Alarms and alh

Carlos Galdino

Last updated: 1 March 2016

- In this tutorial I will talk about alarms and the alarm handler extension (alh). I will build a simple example using the calculator we built in a previous tutorial.
- This tutorial is based on:
    - Records concepts – Alarms specification

https://wiki-ext.aps.anl.gov/epics/index.php/RRM_3-14_Concepts#Alarm_Specification

    - Common record fields

https://wiki-ext.aps.anl.gov/epics/index.php/RRM_3-14_dbCommon#Alarm_Fields

    - Alarm handler documentation

http://www.aps.anl.gov/epics/extensions/alh/index.php

- This tutorial is a hands-on tutorial and I highly recommend you to read the documentation for more information.
- Don't hesitate to send me questions, comments and suggestions.

Editing record fields

We've talked about editing the VAL field in the "build a calculator" tutorial using the `dbpf` command. Now we are going to edit other field related to alarms.

There 2 ways of editing a field: 1) using the `dbpf` command and 2) editing it directly in database file (myCalculator.db). The first method is used to edit fields when the IOC is executing and the changes made are reset to default each time you turn it on and off. The second method is used to edit the default values of a field.

Open the db file (C:\epics\myCalculator\db – Notice I'm on windows). We are going to work on the record number1.

Just to warm up add a description for the record. To do this add the following line to the file:

```
field(DESC, "Number 1 of my beautiful epics calculator")
```

Your edit file should be something like this:

```
record(ai, number1) {
  field(FLNK, "number3")
  field(DESC, "Number 1 of my beautiful epics calculator")
}

record(ai, number2) {
  field(FLNK, "number3")
}

record(calc, number3) {
  field(CALC, "A+B")
```

```
  field(INPA, "number1")
  field(INPB, "number2")
}
```

There might be a lot of lines starting with #!. This lines are VisualDCT parameters, never edit them.

Run the IOC (refer to the “build calculator” tutorial to see how) and use the dbpr command to check the DESC field of record number1.

Alarm fields

It’s easier to understand alarms using examples. We are going to set na alarm on number1 record. If number1.VAL goes higher than 30 I want a MINOR alarm. If it goes higher than 100 I want a MAJOR alarm.

So we are going to set the HIGH and HSV fields to:

HIGH = 30

HSV = MINOR

And the HIHI and HHSV fiedls to:

HIHI = 100

HHSV = MAJOR

To do this add the following lines to number1, like we did in last topic.

```
field(HIGH, 30)
field(HSV, MINOR)
field(HIHI, 100)
field(HHSV, MAJOR)
```

Run the IOC and use the command:

```
dbpr number1 4
```

To check the field you edited (DESC, HIGH, HSV, HIHI, HHSV) and also check the fields STAT and SEVR. You will see that: STAT = NO_ALARM and SEVR = NO_ALARM.

Edit the value of number1 to 35. You can use the MEDM interface we did i a previous tutorial or use the command:

```
dbpf number1 35
```

Check STAT and SEVR again, you will see that STAT = HIGH and SEVR = MINOR. As you may have noticed, STAT stands for the alarm status and ere you have a HIGH ALARM. STAT can have several values depending on the error type (list copied from https://wiki-ext.aps.anl.gov/epics/index.php/RRM_3-14_Concepts#Alarm_Specification):

- NO_ALARM:This record is not in alarm
- READ:An INPUT link failed in the device support
- WRITE:An OUTPUT link failed in the device support
- HIHI:An analog value limit alarm
- HIGH:An analog value limit alarm
- LOLO:An analog value limit alarm

- LOW:An analog value limit alarm
- STATE:An digital value state alarm
- COS:An digital value change of state alarm
- COMM:A device support alarm that indicates the device is not communicating
- TIMEOUT:A device sup alarm that indicates the asynchronous device timed out
- HWLIMIT:A device sup alarm that indicates a hardware limit alarm
- CALC:A record support alarm for calculation records indicating a bad calulation
- SCAN:An invalid SCAN field is entered
- LINK:Soft device support for a link failed:no record, bad field, invalid conversion, INVALID alarm severity on the referenced record.
- SOFT
- BAD_SUB
- UDF
- DISABLE
- SIMM
- READ_ACCESS
- WRITE_ACCESS

The SEVR is the severity of the alarm, here we have a MINOR ALARM. The avaible values are:

- NO_ALARM
- MINOR
- MAJOR
- INVALID -- communication erros

Now change the value of number 1 from 35 to 147, you will see that the alarm STAT is HIHI and the Severity is MAJOR.

Good! Now we have set our first alarm. Use the follwing fields list to check for other alarm limits like the LOW alarm (notice that specfic types of records might have extra fields related to alarm conditions):

HIGH – High alarm limit

HSV – High alarm severity

HIHI - High High alarm limit

HHSV – High High alarm severity

LOW – low alarm limit

LSV – low alarm severity

LOLO – low low alarm limit

LLSV – low low alarm severity

STAT (read only) – Read the alarm status

SEVR (read only) – Read the alarm severity

By now, you don’t need to worry about NSTA, NSEV, ACKS, ACTS and UDF.

NSTA (read only)

NSEV (read only)

NSTA and NSEV are the fields the database access, record support, and device support use to set new alarm status and severity values. Whenever any software component discovers an alarm condition, it uses the following macro function: recGblSetSevr(precord,new_status,new_severity) This ensures that the current alarm severity is set equal to the highest outstanding alarm. The file alarm.h defines all allowed alarm status and severity values – Paragraph copies from https://wiki-ext.aps.anl.gov/epics/index.php/RRM_3-14_dbCommon#Alarm_Fields .

ACKS (read only) - Highest severity unacknowledged alarm

ACKT (read only) - transient acknowledge alarms

UDF (read only) - indicates that the record has never been processed or its value is undfined.

Using ALARM HANDLER (alh)

On Linux download the alh package on
http://www.aps.anl.gov/epics/extensions/alh/index.php

Unzip it in the folder `/opt/epics/extensions/src.` Notice you need to have the extensions package installed. If you don't have, refer to the "Graphic User Interface" tutorial. Add the alh bin folder to the path and use the command `alh` to start the alarm handler.

On Windows, you have already installed it when you downloaded the Windows EPICS Tools to use MEDM in the "Graphic User Interface". Just open the windows epics tools folder and clik in alh.exe.

But don't start it yet, first we will write a configuration file. Open a text editor and write

```
GROUP NULL  maingroup1
       GROUP maingroup1 group1
              CHANNEL group1 number1
              CHANNEL group1 number2
       GROUP maingroup1 group2
              CHANNEL group2 number3
```

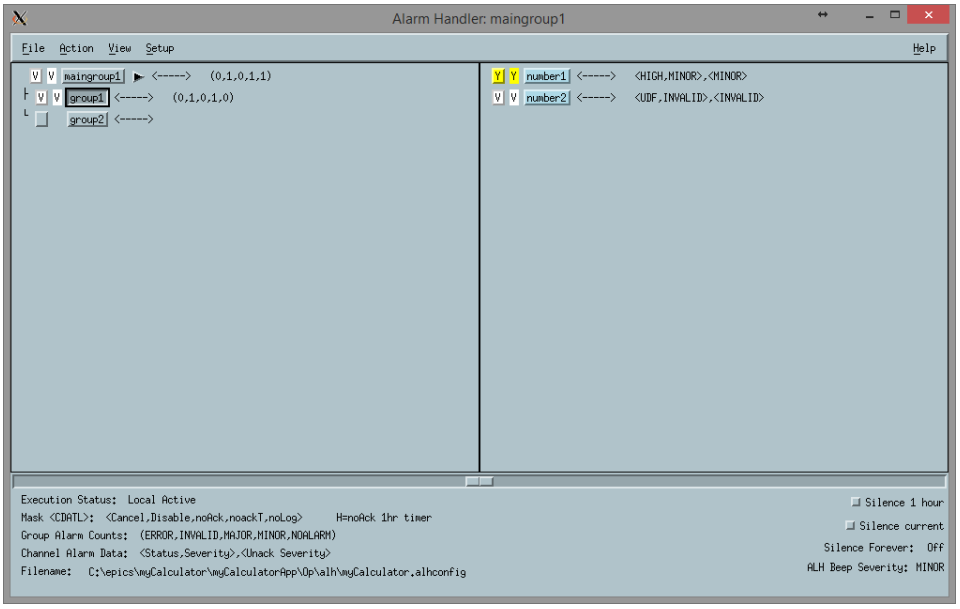
Save this file with the name myCalculator.alhconfig. I recomment it to save it in the path C:\epics\myCalculator\myCalculatorApp\Op\alh. You will probably have to create a new folder called alh.

This folder tell alh that the main group is maingroup1 (notice that NULL afert GROUP means this group has no parets, in other words, this is the "higest group level"). You can only have one main group like this.

The second line sets a group inside maingroup1. This group is called group1 and has the records number1 and number2.

The group group2 has the record number3.

Now, start alh and select the myCalculator.alhconfig file.



You can see the tree we have created with the maingroup1, group1 and group2. You can also see the MINOR ALARM (color yellow) in record number1.

Record number2 might have a UDF alarm because we haven't entered a valid value for it yet. Click the first square before the record name to acknowledge it.

Here we have set the basic steps for using alarms in your applications. Refer to the user guide for more information about the configuration file and about the interface

http://www.aps.anl.gov/epics/EpicsDocumentation/ExtensionsManuals/AlarmHandler/alhUserGuide-1.2.35/ALHUserGuide.html#5_2_2