Docking Station Commands

The Docking Station communicates at 9600 BAUD, No Parity, 8 Data Bits, 2 Stop Bits. Hand shaking
lines are used for flow control.

There are 5 controls used with the Docking Station, Clear, Status, Read Data, Store Status, and Store
Data.

I Clear Command. <ESC> K

This command will clear all the stored data in the DataKey. The Docking Station will respond with an
<ACK> [Hex 06] if the command was completed successfully. A <NAK> [Hex 15] will be sent back if the
clear was not completed successfully. Note: the Larger DataKeys will take much longer to perform this
function.

Sample Visual Basic Code code; (See notes in section VII to use this code)

Public Sub Clear_DataKey()

    Dim i As Integer
    Dim x As String

    Re_Try_It:
    Screen.MousePointer = 11    'Wait
    'Send the clear command.
    Comm1.Output = Chr$(27) + "K"
    i = Do_Timer(7)            'Wait 7 seconds. Was 3 seconds for smaller DataKeys.
    Do
        x = frm_Main.Comml.Input
        i = Do_Timer(0.15)       '0.05
    Loop Until Comm1.InBufferCount = 0
    If x <> "" Then
        If Asc(x) = 6 Then
            'All OK
        Else
            GoTo Bad_Clear
        End If
    Else
        GoTo Bad_Clear
    End If
Else
    GoTo Bad_Clear
End If
'Put new header into DataKey.
    x = Make_Volume("########", 0)
    Comm1.Output = Chr$(27) + "V" + x + Chr$(4)
    i = Do_Timer(4)            'Wait 2 seconds. Was 2 seconds for smaller DataKeys.
    Do
        x = frm_Main.Comml.Input
        i = Do_Timer(0.1)         'Wait 100 milliseconds
    Loop Until Comm1.InBufferCount = 0

Bad_Clear:
    Screen.MousePointer = 0    'Active
    If x <> "" Then
        If Asc(x) = 6 Then
            g_Dialog = "Cleared OK"
            g_Dialog_Type = MB_Icon_Exclamation + MB_OK
            g_Dialog_Title = "DataKey"
            g_Results = MsgBox(g_Dialog, g_Dialog_Type, g_Dialog_Title)
        End If

D3672 - G

Coddng 01/09/08
Else
    g_Dialog = "DataKey did not clear correctly."+chr$(13)
g_Dialog = g_Dialog + "Do you wish to try again?"
g_Dialog_Type = MB_Icon_Stop + MB_Yes_No
g_Dialog_Title = "Clear Failed"
If g_Results = ID_Yes Then
    GoTo Re.Try_It
End If
End If
End Sub

II  Status (Get Volume) Command <ESC> D

This command will get the number of feed lines and the byte count of the data in the DataKey. The Docking Station software starting with version 2.101 will also return the software version.

| Number of un-used (free) record lines ~|  
| Sample Line;                        |

| 00077,00000630100939+00000630100939 |
-------- ----- --------              ----------|
  |-- Software Version                Bytes in Key----|  
  |              | (For Ver. EZ2 3.0 & newer)               |
  |              | (Any text in this area means it is a new style) |
  |  |--Record Lines                     <EOT> [Hex 04]-----|
  |  |--DataKey Status                      |
  |

Sample Visual Basic Code: (See notes in section VII to use this code)

Public Function Get_Volume()

    Dim i As Integer
    Dim k As Double
    Dim l As Double
    Dim Try As Integer
    Dim x As String
    Try = 0

Do_Again:
    On Error GoTo No_Comm
    Comm1.Output = Chr$(27) + "D"
    'Wait for response.
    i = Do_Timer(5)               'Wait 5 seconds
    x = ""
    k = Timer + 3
    Try = Try + 1
    Do
        x = x + Comm1.Input
        l = Do_Timer(2)               'Wait 2 seconds
        Loop Until Comm1.InBufferCount = 0 Or k < Timer
    g_Dock_Version = ""
    i = InStr(x, "+")
    If i = 64 Then
        g_Dock_Version = LTrim(RTrim(Mid$(x, 16, 10)))
        g_Full_String = x
        x = Mid$(x, 65, 10)
    Else
        If Try < 2 Then
            GoTo Do_Again
        Else
            GoTo Done_Volume
        End If
    End If
End If
Docking Station Commands

Get_Volume = (Val(x) - 64) / 117

'If there is a docking station version number then it has new software.
If g_Dock_Version <> "" Then
    g_New_Doc = 1
End If

Done_Volume:
    Exit Function

No_Comm:
    On Error GoTo 0
    g_Dock_Version = ""
    g_Full_String = ""
    Get_Volume = 0
    g_New_Doc = 0
    Resume Done_Volume
End Function

III Read Data Command <ESC> R
This command is used to read the stored data on the DataKey. Send this command only once. The Docking Station will send all the stored data in the DataKey.

Sample of feed line

<table>
<thead>
<tr>
<th>N6</th>
<th>U</th>
<th>G</th>
<th>T</th>
<th>B4</th>
<th>L6</th>
<th>R6</th>
<th>P6</th>
<th>A6</th>
<th>I8</th>
<th>C5</th>
<th>F</th>
<th>D8</th>
<th>H6</th>
<th>E6</th>
<th>Z</th>
<th>M6</th>
<th>W6</th>
<th>m3</th>
<th>t3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

NNNNNN, U, I, t, fccc, IPIPIP, RRRRRR, CCCCCC, WWWWWW, uuuuuuuu, HH:MM, F, mm-dd-yy, hhhhhh, EEEEEE, Z, MMMM00, GGGGGG, mmm, ttt

NNNNNN = Wagon Number
U = U=Undone, D=Done
I = I=Ingredient loading line, P=Corral feeding line
t = t=Truck loaded ingredient (Space)=Mill loaded ingredient
fccc = Batch number, f=Feeding (1-9) ccc=consecutive number
IPIPIP = Ingredient or Corral code
RRRRRR = Recipe code
CCCCCC = Call weight
WWWWWW = Loaded or Delivered weight
uuuuuuuu = User Identification
HH:MM = Time of day (24 hour format)
F = Date format 0=mm-dd-yy 1=yy-mm-dd 2=dd-mm-yy
mm-dd-yy = Date when item was done
hhhhhh = Head count
EEEEEE = Call weight change at the EZ-3500
Z = Zone added per ECR 03-204 12-10-03 BNC
MMMM00 = (Future) Mixer Rotations or Time
GGGGGG = (Future) Gross weight at completion of this line
Mmm = Motion tolerance
ttt = Weight tolerance

Sample Visual Basic Code: (See notes in section VII to use this code)

To use this sample code, the g_File_Path string variable and the g_File_Name string variable must be set to valid values. The data will be saved into a Comma Separated Value type file. The g_File_Name should have an extension of ".CSV". This will allow the data to be easily taken into programs like Microsoft Excel.

If the received data is not to be placed into a database like Microsoft Access, the lines marked with "{" may be deleted.

Private Sub cmd_Receive_Data_Click()
    Dim File_Num As Integer
    Dim i As Integer
    Dim j As Long
    Dim MD_Volume As String
    Dim Total_Bytes As Double
Dim Total_Mess as string
Dim x As String
Dim y As String
Dim z As String

Start_Again:
'Get the volume name from the memory device.
Clear_Buffer
x = Get_Volume

If x = g_Full_String

'No working DataKey found.
g_Dialog = "There was no working DataKey found." + Chr$(13)
g_Dialog = g_Dialog + "Do you wish to try again?"
g_Dialog_Type = MB_Icon_Stop + MB_Yes_No
g_Dialog_Title = "Can not read data from DataKey"
g_Results = MsgBox(g_Dialog, g_Dialog_Type, g_Dialog_Title)
If g_Results = ID_Yes Then
    GoTo Start_Again
End If
Exit Sub
End If

'Check for amount of data in Memory Device.
If Val(Mid$(x, 65, 10)) <= 65 Then
    Exit Sub
Else
    Total_Bytes = Val(Mid$(x, 65, 10)) - 64
End If

'Check if the Memory Device has been to a mixer.
If Left$(x, 8) = "########" Then
    Exit Sub
End If

MD_Volume = x

'Check if the Memory Device has been read in before.
If Left$(x, 8) = "        " Then
    Exit Sub
End If

'Let's read it, it may have completted feed data.
frm_Main.Comm1.Output = Chr$(27) + "R"
x = ""
i = Do_Timer(1) 'Wait 1 second
Do
    x = x + Comm1.Input
    DoEvents
    i = Do_Timer(0.1) 'Wait 100 milliseconds
Loop Until Comm1.InBufferCount = 0

'Let's save the data to the files
File_Num = FreeFile
Total_Mess = x
Open g_File_Path + g_File_Name For Output As #File_Num
Do
    z = ""
i = InStr(x, Chr$(4))
y = Left$(x, i)
'Is the line started correctly?
If Left$(y, 1) <> Chr$(30) Then
    z = z + "-Missing line start format character."
End If
If Mid$(y, 4, 1) <> Chr$(2) Then
    z = z + "-Missing [STX] character."
End If
y = Right$(y, Len(y) - 4)

'Validate the check sum.
c = Check_Sum(Left$(y, Len(y) - 3))
If c < Asc(Mid$(y, Len(y) - 1, 1)) Then
z = z + "-Invalid check sum character."
End If
y = Left$(y, Len(y) - 3)
{
'Save the data into the Records database if it is not an undone record.
If Mid$(y, 8, 1) = "I" Then GoTo No_Save
frm_Records.db_Records.Recordset.AddNew
'Store the Record Event.
frm_Records.txt_Batch_No = Mid$(y, 14, 4)
q_Des_Search = Mid$(y, 64, 8)
frm_Records.txt_Date_Value = Date_Value(Save_Date)
frm_Records.txt_Time_Value = Time_Value(Mid$(y, 56, 5))
frm_Records.txt_Wagon = LTrim(RTrim(Mid$(y, 1, 6)))
frm_Records.txt_User_ID = LTrim(RTrim(Mid$(y, 47, 8)))
frm_Records.txt_Recipe = LTrim(RTrim(Mid$(y, 26, 6)))
frm_Records.txt_Feeding_No = Mid$(y, 14, 1)
frm_Records.txt_Head_Count = Val(LTrim(RTrim(Mid$(y, 73, 6))))
'Check if this is an Ingredient Loading.
If Mid$(y, 10, 1) = "I" Or Mid$(y, 10, 1) = "i" Then
    frm_Records.chk_Ingredient.Value = 1
    frm_Records.txt_Ingredient = LTrim(RTrim(Mid$(y, 19, 6)))
    frm_Records.txt_I_Call_Wt = Val(Mid$(y, 33, 6))
End If
'Check if this is a Pen delivery.
If (Mid$(y, 10, 1) = "P" Or Mid$(y, 10, 1) = "p") Then
    frm_Records.chk_Corral.Value = 1
    frm_Records.txt_Corral = LTrim(RTrim(Mid$(y, 19, 6)))
    frm_Records.txt_C_Call_Wt = Val(Mid$(y, 33, 6))
End If
'Save the record data if it is not an undone line.
frm_Records.db_Records.Recordset.Update
'If there is a note for this feed line add it to the line.
If z <> "" Then
    z = "," + z
    y = Left$(y, Len(y) - 1)
End If
'Save the feed data line.
Print #File_Num, y; z
No_Save:
x = Right$(x, Len(x) - i)
Loop Until Len(x) < 10
Close #File_Num
Screen.MousePointer = 0
'Set the volume to a 'Has Been Read' status.
x = Make_Volume("        ", Val(Mid$(MD_Volume, 10, 6)))
Comm1.Output = Chr$(27) + "V" + x + Chr$(4)
i = Do_Timer(2)  'Wait for 2 seconds
Do
    x = frm_Comml.Input
    i = Do_Timer(0.1)  'Wait for 100 milliseconds
Loop Until Comm1.InBufferCount = 0
End Sub

IV Store Status Command <ESC> V
This command is used to set the number of feed lines in the DataKey and to keep track of the status of the data on the DataKey.
A DataKey that has new feed data on it should have “########” stored in the status field. The EZ-3500 will store “!!!!!!!!!” in that field once this DataKey has been read into the scale. Once the data on the DataKey has been read into the computer’s memory it is a good practice to store “        “ into the status field if the DataKey is not cleared.

The Docking Station will respond with an <ACK> [Hex 06] if the command was completed successfully. A <NAK> [Hex 15] will be sent back if the volume was not stored successfully.

SSSSSSSS,RRRRRR,VVVVVVV,                                ,RRRRR,C
|<----- 31 User Space -------- >|

Line is 65 characters long with check sum.
SSSSSSSS = Status
RRRRRR = Records loaded to DataKey (Two places in string) This includes the format line.
VVVVVVVV = Version number (can be blank spaces when sending data to DataKey)
C = Check Sum

Sample Line;                                                ----- ------- ------- -------
########,00077,EZ2 3.0,                                   ,00077,
--------- ----- --------                                      |
|      |     |-- Software Version    |                  |
|      |         (For Ver. EZ2 3.0 & newer)                        |
|      |                                                 |
|      |--Record Lines ----------------------------------|
|--DataKey Status

Sample Visual Basic Code; (See notes in section VII to use this code)

Public Save_Status()

'Set the volume to a 'Has Not Been Read' status.
x = Make_Volume("########",Number_Of_Records)
Comm1.Output = Chr$(27) + "V" + x + Chr$(4)
i = Do_Timer(2)  'Wait for 2 seconds
Do
x = frm_ Comm1.Input
i = Do_Timer(0.1)  'Wait for 100 milliseconds
Loop Until Comm1.InBufferCount = 0
Do_Timer (4)    'Wait 4 seconds.
Do
x = frm_Main.Commlnput
i = Do_Timer(0.15)
Loop Until Comm1.InBufferCount = 0
If x <> "" Then
If Asc(x) = 6 Then
'All OK
g_Des_Search = "OK"
Else
'Bad
g_Des_Search = "BAD"
End If
Else
'Bad
End If
End If
End Sub

V Store Data Command
This command will send feed lines to the DataKey. The first line sent to a DataKey before any feed lines are sent must be the format line.
The newer style Docking Station, version 2 or later will respond with an <ACK> [Hex 06] if the command was completed successfully. A <NAK> [Hex 15] will be sent back if the data was not stored successfully. Older style Docking Stations do not respond with an <ACK> or <NAK>.

Sample Format line message;
N6 U G T B4 L6 R6 P6 A6 I8 C5 F D8 H6 E6 Z M6 W6 m3 t3

Sample DATA line message;
NNNNNN,U,I,t,fccc,IPIPIP,RRRRRR,CCCCCC,WWWWW,uuuuuuuu,HH:MM,F,mm-dd-yy,hhhhhh,EEEEE;Z,MM;GGGGG,mmm,ttt[C]

<table>
<thead>
<tr>
<th>NN</th>
<th>Number</th>
<th>U</th>
<th>Ingredient loading line, P=Corral feeding line</th>
<th>I</th>
<th>Ingredient or Corral code</th>
</tr>
</thead>
<tbody>
<tr>
<td>U</td>
<td>U=Undone, D=Done</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>I</td>
<td>t=Truck loaded ingredient M=Mill loaded ingredient Space for Pens</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>fccc</td>
<td>f=Feeding (1-9) ccc=consecutive number</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IPIPIP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RRRRRR</td>
<td>Recipe code</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CCCCCC</td>
<td>Call weight</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WWWWWW</td>
<td>Loaded or Delivered weight</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>uuuuuuu</td>
<td>User Identification, or the maximum size load for this recipe</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HH:MM</td>
<td>Time of day (24 hour format)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>Date format 0=mm-dd-yy 1=yy-mm-dd 2=dd-mm-yy</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mm-dd-yy</td>
<td>Date when item was done</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>hhhhhh</td>
<td>Head count</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EEEEE</td>
<td>Call weight change at the EZ-3500, send 0 in this field.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Z</td>
<td>Zone Added per 0 for none or 1 to 9</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MMMMM</td>
<td>(Future)Mixer Rotations or Time</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GGGGGG</td>
<td>(Future)Gross weight at completion of this line</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mmm</td>
<td>Motion tolerance</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ttt</td>
<td>Weight tolerance</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[C]</td>
<td>Carriage Return Dec 13 Hex 0D</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Message is formatted as follows;

Public Function Send_DataKey(The_Message As String, Flag As Integer)

Dim c As Integer 'Check sum.
Dim i As Integer 'For / Next loop counter
Dim j As Double
Dim x As String
Dim y As String

'Add carrage return to the end of the string
The_Message = The_Message + Chr$(13)

c = Check_Sum(The_Message)

'Format line or Data line
If Flag = 0 Then
    'Data Line
    y = "Rd"
Else
    'Format line
y = "RF"
End If
'Make the whole line.
x = Chr$(30) + y + Chr$(2) + The_Message + Chr$(3) + Chr$(c) + Chr$(4)
Select Case g_New_Doc
  Case 0 'Old Docking Station Software
    g_Des_Search = "OK"
    Do
      If Len(x) > 89 Then
        y = Left$(x, 89)
        x = Right$(x, Len(x) - 89)
        If Len(x) = 0 Then x = ""
      Else
        y = x
        x = ""
      End If
      Comm1.Output = y
      i = 1
      Do_Timer(0.01)  'Wait 10 milliseconds
        'Wait until all characters are sent to DataKey.
        Do
          i = 1
          Loop Until Comm1.OutBufferCount = 0
          'Delay 100 milliseconds to allow the DataKey to save the data.
          i = Do_Timer(0.1)  'Wait 100 milliseconds
        Loop Until x = ""
    Loop
  Case 1 'New Docking Station Software
    Do
      If Len(x) > 89 Then
        y = Left$(x, 89)
        x = Right$(x, Len(x) - 89)
        If Len(x) = 0 Then x = ""
      Else
        y = x
        x = ""
      End If
      Comm1.Output = y
      i = Do_Timer(0.01)  'Wait 10 milliseconds
        'Wait until all characters are sent to DataKey.
        Do
          i = 1
          Loop Until Comm1.OutBufferCount = 0
          'Delay 100 milliseconds to allow the DataKey to save the data.
          i = Do_Timer(0.1)  'Wait 100 milliseconds
        Loop Until x = ""
      Do_Timer(5)  'Wait for 5 seconds
      i = frm_Main.Comm1.Input
      'These lines added to test Ack response from docking station.
      If x <> "" Then
        If InStr(x, Chr$(6)) Then
          g_Des_Search = "OK"
        Else
          g_Des_Search = "Bad"
        End If
      Else
        g_Des_Search = "Bad"
      End If
    End If
End Select

Done_Send:
Send_DataKey = g_Des_Search
If Send_DataKey <> "OK" Then
  g_Dialog = "The DataKey must be cleared and feed data reloaded."
  g_Dialog_Type = MB_Icon_Stop + MB_OK
  g_Dialog_Title = "Error Writing to DataKey"
  g_Results = MsgBox(g_Dialog, g_Dialog_Type, g_Dialog_Title)
End If
VI Subroutines and Functions used by the above samples.
(See notes in section VII to use any of this code)

**************************************************************************************
Public Function Do_Timer(Pause As Double)

' This routine will pause for the number of seconds selected by the
' [Pause] value or until there is a character in the input buffer to service.
' [Pause] can be set to a minimum of 0.001 seconds.

'Set the pause flag.
g_Do_Time = 1
Do_The_Time.Enabled = False
Do_The_Time.Interval = Pause * 1000
Do_The_Time.Enabled = True
'It will pause in this loop until the time has expired or there is a
'character in the receive buffer to service.
Do
  DoEvents
Loop Until g_Do_Time = 0 Or Comm1.InBufferCount > 0
Do_The_Time.Enabled = False
End Function
**************************************************************************************
Public Function Make_Volume(Status As String, Count As Long)

Dim c As Integer    'Check sum
Dim x As String     'Build string

'This routine makes a status line that is ready to be sent to the Docking Station. The line is
'formatted as follows;
'SSSSSSSSS,RRRRR,VVVVVVVV,                                 ,RRRRR,C
  |<----- 331 User Space -------- >|
'Line is 65 characters long with check sum.
'SSSSSSSSS = Status
'RRRRR    = Records loaded to DataKey (Two places in string)
'VVVVVVVV = Version number (can be blank spaces)
'C       = Check Sum

x = Left$(Status + String$(8, 32), 8) + "",  'Status
x = x + Format(Count, "00000") + ","     'Record count
x = x + "02.101,"                         'Version number
x = x + "",  'Space takers
x = x + Format(Count, "00000") + ","     'Record count
C = Check_Sum(x)                          'Calculate the check sum
x = x + Chr$(c)                           'Add the check sum

'Return the constructed string.
Make_Volume = x

End Function
**************************************************************************************
Public Function Check_Sum(Message As String)

Dim i As Integer     'For/Next loop counter
Dim j As Integer     'Check sum value

'Check Sum = XOR'ed bits of each character in the message.
'AND'ed with 63 to keep the lower 6 bits.
'OR'ed with 64 to make it a printable character.

'Do the [XOR]
j = 0
For i = 1 To Len(Message)
    j = j Xor Asc(Mid$(Message, i, 1))
Next
'[AND] it to keep the lower 6 bits
j = j And 63
'[OR] it to make it printable.
j = j + 64
'Send the value back.
Check_Sum = j

End Function

**************************************************************************************
Public Sub Clear_Buffer()
    Dim i As Integer
    Dim x As String
    'This routine will clear any characters out of the comm port input buffer.
    On Error GoTo Done_Buffer
    Do
        x = Comm1.Input
        i = Do_Timer(1) 'Wait for 1 second
    Loop Until Comm1.InBufferCount = 0
    Done_Clear:
    On Error GoTo 0
    Exit Sub
    Done_Buffer:
    Resume Done_Clear
End Sub

**************************************************************************************
Function Date_Value(This_Date As Variant)
    Dim x As String
    On Error GoTo Bad_Date
    'Month-Day-Year
    Date_Value = DateDiff("d", "01-01-1990", This_Date)
    Done_Date:
    Exit Function
    Bad_Date:
    Date_Value = 0
    Resume Done_Date
End Function

**************************************************************************************

VII Notes:
1.0 There must be a Timer control placed on a form with the name Do_The_Time.
This control will need the following code;
Private Sub Do_The_Time_Timer()
  'Clear the pause flag.
  g_Do_Time = 0
End Sub

2.0 There must be a communications control placed on a form with the name Comm1.

This control must have the following properties set;
Name : Comm1
Handshaking : 2 – ComRTS
RTSEnable : False
Settings : 9600,n,8,2
All other properties should be OK as the default values

3.0 The following global variable must be defined;
3.1 g_Do_Time declared as an integer.
3.2 g_Dock_Version as a string
3.3 g_Full_String as a string
3.4 g_New_Doc as an integer
3.5 g_Des_Search as string

VIII Sample Files:

1.0 Recipes and Pens by load (Computer designs loads.)

Each load is defined by a batch number. All ingredients to be loaded and all pens to be delivered for this batch are to have the same batch number. The first digit of the batch number is to be the feeding that this batch is for. The next 3 digits can be just a sequential number from 000 to 999 for each days feeding.
All the ingredients used in a recipe are to be given the same batch number. The batch numbers should be numbered from 001 to 999. All the pens for a feeding are to have the same batch number of the feeding multiplied by 1000. For example pens for feeding 2 all would have a batch number of 2000.

### 3.0 Recipes Only (Stationary Mixers and Dump boxes.)

All the ingredients used in a recipe are to be given the same batch number. The batch numbers should be numbered from 001 to 999.

---

**Docking Station Commands**

---

D3672 - G 12

Codding 01/09/08
4.0 Pens Only (Delivery Mixers) (Future Option)

All the pens for a feeding are to have the same batch number of the feeding number multiplied by 1000. For example pens for feeding 2 all would have a batch number of 2000.