{+,-} + NR^{+,-}) \times A_y}$$
$$= \frac{nl^{+,-} - nr^{+,-}}{(nl^{+,-} + nr^{+,-}) \times A_y} \quad ;$$
$$\Delta P_v^{+,-} = \frac{2\sqrt{NL^{+,-} \times NR^{+,-}}}{(NL^{+,-} + NR^{+,-}) \times \sqrt{NL^{+,-} + NR^{+,-}} \times A_y}$$
$$= \frac{2\sqrt{nl^{+,-} \times nr^{+,-}}}{(nl^{+,-} + nr^{+,-}) \times \sqrt{NL^{+,-} + NR^{+,-}} \times A_y \times \sqrt{n^{+,-}}} \quad .$$

Tensor polarization $P_t^{+,-}$ and it's statistic errors $\Delta P_t^{+,-}$ calculated as follows:

Three–marks mode with monitor $M^{+,-,0}$ using (denoted as "tensor3m" in 4.1):

$$rNT^{+,-} := \frac{NT^{+,-}}{NT^0} \; ; \quad rnt^{+,-} := \frac{nt^{+,-}}{nt^0} \; ;$$
$$rM^{+,-} := \frac{M^0}{M^{+,-}} \; ; \quad rm^{+,-} := \frac{m^0}{m^{+,-}} \; ;$$
$$P_t^{+,-} = (rNT^{+,-} \times rM^{+,-} - 1) \times A_{yy}$$
$$= (rnt^{+,-} \times rm^{+,-} - 1) \times A_{yy} \; ;$$
$$\Delta P_t^{+,-} = \frac{A_{yy} \times rM^{+,-} \times \sqrt{NT^{+,-} \times (NT^0)^2 + NT^0 \times (NT^{+,-})^2}}{(NT^0)^2}$$
$$= \frac{A_{yy} \times rm^{+,-} \times \sqrt{nt^{+,-} \times \sqrt{n^0} \times (nt^0)^2 + nt^0 \times \sqrt{n^{+,-}} \times (nt^{+,-})^2}}{\sqrt{n^{+,-} \times n^0} \times (nt^0)^2} \quad .$$

Two–marks mode with monitor $M^{+,-,0}$ using (denoted as "tensor2m" in 4.1):

$$r^{+,-} := \frac{NT^{+,-}}{NT^{-,+}} \times \frac{M^{-,+}}{M^{+,-}} = \frac{nt^{+,-}}{nt^{-,+}} \times \frac{m^{-,+}}{m^{+,-}} \;\; ;$$

$$\Rightarrow r^{+} \equiv 1/r^{-} \;\; ;$$

$$P_t^{+,-} = A_{yy} \times \frac{r^{+,-}-1}{r^{+,-}+1} \;\; ;$$

$$\Rightarrow P_t^{+} \equiv -P_t^{-} \;\; ;$$

$$\Delta r^{+,-} := \frac{M^{-,+}}{M^{+,-}} \times \frac{\sqrt{NT^{+,-}\times(NT^{-,+})^2+NT^{-,+}\times(NT^{+,-})^2}}{(NT^{-,+})^2}$$

$$= \frac{m^{-,+}}{m^{+,-}} \times \frac{\sqrt{nt^{+,-}\times\sqrt{n^{-,+}}\times(nt^{-,+})^2+NT^{-,+}\times\sqrt{n^{+,-}}\times(NT^{+,-})^2}}{\sqrt{n^{+,-}\times n^{-,+}}\times(NT^{-,+})^2} \;\; ;$$

$$\Rightarrow \Delta r^{-} \times (r^{+})^2 \equiv \Delta r^{+} \;\; ;$$

$$\Rightarrow \Delta r^{+} \times (r^{-})^2 \equiv \Delta r^{-} \;\; ;$$

$$\Delta P_t^{+,-} = \frac{2\times A_{yy}\times\Delta r^{+,-}}{(r^{+,-}+1)^2} \;\; ;$$

$$\Rightarrow \Delta P_t^{+} \equiv \Delta P_t^{-} \;\; .$$

Above we assume that statistic errors for $M^{+,-,0}$ are negligible small in comparison with ones for scalers.

4.2 Control module: *polar_cgi* utility

polar_cgi – CGI–style control utility for polarization calculators of the HEP DAQ system.

<FORM ACTION="polar_cgi">[8]

The *polar_cgi* processes HTML <FORM> tag and produces HTML output in responce.

Polarization calculator's launch

If processed form contains input field with NAME="chcomm" and VALUE="No", *polar_cgi* launches polarization calculator (it's name, for example "azhpol", submitted in field with NAME="dir"), so calculator's run will be started, and produces HTML output contains link to calculator's HTML output, named *index.html* by default (see 4.1.1).

Produce form for user comments change

If processed form contains input field with NAME="chcomm" and VALUE="Yes", *polar_cgi* reads a current user comments file (see 4.2.1), using filename submitted in field with NAME="currcommfile" (by default – *current.txt*) and produces HTML output provides form for user comments change (contains hidden input field NAME="chcomm" and VALUE="Change"), where default values obtained from user comments file.

[8]During this chapter angle brackets are part of HTML syntax instead of request to substitute enclosed variable by actual value, as through whole present paper.

User comments change

If processed form contains input field with `NAME="chcomm"` and `VALUE="Change"`, *polar_cgi* writes a user comments file (see 4.2.1), using filename submitted in field with `NAME="currcommfile"` (by default – *current.txt*), and produces HTML output, which redirects back to main HTML form for polarization calculator's launch.

Polarization calculator's stop

If processed form contains input field with `NAME="stop"`, *polar_cgi* sends `SIGTERM` signal to process with PID submitted in field with `NAME="pid"` (such special form may be issued by polarization calculator itself), and produces corresponding HTML output.

View history of already complete polarization calculation runs

If processed form contains input field with `NAME="runnumber"`, *polar_cgi* processes history files (by default – *history.dat* and *history.png*, see 4.1.1) of polarization calculator with name submitted in field with `NAME="dir"`, and produces HTML output contains calculation results and some related information (for example, user comments) corresponding to run with requested number.

If some of mentioned above actions failed, *polar_cgi* produces HTML output with error notification.

Security issues

All HTML forms must have `METHOD="POST"`, except for history viewing form, which must have `METHOD="GET"`. Due to such design we can use different authorization scheme for actions with "control" and "view" nature.

4.2.1 User comments file

`<polarization_calculator> -fcurrent.txt ...`

The *current.txt* is a user comments file for ***azhpol(1)*** polarization calculator of the HEP DAQ system. It produced by CGI–style control utility ***polar_cgi(1)*** in accordance with user submitted HTML form contains input field with `NAME="chcomm"`, in particular at polarization calculator launch. The *current.txt* is read by such polarization calculator at startup, termination, and optionally at obtaining each packet of `<POLARIMETER_NAME>_CYC_END` type.

The *current.txt* consists of one line, contains one or more fields, separated by pipe characters ("|"). Fields filled by `VALUE`s with `NAME="ucomm<N>"`, where `<N>`$= 0, 1, \ldots, N_{max}$ continuously, submitted in the input HTML form.

Currently the following correspondences between fields and meanings are established:

`"ucomm0"` First field is mandatory and represents operation mode of the polarized ions source POLARIS [6]. Valid values are: "`vector`" and "`tensor`".

Second and next fields are optional.

`"ucomm1"` Second field represents polarized beam energy E in MeV/nucleon.

`"ucomm2"` Third field represents potential $U_{spin.p.}$ in Volts.

`"ucomm3"` Fourth field represents current $I_{magn.}$ in Ampers.

`"ucomm4"` Fifth field represents current $I_{sol.}$ in Ampers.

`"ucomm5"` Sixth field represents current $I_{g-pol.}$ in Ampers.

Fields from seventh up to tenth represents values for "+" transition mode (means "1–4" for vector mode and "3–5" for tensor mode) of the polarized ions source POLARIS, namely:

`"ucomm6"` seventh field represents frequency F in MHz;

`"ucomm7"` eighth field represents potential U_{ps} in Volts;

`"ucomm8"` nineth field represents potential U_{fb} in divisions of scale;

`"ucomm9"` tenth field represents current $I_{corr.}$ in Ampers.

Fields from eleventh up to fourteenth represents values for "–" transition mode (means "3–6" for vector mode and "2–6" for tensor mode) of the polarized ions source POLARIS, namely:

`"ucomm10"` eleventh field represents frequency F in MHz;

`"ucomm11"` twelveth field represents potential U_{ps} in Volts;

`"ucomm12"` thirteenth field represents potential U_{fb} in divisions of scale;

`"ucomm13"` fourteenth field represents current $I_{corr.}$ in Ampers.

`"ucomm14"` fifteenth field represents polarimeter angle Θ in degrees.

`"ucomm15"` sixteenth field represents user comments itself (in the narrow meaning – what duty personnel want to say about current polarization calculator's run).

5 HTTP access to polarization calculation results

Polarization calculation results for High Energy Polarimeter of LHE, JINR are accumulated on HTTP server `crab.jinr.ru` (`159.93.18.110`), so user needs only HTTP client (WWW–browser) of any flavour (*Netscape*, *MSIE*, ..., without graphics information – *lynx*).

In according to *azhpol* polarization calculator (see 4.1) functionality three working modes are possible: so called static runs[9], dynamic runs, and viewing of current static run results. Viewing of history of already terminated runs is possible also.

1. Static run mode purpose is to perform authoritative polarization calculations by corresponding polarimeter personnel: in the *static/azhpol* directory – for High Energy Polarimeter. Only one static run for High Energy Polarimeter is permitted simultaneously. Any authorised (by HTTP username/password) user can launch static run, terminate (already started) one, change (at any time) current comments (see 4.2.1). Static run may be launched with termination condition(s) provided (with and without request to start next runs with the same conditions automatically after previous one termination) and without one(s), for all cases an immediate termination (by Stop button in the current run's HTML form) permitted for any authorised user. Following run termination conditions can be requested:
   (a) required number of accelerator bursts reaching,
   (b) required calculation accuracy reaching.

[9]Term "run" introduced in 4.1.

Request for second condition alone not permitted. If requested both, satisfaction of only one enough for run termination. A page `http://crab.jinr.ru/~camac/static` contains start form for static run, after submittion user obtains page with link to static run results `http://crab.jinr.ru/~camac/static/azhpol`. At change comments request user obtains separate page and after submittion returns to start page. Start page also contains form to history view.

2. Dynamic run mode purpose is to perform private polarization calculations by experimental setup personnel. More than one dynamic run for High Energy Polarimeter is permitted simultaneously (limited by system implementation). Any user (possibly not from any host) can launch dynamic run. Only the same user can terminate it (by Stop button in the HTML form corresponding to this dynamic run). Dynamic run may be launched only with termination condition(s) provided (and only without request to start next runs automatically). Dynamic run termination conditions and logic of its application the same as for static run (see above). A page `http://crab.jinr.ru/~camac/dynamic` contains start form for dynamic run, after submittion user obtains page with link to dynamic run results (unpredictable temporary name will be used, so do not forget it or you can't see your dynamic run results). Dynamic run results deleted just after its termination, so be carefull to save (by your HTTP client) dynamic run's final page as soon as possible, if you need it. Due to such nature of dynamic run a change comments and history view services are not provided for it.
3. Purpose of current static run results viewing is to provide authoritative polarization calculation results for experimental setup personnel. Any user (possibly not from any host) can view a corresponding page `http://crab.jinr.ru/~camac/static/azhpol`, which also contains form to history view.

6 HEP DAQ system in work

The High Energy Polarimeter's DAQ system was tested under real work conditions with deuteron polarized beams during the June'2001 run of Dubna Synchrophasotron. Although some bugs are found and fixed and some useful extensions are added, in general HEP DAQ works stably and reliably during approximately seventy hours.

7 Software dependencies and portability

The HEP DAQ system uses subset of third–party software, used by the *qdpb* system, currently as follows:

camac2 package – CAMAC subsystem [4] implementation (see also 2.2).

netpipes package – ready implementation of some service modules (see also [2]).

All software packages mentioned above are free distributable, so does not limits a portability of the HEP DAQ system.

All HEP DAQ code written on C for more or less generic modern UNIX–like environment. All code for user context processes expected to be easy portable to OS's other than FreeBSD.

Packet bodies contents always produced in the little endian byte order.

All code for kernel context organized as devices, pseudo–devices, and loadable kernel modules and implements system calls and hardware interrupt handlers – all are typical kernel objects at modern UNIX–like OS's.

The HEP DAQ system's sources are a some subset of the *qdpb* system's source tree, so all notes about *qdpb* [2] ***make(1)***'ing are valid for the HEP DAQ.

In the current implementation the HEP DAQ system based on the FreeBSD operating system (4.x version at present time). HEP DAQ sources are organized as two archives `polarqdpb-<yyyymmdd>.tar.gz`[10] (for generic polarimetry code) and `azhpolqdpb-<yyyymmdd>.tar.gz` (for specific High Energy Polarimeter code) (please contact `isupov@moonhe.jinr.ru`), both need to be extracted over *qdpb* system [2] source distribution (in such order).

Acknowledgements

Author have a pleasure to thank to Yu.K.Pilipenko for initiation of presented developments, to S.G.Reznikov for useful discussions, help in testing and some code contribution, to L.S.Zolin, G.D.Stoletov and V.N.Zhmyrov for help with polarization calculations, and to A.G.Litvinenko for moral support.

References

[1] Azhgirey L.S. et al. Pribory i Tekhnika Eksperimenta (in Russian), 1997, N.1, p.51.

[2] Gritsaj K.I., Isupov A.Yu. **A trial of distributed portable data acquisition and processing system implementation: the *qdpb* – Data Processing with Branchpoints.** Communication of JINR E10–2001–116, Dubna, 2001.

[3] Gritsaj K.I., Olshevsky V.G. **Software Package for Work with CAMAC in Operating System FreeBSD.** Communication of JINR P10–98–163 (in Russian), Dubna, 1998.

[4] Gritsaj K.I. Private communications.

[5] Haeberli W. Ann. Rev. Nucl. Sci., v.17, 1967, p.373.

[10] `<yyyymmdd>` means some actual year, month, and day numbers.

[6] Anishchenko N.G. et al. In: Proc. of the 5–th Int. Symp. on High Energy Spin Physics (Brookhaven, 1982), AIP Conf. Proc., v.95, AIP, New York, 1983, p.445.
Belushkina A.A. et al. In: Proc. of the 7–th Int. Symp. on High Energy Spin Physics (Protvino, 1986), v.2, IHEP, Serpukhov, 1987, p.215.

Received by Publishing Department
on September 26, 2001.

Исупов А.Ю. E10-2001-198
Реализация системы сбора данных для высокоэнергетического поляриметра ЛВЭ ОИЯИ на основе системы *qdpb*
(система обработки данных с точками ветвления)

Описывается реализация системы сбора данных высокоэнергетического поляриметра ЛВЭ ОИЯИ, построенной на основе системы *qdpb*. Обсуждаются специфические для данной реализации программные модули (зависящие от аппаратного состава и характера собираемой информации высокоэнергетического поляриметра).

Работа выполнена в Лаборатории высоких энергий ОИЯИ.

Сообщение Объединенного института ядерных исследований. Дубна, 2001

Isupov A.Yu. E10-2001-198
DAQ System for High Energy Polarimeter at the LHE, JINR: Implementation Based on the *qdpb*
(Data Processing with Branchpoints) System

Online data acquisition (DAQ) system's implementation for the High Energy Polarimeter (HEP) at the LHE, JINR is described. HEP DAQ is based on the *qdpb* system. Software modules specific for such implementation (HEP data and hardware dependent) are discussed.

The investigation has been performed at the Laboratory of High Energies, JINR.

Communication of the Joint Institute for Nuclear Research. Dubna, 2001

Макет Т.Е.Попеко

Подписано в печать 11.10.2001
Формат 60 × 90/16. Офсетная печать. Уч.-изд. л. 1,9
Тираж 300. Заказ 52895. Цена 1 р. 30 к.

Издательский отдел Объединенного института ядерных исследований
Дубна Московской области