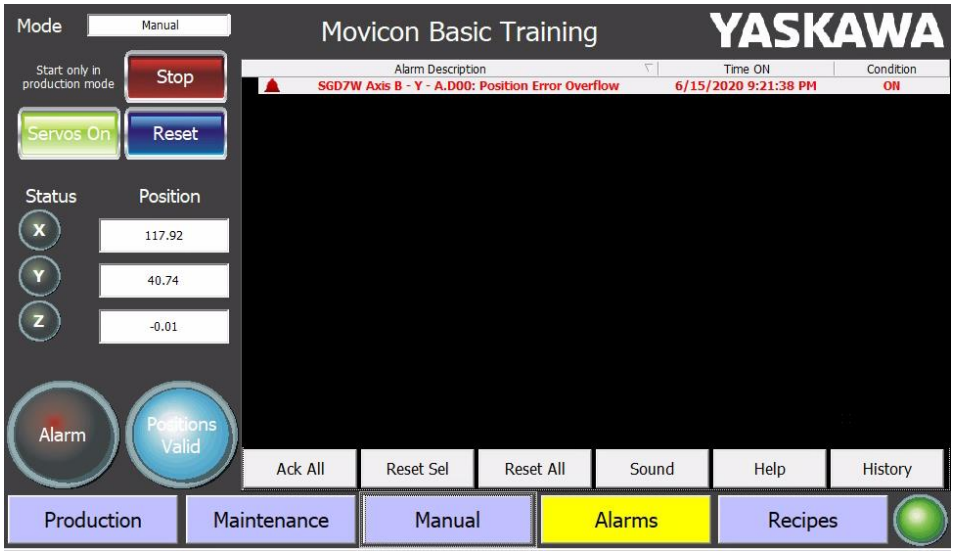
Alarms

Mini-Lab



Requirements

- MP3300iec Demo
- Jog-Zero and Setup screens functional

Lab Overview

This lab document will guide the participant through the following steps:

I. Create the Alarm Window (5 min)
II. HMI-Level Alarms (15 min)
III. MPiec-Level Alarms (25 min)
IV. Create an event to update alarm code descriptions (10 min)
V. (Optional) Clear Retained Alarms at Startup (10 min)

Lab Goal

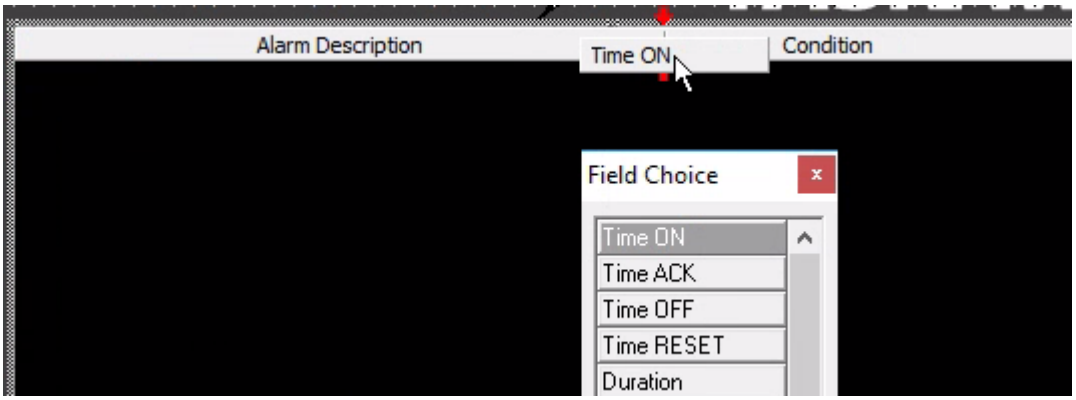
- Operator error alarms are reported using the alarm window
- Control system servo alarms are reported and cleared using the alarm window
- Alarms active during power cycle are automatically reset

I. Create the Alarm Window

A. In the Alarms screen, Insert an Alarm Window from Toolbox → Objects window

B. Resize/remove Columns. Click and drag columns to leave the following:

1. Alarm Description
2. Time ON
3. Condition



C. Set properties as desired

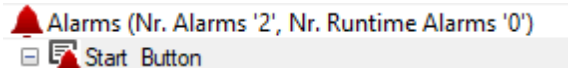
1. Style → Graphic Buttons = deselect
2. Fonts → Item Texts Font = Tahoma(10)
3. (Optional) Style → Deselect Comment Button, Help Button (can be used to launch a help file)

II. Define HMI-Level Alarms

If the operator presses Start when zero cycles remain, produce an alarm

B. Define the alarm (activation condition)

1. Project Explorer ➔ right-click Alarms ➔ Add new Alarm

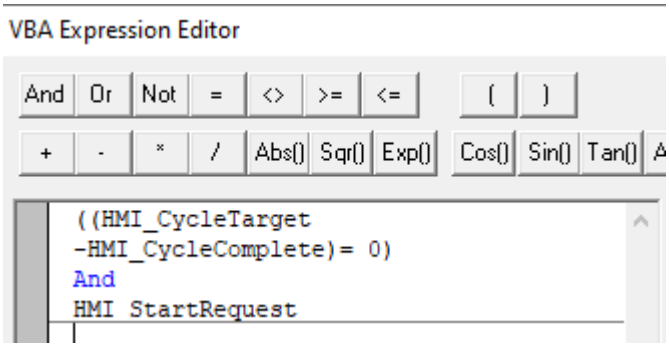


2. Name: `Start_Button`
3. Device Name: `smartPanel`

Device Name will appear as a prefix to the alarm text

4. Alarm Variable: as an expression
   i. ((**HMI_CycleTarget**-**HMI_CycleComplete**) = 0) AND (**HMI_StartRequest**=TRUE)
   ii. Be careful with parenthesis

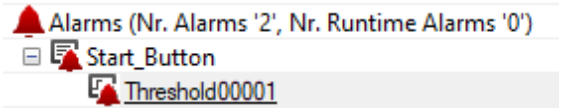
Meaning: If the start button is pressed with 0 cycles remaining



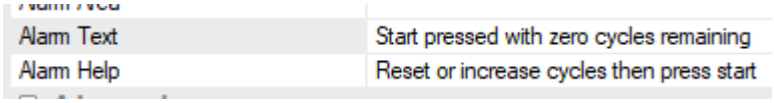
C. Set the alarm threshold

Thresholds run only if alarm condition is active

1. Project Explorer ➔ right-click the specific alarm ➔ "add a new alarm threshold"



2. Type the Alarm Text and Alarm help

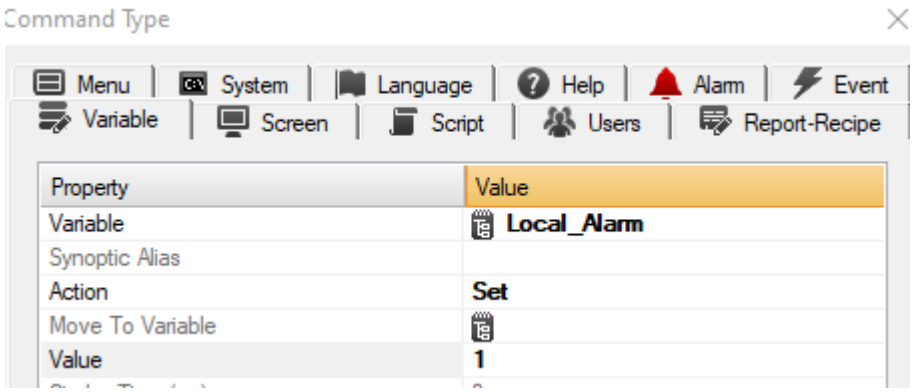


3. Activate the alarm when the alarm variable expression reaches a value of 1 (true)
   i. Properties ➔ Execution ➔ Activation Value = 1
   ii. Activation Condition = major-equal

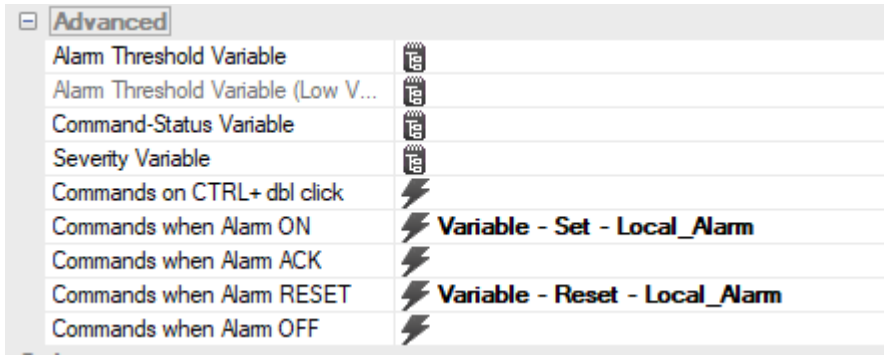
D. Configure the alarm indicator lamp (from the Control panel screen) to turn on when this alarm is active

Turn a variable on and off to indicate local alarm status

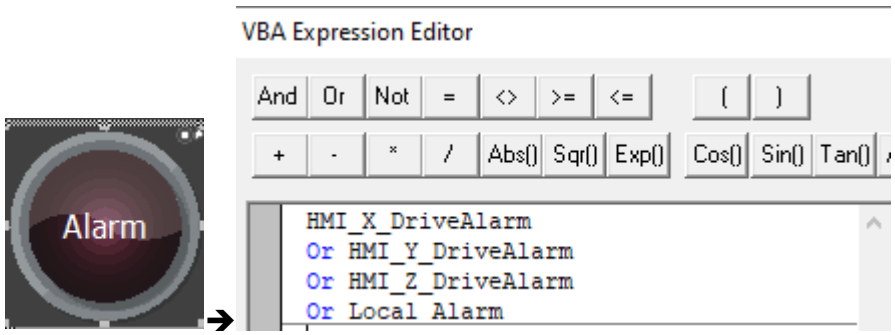
1. **THRESHOLD → PROPERTIES→ EXECUTION → ADVANCED → COMMANDS WHEN ALARM ON**
2. Set a new variable with BIT datatype named **Local_Alarm**
    i. Be sure to set the value to 1



3. **THRESHOLD → PROPERTIES→ EXECUTION → ADVANCED → COMMANDS WHEN ALARM RESET**
    i. Reset the **Local_Alarm** variable in the same way



4. In the Control screen, update the Properties→Command/State Variable so that **Local_Alarm** also causes the Alarm light to illuminate



E. Verify Operation

1. Produce the alarm (Press start when 0 cycles remaining)
2. Double-click the alarm to view the help
3. Acknowledge the alarm
4. Reset the alarm
5. Produce the alarm again
6. View the history
    i. Click on the active alarm
    ii. Click “History”
    iii. Expand the alarm history

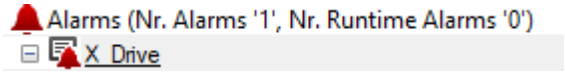
III. Define MPiec-Level Alarms

This instruction shows X-axis drive alarms.

Repeat for other axes drive alarms.

A. Define the alarm (activation condition)

1. Project Explorer ➔ right-click: Alarms ➔ Add new Alarm



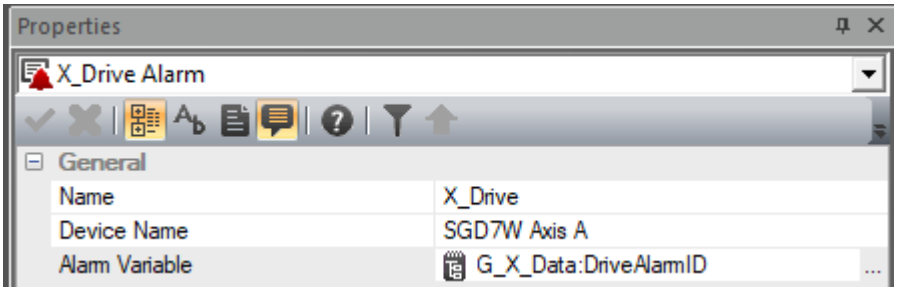
2. Device Name: `SGD7W Axis A`

A prefix to the alarm description

3. Monitor the value of AlarmID from the MPiec

Note: In this project the DriveAlarmID for each axis is not a separate variable, but rather an element of the structure G_X_Data

i. Alarm Variable: **G_X_Data:DriveAlarmID**



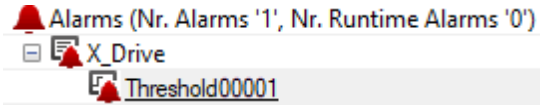
B. Set the alarm threshold conditions

The MPiec constantly updates the **AlarmID** *and* **AlarmMsg** *variables*

AlarmID *will be zero unless there is an alarm.*

Alarm ID *and* **AlarmMsg** *are available to the HMI over the PLCI protocol.*

1. Project Explorer ➔ right-click the alarm ➔ "add a new alarm threshold"



2. Set the Alarm Text as the **AlarmMsg** variable from the MPiec

Note use of %() syntax to display text of a string variable

i. `X- %(G_X_DriveAlarmMsg)`

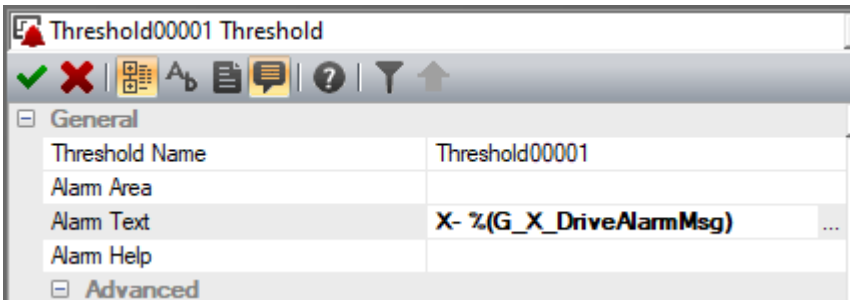
ii. How to display the text of a string variable (it cannot be selected directly)

a. Select the variable from Real Time DB →Variables(Tags)

b. Copy the name field of the variable

c. Paste the name into the threshold Alarm Text

d. Encapsulate in the %() syntax

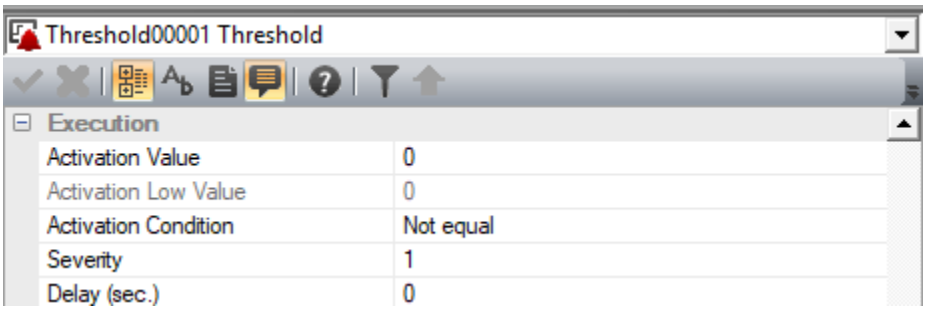


3. Threshold execution AlarmID <> 0

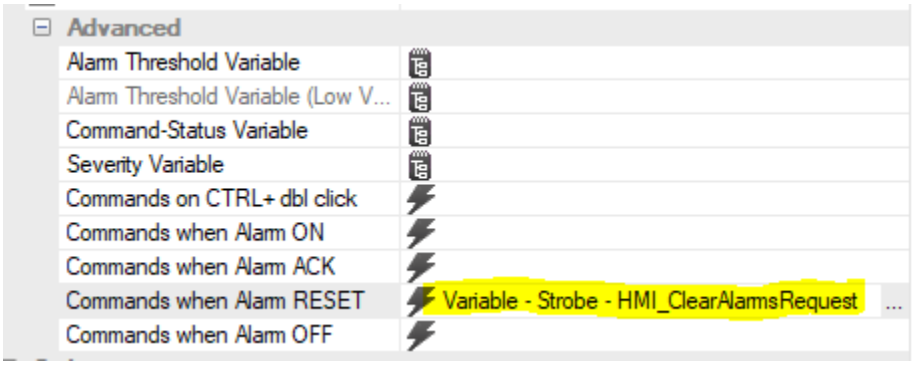
Activate the alarm when the Alarm ID is not zero

i. Activation Value = 0

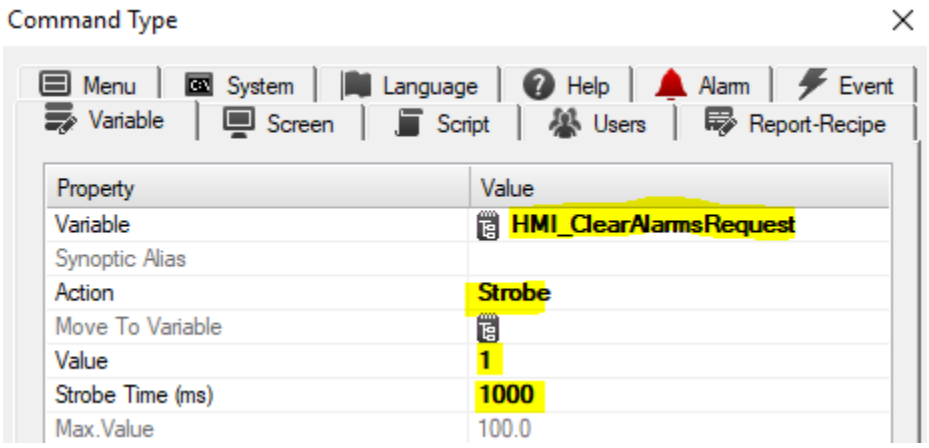
ii. Activation Condition = not equal



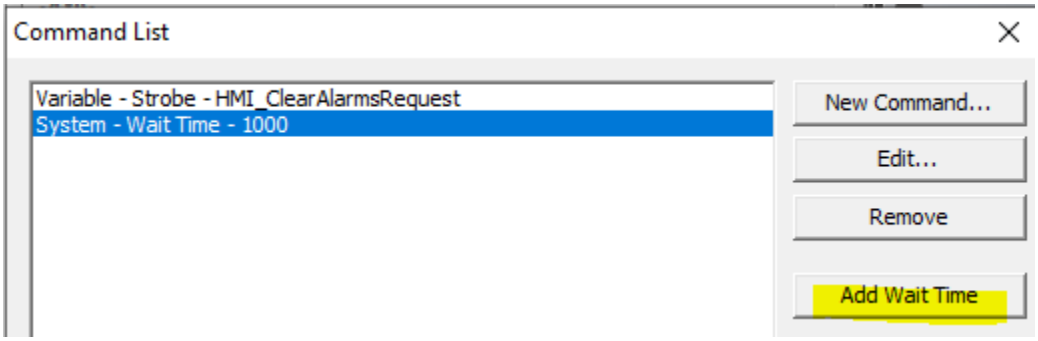
4. Clear the hardware-level alarm with the built-in Reset button
    i. Threshold Properties→Execution→Advanced
    ii. Commands when Alarm RESET



    iii. Strobe for 1000ms the variable **HMI_ClearAlarmsRequest** to Value=1. (1000ms pulse)



    iv. Add a system wait time to prevent the alarm from detection immediately after reset



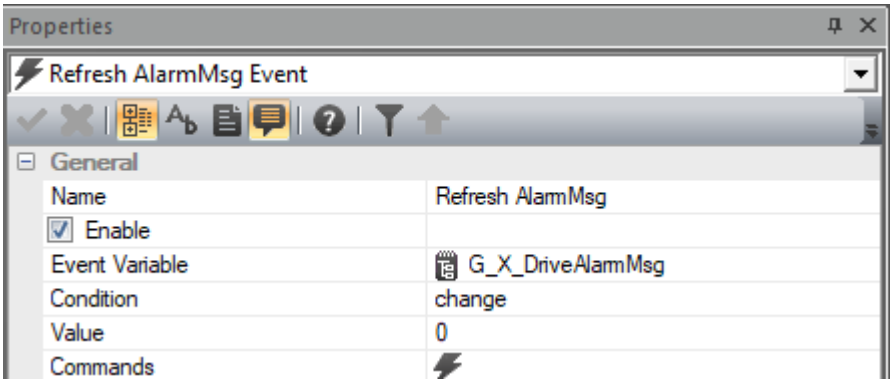
    v. In the Setup screen, change the Clear Alarms button properties also to strobe for 1000ms the variable **HMI_ClearAlarmsRequest** to Value=1
        a. *This button is currently set to hold* **HMI_ClearAlarmsRequest** *= 0 at all times. This will prevent control of the variable from the alarm screen*
        b. Command Type = Execute Commands
        c. Commands On Release = Strobe for 1000ms to value 1

C. Create a “dummy” event to refresh the **AlarmMsg**

The SmartPanel reads the alarm message string over PLCI protocol … only if those variables are used in the HMI at that time.

The event does nothing other than to refresh the value of AlarmMsg

1. Create a new Event named “Refresh AlarmMsg”
2. Set the Event Variable
    i. G_X_DriveAlarmMsg



D. Verify Operation

1. Produce an axis-level drive alarm (jog with Low PE button pressed)
2. Confirm that you can acknowledge and reset the alarm, see history, etc
3. Troubleshooting Tips
    i. If **HMI_ClearAlarmsRequest** is “stuck” true, set it to false in watch or locals.
    ii. Another axis may produce an error. Ignore this or clear it in the setup screen.

IV. Create alarms for Y-Axis and X-Axis

1. Copy, paste from above “X_Drive” alarm and edit to create drive alarms and events for the other axes.
2. Verify the correct message on each axis

End Of Mini-Lab

Troubleshooting Tips

- ☐ **HMI_ClearAlarmsRequest** may become “stuck” with the value of TRUE after Reset All. Set it to FALSE in watch.
- ☐ Check for spelling errors using Refactoring Explorer
- ☐ Check for copy/paste errors

Certification Checklist

- ☐ Produce "start pressed with zero cycles remaining" alarm
- ☐ Create alarm A.D00 on each axis X, Y, and Z (Individually or Simultaneously)
- ☐ Alarm message text displays correctly
- ☐ Clear alarm using the button in Setup screen
- ☐ Clear alarm with ACK-RESET in the Alarm Screen
- ☐ Display alarm history during active alarm

V. (Optional) Allow user to view all alarms log without active alarm

A. Insert a new screen

1. Reduce the dimensions
2. Add a button to close the screen

B. Add a Log Window to the new screen

1. Set Style → filter event type = Alarm messages.
2. Execution--> Max rows to a reasonable value.

C. Create a button for “Alarm Log” on the alarm screen that opens this new screen as “modal - popup”

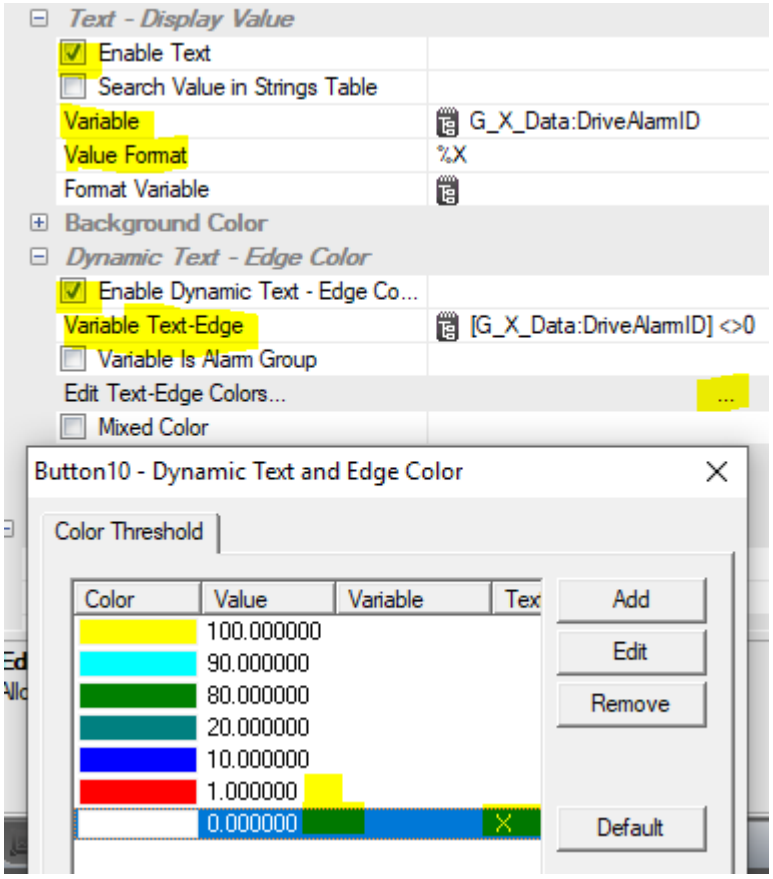
VI. (Optional) Allow user to click on the Alarm button to open the Alarms page

A. Hint: Make the lamp clickable, execute commands to open the screen

VII. (Optional) Display the Alarm Hex Code on the Axis Status Lamp

A. Set the Dynamics as shown

1. HINT: Select all 3 lamps to change properties together, then adjust for axes Y and Z
2. NOTE: Value Format %X forces the hex value to display



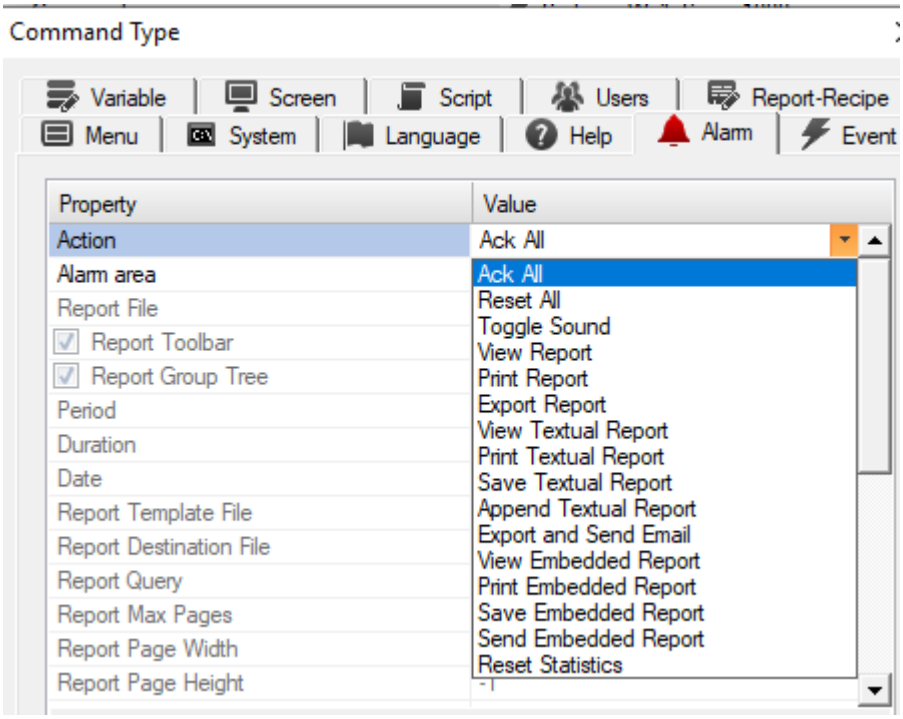
B. Repeat for each status lamp

VIII. (Optional) Automatically Acknowledge and Reset All Alarms Retained at Startup

A. *If the system is power cycled with an active alarm condition, the smartPanel retains this alarm. However, many alarms are cleared by power cycle. This results in an empty alarm condition.*

B. Verify this behavior (cause an alarm, then cycle power)

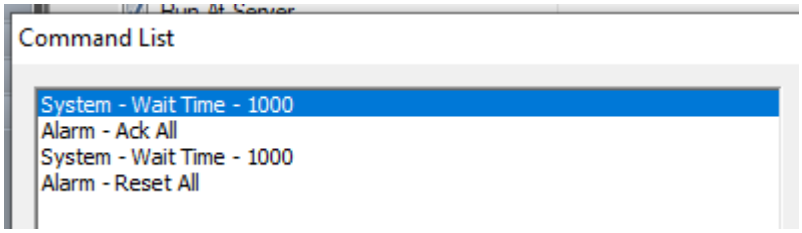
C. *Retained alarms can be cleared automatically at power on using commands after event*



D. Create an event named Startup

1. Event Variable = _SysVar_:ProjectRunning
2. Condition = equal
3. Value = 1

   SysVar:ProjectRunning=1 is always true, but the event is true only once at startup. The command list will run once.

4. Commands
5. 

   i. (the system wait times may not be necessary)

E. Use a Log Window (Toolbox → objects) to show all alarm history

1. Set Style → Filter Event Type = Alarm Messages

IX. Additional alarms detected by HMI

A. *Write a similar alarm for the following conditions*

B. Position not valid when start pressed

C. Servo not on when start pressed

D. Stop request active for an axis when start pressed

E. Axis position is beyond the range set in the Auto screen

X. Additional alarms detected by MPiec

A. *The MPiec can generate controller-level alarms. It can also generate axis-level alarms. These alarms and corresponding messages are detected in this MPiec project using the following variables.*

B. Program the HMI to handle controller-level alarms

1. HMI_ControllerAlarm<>0 G_ControllerAlarmMsg
2. G_X_Data:ControlAlarmID<>0 G_X_ControlAlarmMsg
3. G_Y_Data:ControlAlarmID<>0 G_Y_ControlAlarmMsg
4. G_Z_Data:ControlAlarmID<>0 G_Z_ControlAlarmMsg
5. Axis-level controller alarm code is G_X_Data:ControlAlarmID
6. Axis-level controller alarm message is named G_X_ControlAlarmMsg
7. Produce an axis-level controller alarm by unplugging Mechatrolink-III cable (may not recover)
8. Controller level alarm bit is HMI_ControllerAlarm
9. Controller level alarm message is named HMI_ControllerAlarm
10. Produce a controller alarm