Appendix A: X-Parameter Extractor Code

The code to extract X-parameters within the MUF was implemented as a post-processor in VB.net. The user is given the choice to perform the extraction using the PNA-X or an algorithm which has been implemented within the post-processor. This algorithm can perform the extraction much faster as it can be run on more powerful hardware than the PNA-X and also parallelised (e.g. on compute clusters). This is useful for uncertainty propagation where large numbers of samples need to be processed.

The code listing below contains the functions used to perform the extraction using the PNA-X. This involves ensuring the measurement files have been transferred to storage which the instrument can access (e.g. a network-mounted drive), and that the PNA-X NVNA DCOM library (available from Keysight) has been registered on the computer executing the code.

```
Private Sub PNAX_Initialize_XP_Extraction(myPNAXAddress As String)
Try
        myNVNA = CreateObject("AgilentNVNA.Application", myPNAXAddress)
        If IsNothing(myNVNA) Then
                Throw New System.IO.FileNotFoundException
        End If
        myNVNA.Preset()
        myNVNA.XparameterEnabled = True
        If myNVNA.XparameterEnabled = False Then
                Throw New System.NotSupportedException
        End If
Catch
        Throw ' Pass exception to caller
End Try
End Sub

Private Function PNAX_Extract_XPs(myMDIF As MDIF,
 myPNAXAddress As String, myLocalPath As String,
 myPNAXPath As String) As Object
' Write MDIF file for PNA-X to access
myMDIF.Write(IO.Path.Combine(myLocalPath, "dut.mdf"))
' Perform extraction on PNA-X
Dim success As Boolean = myNVNA.GenerateXParamFromFiles(IO.Path.Combine(
 myPNAXPath, "dut.mdf"), IO.Path.Combine(myPNAXPath, "dut.xnp"), False)
If Not success Then
        Throw New System.IO.InvalidDataException
End If
```

```
' Read in result from PNA-X
Dim myXNP As New MDIF
myXNP.Read(IO.Path.Combine(myLocalPath, "dut.xnp"))
Return myXNP
End Function
```

The custom X-parameter is implemented in the code listing below. The straightforward ".xnp" file generation and formatting is omitted from the end of the listing for brevity as it contains many boilerplate strings.

```
Private Function MUF_Extract_XPs(myMDIF As MDIF,
 normalize_phase As Boolean) As Object

' We can either sweep through AN_1_1 of each stimulus tone and write
' out each block at a time to an xnp file, or build a big array of
' values and then write them out altogether. We use the latter.

' 1. Set blockVAR index to 0
' 1b. Get shape of ET states from first blockVARs
' 2. Increment blockVAR index, if valid get values of indepVARs
' 3.    Get block indices of that set of indepVARs
' 4.    Build index of ET states
' 5.    Extract X-Parameters using this index
' 6. Loop
' 7. Write X-Parameters to file

Dim blockVAR_index As Integer = 0 ' 1. Set blockVAR index to 0
Dim current_block_VARs As HPList = Nothing
Dim ET_vars As String() = {"ssport", "ssfreq", "ssphase"}

' 1b. Get shape of ET states from first blockVARs
Dim ssports As Integer = 1
Dim ssfreqs As Integer = 1
Dim ssphases As Integer = 1
Dim current_ssport As Double = 1
Dim current_ssfreq As Double = 0
Dim current_ssphase As Double = 1
While True
  current_block_VARs = myMDIF.BlockVARs(blockVAR_index)
  For i As Integer = 0 To current_block_VARs.count - 1
    Dim name As String = current_block_VARs.GetHPName(i)
    Dim value As String = current_block_VARs.GetValueDouble(i)
    If ET_vars.Contains(name) Then
      Select Case name
        Case "ssport"
          If value < current_ssport Then
            Exit While
          End If
          If value > current_ssport Then
            ssports = ssports + 1
```

```
            current_ssport = value
          End If
        Case "ssfreq"
          If value > current_ssfreq Then
            ssfreqs = ssfreqs + 1
            current_ssfreq = value
          End If
        Case "ssphase"
          If value > current_ssphase Then
            ssphases = ssphases + 1
            current_ssphase = value
          End If
        Case Else
      End Select
    End If
  Next
  blockVAR_index = blockVAR_index + 1
End While

Dim n_X_params As Integer = ((ssfreqs - 1) * ssports * 2 + 1) *
 (ssfreqs - 1) * ssports 'XFpk, XSpkql, XTpkql
Dim X_params As New ComplexMatrix(myMDIF.BlockCount, n_X_params)
' We'll trim the rows later
Dim X_param_block_indices(myMDIF.BlockCount) As Integer
' And these rows
Dim X_param_index As Integer = 1
blockVAR_index = 0

While True
  ' 2. Increment blockVAR index, if valid get values of indepVARs.
  If (blockVAR_index = myMDIF.BlockCount) Then
    ' We've got through all the stimulus conditions!
    Exit While
  Else

    current_block_VARs = myMDIF.BlockVARs(blockVAR_index)
    Dim indepVar_sweep_array(current_block_VARs.count - 4)
     As MDIF_Var_Sweep
    Dim j As Integer = 0
    For i As Integer = 0 To current_block_VARs.count - 1

      Dim name As String = current_block_VARs.GetHPName(i)
      Dim value As Double = current_block_VARs.GetValueDouble(i)

      ' Unless it's the ET variables...
      If ET_vars.Contains(name) Then
        Continue For
      End If
```

```
    ' Add the indepVar to our sweep object array
    indepVar_sweep_array(j) = New MDIF_Var_Sweep(name, value, value)
    j += 1
Next

' 3.    Get block indices of that set of indepVARs
Dim ET_states As Integer() = myMDIF.GetBlockIndexFromVarRanges(
 indepVar_sweep_array)

' 4.    Build index of ET states
Dim ET_index(ssports - 1, ssfreqs - 1, ssphases - 1) As Integer
Dim index As Integer = 0
For ssport As Integer = 0 To ssports - 1
  For ssfreq As Integer = 0 To ssfreqs - 1
    For ssphase As Integer = 0 To ssphases - 1
      ET_index(ssport, ssfreq, ssphase) = ET_states(index)
      index = index + 1
    Next
  Next
Next

' 5.    Extract X-Parameters using this index

' Fill matrices

Dim B_s(ssports - 1, ssfreqs - 1, ssphases - 1) As ComplexMatrix
Dim A_s(ssports - 1, ssfreqs - 1, ssphases - 1) As ComplexMatrix

For ssport As Integer = 0 To ssports - 1
  For ssfreq As Integer = 0 To ssfreqs - 1
    For ssphase As Integer = 0 To ssphases - 1
      Dim block_index As Integer = ET_index(ssport, ssfreq,
       ssphase)
      Dim this_block As RealMatrix = myMDIF.BlockMatrix(
       block_index).CreateRealMatrix
      Dim A As New ComplexMatrix(ssfreqs - 1, ssports)
      Dim B As New ComplexMatrix(ssfreqs - 1, ssports)
      Dim P As Complex
      P = toComplex(this_block.Rarray(0, 1),
       this_block.Rarray(0, 2))
      P = P / Abs(P)
      For port As Integer = 0 To ssports - 1
        For freq As Integer = 0 To ssfreqs - 2
          ' this_block: freq, A1 real, A1 imag,
          ' B1 real, B1 imag, A2 real, A2 imag.
          ' Add one to complex matrix indices because
          ' they are 1-indexed
          A(freq + 1, port + 1) = toComplex(
           this_block.Rarray(freq, port * 4 + 1),
```

```
            this_block.Rarray(freq, port * 4 + 2))
          B(freq + 1, port + 1) = toComplex(
            this_block.Rarray(freq, port * 4 + 3),
            this_block.Rarray(freq, port * 4 + 4))
          A(freq + 1, port + 1) = A(freq + 1, port + 1) +
           New Complex(1.0E-17 * (port + 1), 1.0E-17)
          B(freq + 1, port + 1) = B(freq + 1, port + 1) +
           New Complex(1.0E-17 * (port + 1), 1.0E-17)
        Next
      Next
      A_s(ssport, ssfreq, ssphase) = A
      B_s(ssport, ssfreq, ssphase) = B
    Next
  Next
Next

' Next step
Dim X_columns As Integer = (ssfreqs - 1) * ssports * 2 + 1 - 1
'-1 as we are fitting XSpk11 and XTpk11 together
Dim X As New ComplexMatrix(ssports * ssfreqs * ssphases - ssphases,
 X_columns) ' Implicit -1 as we don't include ET on A11
Dim Y As New ComplexMatrix(ssports * ssfreqs * ssphases - ssphases)
Dim ET_i As Integer
Dim A0 As New Complex(0, 0)
Dim A0s As New ComplexMatrix(ssfreqs - 1, ssports)
Dim s As New ComplexMatrix(X_columns)

' Calculate A0
Dim OPT_average_A0 As Boolean = False
If OPT_average_A0 Then
  For ET_port As Integer = 0 To ssports - 1
    For ET_phase As Integer = 0 To ssphases - 1
      A0 = A0 + A_s(ET_port, 0, ET_phase)(1, 1) /
       (ssports * ssphases)
    Next
  Next
  Else
  A0 = A0 + A_s(0, 0, 0)(1, 1)
End If

For port As Integer = 0 To ssports - 1
  For freq As Integer = 0 To ssfreqs - 2
    ET_i = 1
    For ssport As Integer = 0 To ssports - 1
      For ssfreq As Integer = 0 To ssfreqs - 1
        If ssport = 0 And ssfreq = 1 Then Continue For
        For ssphase As Integer = 0 To ssphases - 1
          Y(ET_i) = B_s(ssport, ssfreq, ssphase)(freq + 1,
           port + 1)
```

```
        X(ET_i, 1) = toComplex(1, 0)
        For a_port As Integer = 0 To ssports - 1
          For a_freq As Integer = 0 To ssfreqs - 2
            If a_port = 0 And a_freq = 0 Then
            X(ET_i, (a_port * (ssfreqs - 1) + a_freq) + 2) =
             (A_s(ssport, ssfreq, ssphase)(a_freq + 1,
             a_port + 1) - A0) + New Complex(1.0E-17, 1.0E-17)
            Else
            X(ET_i, (a_port * (ssfreqs - 1) + a_freq) + 2) =
             A_s(ssport, ssfreq, ssphase)(a_freq + 1, a_port + 1)
            X(ET_i, (a_port * (ssfreqs - 1) + a_freq) +
             (ssports * ssfreqs - 1)) = Conj(A_s(ssport, ssfreq,
             ssphase)(a_freq + 1, a_port + 1))
            End If
          Next
        Next
        ET_i += 1
      Next
    Next
  Next

  ' LSE
  s = ((ConjTranspose(X) * X) ^ -1) * (ConjTranspose(X) * Y)

  'XF
  'X_params(X_param_index, port * (ssfreqs - 1) + freq + 1) = s(1)
  Dim XF As New Complex(0, 0)
  XF = B_s(0, 0, 0)(freq + 1, port + 1)
  For ET_port As Integer = 0 To ssports - 1
    For ET_freq As Integer = 0 To ssfreqs - 2
      If ET_port = 0 And ET_freq = 0 Then
        XF = XF - s(2 + (ET_port * (ssfreqs - 1) + ET_freq)) *
         (A_s(0, 0, 0)(freq + 1, port + 1) - A0)
      Else
        XF = XF - s(2 + (ET_port * (ssfreqs - 1) + ET_freq)) *
         A_s(0, 0, 0)(freq + 1, port + 1)
        XF = XF - s(1 + (ET_port * (ssfreqs - 1) + ET_freq) +
         (ssports * (ssfreqs - 1))) * Conj(A_s(0, 0, 0)(freq + 1,
          port + 1))
      End If
    Next
  Next

  X_params(X_param_index, port * (ssfreqs - 1) + freq + 1) = XF
  For q As Integer = 0 To ssports - 1
    For l As Integer = 0 To ssfreqs - 2
      If q = 0 And l = 0 Then
        'XSpk11 = XS + XT
        X_params(X_param_index, ssports * (ssfreqs - 1) + port *
```

```
                (ssfreqs - 1) * ssports * (ssfreqs - 1) + freq *
                ssports * (ssfreqs - 1) + q * (ssfreqs - 1) + l + 1) =
                s(2 + (q * (ssfreqs - 1) + l))
              'XTpk11 = 0
              X_params(X_param_index, ssports * (ssfreqs - 1) + port *
                (ssfreqs - 1) ^ 2 * ssports + freq * ssports *
                (ssfreqs - 1) + q * (ssfreqs - 1) + l + 1 +
                (ssfreqs - 1) ^ 2 * ssports ^ 2) = New Complex(0, 0)
            Else
              'XS
              X_params(X_param_index, ssports * (ssfreqs - 1) + port *
                (ssfreqs - 1) * ssports * (ssfreqs - 1) + freq *
                ssports * (ssfreqs - 1) + q * (ssfreqs - 1) + l + 1) =
                s(2 + (q *  (ssfreqs - 1) + l))
              'XT
              X_params(X_param_index, ssports * (ssfreqs - 1) + port *
              (ssfreqs - 1) ^ 2 * ssports + freq * ssports *
              (ssfreqs - 1) + q * (ssfreqs - 1) + l + 1 +
              (ssfreqs - 1) ^ 2 * ssports ^ 2) = s(1 + (q *
              (ssfreqs - 1) + l) + (ssports * (ssfreqs - 1)))
            End If
          Next
        Next

      Next
    Next

    X_param_block_indices(X_param_index - 1) = blockVAR_index
    X_param_index = X_param_index + 1
    blockVAR_index = ET_states(ET_states.Length - 1) + 1
    'Jump next loop index to next set of indepVARs

  End If

End While
```