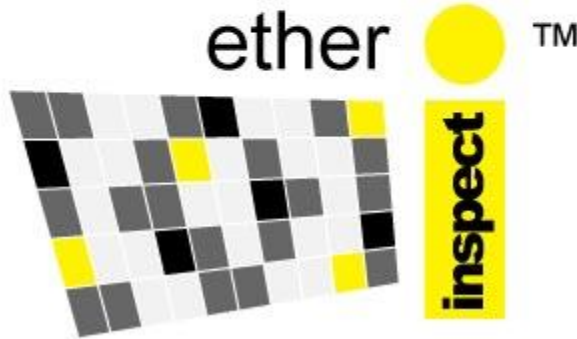


COGNEX



EI11 SCRIPTING

Ether-Inspect scripting is a feature that allows users to customize vision applications in multiple ways. This document will describe the differences between the older EI6 (C2D2) & EI9 (EW19) scripting environment and the new EI11 system. It is intended to assist users that are familiar with EI6 scripting transition to EI11. Several examples of common uses will be shown as done in both versions. Significantly improved functionality has been added to enhance performance and capabilities. EI11 features a C# programming environment with access to all vision tools and features. EI6 & 9 utilized a basic VBScript with limited access to tool results. This is no longer supported in Windows 10 64-bit applications. For programmers, the new C# format will be very familiar and will allow significantly improved functionality.

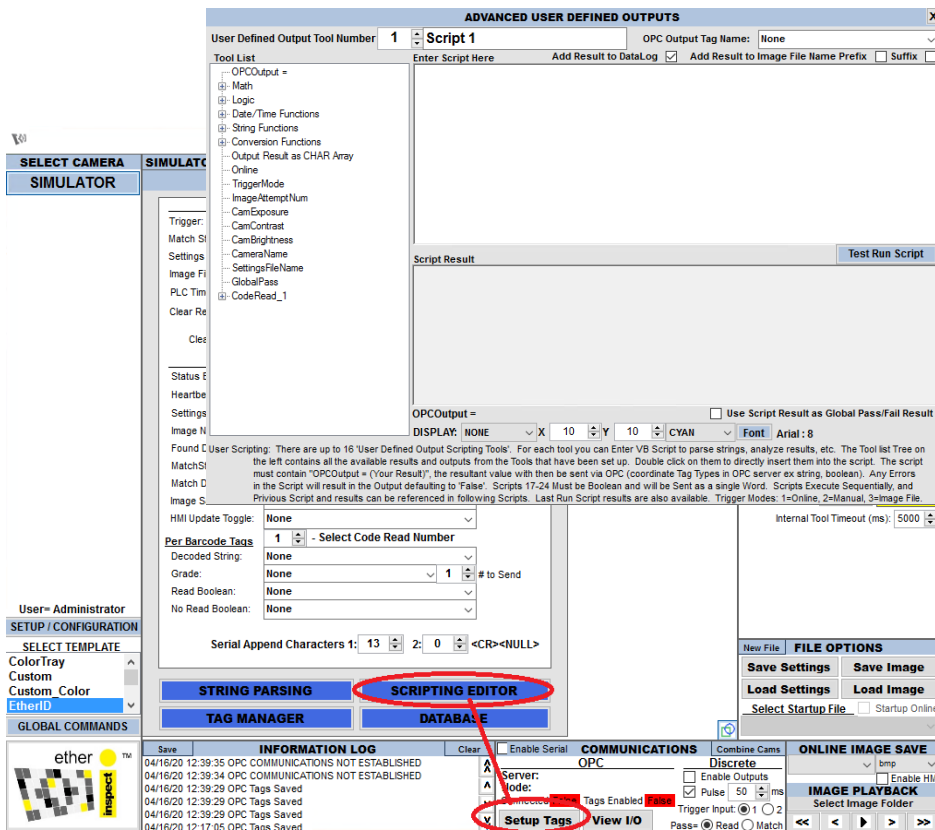
Primary Differences

- EI11 scripting is in C#. EI6&9 scripting was VBScript.
This is a much more powerful and commonly used programming environment.
- EI11 has direct access to all vision tool results. EI6&9 was limited to hard coded references.
Integrators will have direct access to the vision tools .net framework.
- EI6&9 was limited to 200 available scripts. There is no limit in EI11.
- EI6&9 scripts each had 1 result value or did 1 task (OPCOutput =). EI11 scripts can be written to do many different functions. This allows developers more flexibility. A single script could be used for all PLC writes, graphics, etc.
- EI11 scripts need to be compiled. There is a button for this at the top. It becomes fully executable code.
- EI11 has debugging capabilities. There was none in EI6&9.

TIPS

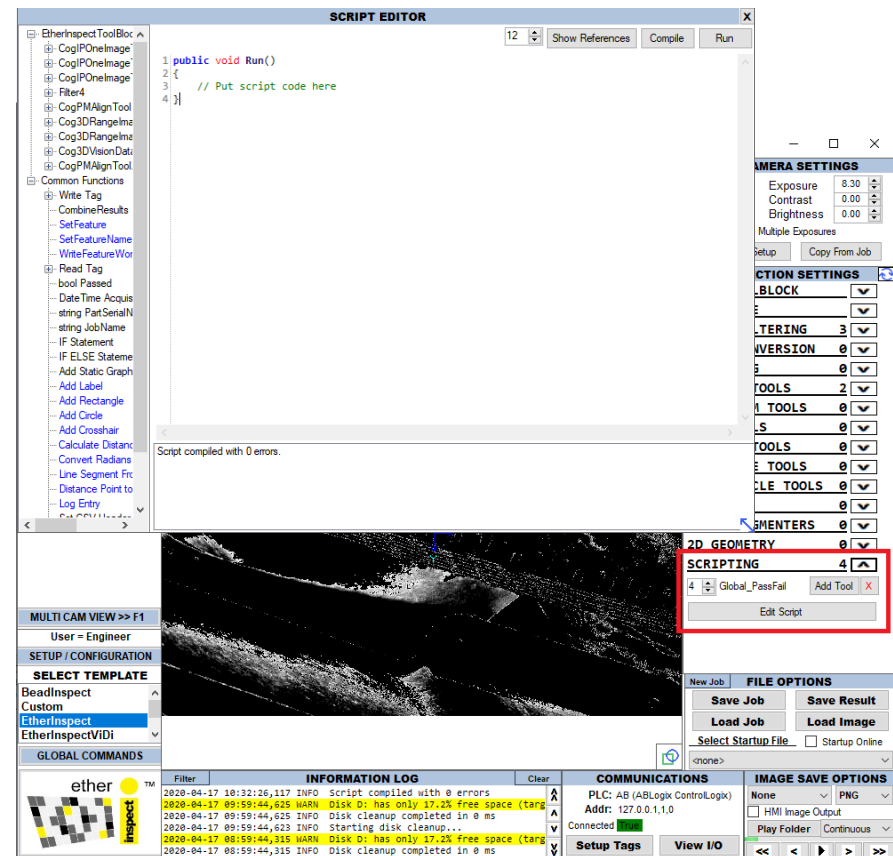
- While all scripting needs could be done in a single C# script configuration, it may be advisable to make multiple scripts that handle different task. As an example, one script that does all PLC communications writes, and another that does all graphics displays, etc.
- The Tool Tree has numerous code builder dialogs that can help scripting. More capability will continuously be included. Any suggestions for additional features here will be considered.

Where to Access Scripting



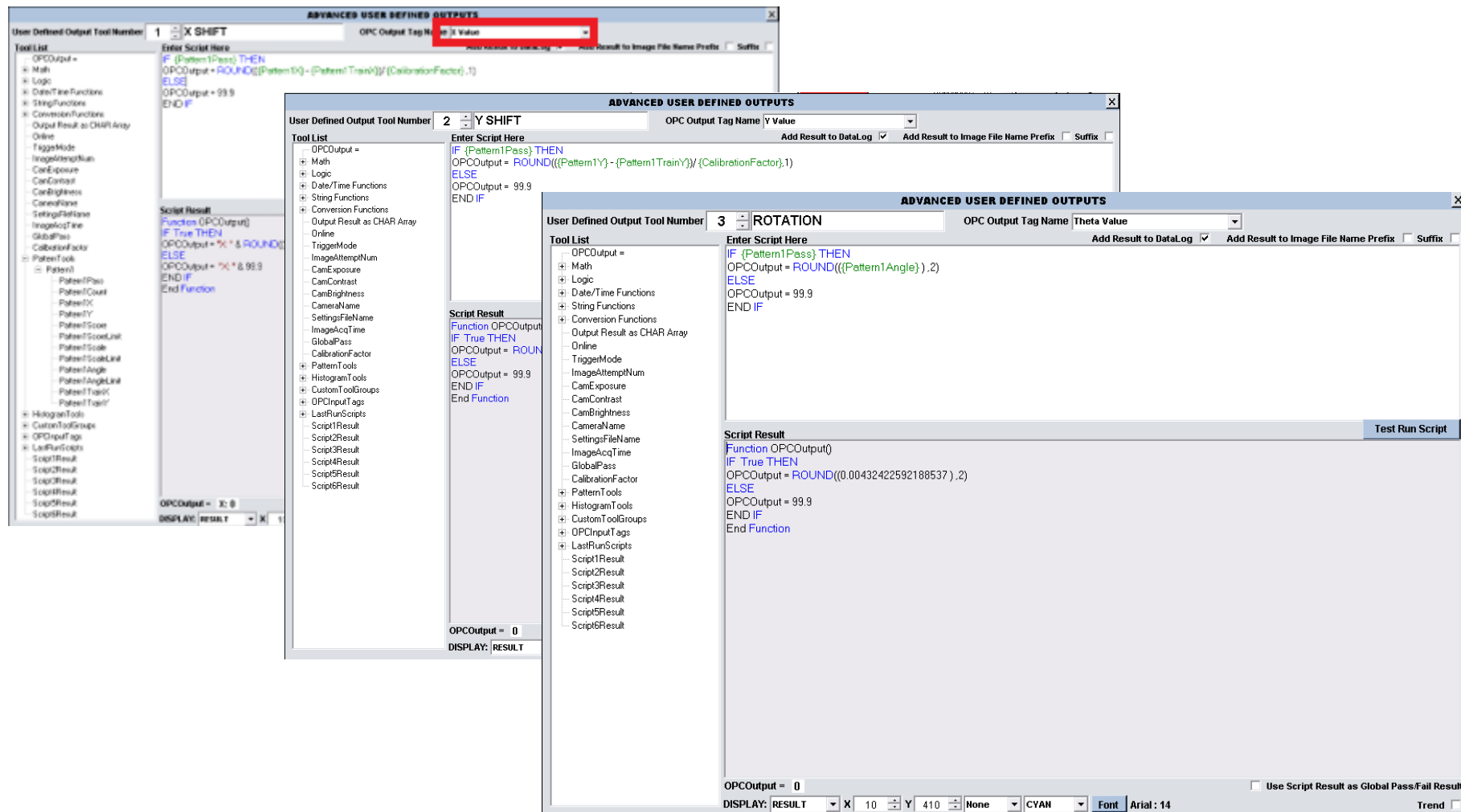
El6 Scripting

In EI6, access to scripting was located in the Communications section under the Setup Tags interface. In EI11, it is found at the bottom of the Inspection Settings section on the right where Vision tools are configured.



El11 Scripting

EI6 WRITE TO PLC



This example is sending 2D Robot guidance values to a PLC (X, Y & Θ) from a Pattern find tool. In EI6, 3 scripts would be written to calculate each value and would link to the respective Output communications Tag.

EI11 WRITE TO PLC



The same task in EI11 will require only 1 script as above. The references for the tool results can be added as shown. The 3 output writes associate the tags with respective values in the script. Note that using this structure will allow for access to ALL tool result items, not just the ones directly available via references hard coded into the tool tree on the left.

EI6 GRAPHIC LABLES

SIMULATOR 20 Camera Connection Software License (QXW Cognex License)

SELECT CAMERA SIMULATOR

Camera: (Simulator) 20 / -20 ms

Last Valid Code #1 Read View Counts Log Results Enable I/O Monitoring

Symbology: DataMatrix Last Find: 04736298AA Match String: Select Match: Product List

TRIGGER OPTIONS

Select Mode

External External / Timed Network Network / Timed External / Soft Stop on Read Timed

Inter-Trigger Delay 1,000 Debounce 10

CAMERA SETTINGS

Exposure: 8.30 Contrast: 0.00 Brightness: 0.00 Orientation: 0°

Multi-Exp Image Size

CODE SEARCH SETTINGS

ADVANCED USER DEFINED OUTPUTS

User Defined Output Tool Number 1 Script 1

OPC Output Tag Name: None

Add Result to DataLog Add Result to Image File Name Prefix Suffix

Tool List

- OPCOutput =
- Math
- Logic
- Date/Time Functions
- String Functions
- Conversion Functions
- Output Result as CHAR Array
- Online
- TriggerMode
- ImageAttemptNum
- CamExposure
- CamContrast
- CamBrightness
- CameraName
- SettingFileName
- GlobalPass
- CodeRead_1
- OPCInputTags
- LastRunScript
- Script1Result

Script 1

Enter Script Here

OPCOutput = NOW

Script Result

Test Run Script

XPCOutput =

DISPLAY: RESULT X 10 Y 10 CYAN Font Arial: 11.25

Use Script Result as Global Pass/Fail Result

User= Administrator

SETUP / CONFIGURATION

SELECT TEMPLATE

EtherID

EtherID_Verify

EtherInspect

EtherInspect_Color

GLOBAL COMMANDS

Save

04/02/17 08:38:23 OPC COMMU

ether inspect

Start EtherInspect - Microsof... Ether-Inspect MAX (Ap... Quality WorX Database ... SIMULATOR 20 Camera... Fail

Pass Read Match 9:16 AM

In EI6, graphics could be added to the image display by configuring the Display settings at the bottom of the scripting area. In this example, it will add a Date & Time graphic on the image. Once set, the properties of the graphic are not adjustable while online

EI11 GRAPHIC LABLES

4/22/2020 4:22:14 PM

<Default>

IN: CALIBR IMAGE

SCRIPT EDITOR

12 Show References Compile Run

EtherInspectToolBlock
Common Functions
Write Tag
CombineResults
SetFeature
SetFeatureName
WriteFeatureWord
Read Tag
bool Passed
DateTime AcquisitionTime
string PartSerialNumber
string JobName
IF Statement
IF ELSE Statement
Add Static Graphic
Add Label
Add Rectangle
Add Circle
Add Crosshair
Calculate Distance
Convert Radians to Degrees
Line Segment From Points
Distance Point to Line
Log Entry
Set CSV Header
Append CSV Entry
QLS INI Configuration Error
Camera.DatabaseSection not defi

```
1 public void Run()  
2 {  
3     // Put script code here  
4     string Now = JobResult.AcquisitionTime.ToString();  
5     AddLabel(Now,10,10,CogColorConstants.Cyan,CogGraphicLabelAlignmentConstants.TopLeft,null,14,CogColorConstants.None,"@");  
6 }  
7
```

CODE BUILDER - ADD LABEL

Text

X Coord 10

Y Coord 10

Color Cyan

Alignment TopLeft

Font Size 14

Background Color None

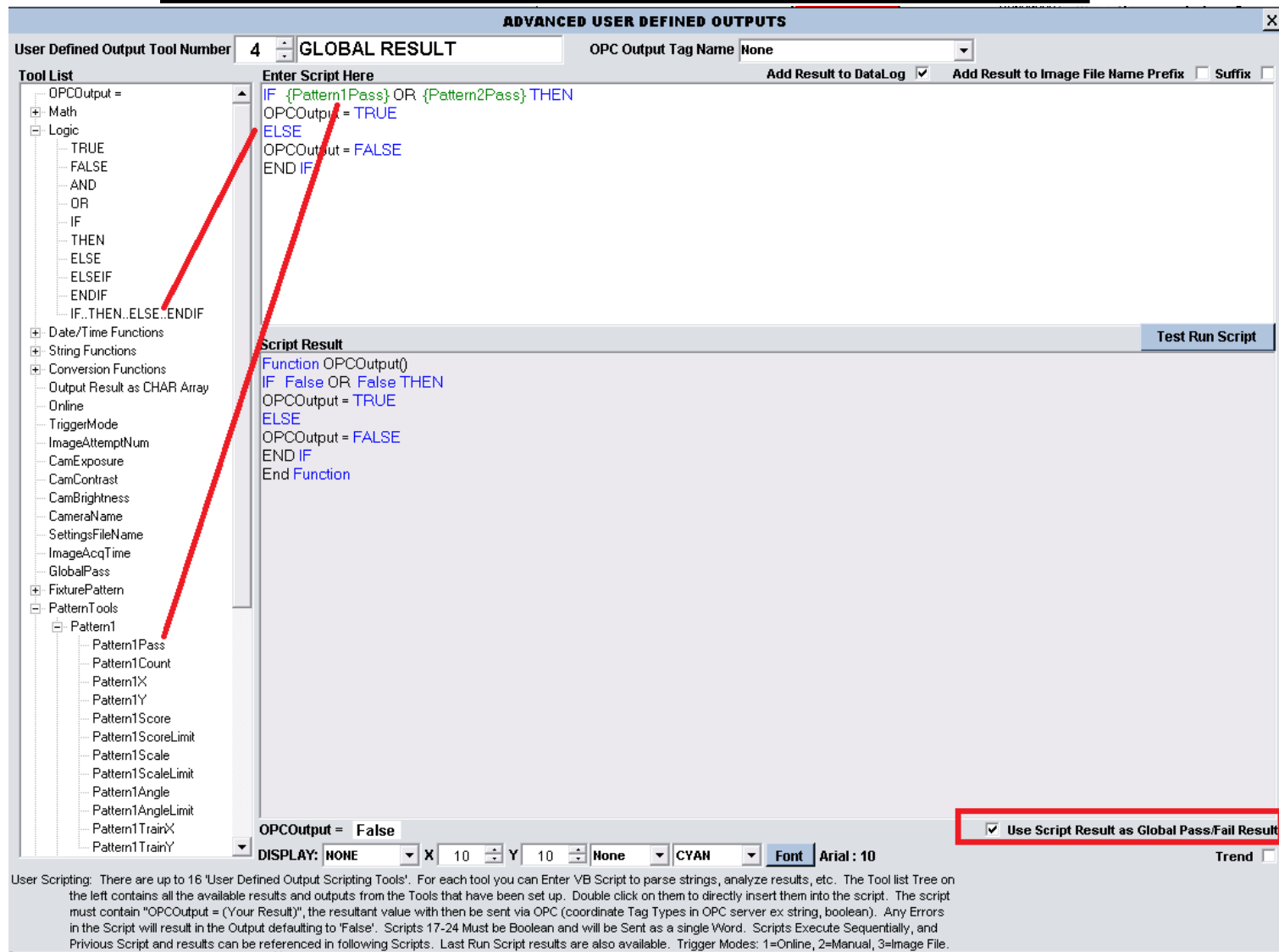
Image Coordinate Space Name @

OK Cancel

Script compiled with 0 errors.

The Date and Time in EI11 can be displayed as shown above. The Acquisition time will be used and will match what is associated with the image save. Use Add Label from the tool tree and the code builder pop up to create the initial code. Note that the variables in the code can be changed dynamically (ie. Color, Font Size, etc)

EI6 GLOBAL PASS FAIL OVERRIDE



In EI6, the default Global Pass/Fail result of an inspection can be controlled via script. A Boolean result (True/False) script needs to be written and the checkbox highlighted enabled. In this example, the inspection will result is a Pass if either of 2 Pattern tools has an acceptable result.

EI11 GLOBAL PASS FAIL OVERRIDE



In EI11, the JobResult.Passed property can be controlled within the script. The above code would accomplish the same result of either Pattern tool being an acceptable final result.

EI6 LOG DATA TO CSV

The screenshot displays the Ether-Inspect MAX software interface, version 1.0.0.194, with a 20 Camera Connection Software License (QWX Cognex License). The main window is titled "ETHER_ID_MATRIX1: Attest" and shows camera information: Camera: Basler acA640-90, IP: 192.168.3.1. The "INSPECTION RESULT" section shows "Global Pass/Fail: FAIL" and "View Counts" button. The "TRIGGER OPTIONS" section includes "Select Mode" (External, Network, External / Soft, Timed) and "Go Live" button. The "CAMERA SETTINGS" section includes "Exposure" (100.0), "Contrast" (0.70), "Brightness" (0.00), and "Orientation" (0°). The "OPC TAGS SETUP" dialog is open, showing "User Defined Output Tool Number 1" and "SCORE". The "OPC Output Tag Name" is set to "None". The "Add Result to DataLog" checkbox is checked. The "Script Result" section shows the script: `OPCOutput = {Pattern1Score}`. The "Tool List" on the left includes: OPCOutput, Math, Logic, Date/Time Functions, String Functions, Conversion Functions, Output Result as CHAR Array, Online, TriggerMode, ImageAttemptNum, CamExposure, CamContrast, CamBrightness, CameraName, SettingsFileName, ImageAcqTime, GlobalPass, FixturePattern, PatternTools, OPCInputTags, LastRunScripts, Script1Result, Script2Result, Script3Result. The "MULTI CAM VIEW >> F1" button is visible at the bottom left. The "User" is set to "Administrator".

The "ADVANCED USER DEFINED OUTPUTS" dialog is also open, showing "User Defined Output Tool Number 2" and "SCALE". The "OPC Output Tag Name" is set to "None". The "Add Result to DataLog" checkbox is checked. The "Script Result" section shows the script: `OPCOutput = {Pattern1Scale}`. The "Tool List" on the left includes: OPCOutput, Math, Logic, Date/Time Functions, String Functions, Conversion Functions, Output Result as CHAR Array, Online, TriggerMode, ImageAttemptNum, CamExposure, CamContrast, CamBrightness, CameraName, SettingsFileName, ImageAcqTime, GlobalPass, FixturePattern, PatternTools, OPCInputTags, LastRunScripts, Script1Result, Script2Result, Script3Result.

The "ADVANCED USER DEFINED OUTPUTS" dialog is also open, showing "User Defined Output Tool Number 3" and "ANGLE". The "OPC Output Tag Name" is set to "None". The "Add Result to DataLog" checkbox is checked. The "Script Result" section shows the script: `OPCOutput = {Pattern1Angle}`. The "Tool List" on the left includes: OPCOutput, Math, Logic, Date/Time Functions, String Functions, Conversion Functions, Output Result as CHAR Array, Online, TriggerMode, ImageAttemptNum, CamExposure, CamContrast, CamBrightness, CameraName, SettingsFileName, ImageAcqTime, GlobalPass, FixturePattern, PatternTools, OPCInputTags, LastRunScripts, Script1Result, Script2Result, Script3Result.

Logging results to a CSV file in EI6 required enabling the main Log Results checkbox in Ether Inspect and additionally the checkbox shown for each script result to log. In this example, 3 scripts are used to log a Pattern tool score, scale & angle results. The name of each script will become the header.

EI11 LOG DATA TO CSV

The screenshot displays the EI11 Script Editor interface. On the left, the 'Tool Tree' lists various functions. Two red arrows originate from the 'Set CSV Header' and 'Append CSV Entry' functions and point to their respective code lines in the script. The script, located in the 'SCRIPT EDITOR' window, contains the following code:

```
1 public void Run()
2 {
3     PMAAlignToolResult CogPMAAlignTool1Result = (PMAAlignToolResult)GetToolResultObject("CogPMAAlignTool1");
4     double FoundScore = CogPMAAlignTool1Result.FoundScore;
5     double FoundAngleDegrees = CogPMAAlignTool1Result.FoundAngleDegrees;
6     double FoundScale = CogPMAAlignTool1Result.FoundScale;
7
8     SetCSVHeader("SCORE", "SCALE", "ANGLE"); // Timestamp, Pass-Fail already included
9     AppendToCSV(FoundScore, FoundScale, FoundAngleDegrees);
10 }
11
12
```

Below the script editor, a file explorer window shows the directory structure, with the file '2020-04-17.csv' highlighted. To the right, a spreadsheet window displays the data logged to the CSV file. The spreadsheet has columns for Timestamp, Pass-Fail, SCORE, SCALE, and ANGLE. The data rows show the results of the alignment process for various timestamps.

Timestamp	Pass-Fail	SCORE	SCALE	ANGLE
2020-04-17_13-10-54.235	Fail	0.686857	0.99788	0.297169
2020-04-17_13-10-54.979	Fail	0.791272	0.997369	-0.28422
2020-04-17_13-10-55.150	Fail	0.813446	0.995936	0.252207
2020-04-17_13-10-55.296	Fail	0.814121	1.007961	0.154176
2020-04-17_13-10-57.478	Fail	0.765927	1.019512	0.151268
2020-04-17_13-10-57.630	Fail	0.708063	1.017938	-0.71507
2020-04-17_13-10-57.785	Fail	0.860162	1.006257	0.070856
2020-04-17_13-10-58.023	Fail	0.595226	0.987079	2.538907
2020-04-17_13-10-58.186	Fail	0.687795	0.973572	3.217569

In EI11 there are code building links in the tool tree for Set CSV Header and Append CSV Entry. Add as many commas as necessary to include more values in the log file.

EI6 FEATURE RESULTS

SELECT CAMERA

SIMULATOR

SIMULATOR :

Camera : (Simulator)

INSPECTION RESULTS

Global Pass/Fail: **PASS**

Fixture Pattern: ■

Meas Tools: ■

OCV Tools: ■

Log Results ☐

Enable I/O Monitoring ☐

View Mode

LIGHT ON

TRIGGER OPTIONS

Select Mode

☐ External
 ☐ External / Timed
 ☒ Network
 ☐ Network / Timed
 ☐ External / Soft
 ☐ Stop on Pass
 ☐ Timed

Inter-Trigger Delay 3,000

Debounce 10

CAMERA SETTINGS

Exposure 8.30

Contrast 0.00

Brightness 0.00

Orientation 0°

Multi-Exp Image Size

OPC TAGS SETUP

ADVANCED USER DEFINED OUTPUTS

User Defined Output Tool Number 17

Script 17

OPC Output Tag Name Feature Results

Tool List

- OPCOutput =
- Math
- Logic
- Date/Time Functions
- String Functions
- Conversion Functions
- Output Result as CHAR Array
- Online
- TriggerMode
- ImageAttemptNum
- CamExposure
- CamContrast
- CamBrightness
- CameraName
- SettingsFileName
- GlobalPass
- PatternTools
 - Pattern 1
 - Pattern 1Pass
 - Pattern 1Count
 - Pattern 1X
 - Pattern 1Y
 - Pattern 1Score
 - Pattern 1ScoreLimit
 - Pattern 1Scale
 - Pattern 1ScaleLimit
 - Pattern 1Angle

Enter Script Here

Add Result to DataLog ☒

Add Result to Image File Name Prefix ☐

Suffix ☐

```

IF {Pattern1Pass} OR {Pattern2Pass} THEN
OPCOutput = TRUE
ELSE
OPCOutput = FALSE
END IF
          
```

Script Result

Test Run Script

```

Function OPCOutput()
IF False OR False THEN
OPCOutput = TRUE
ELSE
OPCOutput = FALSE
END IF
End Function
          
```

OPCOutput = False

Use Script Result as Global Pass/Fail Result ☐

DISPLAY: NONE

X 10

Y 10

None

CYAN

Font Arial : 10

Trend ☐

User= Administrator

SETUP / CONFIGURATION

SELECT TEMPLATE

EtherID

EtherID_Verify

EtherInspect

EtherInspect_Color

GLOBAL COMMANDS

ether

inspect

User Scripting: There are up to 16 'User Defined Output Scripting Tools'. For each tool you can Enter VB Script to parse strings, analyze results, etc. The Tool list Tree on the left contains all the available results and outputs from the Tools that have been set up. Double click on them to directly insert them into the script. The script must contain "OPCOutput = (Your Result)", the resultant value will then be sent via OPC (coordinate Tag Types in OPC server ex string, boolean). Any Errors in the Script will result in the Output defaulting to 'False'. Scripts 17-24 Must be Boolean and will be Sent as a single Word. Scripts Execute Sequentially, and Previous Script and results can be referenced in following Scripts. Last Run Script results are also available. Trigger Modes: 1=Online, 2=Manual, 3=Image File.

Individual Boolean Feature results can be sent to the PLC to sync with the Ford FAST function block. In EI6, Up to 8 features can be sent via a single WORD output tag. Scripts 17-24 are all combined to one Output tag assignment for this purpose. Each of these 8 scripts need to be configured with a Boolean result for the respective feature. Note, each feature uses 2 bits of the word (Pass & Fail bits for each)

EI11 FEATURE RESULTS

The screenshot displays the EI11 software interface, which is divided into three main sections:

- TAG STATES [SIMULATOR]:** This section on the left lists various system states and features.
 - State: Ready
 - PLCTrigger: ☐
 - Status Word: 0
 - (00) Online: ☐
 - (01) Toggle: ☐
 - (03) Busy: ☐
 - (04) QualitySubmitErr: ☐
 - (05) ImageError: ☐
 - (06) TriggerReady: ☐
 - (07) ExposureComplete: ☐
 - (10) GlobalPass: ☐
 - (11) GlobalFail: ☐
 - (12) ResultsReady: ☐
 - (13) LightingNOK: ☐
 - (14) TurnLightOn: ☐
 - FEATURES**
 - Feature1: ☒ (indicated by a green square)
 - CUSTOM OUTPUTS**
 - FeatureResults: 0
 - CUSTOM INPUTS**
- SCRIPT EDITOR:** This section on the right contains a script for the 'Run()' function.


```

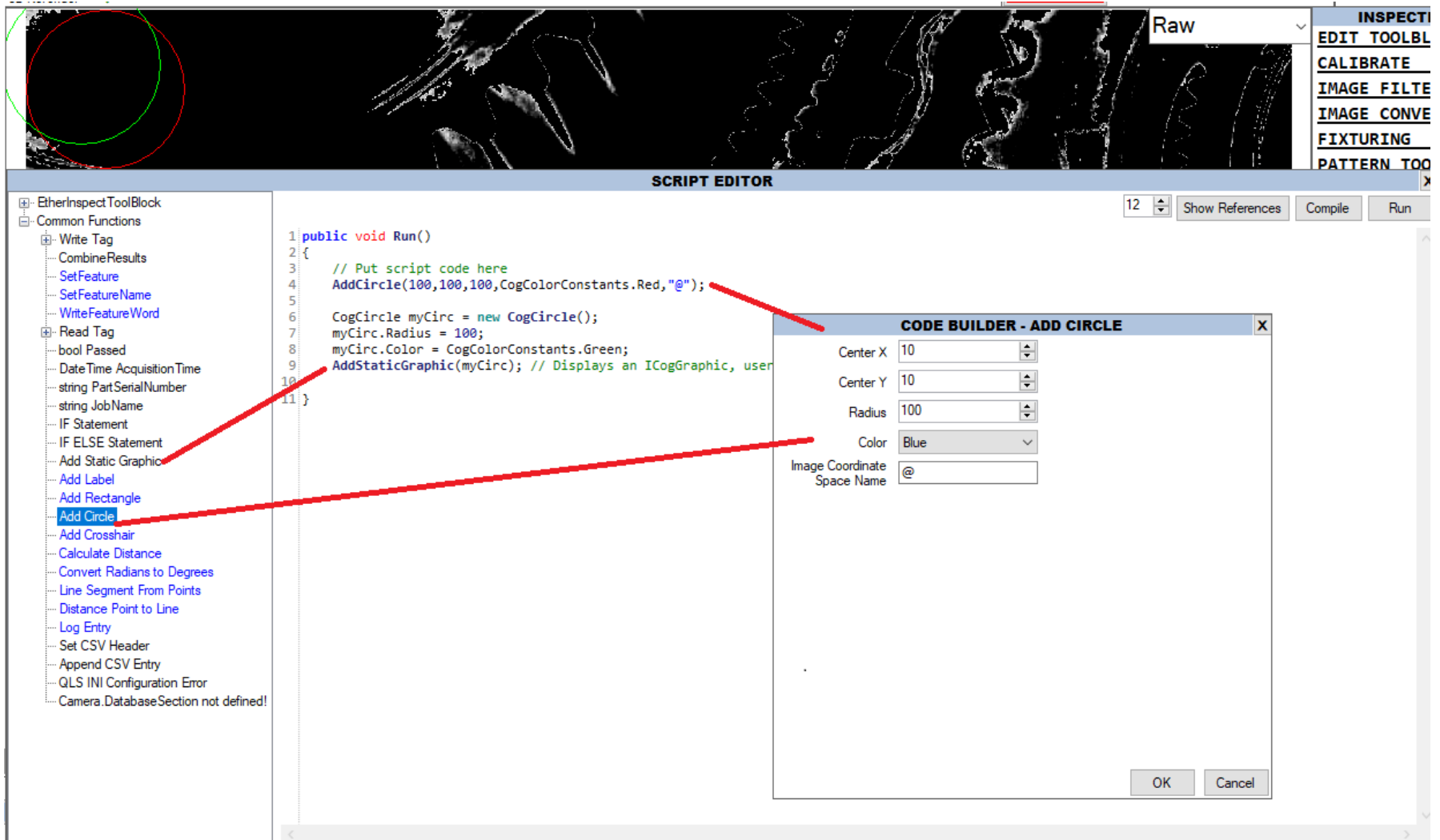
1 public void Run()
2 {
3     // Put script code here
4     SetFeatureName(0, "Feature1");
5     bool CogPMAAlignTool1Passed = GetToolPassFail("CogPMAAlignTool1");
6     bool CogPMAAlignTool2Passed = GetToolPassFail("CogPMAAlignTool2");
7     SetFeature(0, CogPMAAlignTool1Passed || CogPMAAlignTool2Passed);
8     WriteFeatureWord("FeatureResults", 0);
9 }
      
```

 Below the script, a status bar indicates "Script compiled with 0 errors."
- Tree View:** A central tree view shows the hierarchy of tool blocks.
 - EtherInspectToolBlock
 - CogIPOnelImageTool1
 - CogIPOnelImageTool2
 - CogIPOnelImageTool3
 - Filter4
 - CogPMAAlignTool1
 - VisionPro Tool Reference
 - CogPMAAlignTool1 Tool Reference
 - Pass-Fail Result
 - CogPMAAlignTool2
 - Common Functions
 - Write Tag
 - Combine Results
 - SetFeature
 - SetFeatureName
 - WriteFeatureWord
 - Read Tag
 - bool Passed
 - Date Time AcquisitionTime
 - string PartSerialNumber
 - string JobName
 - IF Statement
 - IF ELSE Statement
 - Add Static Graphic
 - Add Label
 - Add Rectangle
 - Add Circle
 - Add Crosshair
 - Calculate Distance
 - Convert Radians to Degrees

Red arrows in the image point from the script editor to the tree view and tag states, indicating the flow of data and logic. Specifically, arrows point from 'SetFeatureName' to 'Feature1', from 'SetFeature' to 'FeatureResults', and from 'WriteFeatureWord' to 'FeatureResults'.

In EI11, the same example can be done as shown. 8 features can be set for each Word write. Features 0-7 will write with Word 0. Additional features can be easily added in this format (Features 8-15 go with Word 1, etc). Add logic for as many features as necessary. Additionally, if the Features are named, they will display in the View IO window as shown.

EI11 GRAPHICS



EI11 has additional functionality to create graphics on the image (in addition to text). There are a few Code Builder links in the tool tree such as Add Circle as shown. Use AddStaticGraphic to create any custom Cognex graphics or result graphics from vision tools. The properties can be changed dynamically in code as well. Example would be red/green highlight boxes around key inspections.

EI11 DEBUGGING

The screenshot displays the EI11 Script Editor interface. On the left is a tree view of the 'EtherInspect ToolBlock' containing various functions like 'Write Tag', 'Read Tag', and 'Log Entry'. The main area shows a C# script with a try/catch block. A red arrow points from the 'Log Entry' function in the tree to the 'Log.ErrorFormat' call in the script. Another red arrow points from the 'Log Entry' function to the 'CODE BUILDER - LOG ENTRY' window, which shows an error message 'Exception doing division'. A third red arrow points from the 'Log Entry' function to the 'INFORMATION LOG' at the bottom, which shows an error message 'Exception doing division: Attempted to divide by zero.'.

SCRIPT EDITOR

12 Show References Compile Run

```
1 public void Run()
2 {
3     try
4     {
5         int y = 0;
6         int x = 10/y;
7     }
8     catch (Exception ex)
9     {
10        Log.ErrorFormat("Exception doing division: {0}", ex.Message);
11    }
12 }
13
14
```

CODE BUILDER - LOG ENTRY

Level: Error
Message: Exception doing division

INFORMATION LOG

2020-04-23 12:01:50,610 INFO Script compiled with 0 errors
2020-04-23 12:01:49,146 ERROR Exception doing division: Attempted to divide by zero.

EI11 has several debugging capabilities, None were available in EI6.

- When the script is compiled the field beneath the script will display line numbers and errors in the code.
- Try/Catch code can be used to trap errors.
- Log Entry can be used to put messages in the main Ether Inspect Information Log box
- 'Run' the script will execute the code, including writes to the PLC. EI6 required camera to be triggered online to test comms

EI11 PLC INPUT

The screenshot displays the 'SCRIPT EDITOR' interface. On the left, a tree view lists variables and functions. The main area shows a C# script for 'CogPMAAlignTool1'. Red arrows indicate the mapping between script variables and the tree view:

- `double FoundScore` in the script points to `double FoundScore` in the tree view.
- `short LimitA` in the script points to `short LimitA` in the tree view.
- `short LimitB` in the script points to `short LimitB` in the tree view.

The script code is as follows:

```
1 public void Run()
2 {
3     // Put script code here
4     PMAAlignToolResult CogPMAAlignTool1Result = (PMAAlignToolResult)GetToolResultObject("CogPMAAlignTool1");
5     double FoundScore = CogPMAAlignTool1Result.FoundScore;
6     short LimitA = (short)GetTag("LimitA").GetValue();
7     short LimitB = (short)GetTag("LimitB").GetValue();
8
9     if(FoundScore > LimitA && FoundScore < LimitB) // Condition
10    {
11        // Code to execute if condition true
12        JobResult.Passed = true;
13    }
14    else
15    {
16        // Code to execute if condition false
17        JobResult.Passed = false;
18    }
19 }
20 }
```

The bottom status bar indicates: 'Script compiled with 0 errors.'

EI11 has access to the PLC Input tag values. In this example, 2 limit values are received from the PLC (LimitA & LimitB) and used to determine the Global Pass/Fail result for the inspection of a Pattern Tool.