# Waveform Checker

Date Started: (July 02, 2024)

Date Finished: (2024)

Tester: JHG

Project Information:

|  |  |
| --- | --- |
| **Item Name** | **Details** |
| File Type | Excel File with Macro Enabled (.xlsm) |
| Language | Visual Basic in Excel |
| Software Revision | 1.0 |
| Path: | \\fs-mcs\testlab\main\TestData\Asia\_Office\_Data\Software Base Check\Waveform,PQ |

Description:

Software that evaluates waveform captures for shark and nexus series. Using omicron as the input power source for meters, the tool State Sequencer is used to create a series of source configurations that can have different durations. Great tool to create series of sag and surge events that will trigger waveform and PQ captures. These series of events will be the reference for cross-checking the events generated by the meter.

Main Parts:

1. Main Sheet – Contains the summary of meter types, results and buttons. This is where the control of checker happens.

A screenshot of a computer

Description automatically generated

Sub Part:

1. Meter Column – Initial version only contains the following meters:

* Shark 200, Shark 250, Shark 270, Nexus 1450, and Nexus 1500+

1. Buttons:
   * Import – used to open a selected file, copy its content and paste it to the designated meter sheet. These logs are then arranged ascendingly to match the series of events.
   * Process – used to execute current existing logics to evaluate the logs of selected imported logs. Then, the Result value will change from “Stored” to either “Satisfactory” or “Failed”.
   * Clear – can be used to clear a specific row and specific sheet. Example is when I click cell B10 and Clear right after, the cells B10 to C10 will be empty including the “Nexus 1500+” Sheet. Main purpose is for reusability of the main program.
2. Result Column – can be in states: “Stored”, “Satisfactory”, and “Failed”

A screenshot of a computer

Description automatically generated

* Stored – means that a csv file of that meter is already imported and pasted in designated “meter” sheet. This state will be written after a successful log import using “Import” button.
* Satisfactory – means that the expected events match the events in the imported logs.
* Failed - means that the some/all expected events do not match the events in the imported logs.

1. Findings Column – will only be filled if the result is “Failed” else empty. Can be:
   * Failed due to unequal number of events – happens when the actual logs count is higher than the expected number of events.
   * Failed due to different events – happens when there is an equal number of events but there is a specific event in time that is different.
   * Failed due to different durations (not yet fully established/refurbished?)
2. Sequence of Events Sheet – Contains the sequence of events that are in the State Sequencer ***excluding*** the Normal States. In some Shark meter series, normal events are not recorded.

A screenshot of a spreadsheet

Description automatically generated

In this Configuration the actual sequence is :

(Normal means All V=120V and C = 5A)

* Normal (1 min )
* Va Sag (1s)
* Normal (2 mins)
* Vb Sag (1s)
* Normal (2 mins)
* Vc Sag (1s)
* Normal (2 mins)
* Ia Sag (1s)
* Normal (2 mins)
* Ib Sag (1s)
* Normal (2 mins)
* Ic Sag (1s)
* Normal (2 mins)
* …and so on..

1. Comparison Table Sheet – a summary containing all waveform events for all meters that are parallel to omicron source.

A screenshot of a computer

Description automatically generated

1. Meters’ Sheet – a specific sheet named after the model of the meter. Initially they are blank unless the user imports a log of that meter.

A screenshot of a computer

Description automatically generated

This is the space where the logs will be pasted and arranged. Also, this is where the evaluation of logs is pasted (unless suggested to move).

This is what it looks like after importing. Just pure raw data.A screenshot of a computer

Description automatically generated

This is what it looks like after processing.

A screenshot of a computer

Description automatically generated

*Column B* will contain Expected values and *Column C* will contain all events that are not Normal from the sheet itself. *Green* coloring is automatic and indicates that the logs match the expected values.

Logic:

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer

Description automatically generated

Codes:

Private Sub MeterCheck()

MeterType = "" 'Making it empty first

StatAdd = ""

If ActiveCell.Address = "$B$6" Then

MeterType = "Shark 200"

StatAdd = ActiveCell.Address

ElseIf ActiveCell.Address = "$B$7" Then

MeterType = "Shark 250"

StatAdd = ActiveCell.Address

ElseIf ActiveCell.Address = "$B$8" Then

MeterType = "Shark 270"

StatAdd = ActiveCell.Address

ElseIf ActiveCell.Address = "$B$9" Then

MeterType = "Nexus 1450"

StatAdd = ActiveCell.Address

ElseIf ActiveCell.Address = "$B$10" Then

MeterType = "Nexus 1500+"

StatAdd = ActiveCell.Address

Else

MeterType = "Error"

MsgBox "Meter does not exist!"

Exit Sub

End If

End Sub

Public Sub BImpo()

'------------------------------------------- Initialization

Dim CsvF As String 'csv dir

Set exApp = New Excel.Application

'------------------------------------------- Process

'----- Opening Base File

CsvF = Application.GetOpenFilename()

If CsvF = "False" Then

MsgBox "No Selected File"

Cont = False

Exit Sub

End If

If (Right(CsvF, 4)) <> ".csv" Then

MsgBox "Invalid File"

Cont = False

Exit Sub

End If

Set csvWB = exApp.Workbooks.Open(CsvF)

'----- Copying Base

csvWB.ActiveSheet.Activate

csvWB.ActiveSheet.Range("A1").Activate

Call MeterCheck

If MeterType = "Error" Then

Cont = False

Exit Sub

End If

'----Coordinates

R = 0

Co = 0

R = csvWB.ActiveSheet.UsedRange.Rows.Count

Co = csvWB.ActiveSheet.UsedRange.Columns.Count

'-------------------Check if MeterType is Histo, if True then Sort

csvWB.ActiveSheet.Range("A1", Cells(R, Co).Address()).Copy

csvWB.Close 'Making sure the CsvFile is terminated

Cont = True 'Flagging that there is an item in the Clipboard

'MsgBox StatAdd

'MsgBox "Done Copying"

End Sub

Public Sub BPaster()

Set Compa = ActiveWorkbook.Sheets(MeterType)

'----- Pasting

Compa.Activate

'If Compa.Range("A1") = "" Then

Compa.Range("E1").Activate

'Compa.Range("A1") = "Meter 1"

'Compa.Range("A2").Activate

'Else

'Compa.Range(Range("A1").End(xlDown).Address).Offset(2).Activate

'ActiveCell = "Meter 1"

'Compa.Range(Range("A1").End(xlDown).Address).Offset(3).Activate

'End If

Compa.Paste

Compa.Range("E1", Cells(R, 21).Address()).Sort Key1:=Range("E1"), Order1:=xlAscending, Header:=xlYes

ActiveWorkbook.Sheets("Main").Activate

ActiveWorkbook.Sheets("Main").Range(StatAdd).Value = "Stored"

'MsgBox "Done Pasting"

End Sub

Public Sub Clears()

Dim StatsAdd As String

Call MeterCheck

If MeterType = "Error" Then

Cont = False

Exit Sub

End If

Set Compa = ActiveWorkbook.Sheets(MeterType)

StatsAdd = ActiveCell.Address

Compa.Activate

'Filee.Show 'Safety Prompt

'If Cleardata = True Then

Compa.UsedRange.ClearContents

Compa.UsedRange.ClearFormats

Cleardata = False

'Else

'Exit Sub

'End If

ActiveWorkbook.Sheets("Main").Activate

ActiveWorkbook.Sheets("Main").Range(StatsAdd).Value = ""

ActiveWorkbook.Sheets("Main").Range(StatsAdd).Offset(, 1).Value = ""

'MsgBox "Done Clearing"

End Sub

CurrStat = ActiveCell.Value

If ActiveCell.Address = "$B$6" And CurrStat = "Stored" Then

MType = "Shark 200"

ResAdd = ActiveCell.Address

ElseIf ActiveCell.Address = "$B$7" And CurrStat = "Stored" Then

MType = "Shark 250"

ResAdd = ActiveCell.Address

ElseIf ActiveCell.Address = "$B$8" And CurrStat = "Stored" Then

MType = "Shark 270"

ResAdd = ActiveCell.Address

ElseIf ActiveCell.Address = "$B$9" And CurrStat = "Stored" Then

MType = "Nexus 1450"

ResAdd = ActiveCell.Address

ElseIf ActiveCell.Address = "$B$10" And CurrStat = "Stored" Then

MType = "Nexus 1500+"

ResAdd = ActiveCell.Address

Else

MsgBox "Select the correct/non-empty cell."

Exit Sub

End If

Set ComS = ActiveWorkbook.Sheets("Comparison Table")

Set SciS = ActiveWorkbook.Sheets("Sequence Of Events")

Set ThisW = ActiveWorkbook.Sheets(MType)

Set Ma = ActiveWorkbook.Sheets("Main")

events = ThisW.Range("I2:I100")

Dim even As Integer

Rr = 0

For even = 1 To UBound(events)

If Right(events(even, 1), 6) <> "Normal" And events(even, 1) <> "" Then

Rr = Rr + 1

End If

Next even

'--------------------------------------------Pasting Comparator

Dim iRange As Range

Dim iCells As Range

Set iRange = ThisW.Range("B1", Cells(Rr + 4, 3).Address())

For Each iCells In iRange

iCells.BorderAround \_

LineStyle:=xlContinuous, \_

Weight:=xlThin

Next iCells

ThisW.Columns("B:C").HorizontalAlignment = xlCenter

'---------------------------------- Expected

Dim cnt As Integer

cnt = Application.WorksheetFunction.CountA(SciS.Range("F8:F1000")) 'sci

ThisW.Range("B1:B2", "B4").Interior.Color = RGB(256, 236, 156)

ThisW.Range("B3").Interior.Color = RGB(255, 255, 255)

ThisW.Range("B1").Value = "Expected"

ThisW.Range("B2").Value = "Duration"

ThisW.Range("B3").Value = SciS.Range("F6").Value

ThisW.Range("B4").Value = "Events"

ThisW.Range("B5", Cells(cnt + 4, 2).Address()).Value = SciS.Range("F8", Cells(cnt + 7, 6).Address()).Value

'--------------------------------- Actual

Dim eve As Integer

ThisW.Range("C1:C2", "C4").Interior.Color = RGB(208, 236, 252)

ThisW.Range("C3").Interior.Color = RGB(255, 255, 255)

ThisW.Range("C1").Value = "Actual"

ThisW.Range("C2").Value = "Duration"

ThisW.Range("C3").Value = Application.WorksheetFunction.Round(Application.WorksheetFunction.Average(ThisW.Range("H2", Cells(Rr, 8).Address())), 0) & " ms"

ThisW.Range("C4").Value = "Events"

ThisW.Range("C4").Font.Color = vbBlack

Dim Val As Integer

Val = 0

For eve = 1 To UBound(events)

If Right(events(eve, 1), 6) <> "Normal" And events(eve, 1) <> "" Then

Val = Val + 1

'MsgBox Val

ThisW.Range(Cells(4 + Val, 3).Address()).Value = events(eve, 1)

End If

Next eve

'-----------------------------------Evaluating Part

Dim Dur1 As String

Dim Dur2 As String

Dim Tr As Boolean

Dim i As Integer

Dim Err As Integer

Dim ErrM As String

E = Application.WorksheetFunction.CountA(ThisW.Range("B5:B5000"))

A = Application.WorksheetFunction.CountA(ThisW.Range("C5:C5000"))

Dur1 = ThisW.Range("B3").Value

Dur2 = ThisW.Range("C3").Value

Err = 0

'----------------------------Coloration

For i = 1 To Rr

If ThisW.Range(Cells(4 + i, 2).Address()).Value <> ThisW.Range(Cells(4 + i, 3).Address()).Value Then

ThisW.Range(Cells(4 + i, 3).Address()).Interior.Color = RGB(205, 92, 92)

'ThisW.Range(Cells(4 + i, 3).Address()).Font.Color = RGB(156, 0, 6)

Err = Err + 1

ElseIf ThisW.Range(Cells(4 + i, 2).Address()).Value = ThisW.Range(Cells(4 + i, 2).Address()).Value Then

ThisW.Range(Cells(4 + i, 3).Address()).Interior.Color = RGB(144, 238, 144)

End If

Next i

'----------------------Count errors

If Err = 0 Then

ErrM = ""

Else

ErrM = CStr(Err) & " Events"

End If

'----------------------- Higlight Durations

If Dur1 <> Dur2 Then

ThisW.Range("C3").Interior.Color = RGB(205, 92, 92)

Else

ThisW.Range("C3").Interior.Color = RGB(144, 238, 144)

End If

'---------------------------Decisioning

If E <> A Then ' Unequal # of Events

Ma.Range(ResAdd).Value = "Failed"

Ma.Range(ResAdd).Offset(, 1).Value = "Unequal # of Events | " & ErrM

ElseIf Dur1 <> Dur2 And Err = 0 Then 'Diff Duration

Ma.Range(ResAdd).Value = "Failed"

Ma.Range(ResAdd).Offset(, 1).Value = "Unequal Durations"

ElseIf E = A And Err > 0 Then

Ma.Range(ResAdd).Value = "Failed"

Ma.Range(ResAdd).Offset(, 1).Value = "Different " & ErrM

ElseIf E = A And Err = 0 Then

Ma.Range(ResAdd).Value = "Satisfactory"

End If

End Sub