3x1x1 SlowControlDB access from WA105soft

Status report

Yuriy Onishchuk

Kiev Taras Shevchenko National University

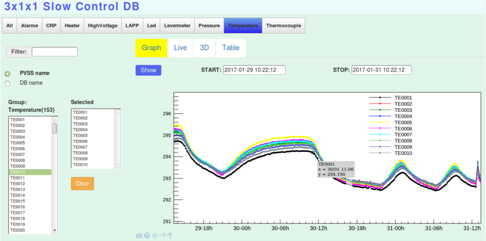
01-Feb-2016

General remarks

- Argon filling of the 3x1x1 assembly will happen during short period
- Copy of Slow Control database to use in off-line analysis is created
- Web-display of the Slow Control DB sensors is in operation status
- Access to sensor values will allow people performing own analysis on the 3x1x1 data
- Library allowing to query the database from Qscan is under design

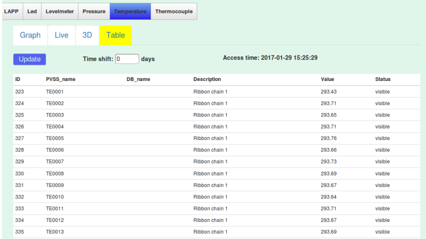
3x1x1 Slow Control DB short description

- mysql based DB consists of ∼660 tables
- Each table describes an individual sensor and has simple structure with two fields: date and value
- Slow and Fast modes to put values into the DB: ∼600 and ∼$6 - 30$ sec. Automatic switching between modes depends on value changes and occurs independently for each sensor
- Additional table PARAM_NAME determines an internal relationship between sensor groups and short descriptions
- Stand-alone and web variants of the DisplayDB have been designed using C++ and Python mysql API
- Web DisplayDB (https://wa105data.web.cern.ch/wa105data/) is a good tool to study sensor evidence and clarify own knowledge: multi-graphs for temporal evolution, alive monitoring, lookup table description, $3D$ view



START: 2017-01-29 10:22:12

STOP: 2017-01-31 10:22:12



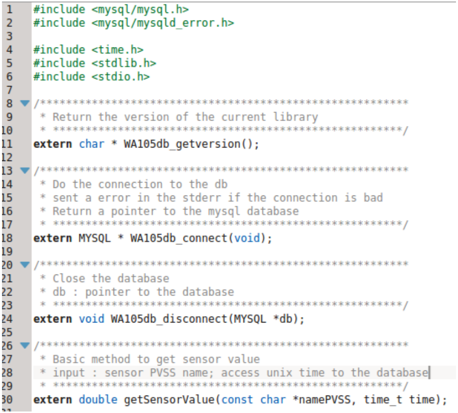
Time shift: 0 days

Access time: 2017-01-29 15:25:29

ID	PVSS_name	DB_name	Description	Value	Status
323	TE0001		Ribbon chain 1	293.43	visible
324	TE0002		Ribbon chain 1	293.71	visible
325	TE0003		Ribbon chain 1	293.65	visible
326	TE0004		Ribbon chain 1	293.71	visible
327	TE0005		Ribbon chain 1	293.76	visible
328	TE0006		Ribbon chain 1	293.66	visible
329	TE0007		Ribbon chain 1	293.73	visible
330	TE0008		Ribbon chain 1	293.69	visible
331	TE0009		Ribbon chain 1	293.67	visible
332	TE0010		Ribbon chain 1	293.64	visible
333	TE0011		Ribbon chain 1	293.71	visible
334	TE0012		Ribbon chain 1	293.67	visible
335	TE0013		Ribbon chain 1	293.69	visible

Library libwa105db

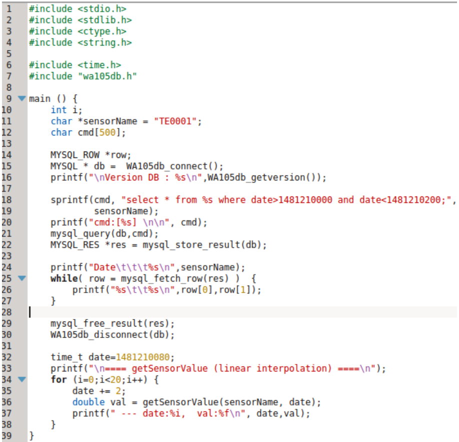
- libwa105db contains statically and dynamically compiled libraries (.o, .a and .so) and proper .c, .h source files
- Basic getter-function, getSensorValue(), returns sensor value wrt PVSS sensor name and unix-time as result of linear interpolation between closest time knots.
- Switched off sensors returns 0. Off criterion is absence of time knots within 1800 s.
- Header file, wa105db.h, shows how to use library functions



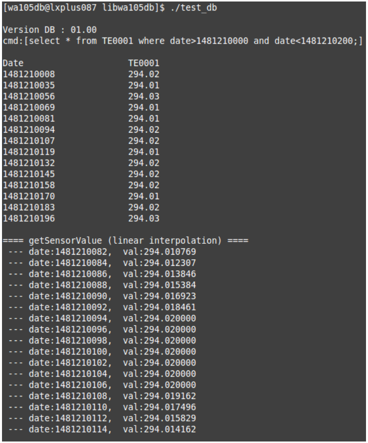
```
#include <mysql/mysql.h>
#include <mysql/mysqld_error.h>

#include <time.h>
#include <stdlib.h>
#include <stdio.h>

/************************************************************
 * Return the version of the current library
 * *********************************************************/
extern char * WA105db_getversion();

/************************************************************
 * Do the connection to the db
 * sent a error in the stderr if the connection is bad
 * Return a pointer to the mysql database
 * *********************************************************/
extern MYSQL * WA105db_connect(void);

/************************************************************
 * Close the database
 * db : pointer to the database
 * *********************************************************/
extern void WA105db_disconnect(MYSQL *db);

/************************************************************
 * Basic method to get sensor value
 * input : sensor PVSS name; access unix time to the database
 * *********************************************************/
extern double getSensorValue(const char *namePVSS, time_t time);
```

Library functions test

- Test file, test_db.c, shows variants of library function usage



```c
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <string.h>

#include <time.h>
#include "wa105db.h"

main () {
    int i;
    char *sensorName = "TE0001";
    char cmd[500];

    MYSQL_ROW *row;
    MYSQL * db =  WA105db_connect();
    printf("\nVersion DB : %s\n",WA105db_getversion());

    sprintf(cmd, "select * from %s where date>1481210000 and date<1481210200;",
            sensorName);
    printf("cmd:[%s] \n\n", cmd);
    mysql_query(db,cmd);
    MYSQL_RES *res = mysql_store_result(db);

    printf("Date\t\t\t%s\n",sensorName);
    while( row = mysql_fetch_row(res) )  {
        printf("%s\t\t%s\n",row[0],row[1]);
    }

    mysql_free_result(res);
    WA105db_disconnect(db);

    time_t date=1481210080;
    printf("\n==== getSensorValue (linear interpolation) ====\n");
    for (i=0;i<20;i++) {
        date += 2;
        double val = getSensorValue(sensorName, date);
        printf(" --- date:%i,  val:%f\n", date,val);
    }
}
```



```
[wa105db@lxplus087 libwa105db]$ ./test_db

Version DB : 01.00
cmd:[select * from TE0001 where date>1481210000 and date<1481210200;]

Date                    TE0001
1481210008              294.02
1481210035              294.01
1481210056              294.03
1481210069              294.01
1481210081              294.01
1481210094              294.02
1481210107              294.02
1481210119              294.01
1481210132              294.02
1481210145              294.02
1481210158              294.02
1481210170              294.01
1481210183              294.02
1481210196              294.03

==== getSensorValue (linear interpolation) ====
 --- date:1481210082,  val:294.010769
 --- date:1481210084,  val:294.012307
 --- date:1481210086,  val:294.013846
 --- date:1481210088,  val:294.015384
 --- date:1481210090,  val:294.016923
 --- date:1481210092,  val:294.018461
 --- date:1481210094,  val:294.020000
 --- date:1481210096,  val:294.020000
 --- date:1481210098,  val:294.020000
 --- date:1481210100,  val:294.020000
 --- date:1481210102,  val:294.020000
 --- date:1481210104,  val:294.020000
 --- date:1481210106,  val:294.020000
 --- date:1481210108,  val:294.019162
 --- date:1481210110,  val:294.017496
 --- date:1481210112,  val:294.015829
 --- date:1481210114,  val:294.014162
```

- Qscan/WA105Soft user could use the libwa105db library in two ways:
  - Develop own approach to process DB311 data after the database connection/disconnection. This approach based on detailed knowledge of mysql C++ API
  - Simplified way using getSensorValue() function

Conclusions & Next steps

Conclusions

- Draft variant of the library allowing to query the database from Qscan/WA105Soft has desined
- Short decription of the library has been done
- Test example showed variants of the library usage

Next steps

- More sophisticated library functions design
- Continue process of sensor description

Many thanks to
Thierry for his help