ENGINEERING CHANGE NOTICE

Page 1 of 21. ECN **638795**Proj.
ECN

2. ECN Category (mark one) Supplemental ☐ Direct Revision ☒ Change ECN ☐ Temporary ☐ Standby ☐ Supersedeure ☐ Cancel/Void ☐		3. Originator's Name, Organization, MSIN, and Telephone No. JOHN HUBER, 77B60, T4-07, 373-5549		4. USQ Required? [X] Yes [] No TF-98-1260 Rev 0		5. Date 12/21/98	
		6. Project Title/No./Work Order No. System Requirements - AY-102 Densitometer		7. Bldg./Sys./Fac. No. 241-AY		8. Approval Designator Q	
		9. Document Numbers Changed by this ECN (includes sheet no. and rev.) HNF-1569 Rev 0		10. Related ECN No(s). NONE		11. Related PO No. N/A	
12a. Modification Work [] Yes (fill out Blk. 12b) [X] No (NA Blks. 12b, 12c, 12d)		12b. Work Package No. N/A		12c. Modification Work Complete N/A Design Authority/Cog. Engineer Signature & Date		12d. Restored to Original Condition (Temp. or Standby ECN only) N/A Design Authority/Cog. Engineer Signature & Date	
13a. Description of Change Replace HNF-1569 Rev 0 with HNF-1569 Rev 1 attached.							
13b. Design Baseline Document? [] Yes [X] No							
14a. Justification (mark one) Criteria Change [] Design Improvement [X] Environmental [] Facility Deactivation [] As-Found [] Facilitate Const [] Const. Error/Omission [] Design Error/Omission []							
14b. Justification Details Revision 1 incorporates software change control methods and a source code listing in Appendix A. An independent review was performed per IP-0842, Vol IV, Section 4.24. This document revision will not change the Hanford Site collective dose since it has no effect on radiological sources, contamination control, or shielding. (12/4-98)							
15. Distribution (include name, MSIN, and no. of copies) JH HUBER T4-07 (1 ADV) JW Bailey S5-05 WL Adams S5-12 Central Files B1-07						RELEASE STAMP DATE: STA: 4 MAY 17 1998 ID: 2	

ENGINEERING CHANGE NOTICE

Page 2 of 2

1. ECN (use no. from pg. 1)

638795

16. Design Verification Required	17. Cost Impact				18. Schedule Impact (days)	
	ENGINEERING			CONSTRUCTION		
	☐ Yes	Additional	☐ \$	Additional	☐ \$	Improvement ☐
	☒ No	Savings	☐ \$ N/A	Savings	☐ \$ N/A	Delay ☐ N/A

19. Change Impact Review: Indicate the related documents (other than the engineering documents identified on Side 1) that will be affected by the change described in Block 13. Enter the affected document number in Block 20.

SDD/DD	☐	Seismic/Stress Analysis	☐	Tank Calibration Manual	☐
Functional Design Criteria	☐	Stress/Design Report	☐	Health Physics Procedure	☐
Operating Specification	☐	Interface Control Drawing	☐	Spares Multiple Unit Listing	☐
Criticality Specification	☐	Calibration Procedure	☒	Test Procedures/Specification	☐
Conceptual Design Report	☐	Installation Procedure	☐	Component Index	☐
Equipment Spec.	☐	Maintenance Procedure	☐	ASME Coded Item	☐
Const. Spec.	☐	Engineering Procedure	☐	Human Factor Consideration	☐
Procurement Spec.	☐	Operating Instruction	☐	Computer Software	☐
Vendor Information	☐	Operating Procedure	☐	Electric Circuit Schedule	☐
OM Manual	☐	Operational Safety Requirement	☐	ICRS Procedure	☐
FSAR/SAR	☐	IEFD Drawing	☐	Process Control Manual/Plan	☐
Safety Equipment List	☐	Cell Arrangement Drawing	☐	Process Flow Chart	☐
Radiation Work Permit	☐	Essential Material Specification	☐	Purchase Requisition	☐
Environmental Impact Statement	☐	Fac. Proc. Samp. Schedule	☐	Tickler File	☐
Environmental Report	☐	Inspection Plan	☐	NONE	☒
Environmental Permit	☐	Inventory Adjustment Request	☐		☐

20. Other Affected Documents: (NOTE: Documents listed below will not be revised by this ECN.) Signatures below indicate that the signing organization has been notified of other affected documents listed below.

Document Number/Revision

Document Number/Revision

Document Number Revision

None

21. Approvals

[illegible]

S

System Requirements for One-Time-Use Enraf Control Panel Software

John H. Huber

PO BOX 1500, Richland, WA 99352

U.S. Department of Energy Contract DE-AC06-96RL13200

EDT/ECN: 638795

UC: 512 and 2030

Org Code: 74800

Charge Code: 105117/FA20

B&R Code: EW3120071

Total Pages: 252

Key Words: Enraf, Densitometer, Software, Control, Panel, Gauge, Gage, ATG, 854, Level, LIT, Level Indicating Transmitter, Density Indicating Transmitter, Logger

Abstract: This document describes requirements and documentation for software intended for remote operation of Enraf densitometers -- Enraf Control Panel.

* Visual Basic is a registered trademark of Microsoft Corporation.

TRADEMARK DISCLAIMER. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof or its contractors or subcontractors.

Printed in the United States of America. To obtain copies of this document, contact: Document Control Services, P.O. Box 950, Mailstop H6-08, Richland WA 99352, Phone (509) 372-2420; Fax (509) 376-4989.

Release Approval Date 5/13/99

Approved for Public Release

RECORD OF REVISION

(1) Document Number

HNF-1569

Page 1

(2) Title

System Requirements for One-Time-Use Enraf Control Panel Software

CHANGE CONTROL RECORD

(3) Revision

(4) Description of Change - Replace, Add, and Delete Pages

Authorized for Release

(5) Cog. Engr.

(6) Cog. Mgr.

Date

0

(7) EDT 609607

JH Huber

RE Larson

1

RS.

ECN 638795 - Rev 1 replaces rev 0 in its entirety.

JH Huber

RE Larson

[Signature] 4-4-99

[Signature] 5/5/99

CONTENTS

1.0	INTRODUCTION	3
1.1	<u>PURPOSE</u>	3
1.2	<u>SCOPE</u>	3
2.0	SYSTEM PROJECT MANAGEMENT PLANNING	3
2.1	<u>DELIVERABLES</u>	3
2.2	<u>RESPONSIBILITIES</u>	3
3.0	ACCEPTANCE CRITERIA	4
3.1	<u>SOFTWARE INTERFACE</u>	4
3.2	<u>FUNCTIONALITY</u>	4
3.3	<u>SYSTEM REQUIREMENTS AND BASELINE</u>	4
4.0	ALTERNATIVES ANALYSIS	5
5.0	SYSTEM DESIGN DESCRIPTION	6
6.0	SYSTEM USER DOCUMENTATION	6
7.0	SYSTEM CONFIGURATION MANAGEMENT	6
8.0	VERIFICATION (REVIEWS)	8
9.0	VERIFICATION & VALIDATION	8
10.0	REFERENCES	8
	APPENDIX A SOURCE CODE LISTING	A-1

SYSTEM REQUIREMENTS FOR ONE-TIME-USE ENRAF CONTROL PANEL SOFTWARE

1.0 INTRODUCTION

An Enraf Densitometer is installed on tank 241-AY-102. The Densitometer will frequently be tasked to obtain and log density profiles. The activity can be effected a number of ways. Enraf Incorporated provides a software package called "Logger18" to its customers for the purpose of in-shop testing of their gauges. Logger18 is capable of accepting an input file which can direct the gauge to obtain a density profile for a given tank level and bottom limit. Logger18 is a complex, DOS based program which will require trained technicians and/or tank farm entries to obtain the data. ALARA considerations have prompted the development of a more user-friendly, computer-based interface to the Enraf densitometers.

1.1 Purpose

This document records the plan by which this new Enraf data acquisition software will be developed, reviewed, verified, and released.

1.2 Scope

This plan applies to the development and implementation of a one-time-use software program, which will be called "Enraf Control Panel." The software will be primarily used for remote operation of Enraf Densitometers for the purpose of obtaining and logging tank product density profiles.

2.0 SYSTEM PROJECT MANAGEMENT PLANNING

2.1 Deliverables

2.1.1 At completion of this software development project, a Graphical User Interface to Enraf Densitometers will be provided to Tank Farms Project W-320 (C-106 Retrieval). The software package will include this document, a software Acceptance Test Report, and executable software. Preliminary testing of the software shall be provided in the shop/desktop via Acceptance Test Procedure HNF-1991. In-service testing will be performed along with the system Operational Test Procedure OTP-320-010. Subsequent testing of software revisions will be in accordance with requirements set forth in Section 7.0, Software Configuration Management.

2.2 Responsibilities

2.2.1 Single-Shell Tank Engineering will provide the system developer (J. H. Huber). Mr. Huber will provide the software and all related documentation as required per

HNF-PRO-309 "Computer Software Quality Assurance Requirements." Mr. Huber shall also conduct software related testing and assist with the in-service operational testing and software revision requirements that may arise.

- 2.2.2 J. W. Lenstch of Numatec Hanford Company, Disposal Projects (or his delegate) will provide funding and project direction for this activity as well as cost and schedule tracking. Development status will be reported to the designated Project Engineer upon request. The Disposal Projects organization shall also provide or designate and fund supporting organizations/individuals required to effect in-field testing and operation.

3.0 ACCEPTANCE CRITERIA

3.1 Software Interface

The software shall provide a user-friendly interface as compared to the existing Enraf Logger program. The program shall be largely point-and-click motivated.

3.2 Functionality

The software shall provide all functionality delineated in the System Requirements section (3.3) of this document.

3.3 System Requirements and Baseline

- 3.3.1 The program shall be capable of causing an Enraf Series 854 Level Detector to obtain a tank product density profile and sludge level as described in HNF-SD-WM-PCP-013 Rev. 1 "Tank 241-C-106 Waste Retrieval Sluicing Sys Process Control Plan" within the capabilities of the gauge itself.
- 3.3.2 The program shall be capable of saving data to an ASCII text file, listing level vs. density information, date and time. Levels shall be recorded in inches. Densities shall be recorded in kg/m^3 .
- 3.3.3 The program shall be executable on any IBM compatible personal computer running Microsoft Windows 95/98/NT. The computer shall be assumed to have a minimum of 16 Megabytes of RAM, 20 Megabytes of available hard disk space, an available serial port (Comport) and a local, stand-alone printer.
- 3.3.4 The program shall not be designed to interface directly with the Hanford Local Area Network (HLAN) or the Tank Monitor and Control System (TMACS).
- 3.3.5 The user shall be assumed to be a Certified Tank Farm Operator or Instrument Technician with average computer skills.

4.0 ALTERNATIVES ANALYSIS

The following alternative solutions to producing this program were considered. Justifications for not selecting the alternative are also provided.

- 4.1 *Use the Enraf PET for exclusive gauge operation (NOT selected).* The Portable Enraf Terminal is virtually a keyboard used to enter commands into the gauge. The PET must be used at the gauge rather than remotely. It was determined that since density profile scans can take 25-minutes or more, ALARA considerations would be in question.
- 4.2 *Use the Enraf Logger program as supplied by the vendor (NOT selected).* The Enraf Logger program was created by Enraf technicians, for Enraf technicians. Although it is fully capable of fulfilling the basic functional requirements, it is not user friendly or secure. The Logger18 program allows certain operations which should not be permitted during densitometer operation and data logging. Also, in order to obtain density profiles with Logger18 it requires manual editing of certain text files, which then become the program input. There is no capability for on screen input and execution of density profile scans.
- 4.3 *Modify the Enraf Logger program to be more user friendly (NOT selected).* Enraf has provided the Logger program source code to Lockheed Martin Hanford Company free of charge (but have retained copyright restrictions). The program was written in Microsoft QuickBASIC Professional Edition 7.1. This software is no longer maintained by Microsoft. A copy of the standard QuickBASIC 4.5 edition was obtained, but there appears to be some compatibility problems with code syntax. The code is also very complex and is highly annotated in Dutch (not English). Some translation has been effected using a Dutch-English dictionary, but many sentence constructions have precluded literal (word-by-word) translation of critical portions of the code. It is not assumed that funding is available to contract a professional Dutch translator or the direct assistance of Enraf Inc., The Netherlands.
- 4.4 *Purchase ENTIS software from Enraf Inc (NOT selected).* Enraf provides a software-based tank inventory system with a slightly more user friendly interface. However, the cost of this software is high and is far more powerful and complex than is required for project W-320.
- 4.5 *Convert only the required portions of the Enraf Logger software to Visual Basic (Selected).* Visual Basic 4.0 (and on) is a very powerful and user friendly programming tool. It is packaged with pre-written code (libraries) for serial communications, and Graphical User Interfaces (GUI or a "Windows look and feel"). Out of dozens of routines in the Logger18 code, there are only a handful that are actually needed for densitometer operation. Much of the Logger18 code is screen graphics manipulation, which is automatically handled in Visual Basic. Programming time is short. The GUI can be easily modified to fit the operator's needs. Routines can be easily added to enhance its functionality.

Conversion of the Enraf Logger software was selected to be the best choice not only for the benefit to the operators and ALARA, but ease of programming and the plethora of programming options available.

5.0 SYSTEM DESIGN DESCRIPTION

- 5.1 The system design description has been determined to be not required for this application. SDD's are understood to be necessary for complex software applications. However, this application simply reads and sends ASCII strings from and to a proprietary instrument via a GUI. The subroutines were largely copied from vendor provided source code and therefore remain the property of Enraf Inc. This software will not be modified without further documentation or revisions to this document.

6.0 SYSTEM USER DOCUMENTATION

The system user documentation will be a combination of the Densitometer Operational Procedure OTP-320-010, the Densitometer Maintenance and Operation procedure 7-MISC-238 and user help files provided with the application. A supporting document describing the software and its use will not be produced or necessary.

7.0 SYSTEM CONFIGURATION MANAGEMENT

Since this is considered to be single-use software, a listing of all source programs related to this software fulfills System Configuration Management requirements. A source code listing of the most recent minor release can be found in Appendix A. The listing of the most recent revision shall be kept at directory

\\DS002\H0082654\EnrafControlPanel.

However, in an effort to better document and track changes of the program, a change control process has been developed.

- 7.1 Changes to the software will be addressed as follows. Each change shall be determined by the cognizant engineer to fall within one of the following categories.

7.1.1 Revision.

Revisions are defined to be those changes that

- Effect the GUI such that no user re-training is required.
- Only increment the tertiary portion of the version number (e.g. v1.1.1 to v1.1.2).
- Are documented with a file named "README.TXT" that provides a

revision history and description of the revision along with any known bugs or minor issues with the program. This file is bundled with each setup program and placed in the application directory.

- May require informal, mock-up testing, at most to verify program functionality prior to field deployment. This requirement is determined by the cognizant engineer. A log of informal testing shall be recorded in a numbered Notebook/Logbook. The Notebook/Logbook number shall be recorded in the "README.TXT" file.
- Shall not effect system baseline functionality.

7.1.2 Minor Release

Minor releases are defined to be those changes that

- Effect the GUI such that some user re-training is required.
- Increment the secondary portion of the version number (e.g. v1.1.1 to 1.2.0)
- Are documented with a file named "README.TXT" that provides a revision history and description of the revision along with any known bugs or minor issues with the program. This file is bundled with each setup program and placed in the application directory.
- Require informal, mock-up testing, at most to verify program functionality prior to field deployment. The cognizant engineer determines this requirement. A log of informal testing shall be recorded in a numbered Notebook/Logbook. The Notebook/Logbook number shall be recorded in the "README.TXT" file.
- Require additional in-service testing to verify field data acquisition capability and acceptability. Test procedure may be documented in the work steps of the job control system work package. The work steps will contain sign-offs for acceptability.
- Shall not effect baseline functionality.
- An Engineering Change Notice shall be issued against this document to include the new source code listing.

7.1.3 Major Release

Major Releases are defined to be those changes that

- Effect the GUI such that extensive user re-training is required.
- Increment the first portion of the version number (e.g. v1.1.1 to v2.0.0)
- Require formal Operational Testing, the results of which, including exceptions are documented in an Operational Test Report supporting document.
- May effect baseline functionality.
- An Engineering Change Notice shall be issued against this document to include the new source code listing.
- A Test Report shall be issued documenting acceptance of the new release.

8.0 VERIFICATION (REVIEWS)

Reviews of this and related documents for this project shall be performed in accordance with HNF-PRO-233, "Review and Approval of Documents" (formerly WHC-CM-3-5, Section 12.7). Documents shall be released/reviewed via Engineering Data Transmittal (HNF-PRO-224) and released per engineering document release requirements (HNF-PRO-440, and HNF-PRO-1819).

9.0 VERIFICATION & VALIDATION

As a minimum, for initial deployment of the software program, a software Acceptance Test Procedure (ATP) shall be prepared and performed by the system developer. The test shall be witnessed by a software quality assurance representative. Test execution results shall be prepared and released following the test.

10.0 REFERENCES

HNF-PRO-309 "Computer Software Quality Assurance Requirements"

HNF-PRO-233, "Review and Approval of Documents"

HNF-SD-WM-PCP-013 Rev. 1 "Tank 241-C-106 Waste Retrieval Sluicing Sys Process Control Plan"

HNF-PRO-1819 "PHMC Engineering Requirements"

Appendix A

Source Code Listing

HNF-1569
Revision 1

```
=====
Begin abtglobal.BAS
Attribute VB_Name = "Global"
Option Explicit

#If Win16 Then
    Declare Function ShellAbout Lib "shell.dll" (ByVal hWnd As Integer, ByVal szApp As String, ByVal szOtherStuff As String, ByVal hIcon As Integer) As Long
#ElseIf Win32 Then
    Declare Function ShellAbout Lib "shell32.dll" Alias "ShellAboutA" (ByVal hWnd As Long, ByVal szApp As String, ByVal szOtherStuff As String, ByVal hIcon As Long) As Long
#End If

Sub AppAbout(AppForm As Form, szTitle As String, szOtherInfo As String)
    Call ShellAbout(AppForm.hWnd, szTitle, szOtherInfo, AppForm.Icon)
End Sub
```

End abtglobal.BAS

```
=====
Begin frmAbout.frm

VERSION 5.00
Begin VB.Form frmAbout
    BorderStyle = 3 'Fixed Dialog
    Caption = "About MyApp"
    ClientHeight = 3564
    ClientLeft = 2340
    ClientTop = 1932
    ClientWidth = 5736
    ClipControls = 0 'False
    LinkTopic = "Form2"
    MaxButton = 0 'False
    MinButton = 0 'False
    ScaleHeight = 2453.724
    ScaleMode = 0 'User
    ScaleWidth = 5380.766
    ShowInTaskbar = 0 'False
    Begin VB.PictureBox picIcon
        AutoSize = -1 'True
        ClipControls = 0 'False
        Height = 432
        Left = 240
        Picture = "frmAbout.frx":0000
        ScaleHeight = 263.118
        ScaleMode = 0 'User
        ScaleWidth = 263.118
        TabIndex = 1
        Top = 240
        Width = 432
    End
    Begin VB.CommandButton cmdOK
        Cancel = -1 'True
        Caption = "OK"
        Default = -1 'True
        Height = 345
        Left = 4245
        TabIndex = 0
        Top = 2625
        Width = 1260
    End
    Begin VB.CommandButton cmdSysInfo
        Caption = "&System Info..."
        Height = 345
        Left = 4260
        TabIndex = 2
        Top = 3075
        Width = 1245
    End
    Begin VB.Line Linel
        BorderColor = &H00808080&
        BorderStyle = 6 'Inside Solid
    End
End
```

HNF-1569
Revision 1

```

Index      = 1
X1         = 84.426
X2         = 5309.473
Y1         = 1687.452
Y2         = 1687.452
End
Begin VB.Label lblDescription
Caption    = "App Description"
ForeColor  = &H00000000&
Height     = 1170
Left       = 1050
TabIndex  = 3
Top        = 1125
Width      = 3885
End
Begin VB.Label lblTitle
Caption    = "Application Title"
ForeColor  = &H00000000&
Height     = 480
Left       = 1050
TabIndex  = 5
Top        = 240
Width      = 3885
End
Begin VB.Line Linel
BorderColor = &H00FFFFFF&
BorderWidth = 2
Index       = 0
X1          = 98.497
X2          = 5309.473
Y1          = 1697.779
Y2          = 1697.779
End
Begin VB.Label lblVersion
Caption    = "Version"
Height     = 225
Left       = 1050
TabIndex  = 6
Top        = 780
Width      = 3885
End
Begin VB.Label lblDisclaimer
Caption    = "Warning: ..."
ForeColor  = &H00000000&
Height     = 825
Left       = 255
TabIndex  = 4
Top        = 2625
Width      = 3870
End
End
Attribute VB_Name = "frmAbout"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
' =====
' Program Title:   Enraf Control Panel
' Author:         John H. Huber
' Platform:       Windows 95 (32-bit)
' Software:       MS Visual Basic 5.0 (originally written
'                 in 3.0, then converted to 4.0, and finally
'                 5.0 by the time it went into production)
'
' Version:        1.2
' System Req.     HNF-1569
' ATP:            HNF-1991
' ATR:            HNF-1992
' OTP:            OTP-320-010
' OTR:            HNF-2965
' Company:        Lockheed Martin Hanford Co.
' =====

```

HNF-1569
Revision 1

Option Explicit

```
' Reg Key Security Options...
Const READ_CONTROL = &H20000
Const KEY_QUERY_VALUE = &H1
Const KEY_SET_VALUE = &H2
Const KEY_CREATE_SUB_KEY = &H4
Const KEY_ENUMERATE_SUB_KEYS = &H8
Const KEY_NOTIFY = &H10
Const KEY_CREATE_LINK = &H20
Const KEY_ALL_ACCESS = KEY_QUERY_VALUE + KEY_SET_VALUE + _
    KEY_CREATE_SUB_KEY + KEY_ENUMERATE_SUB_KEYS + _
    KEY_NOTIFY + KEY_CREATE_LINK + READ_CONTROL

' Reg Key ROOT Types...
Const HKEY_LOCAL_MACHINE = &H80000002
Const ERROR_SUCCESS = 0
Const REG_SZ = 1                ' Unicode nul terminated string
Const REG_DWORD = 4            ' 32-bit number

Const gREGKEYSYSINFOLOC = "SOFTWARE\Microsoft\Shared Tools Location"
Const gREGVALSYSINFOLOC = "MSINFO"
Const gREGKEYSYSINFO = "SOFTWARE\Microsoft\Shared Tools\MSINFO"
Const gREGVALSYSINFO = "PATH"

Private Declare Function RegOpenKeyEx Lib "advapi32" Alias "RegOpenKeyExA" (ByVal hKey As Long,
ByVal lpSubKey As String, ByVal ulOptions As Long, ByVal samDesired As Long, ByRef phkResult As
Long) As Long
Private Declare Function RegQueryValueEx Lib "advapi32" Alias "RegQueryValueExA" (ByVal hKey As
Long, ByVal lpValueName As String, ByVal lpReserved As Long, ByRef lpType As Long, ByVal lpData As
String, ByRef lpcbData As Long) As Long
Private Declare Function RegCloseKey Lib "advapi32" (ByVal hKey As Long) As Long

Private Sub cmdSysInfo_Click()
    Call StartSysInfo
End Sub

Private Sub cmdOK_Click()
    Unload Me
End Sub

Private Sub Form_Load()
    Dim m$
    Me.Caption = "About " & App.title
    lblVersion.Caption = "Version " & App.Major & "." & App.Minor & "." & App.Revision
    lblTitle.Caption = App.title
    m = "Enraf Control Panel is designed specifically" + Chr(13) + Chr(10)
    m = m + "for operation of Enraf Series 854 Level Gauges" + Chr(13) + Chr(10)
    m = m + "and densitometers. If you have questions about this" + Chr(13) + Chr(10)
    m = m + "program, contact John Huber (john_h_huber@rl.gov)"
    lblDescription.Caption = m
    m = "Warning: Unauthorized use prohibited."
    m = m & vbCrLf & "Copyright (c) 1997, 1998, 1999 DOE Hanford"
    lblDisclaimer = m
End Sub

Public Sub StartSysInfo()
    On Error GoTo SysInfoErr

    Dim rc As Long
    Dim SysInfoPath As String

    ' Try To Get System Info Program Path\Name From Registry...
    If GetKeyValue(HKEY_LOCAL_MACHINE, gREGKEYSYSINFO, gREGVALSYSINFO, SysInfoPath) Then
        ' Try To Get System Info Program Path Only From Registry...
    ElseIf GetKeyValue(HKEY_LOCAL_MACHINE, gREGKEYSYSINFOLOC, gREGVALSYSINFOLOC, SysInfoPath) Then
        ' Validate Existence Of Known 32 Bit File Version
        If (Dir(SysInfoPath & "\MSINFO32.EXE") <> "") Then
            SysInfoPath = SysInfoPath & "\MSINFO32.EXE"
        End If
    End If
End Sub
```

HNF-1569
Revision 1

```

' Error - File Can Not Be Found...
Else
    GoTo SysInfoErr
End If
' Error - Registry Entry Can Not Be Found...
Else
    GoTo SysInfoErr
End If

Call Shell(SysInfoPath, vbNormalFocus)

Exit Sub
SysInfoErr:
MsgBox "System Information Is Unavailable At This Time", vbOKOnly
End Sub

Public Function GetKeyValue(KeyRoot As Long, KeyName As String, SubKeyRef As String, ByRef KeyVal As
String) As Boolean
    Dim i As Long                ' Loop Counter
    Dim rc As Long               ' Return Code
    Dim hKey As Long             ' Handle To An Open Registry Key
    Dim hDepth As Long           '
    Dim KeyValType As Long       ' Data Type Of A Registry Key
    Dim tmpVal As String         ' Tempory Storage For A Registry Key
    Value
    Dim KeyValSize As Long       ' Size Of Registry Key Variable
    '-----
    ' Open RegKey Under KeyRoot {HKEY_LOCAL_MACHINE...}
    '-----
    rc = RegOpenKeyEx(KeyRoot, KeyName, 0, KEY_ALL_ACCESS, hKey) ' Open Registry Key

    If (rc <> ERROR_SUCCESS) Then GoTo GetKeyError                ' Handle Error...

    tmpVal = String$(1024, 0)                ' Allocate Variable Space
    KeyValSize = 1024                        ' Mark Variable Size

    '-----
    ' Retrieve Registry Key Value...
    '-----
    rc = RegQueryValueEx(hKey, SubKeyRef, 0, _
        KeyValType, tmpVal, KeyValSize) ' Get/Create Key Value

    If (rc <> ERROR_SUCCESS) Then GoTo GetKeyError                ' Handle Errors

    If (Asc(Mid(tmpVal, KeyValSize, 1)) = 0) Then                ' Win95 Adds Null Terminated String...
        tmpVal = Left(tmpVal, KeyValSize - 1)                  ' Null Found, Extract From String
    Else                                                         ' WinNT Does NOT Null Terminate
String...
        tmpVal = Left(tmpVal, KeyValSize)                        ' Null Not Found, Extract String Only
    End If
    '-----
    ' Determine Key Value Type For Conversion...
    '-----
    Select Case KeyValType
    Case REG_SZ                ' Search Data Types...
        KeyVal = tmpVal        ' String Registry Key Data Type
    Case REG_DWORD              ' Copy String Value
        For i = Len(tmpVal) To 1 Step -1
            KeyVal = KeyVal + Hex(Asc(Mid(tmpVal, i, 1))) ' Double Word Registry Key Data Type
        Next                  ' Convert Each Bit
        KeyVal = Format$("&h" + KeyVal)                    ' Build Value Char. By Char.
    End Select                ' Convert Double Word To String

    GetKeyValue = True        ' Return Success
    rc = RegCloseKey(hKey)    ' Close Registry Key
    Exit Function             ' Exit

GetKeyError:                ' Cleanup After An Error Has Occured...
    KeyVal = ""               ' Set Return Val To Empty String
    GetKeyValue = False       ' Return Failure
    rc = RegCloseKey(hKey)    ' Close Registry Key

```


HNF-1569
Revision 1

End Function

```
Private Sub lblDescription_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
' Responds to Ctrl-Shift-Left Click, three times
' The purpose of this sub is to extract passwords in the event
' they are forgotten. Avoids the need for using Regedit to find
' the passwords (they are not encrypted).
```

```
If Button <> 1 Then Exit Sub
If Shift <> 3 Then Exit Sub
Static CCount%
CCount = CCount + 1
If CCount = 3 Then
```

```
Dim m$
```

```
m = "Admin Password = " & AdminPword & vbCrLf
m = m & "Service Password = " & ServPword & vbCrLf
m = m & "Main Password = " & MainPword & vbCrLf
```

```
lblDescription.Caption = m
```

```
ElseIf CCount >= 4 Then
    m = "Enraf Control Panel is designed specifically" + Chr(13) + Chr(10)
    m = m + "for operation of Enraf Series 854 Level Gauges" + Chr(13) + Chr(10)
    m = m + "and densitometers. If you have questions about this" + Chr(13) + Chr(10)
    m = m + "program, contact John Huber (john_h_huber@rl.gov)"
    lblDescription.Caption = m
    CCount = 0
```

```
End If
```

```
End Sub
```

End frmAbout.frm

```
=====
Begin frmCalc.frm
```

```
VERSION 5.00
```

```
Begin VB.Form frmCalc
```

```
    Caption           = "Buoyancy Calculator"
    ClientHeight       = 3660
    ClientLeft         = 45
    ClientTop          = 270
    ClientWidth        = 5670
    LinkTopic          = "Form1"
    ScaleHeight        = 3660
    ScaleWidth         = 5670
    StartUpPosition    = 2 'CenterScreen
```

```
Begin VB.Frame Frame2
```

```
    Caption           = "Output log"
    Height            = 3132
    Left              = 2520
    TabIndex          = 13
    Top               = 120
    Width             = 3012
```

```
Begin VB.CommandButton cmdUse
```

```
    Caption           = "&Use for Sediment Check"
    Enabled            = 0 'False
    Height            = 252
    Left              = 720
    TabIndex          = 17
    Top               = 2760
    Width             = 2172
```

```
End
```

```
Begin VB.TextBox txtRWT
```

```
    Height           = 288
    Left             = 1560
    TabIndex         = 15
    Top              = 2280
    Width            = 1332
```

HNF-1569
Revision 1

```
End
Begin VB.TextBox txtOut
    Height       = 1932
    Left         = 120
    MultiLine    = -1 'True
    ScrollBars   = 3 'Both
    TabIndex     = 14
    Top          = 240
    Width        = 2772
End
Begin VB.Label Label4
    Alignment    = 1 'Right Justify
    Caption      = "Recommended Wire Tension (S2)"
    Height       = 372
    Left         = 120
    TabIndex     = 16
    Top          = 2280
    Width        = 1332
End
End
Begin VB.CommandButton cmdPrint
    Caption      = "&Print"
    Height       = 252
    Left         = 4560
    TabIndex     = 10
    ToolTipText  = "Print output log to printer"
    Top          = 3360
    Width        = 972
End
Begin VB.CommandButton cmdClose
    Caption      = "&Close"
    Height       = 252
    Left         = 120
    TabIndex     = 9
    Top          = 3360
    Width        = 852
End
Begin VB.CommandButton cmdClear
    Caption      = "C&lear"
    Height       = 252
    Left         = 1200
    TabIndex     = 8
    ToolTipText  = "Clear entries"
    Top          = 3360
    Width        = 852
End
Begin VB.CommandButton cmdCalc
    Caption      = "C&alculate"
    Height       = 252
    Left         = 2640
    TabIndex     = 7
    ToolTipText  = "Perform calculations"
    Top          = 3360
    Width        = 1092
End
Begin VB.Frame Frame1
    Caption      = "Input Values"
    Height       = 3132
    Left         = 120
    TabIndex     = 0
    Top          = 120
    Width        = 2292
Begin VB.CommandButton cmdGetDV
    Caption      = "G&et"
    Height       = 252
    Left         = 240
    TabIndex     = 12
    ToolTipText  = "Gets DV from Gauge"
    Top          = 1440
    Width        = 372
End
End
```

HNF-1569
Revision 1

```
Begin VB.CommandButton cmdGetDW
    Caption       = "&Get"
    Height        = 252
    Left          = 240
    TabIndex      = 11
    ToolTipText   = "Gets DW from Gauge"
    Top           = 720
    Width         = 372
End
Begin VB.TextBox txtSPG
    Height        = 288
    Left          = 1320
    TabIndex      = 3
    Text          = "0"
    ToolTipText   = "Enter Specific Gravity of Tank Product"
    Top           = 2160
    Width         = 852
End
Begin VB.TextBox txtDV
    Height        = 288
    Left          = 720
    TabIndex      = 2
    Text          = "0"
    ToolTipText   = "Enter a value or click GET"
    Top           = 1440
    Width         = 1452
End
Begin VB.TextBox txtDW
    Height        = 288
    Left          = 720
    TabIndex      = 1
    Text          = "0"
    ToolTipText   = "Enter a value or click GET"
    Top           = 720
    Width         = 1452
End
Begin VB.Label Label3
    Caption       = "SpG of Liquid:"
    BeginProperty Font
        Name       = "MS Sans Serif"
        Size       = 8.25
        Charset    = 0
        Weight     = 700
        Underline   = 0   'False
        Italic      = 0   'False
        Strikethrough = 0   'False
    EndProperty
    Height        = 252
    Left          = 120
    TabIndex      = 6
    Top           = 1920
    Width         = 1452
End
Begin VB.Label Label2
    Caption       = "Displacer Vol. (DV):"
    BeginProperty Font
        Name       = "MS Sans Serif"
        Size       = 8.25
        Charset    = 0
        Weight     = 700
        Underline   = 0   'False
        Italic      = 0   'False
        Strikethrough = 0   'False
    EndProperty
    Height        = 252
    Left          = 120
    TabIndex      = 5
    Top           = 1200
    Width         = 1692
End
Begin VB.Label Label1
```

HNF-1569
Revision 1

```

Caption      = "Displacer Wt (DW):"
BeginProperty Font
    Name      = "MS Sans Serif"
    Size      = 8.25
    Charset   = 0
    Weight    = 700
    Underline = 0 'False
    Italic    = 0 'False
    Strikethrough = 0 'False
EndProperty
Height       = 252
Left        = 120
TabIndex    = 4
Top         = 480
Width       = 1692
End
End
Attribute VB_Name = "frmCalc"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
Option Explicit

Private Sub cmdCalc_Click()
' Calculates the optimum wire tension setting for S2
' such that the displacer will sink into the liquid interface
' without excessively tipping over at the sludge or tank bottom
' interface

On Error Resume Next
Dim DV As Double
Dim DW As Double
Dim SpG As Double
Dim BuoyantForce As Double
Dim WireTension As Double
Dim RecWireTension As String
Dim CR
CR = Chr(13) + Chr(10)

If txtDV.text = "" Then txtDV.text = "0"
If txtDW.text = "" Then txtDW.text = "0"
If txtSPG.text = "" Then txtSPG.text = "0"
DV = CDBl(txtDV.text)
DW = CDBl(txtDW.text)
SpG = CDBl(txtSPG.text)

'note, dimensions are in grams and cm
BuoyantForce = SpG * DV

WireTension = DW - BuoyantForce

RecWireTension = Str(WireTension - 25)
txtRWT.text = RecWireTension

'Display results
txtOut.text = txtOut.text + "Calculation" + CR
txtOut.text = txtOut.text + Time$ + " " + Date$ + CR
txtOut.text = txtOut.text + "=====" + CR
txtOut.text = txtOut.text + "BuoyantForce = " + Str(BuoyantForce) + CR + CR
txtOut.text = txtOut.text + "Displacer wt" + CR
txtOut.text = txtOut.text + "at neutral buoyancy = " + Str(WireTension) + CR + CR
txtOut.text = txtOut.text + "Recommended Wire Tension " + CR
txtOut.text = txtOut.text + "setting (S2) for sending " + CR
txtOut.text = txtOut.text + "displacer to tank bottom: " + CR
txtOut.text = txtOut.text + "      " + RecWireTension + CR
txtOut.text = txtOut.text + ">>>>>END CALCULATION<<<<<<" + CR + CR

```

```
End Sub

Private Sub cmdClear_Click()
txtDW.text = "0"
txtDV.text = "0"
txtSPG.text = "0"
End Sub

Private Sub cmdClose_Click()
Unload Me
End Sub

Private Sub cmdGetDV_Click()
' Places the actual DV value in the calculator screen

' if not connected, then connect
If gaugeChecked = 0 Then
    checkGageAddress ta, ja
End If

If ja = 0 Then
    Exit Sub
End If

    Dim echo$, answer$
    ciucom "ZDV", echo, status
    decode "DV", echo, 64, answer
    txtDV.text = answer

End Sub

Private Sub cmdGetDW_Click()
' Places the actual DW value in the calculator screen
If gaugeChecked = 0 Then
    checkGageAddress ta, ja
End If

If ja = 0 Then
    Exit Sub
End If

    Dim echo$, answer$
    ciucom "ZDW", echo, status
    decode "DW", echo, 64, answer
    txtDW.text = answer

End Sub

Private Sub cmdPrint_Click()
Printer.Print txtOut.text
Printer.EndDoc
End Sub

Private Sub cmdUse_Click()
' Sends the calculated value to the input box in the
' frmDensity.GetSedimentLevel routine.
frmCalc.Hide
cmdUse.Enabled = False
End Sub

Private Sub Form_Load()
' Preloads the calculator with DW & DV Values from the
' gauge.

If gaugeChecked <> 0 Then
```

```

Dim echo$, answer$
ciucom "ZDW", echo, status
decode "DW", echo, 64, answer
txtDW.text = answer

ciucom "ZDV", echo, status
decode "DV", echo, 64, answer
txtDV.text = answer

End If

txtSPG.text = LastSpG

End Sub

Private Sub Form_Unload(Cancel As Integer)
' Save the SpG used
LastSpG = txtSPG.text
End Sub

Private Sub txtOut_Change()
' Make sure txt box arrows down to bottom.
txtOut.SelStart = Len(txtOut.text)
End Sub

End frmCalc.frm
=====
Begin frmCFG.frm

VERSION 5.00
Begin VB.Form frmCFG
BorderStyle = 1 'Fixed Single
Caption = "Current Gauge Setup/Configuration"
ClientHeight = 4200
ClientLeft = 30
ClientTop = 270
ClientWidth = 4380
LinkTopic = "Form1"
MaxButton = 0 'False
MinButton = 0 'False
ScaleHeight = 4200
ScaleWidth = 4380
StartupPosition = 3 'Windows Default
Begin VB.CommandButton cmdPrint
Caption = "&Print"
Height = 252
Left = 1440
TabIndex = 4
Top = 3840
Width = 1332
End
Begin VB.CommandButton cmdAbort
Caption = "&Abort"
Enabled = 0 'False
Height = 252
Left = 1440
TabIndex = 3
Top = 3600
Width = 1332
End
Begin VB.CommandButton cmdMC
Caption = "&Make Changes"
Height = 252
Left = 2760
TabIndex = 2
Top = 3600
Width = 1332
End
Begin VB.CommandButton cmdClose
Caption = "&Close"
Height = 252
Left = 240
TabIndex = 1

```

```

        Top          = 3600
        Width        = 1212
    End
    Begin VB.TextBox txtCFG
        Height        = 3372
        Left          = 120
        MultiLine     = -1 'True
        ScrollBars    = 3 'Both
        TabIndex      = 0
        Top           = 120
        Width         = 4092
    End
End
Attribute VB_Name = "frmCFG"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
Sub cmdAbort_Click()
    Abort = 1
End Sub
Sub cmdClose_Click()
    cmdAbort_Click
Unload Me
Abort = 0
End Sub

Private Sub cmdMC_Click()
    ' Opens the Service screen to make changes to the gauge.
    If ServPwordFlag = 1 Then
        frmSPW.Label2.Caption = ServPword
        frmSPW.Show 1
    If frmSPW.Label2.Caption = "GO" Then
        frmSPW.Label2.Caption = ""
        Unload frmSPW
        frmService.Show
    End If
    Else
        frmService.Show
    End If
End Sub

Private Sub cmdPrint_Click()
    Printer.Print txtCFG.text
    Printer.EndDoc
End Sub

Private Sub Form_Load()
    frmCFG.Left = 0.25 * frmCFG.Width 'frmMain.Left
    frmCFG.Top = frmMain.Top + frmMain.Height
End Sub

Private Sub txtCFG_Change()
    txtCFG.SelStart = Len(txtCFG.text)
End Sub

                                End frmCFG.frm
=====
                                Begin FRMCIU.FRM
VERSION 5.00
Begin VB.Form frmCIUSetup
    Appearance      = 0 'Flat
    BorderStyle     = 4 'Fixed ToolWindow
    Caption         = "CIU Setup"
    ClientHeight    = 2880
    ClientLeft      = 1410
    ClientTop       = 2490
    ClientWidth     = 6630
    BeginProperty Font
        Name        = "MS Sans Serif"
        Size        = 8.25
        Charset     = 0
    EndProperty

```

HNF-1569
Revision 1

```
Weight      = 700
Underline   = 0 'False
Italic      = 0 'False
Strikethrough = 0 'False
EndProperty
ForeColor   = &H80000008&
HelpContextID = 200
LinkTopic   = "Form1"
MaxButton   = 0 'False
PaletteMode = 1 'UseZOrder
ScaleHeight = 2880
ScaleWidth  = 6630
ShowInTaskbar = 0 'False
Tag         = "CIU Setup Form"
Visible     = 0 'False
Begin VB.Frame Frame1
    Caption   = "Communications Setup"
    Height    = 2052
    HelpContextID = 220
    Left      = 120
    TabIndex  = 3
    Top       = 120
    Width     = 6372
    Begin VB.TextBox txtTurnAround
        Height    = 285
        Left      = 5160
        TabIndex  = 15
        Tag       = "Turn Around"
        Text      = "900"
        ToolTipText = "Time CIU waits before checking buffer for gauge response"
        Top       = 1560
        Width     = 975
    End
    Begin VB.TextBox txtScanRate
        Height    = 285
        Left      = 5160
        TabIndex  = 14
        Tag       = "Scan Rate"
        Text      = "3"
        ToolTipText = "Interval between level scans"
        Top       = 480
        Width     = 975
    End
    Begin VB.TextBox txtGaugeAdd
        Height    = 285
        Left      = 2040
        TabIndex  = 11
        Tag       = "Gauge Address"
        Text      = "10"
        ToolTipText = "Address of Gauge, 0 thru 99"
        Top       = 1560
        Width     = 975
    End
    Begin VB.TextBox txtCIUAddress
        Height    = 285
        Left      = 2040
        TabIndex  = 10
        Tag       = "CIU Address"
        Text      = "1"
        ToolTipText = "Address of CIU, 0 thru 9"
        Top       = 1200
        Width     = 975
    End
    Begin VB.TextBox txtCOMPort
        Height    = 285
        Left      = 2040
        TabIndex  = 9
        Tag       = "Communications Port"
        Text      = "2"
        ToolTipText = "RS232C Connection Port"
        Top       = 840
    End
End
```


HNF-1569
Revision 1

```
Width          = 975
End
Begin VB.TextBox txtBaud
    Height      = 285
    Left        = 2040
    TabIndex    = 8
    Tag         = "Baud Rate"
    Text        = "2400"
    ToolTipText = "Transmission speed at gauge"
    Top         = 480
    Width       = 975
End
Begin VB.Label Label12
    Alignment    = 1 'Right Justify
    Caption      = "Turn Around (msec)"
    Height       = 252
    Left         = 3360
    TabIndex    = 13
    ToolTipText  = "Time CIU waits before checking buffer for gauge response"
    Top          = 1560
    Width        = 1692
End
Begin VB.Label Label11
    Alignment    = 1 'Right Justify
    Caption      = "Scan Rate (sec)"
    Height       = 252
    Left         = 3360
    TabIndex    = 12
    ToolTipText  = "Interval between level scans"
    Top          = 480
    Width        = 1692
End
Begin VB.Label Label10
    Appearance   = 0 'Flat
    Caption      = "Gauge Address (TA)"
    ForeColor    = &H80000008&
    Height       = 252
    Left         = 120
    TabIndex    = 7
    ToolTipText  = "Address of Gauge, 0 thru 99"
    Top          = 1560
    Width        = 1812
End
Begin VB.Label Label9
    Appearance   = 0 'Flat
    Caption      = "CIU Address"
    ForeColor    = &H80000008&
    Height       = 252
    Left         = 840
    TabIndex    = 6
    ToolTipText  = "Address of CIU, 0 thru 9"
    Top          = 1200
    Width        = 1092
End
Begin VB.Label Label4
    Appearance   = 0 'Flat
    Caption      = "Com Port"
    ForeColor    = &H80000008&
    Height       = 252
    Left         = 1080
    TabIndex    = 5
    ToolTipText  = "RS232C Connection Port"
    Top          = 840
    Width        = 852
End
Begin VB.Label Label3
    Appearance   = 0 'Flat
    Caption      = "Baud Rate"
    ForeColor    = &H80000008&
    Height       = 252
    Left         = 960
```

HNF-1569
Revision 1

```
        TabIndex      = 4
        ToolTipText    = "Transmission speed at gauge"
        Top            = 480
        Width          = 972
    End
End
Begin VB.CommandButton cmdCIUSetAdvanced
    Appearance      = 0 'Flat
    BackColor       = &H80000005&
    Caption         = "&Advanced"
    Height          = 495
    Left            = 4200
    TabIndex        = 2
    Top             = 2280
    Width           = 1335
End
Begin VB.CommandButton cmdCIUSetCancel
    Appearance      = 0 'Flat
    BackColor       = &H80000005&
    Caption         = "&Cancel"
    Height          = 495
    Left            = 2760
    TabIndex        = 0
    Top             = 2280
    Width           = 1095
End
Begin VB.CommandButton cmdCIUSetOK
    Appearance      = 0 'Flat
    BackColor       = &H80000005&
    Caption         = "&OK"
    Height          = 495
    Left            = 1080
    TabIndex        = 1
    Top             = 2280
    Width           = 1095
End
End
Attribute VB_Name = "frmCIUSetup"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
Private Sub cmdCIUSetAdvanced_Click()

    frmCOMMAAdv.Show 1

End Sub

Private Sub cmdCIUSetCancel_Click()
    frmCIUSetup.Hide
End Sub

Private Sub cmdCIUSetOK_Click()
    'Validate entries
    If CInt(Trim(txtGaugeAdd.text)) < 0 Or CInt(Trim(txtGaugeAdd.text)) > 99 Then
        MsgBox "Bad Gauge Address entered!"
        Exit Sub
    End If
    If Trim(txtBaud.text) <> "1200" Then
        If Trim(txtBaud.text) <> "2400" Then
            MsgBox "Bad Baud rate entered!"
            Exit Sub
        End If
    End If
    If CInt(Trim(txtCOMPort.text)) < 1 Or CInt(Trim(txtCOMPort.text)) > 4 Then
        MsgBox "Bad Comport entered!"
        Exit Sub
    End If
    If CDbl(Trim(txtScanRate.text)) < 0.5 Or CDbl(Trim(txtScanRate.text)) > 65 Then
        MsgBox "Bad Scan Rate entered!"
    End If
End Sub
```

HNF-1569
Revision 1

```
Exit Sub
End If
If CDb1(Trim(txtTurnAround.text)) <= 200 Or CDb1(Trim(txtTurnAround.text)) > 4000 Then
    MsgBox "Bad Turn Around entered!"
Exit Sub
End If
If gettingScans = 1 Then
    frmMain.cmdDisconnect_Click
End If

ciuclose

    ciuadres = Trim(UCASE(txtCIUAddress.text))
    ta = Trim(txtGaugeAdd.text)
    Baudrate = Trim(txtBaud.text)
    Compoort = Trim(txtCOMPort.text)
    scanRate = CDb1(Trim(txtScanRate.text)) * 1000
    turn_around = CDb1(Trim(txtTurnAround.text))
    frmCIUSetup.Hide

End Sub

Private Sub Form_Load()
txtCIUAddress.text = ciuadres
txtBaud.text = Baudrate
txtCOMPort.text = Compoort
txtGaugeAdd.text = ta
txtScanRate.text = Str(scanRate / 1000)
txtTurnAround.text = Str(turn_around)

End Sub

Private Sub Form_QueryUnload(Cancel As Integer, UnloadMode As Integer)
    Select Case UnloadMode
        Case vbFormControlMenu
            frmCIUSetup.Hide
            Cancel = True
        Case vbFormCode
            End
        Case vbAppWindows
            End
        Case vbAppTaskManager
            End
        Case vbFormMDIForm
            End
    End Select
End Sub

Private Sub Text1_Change()

End Sub

End FRMCIU.FRM
=====
Begin FRMCOMMA.FRM

VERSION 5.00
Begin VB.Form frmCOMMAAdv
    Appearance       = 0 'Flat
    BorderStyle      = 4 'Fixed ToolWindow
    Caption          = "Comm Setup - advanced"
    ClientHeight     = 2625
    ClientLeft       = 2880
    ClientTop        = 1410
    ClientWidth      = 2475
    BeginProperty Font
        Name          = "MS Sans Serif"
        Size          = 8.25
        Charset       = 0
    EndProperty
End Form
```

HNF-1569
Revision 1

```

Weight      = 700
Underline   = 0 'False
Italic      = 0 'False
Strikethrough = 0 'False
EndProperty
ForeColor   = &H00C0C0C0&
HelpContextID = 210
LinkTopic   = "Form1"
MaxButton   = 0 'False
PaletteMode = 1 'Use2Order
ScaleHeight = 2625
ScaleWidth  = 2475
ShowInTaskbar = 0 'False
Tag         = "Advanced CIU Setup"
Begin VB.Frame Frame1
    Caption   = "Comm Setup"
    Height    = 1932
    HelpContextID = 310
    Left      = 120
    TabIndex  = 2
    Top       = 120
    Width     = 2172
    Begin VB.TextBox txtCIUWait
        Height    = 288
        Left      = 1200
        TabIndex  = 10
        Text       = "0.5"
        ToolTipText = "Tenths of a second"
        Top       = 1440
        Width     = 612
    End
    Begin VB.TextBox txtStopBits
        Height    = 285
        Left      = 1200
        TabIndex  = 5
        Tag        = "Stop Bits"
        Text       = "1"
        Top       = 1080
        Width     = 615
    End
    Begin VB.TextBox txtDataBits
        Height    = 285
        Left      = 1200
        TabIndex  = 4
        Tag        = "Data Bits"
        Text       = "7"
        Top       = 720
        Width     = 615
    End
    Begin VB.TextBox txtParity
        Height    = 285
        Left      = 1200
        TabIndex  = 3
        Tag        = "Parity"
        Text       = "0"
        Top       = 360
        Width     = 615
    End
    End
    Begin VB.Label Label1
        Alignment  = 1 'Right Justify
        Caption    = "CIU Wait"
        Height     = 255
        Left       = 120
        TabIndex   = 9
        ToolTipText = "Time the program waits after receiving a CIU busy character"
        Top        = 1560
        Width      = 975
    End
    End
    Begin VB.Label Label6
        Alignment  = 1 'Right Justify
        Appearance = 0 'Flat

```

```

Caption      = "Stop Bits"
ForeColor    = &H80000008&
Height       = 255
Left         = 240
TabIndex     = 8
Top          = 1080
Width        = 855
WordWrap     = -1 'True
End
Begin VB.Label Label5
    Alignment  = 1 'Right Justify
    Appearance = 0 'Flat
    Caption    = "Data Bits"
    ForeColor  = &H80000008&
    Height     = 255
    Left       = 240
    TabIndex   = 7
    Top        = 720
    Width      = 855
    WordWrap   = -1 'True
End
Begin VB.Label Label4
    Alignment  = 1 'Right Justify
    Appearance = 0 'Flat
    Caption    = "Parity"
    ForeColor  = &H80000008&
    Height     = 255
    Left       = 480
    TabIndex   = 6
    Top        = 360
    Width      = 615
    WordWrap   = -1 'True
End
End
Begin VB.CommandButton cmdCOMMSetsCan
    Appearance = 0 'Flat
    BackColor  = &H80000005&
    Caption    = "&Cancel"
    Height     = 375
    Left       = 1320
    TabIndex   = 1
    Top        = 2160
    Width      = 975
End
Begin VB.CommandButton cmdCOMMSetsOK
    Appearance = 0 'Flat
    BackColor  = &H80000005&
    Caption    = "&OK"
    Height     = 375
    Left       = 120
    TabIndex   = 0
    Top        = 2160
    Width      = 975
End
End
Attribute VB_Name = "frmCOMMAAdv"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
Private Sub cmdCOMMSetsCan_Click()

    frmCOMMAAdv.Hide

End Sub

Private Sub cmdCOMMSetsOK_Click()

If gettingScans = 1 Then
    frmMain.cmdDisconnect_Click
    Delay (0.5)

```

```
ciuclose
End If

DataBits = txtDataBits.text
StopBits = txtStopBits.text
Parity = txtParity.text

busy_waittime = CDb1(txtCIUWait.text)

frmCOMMAAdv.Hide

End Sub

Private Sub Form_Load()
txtDataBits.text = DataBits
txtStopBits.text = StopBits
txtParity.text = Parity
txtCIUWait.text = busy_waittime
End Sub

Private Sub Form_QueryUnload(Cancel As Integer, UnloadMode As Integer)
Select Case UnloadMode
Case vbFormControlMenu
'frmCOMMAAdv.Hide
'Cancel = True
Cancel = False
Case vbFormCode
End
Case vbAppWindows
End
Case vbAppTaskManager
End
Case vbFormMDIForm
End
End Select
End Sub
```

End FRMCOMMA.FRM

```
=====
Begin frmDensity.frm

VERSION 5.00
Object = "{6B7E6392-850A-101B-AFC0-4210102A8DA7}#1.2#0"; "COMCTL32.OCX"
Object = "{FE0065C0-1B7B-11CF-9D53-00AA003C9CB6}#1.0#0"; "COMCT232.OCX"
Begin VB.Form frmDensity
BorderStyle = 3 'Fixed Dialog
Caption = "Get Density Data"
ClientHeight = 4380
ClientLeft = 3075
ClientTop = 5325
ClientWidth = 6885
HelpContextID = 300
LinkTopic = "Form1"
MaxButton = 0 'False
MinButton = 0 'False
PaletteMode = 1 'UseZOrder
ScaleHeight = 4380
ScaleWidth = 6885
ShowInTaskbar = 0 'False
Tag = "Density Profiles"
Begin VB.CommandButton Command2
Caption = "Command2"
Height = 195
Left = 960
TabIndex = 27
Top = 120
Visible = 0 'False
Width = 375
```

```

End
Begin VB.CommandButton cmdGetDensityFull
    Appearance      = 0 'Flat
    BackColor       = &H80000005&
    Caption         = "&Go"
    BeginProperty Font
        Name        = "MS Sans Serif"
        Size        = 8.25
        Charset     = 0
        Weight      = 700
        Underline   = 0 'False
        Italic      = 0 'False
        Strikethrough = 0 'False
    EndProperty
    Height          = 255
    HelpContextID   = 340
    Left            = 120
    TabIndex       = 26
    ToolTipText     = "Starts a WRSS IP Scan"
    Top             = 3480
    Visible        = 0 'False
    Width          = 732
End
Begin VB.CommandButton Command1
    Caption         = "Command1"
    Enabled        = 0 'False
    Height         = 192
    Left           = 120
    TabIndex      = 23
    Top            = 120
    Visible       = 0 'False
    Width         = 732
End
Begin VB.CommandButton cmdUN
    Caption         = "&UN"
    BeginProperty Font
        Name        = "Arial Narrow"
        Size        = 8.25
        Charset     = 0
        Weight      = 400
        Underline   = 0 'False
        Italic      = 0 'False
        Strikethrough = 0 'False
    EndProperty
    Height          = 252
    Left            = 960
    TabIndex       = 21
    ToolTipText     = "Manually Unlock Displacer"
    Top             = 2880
    Width          = 372
End
Begin VB.CommandButton cmdFR
    Caption         = "&FR"
    BeginProperty Font
        Name        = "Arial Narrow"
        Size        = 8.25
        Charset     = 0
        Weight      = 400
        Underline   = 0 'False
        Italic      = 0 'False
        Strikethrough = 0 'False
    EndProperty
    Height          = 252
    Left            = 480
    TabIndex       = 20
    ToolTipText     = "Manually Halt Displacer"
    Top             = 2880
    Width          = 495
End
Begin VB.CommandButton cmdCA
    Caption         = "C&A"

```

```

BeginProperty Font
    Name      = "Arial Narrow"
    Size      = 8.25
    Charset   = 0
    Weight    = 400
    Underline  = 0 'False
    Italic    = 0 'False
    Strikethrough = 0 'False
EndProperty
Height      = 252
Left       = 120
TabIndex   = 19
ToolTipText = "Manually Raise Displacer"
Top        = 2880
Width      = 372
End
Begin VB.CommandButton cmdPrintLog
    Caption      = "Print Log"
    Height       = 252
    Left        = 4440
    TabIndex     = 18
    Top         = 3480
    Width       = 1332
End
Begin ComctlLib.ProgressBar pBar
    Height       = 252
    Left        = 4200
    TabIndex     = 12
    Top         = 3840
    Visible      = 0 'False
    Width       = 2532
    _ExtentX    = 4471
    _ExtentY    = 450
    _Version    = 327682
    Appearance   = 1
End
Begin VB.TextBox txtStatus
    BackColor    = &H0080FFFF&
    Height       = 288
    Left        = 120
    TabIndex     = 11
    ToolTipText  = "Status window"
    Top         = 3840
    Width       = 3972
End
Begin VB.Frame Frame1
    Caption      = "Output Log"
    BeginProperty Font
        Name      = "MS Sans Serif"
        Size      = 8.25
        Charset   = 0
        Weight    = 700
        Underline  = 0 'False
        Italic    = 0 'False
        Strikethrough = 0 'False
    EndProperty
    Height      = 3252
    Left       = 3480
    TabIndex   = 8
    Tag        = "Density Output"
    Top        = 120
    Width      = 3252
    Begin VB.TextBox txtData
        BeginProperty Font
            Name      = "Courier New"
            Size      = 7.5
            Charset   = 0
            Weight    = 400
            Underline  = 0 'False
            Italic    = 0 'False
            Strikethrough = 0 'False

```


HNF-1569
Revision 1

```

EndProperty
Height      = 2892
HelpContextID = 360
Left       = 120
Multiline  = -1 'True
ScrollBars = 3 'Both
TabIndex   = 9
ToolTipText = "Record of CIU/Gauge communications"
Top        = 240
Width      = 3012
End
End
Begin VB.CommandButton cmdClose
Caption      = "Close"
Height      = 252
Left        = 2640
TabIndex    = 7
ToolTipText  = "Closes this window"
Top         = 3480
Width       = 732
End
Begin VB.CommandButton cmdAbortDensity
Appearance   = 0 'Flat
BackColor    = &H80000005&
Caption      = "&Abort"
Enabled      = 0 'False
BeginProperty Font
Name         = "MS Sans Serif"
Size        = 8.25
Charset      = 0
Weight       = 700
Underline    = 0 'False
Italic       = 0 'False
Strikethrough = 0 'False
EndProperty
Height      = 252
HelpContextID = 350
Left        = 1320
TabIndex    = 6
ToolTipText  = "Stops the current activity & Freezes displacer"
Top         = 3480
Width       = 852
End
Begin VB.CommandButton cmdGetDensity
Appearance   = 0 'Flat
BackColor    = &H80000005&
Caption      = "&Go"
BeginProperty Font
Name         = "MS Sans Serif"
Size        = 8.25
Charset      = 0
Weight       = 700
Underline    = 0 'False
Italic       = 0 'False
Strikethrough = 0 'False
EndProperty
Height      = 255
HelpContextID = 340
Left        = 120
TabIndex    = 5
ToolTipText  = "Starts an IP Scan"
Top         = 3480
Width       = 732
End
Begin VB.Frame Frame2
Caption      = "Interface Profile"
BeginProperty Font
Name         = "MS Sans Serif"
Size        = 8.25
Charset      = 0
Weight       = 700

```

HNF-1569
Revision 1

```

        Underline      = 0  'False
        Italic         = 0  'False
        Strikethrough  = 0  'False
EndProperty
Height      = 3252
Left       = 1440
TabIndex   = 0
Tag        = "Density Input"
Top        = 120
Width      = 1932
Begin ComCtl2.UpDown UpDown3
    Height      = 288
    Left       = 1453
    TabIndex    = 30
    Top        = 1680
    Width      = 240
    _ExtentX    = 423
    _ExtentY    = 503
    _Version    = 327680
    BuddyControl = "txtInterval"
    BuddyDispid = 196631
    OrigLeft    = 1440
    OrigTop     = 1680
    OrigRight   = 1680
    OrigBottom  = 1935
    SyncBuddy   = -1 'True
    BuddyProperty = 65547
    Enabled     = -1 'True
End
Begin ComCtl2.UpDown UpDown2
    Height      = 288
    Left       = 1453
    TabIndex    = 29
    Top        = 1080
    Width      = 240
    _ExtentX    = 423
    _ExtentY    = 503
    _Version    = 327680
    BuddyControl = "txtSed"
    BuddyDispid = 196632
    OrigLeft    = 1440
    OrigTop     = 1080
    OrigRight   = 1680
    OrigBottom  = 1335
    Max         = 1000
    SyncBuddy   = -1 'True
    BuddyProperty = 65547
    Enabled     = -1 'True
End
Begin ComCtl2.UpDown UpDown1
    Height      = 285
    Left       = 1440
    TabIndex    = 28
    Top        = 480
    Width      = 240
    _ExtentX    = 423
    _ExtentY    = 503
    _Version    = 327680
    BuddyControl = "txtDensityStart"
    BuddyDispid = 196633
    OrigLeft    = 1440
    OrigTop     = 480
    OrigRight   = 1680
    OrigBottom  = 735
    Max         = 1000
    SyncBuddy   = -1 'True
    BuddyProperty = 65547
    Enabled     = -1 'True
End
Begin VB.CheckBox chkSP
    Caption      = "Single Point"

```

HNF-1569
Revision 1

```
BeginProperty Font
  Name      = "MS Sans Serif"
  Size      = 8.25
  Charset   = 0
  Weight    = 700
  Underline = 0 'False
  Italic    = 0 'False
  Strikethrough = 0 'False
EndProperty
Height      = 252
Left        = 360
TabIndex    = 22
ToolTipText = "Check for one measurement at the start level only"
Top         = 2280
Width       = 1452
End
Begin VB.CommandButton cmdGetSedimentLevel
  Caption      = "Get"
  Height       = 252
  Left         = 240
  TabIndex     = 17
  Top          = 1080
  Width        = 492
End
Begin VB.OptionButton optDown
  Caption      = "Down"
  Height       = 252
  Left         = 960
  TabIndex     = 15
  Top          = 2880
  Value        = -1 'True
  Width        = 732
End
Begin VB.OptionButton optUp
  Caption      = "Up"
  Height       = 252
  Left         = 240
  TabIndex     = 14
  Top          = 2880
  Width        = 615
End
Begin VB.TextBox txtInterval
  Alignment    = 1 'Right Justify
  BackColor    = &H00C0C0C0&
  Height       = 288
  HelpContextID = 330
  Left         = 840
  TabIndex     = 4
  Text         = "12"
  ToolTipText  = "Distance between measurements."
  Top          = 1680
  Width        = 612
End
Begin VB.TextBox txtSed
  Alignment    = 1 'Right Justify
  BackColor    = &H00C0C0C0&
  Height       = 288
  HelpContextID = 320
  Left         = 840
  TabIndex     = 3
  Text         = "1"
  ToolTipText  = "Enter Sediment or Bottom of Tank Level"
  Top          = 1080
  Width        = 612
End
Begin VB.TextBox txtDensityStart
  Alignment    = 1 'Right Justify
  BackColor    = &H00C0C0C0&
  Height       = 285
  HelpContextID = 310
  Left         = 840
```

HNF-1569
Revision 1

```

    TabIndex      = 2
    Text          = "156.00"
    ToolTipText   = "Must enter a Start Level"
    Top           = 480
    Width         = 615
End
Begin VB.Line Line2
    BorderColor   = &H80000014&
    X1            = 120
    X2            = 1800
    Y1            = 2160
    Y2            = 2160
End
Begin VB.Line Line1
    X1            = 120
    X2            = 1800
    Y1            = 2160
    Y2            = 2160
End
Begin VB.Label Label3
    Caption       = "Scan Direction"
    BeginProperty Font
        Name      = "MS Sans Serif"
        Size      = 8.25
        Charset   = 0
        Weight    = 700
        Underline  = 0 'False
        Italic     = 0 'False
        Strikethrough = 0 'False
    EndProperty
    Height       = 252
    Left         = 120
    TabIndex     = 16
    Top          = 2640
    Width        = 1332
End
Begin VB.Label Label8
    Caption       = "Sediment Level (in)"
    BeginProperty Font
        Name      = "MS Sans Serif"
        Size      = 8.25
        Charset   = 0
        Weight    = 700
        Underline  = 0 'False
        Italic     = 0 'False
        Strikethrough = 0 'False
    EndProperty
    Height       = 252
    Left         = 120
    TabIndex     = 13
    Top          = 840
    Width        = 1692
End
Begin VB.Label Label1
    Caption       = "Interval (inches)"
    BeginProperty Font
        Name      = "MS Sans Serif"
        Size      = 8.25
        Charset   = 0
        Weight    = 700
        Underline  = 0 'False
        Italic     = 0 'False
        Strikethrough = 0 'False
    EndProperty
    Height       = 252
    Left         = 120
    TabIndex     = 10
    Top          = 1440
    Width        = 1452
End
Begin VB.Label Label2
```

HNF-1569
Revision 1

```

Appearance      = 0 'Flat
Caption         = "Start Level (inches)"
BeginProperty Font
    Name        = "MS Sans Serif"
    Size        = 8.25
    Charset     = 0
    Weight      = 700
    Underline   = 0 'False
    Italic      = 0 'False
    Strikethrough = 0 'False
EndProperty
ForeColor       = &H80000008&
Height         = 252
Left           = 120
TabIndex       = 1
Top            = 240
Width          = 1692
End
End
Begin VB.CommandButton cmdTankProfile
    Appearance    = 0 'Flat
    BackColor     = &H80000005&
    Caption       = "&Go"
    Enabled       = 0 'False
    BeginProperty Font
        Name      = "MS Sans Serif"
        Size      = 8.25
        Charset   = 0
        Weight    = 700
        Underline = 0 'False
        Italic    = 0 'False
        Strikethrough = 0 'False
    EndProperty
    Height        = 255
    HelpContextID = 340
    Left         = 120
    TabIndex     = 25
    ToolTipText   = "Starts a TP Scan"
    Top          = 3480
    Visible      = 0 'False
    Width        = 732
End
Begin VB.CommandButton cmdGetDensity2
    Appearance    = 0 'Flat
    BackColor     = &H80000005&
    Caption       = "&Go"
    BeginProperty Font
        Name      = "MS Sans Serif"
        Size      = 8.25
        Charset   = 0
        Weight    = 700
        Underline = 0 'False
        Italic    = 0 'False
        Strikethrough = 0 'False
    EndProperty
    Height        = 255
    HelpContextID = 340
    Left         = 120
    TabIndex     = 24
    ToolTipText   = "Starts a Program Driven Scan"
    Top          = 3480
    Visible      = 0 'False
    Width        = 732
End
Begin VB.Image picMtr
    Height        = 2172
    Index         = 58
    Left         = 120
    Picture       = "frmDensity.frx":0000
    Stretch       = -1 'True
    Top          = 360

```

HNF-1569
Revision 1

```
Width          = 1212
End
Begin VB.Image picMtr
    Height      = 2172
    Index       = 57
    Left        = 120
    Picture     = "frmDensity.frx":D58E
    Stretch     = -1 'True
    Top         = 360
    Width       = 1212
End
Begin VB.Image picMtr
    Height      = 2172
    Index       = 56
    Left        = 120
    Picture     = "frmDensity.frx":1AB1C
    Stretch     = -1 'True
    Top         = 360
    Width       = 1212
End
Begin VB.Image picMtr
    Height      = 2172
    Index       = 55
    Left        = 120
    Picture     = "frmDensity.frx":280AA
    Stretch     = -1 'True
    Top         = 360
    Width       = 1212
End
Begin VB.Image picMtr
    Height      = 2172
    Index       = 54
    Left        = 120
    Picture     = "frmDensity.frx":35638
    Stretch     = -1 'True
    Top         = 360
    Width       = 1212
End
Begin VB.Image picMtr
    Height      = 2172
    Index       = 53
    Left        = 120
    Picture     = "frmDensity.frx":42BC6
    Stretch     = -1 'True
    Top         = 360
    Width       = 1212
End
Begin VB.Image picMtr
    Height      = 2172
    Index       = 52
    Left        = 120
    Picture     = "frmDensity.frx":50154
    Stretch     = -1 'True
    Top         = 360
    Width       = 1212
End
Begin VB.Image picMtr
    Height      = 2172
    Index       = 51
    Left        = 120
    Picture     = "frmDensity.frx":5D6E2
    Stretch     = -1 'True
    Top         = 360
    Width       = 1212
End
Begin VB.Image picMtr
    Height      = 2172
    Index       = 50
    Left        = 120
    Picture     = "frmDensity.frx":6AC70
    Stretch     = -1 'True
```

HNF-1569
Revision 1

```
Top          = 360
Width        = 1212
End
Begin VB.Image picMtr
    Height    = 2172
    Index     = 49
    Left      = 120
    Picture    = "frmDensity.frx":781FE
    Stretch    = -1 'True
    Top       = 360
    Width     = 1212
End
Begin VB.Image picMtr
    Height    = 2172
    Index     = 48
    Left      = 120
    Picture    = "frmDensity.frx":8578C
    Stretch    = -1 'True
    Top       = 360
    Width     = 1212
End
Begin VB.Image picMtr
    Height    = 2172
    Index     = 47
    Left      = 120
    Picture    = "frmDensity.frx":92D1A
    Stretch    = -1 'True
    Top       = 360
    Width     = 1212
End
Begin VB.Image picMtr
    Height    = 2172
    Index     = 46
    Left      = 120
    Picture    = "frmDensity.frx":A02A8
    Stretch    = -1 'True
    Top       = 360
    Width     = 1212
End
Begin VB.Image picMtr
    Height    = 2172
    Index     = 45
    Left      = 120
    Picture    = "frmDensity.frx":AD836
    Stretch    = -1 'True
    Top       = 360
    Width     = 1212
End
Begin VB.Image picMtr
    Height    = 2172
    Index     = 44
    Left      = 120
    Picture    = "frmDensity.frx":BADC4
    Stretch    = -1 'True
    Top       = 360
    Width     = 1212
End
Begin VB.Image picMtr
    Height    = 2172
    Index     = 43
    Left      = 120
    Picture    = "frmDensity.frx":C8352
    Stretch    = -1 'True
    Top       = 360
    Width     = 1212
End
Begin VB.Image picMtr
    Height    = 2172
    Index     = 42
    Left      = 120
    Picture    = "frmDensity.frx":D58E0
```

```
Stretch      = -1 'True
Top          = 360
Width        = 1212
End
Begin VB.Image picMtr
Height       = 2172
Index        = 41
Left         = 120
Picture      = "frmDensity.frx":E2E6E
Stretch      = -1 'True
Top          = 360
Width        = 1212
End
Begin VB.Image picMtr
Height       = 2172
Index        = 40
Left         = 120
Picture      = "frmDensity.frx":F03FC
Stretch      = -1 'True
Top          = 360
Width        = 1212
End
Begin VB.Image picMtr
Height       = 2172
Index        = 39
Left         = 120
Picture      = "frmDensity.frx":FD98A
Stretch      = -1 'True
Top          = 360
Width        = 1212
End
Begin VB.Image picMtr
Height       = 2172
Index        = 38
Left         = 120
Picture      = "frmDensity.frx":10AF18
Stretch      = -1 'True
Top          = 360
Width        = 1212
End
Begin VB.Image picMtr
Height       = 2172
Index        = 37
Left         = 120
Picture      = "frmDensity.frx":1184A6
Stretch      = -1 'True
Top          = 360
Width        = 1212
End
Begin VB.Image picMtr
Height       = 2172
Index        = 36
Left         = 120
Picture      = "frmDensity.frx":125A34
Stretch      = -1 'True
Top          = 360
Width        = 1212
End
Begin VB.Image picMtr
Height       = 2172
Index        = 35
Left         = 120
Picture      = "frmDensity.frx":132FC2
Stretch      = -1 'True
Top          = 360
Width        = 1212
End
Begin VB.Image picMtr
Height       = 2172
Index        = 34
Left         = 120
```


HNF-1569
Revision 1

```
Picture      = "frmDensity.frx":140550
Stretch      = -1 'True
Top          = 360
Width        = 1212
End
Begin VB.Image picMtr
Height       = 2172
Index        = 33
Left         = 120
Picture      = "frmDensity.frx":14DADE
Stretch      = -1 'True
Top          = 360
Width        = 1212
End
Begin VB.Image picMtr
Height       = 2172
Index        = 32
Left         = 120
Picture      = "frmDensity.frx":15B06C
Stretch      = -1 'True
Top          = 360
Width        = 1212
End
Begin VB.Image picMtr
Height       = 2172
Index        = 31
Left         = 120
Picture      = "frmDensity.frx":1685FA
Stretch      = -1 'True
Top          = 360
Width        = 1212
End
Begin VB.Image picMtr
Height       = 2172
Index        = 30
Left         = 120
Picture      = "frmDensity.frx":175B88
Stretch      = -1 'True
Top          = 360
Width        = 1212
End
Begin VB.Image picMtr
Height       = 2172
Index        = 29
Left         = 120
Picture      = "frmDensity.frx":183116
Stretch      = -1 'True
Top          = 360
Width        = 1212
End
Begin VB.Image picMtr
Height       = 2172
Index        = 28
Left         = 120
Picture      = "frmDensity.frx":1906A4
Stretch      = -1 'True
Top          = 360
Width        = 1212
End
Begin VB.Image picMtr
Height       = 2172
Index        = 27
Left         = 120
Picture      = "frmDensity.frx":19DC32
Stretch      = -1 'True
Top          = 360
Width        = 1212
End
Begin VB.Image picMtr
Height       = 2172
Index        = 26
```

HNF-1569
Revision 1

```

Left          = 120
Picture       = "frmDensity.frx":1AB1C0
Stretch      = -1 'True
Top          = 360
Width        = 1212
End
Begin VB.Image picMtr
    Height     = 2172
    Index      = 25
    Left       = 120
    Picture    = "frmDensity.frx":1B874E
    Stretch    = -1 'True
    Top        = 360
    Width      = 1212
End
Begin VB.Image picMtr
    Height     = 2172
    Index      = 24
    Left       = 120
    Picture    = "frmDensity.frx":1C5CDC
    Stretch    = -1 'True
    Top        = 360
    Width      = 1212
End
Begin VB.Image picMtr
    Height     = 2172
    Index      = 23
    Left       = 120
    Picture    = "frmDensity.frx":1D326A
    Stretch    = -1 'True
    Top        = 360
    Width      = 1212
End
Begin VB.Image picMtr
    Height     = 2172
    Index      = 22
    Left       = 120
    Picture    = "frmDensity.frx":1E07F8
    Stretch    = -1 'True
    Top        = 360
    Width      = 1212
End
Begin VB.Image picMtr
    Height     = 2172
    Index      = 21
    Left       = 120
    Picture    = "frmDensity.frx":1EDD86
    Stretch    = -1 'True
    Top        = 360
    Width      = 1212
End
Begin VB.Image picMtr
    Height     = 2172
    Index      = 20
    Left       = 120
    Picture    = "frmDensity.frx":1FB314
    Stretch    = -1 'True
    Top        = 360
    Width      = 1212
End
Begin VB.Image picMtr
    Height     = 2172
    Index      = 19
    Left       = 120
    Picture    = "frmDensity.frx":2088A2
    Stretch    = -1 'True
    Top        = 360
    Width      = 1212
End
Begin VB.Image picMtr
    Height     = 2172

```

HNF-1569
Revision 1

```
Index      = 18
Left       = 120
Picture    = "frmDensity.frx":215E30
Stretch    = -1 'True
Top        = 360
Width      = 1212
End
Begin VB.Image picMtr
    Height   = 2172
    Index    = 17
    Left     = 120
    Picture  = "frmDensity.frx":2233BE
    Stretch  = -1 'True
    Top      = 360
    Width    = 1212
End
Begin VB.Image picMtr
    Height   = 2172
    Index    = 16
    Left     = 120
    Picture  = "frmDensity.frx":23094C
    Stretch  = -1 'True
    Top      = 360
    Width    = 1212
End
Begin VB.Image picMtr
    Height   = 2172
    Index    = 15
    Left     = 120
    Picture  = "frmDensity.frx":23DEDA
    Stretch  = -1 'True
    Top      = 360
    Width    = 1212
End
Begin VB.Image picMtr
    Height   = 2172
    Index    = 14
    Left     = 120
    Picture  = "frmDensity.frx":24B468
    Stretch  = -1 'True
    Top      = 360
    Width    = 1212
End
Begin VB.Image picMtr
    Height   = 2172
    Index    = 13
    Left     = 120
    Picture  = "frmDensity.frx":2589F6
    Stretch  = -1 'True
    Top      = 360
    Width    = 1212
End
Begin VB.Image picMtr
    Height   = 2172
    Index    = 12
    Left     = 120
    Picture  = "frmDensity.frx":265F84
    Stretch  = -1 'True
    Top      = 360
    Width    = 1212
End
Begin VB.Image picMtr
    Height   = 2172
    Index    = 11
    Left     = 120
    Picture  = "frmDensity.frx":273512
    Stretch  = -1 'True
    Top      = 360
    Width    = 1212
End
Begin VB.Image picMtr
```

HNF-1569
Revision 1

```
Height      = 2172
Index       = 10
Left        = 120
Picture     = "frmDensity.frx":280AA0
Stretch     = -1 'True
Top         = 360
Width       = 1212
End
Begin VB.Image picMtr
Height      = 2172
Index       = 9
Left        = 120
Picture     = "frmDensity.frx":28E02E
Stretch     = -1 'True
Top         = 360
Width       = 1212
End
Begin VB.Image picMtr
Height      = 2172
Index       = 8
Left        = 120
Picture     = "frmDensity.frx":29B5BC
Stretch     = -1 'True
Top         = 360
Width       = 1212
End
Begin VB.Image picMtr
Height      = 2172
Index       = 7
Left        = 120
Picture     = "frmDensity.frx":2A8B4A
Stretch     = -1 'True
Top         = 360
Width       = 1212
End
Begin VB.Image picMtr
Height      = 2172
Index       = 6
Left        = 120
Picture     = "frmDensity.frx":2B60D8
Stretch     = -1 'True
Top         = 360
Width       = 1212
End
Begin VB.Image picMtr
Height      = 2172
Index       = 5
Left        = 120
Picture     = "frmDensity.frx":2C3666
Stretch     = -1 'True
Top         = 360
Width       = 1212
End
Begin VB.Image picMtr
Height      = 2172
Index       = 4
Left        = 120
Picture     = "frmDensity.frx":2D0BF4
Stretch     = -1 'True
Top         = 360
Width       = 1212
End
Begin VB.Image picMtr
Height      = 2172
Index       = 3
Left        = 120
Picture     = "frmDensity.frx":2DE182
Stretch     = -1 'True
Top         = 360
Width       = 1212
End
```

```

Begin VB.Image picMtr
    Height      = 2172
    Index       = 2
    Left        = 120
    Picture     = "frmDensity.frx":2EB710
    Stretch     = -1 'True
    Top         = 360
    Width       = 1212
End
Begin VB.Image picMtr
    Height      = 2172
    Index       = 1
    Left        = 120
    Picture     = "frmDensity.frx":2F8C9E
    Stretch     = -1 'True
    Top         = 360
    Width       = 1212
End
Begin VB.Image picMtr
    Height      = 2172
    Index       = 0
    Left        = 120
    Picture     = "frmDensity.frx":30622C
    Stretch     = -1 'True
    Top         = 360
    Width       = 1212
End
Begin VB.Menu mnuSelect
    Caption     = "&Select Method"
    Begin VB.Menu mnuSelectIP
        Caption  = "IP Type Scan"
        Checked  = -1 'True
    End
    Begin VB.Menu mnuSelectProgram
        Caption  = "&Program Scan"
    End
    Begin VB.Menu mnuSelectTP
        Caption  = "&TP Type Scan"
        Enabled  = 0 'False
        Visible  = 0 'False
    End
    Begin VB.Menu mnuSelectFull
        Caption  = "&WRSS Full Auto"
    End
End
Begin VB.Menu mnuOptions
    Caption     = "&Options"
    Begin VB.Menu mnuOptionsIPsed
        Caption  = "&Scan to Sed Level"
        Checked  = -1 'True
    End
    Begin VB.Menu mnuOptionsIPBottom
        Caption  = "Scan to &Tank Bottom"
    End
End
End
Attribute VB_Name = "frmDensity"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
Option Explicit
Dim OldStart$
Dim OldSed$
Dim OldInterval$
Dim DisabledLevelScan%
Dim GettingSediment%
Dim GettingDensities%

```

```
Private Sub chkSP_Click()  
Debug.Print "chkSP.value= "; chkSP.Value  
If chkSP.Value = 1 Then 'checked  
    Debug.Print "Must be checked..."  
    Interval = txtInterval.Text  
    txtInterval.Text = ""  
Else 'if unchecked  
    Debug.Print "Must be unchecked..."  
    txtInterval.Text = Interval  
End If  
End Sub  
  
Private Sub cmdAbortDensity_Click()  
    Dim echo$  
    Dim message As String  
    Dim Reply As Integer  
  
    message = "Warning: Are you sure "  
    message = message + "you want to ABORT?"  
  
    Reply = MsgBox(message, vbYesNo + vbExclamation, "Abort???" )  
    If Reply = 6 Then  
        'Abort Density Measurement.  
        AbortDensity = 1  
        ciucom "ZFR", echo, status  
        cmdClose.Enabled = True  
        Exit Sub  
    End If  
End Sub  
  
Private Sub cmdCA_Click()  
    ' Sends a CA to the gauge  
  
    Dim m$, R$, echo$  
    If AbortDensity = 1 Then  
        ciucom "ZCA", echo, status  
        txtData.Text = txtData.Text & vbNewLine & "Manual command CA Sent..." & vbNewLine  
        txtStatus.Text = "Sending CA command to gauge..."  
    ElseIf GettingDensities = 1 Then  
        m = "Warning! This will abort density measurement." & vbNewLine  
        m = m & "Do you wish to abort?"  
        R$ = MsgBox(m, vbYesNo + vbExclamation, "Abort?" )  
        If R = 6 Then  
            AbortDensity = 1  
            ciucom "ZCA", echo, status  
            DisabledLevelScan = 1  
            txtData.Text = txtData.Text & vbNewLine & "Manual command CA Sent..." & vbNewLine  
            txtStatus.Text = "Sending CA command to gauge..."  
        End If  
    ElseIf GettingSediment = 1 Then  
        m = "Warning! This will abort sediment check." & vbNewLine  
        m = m & "Do you wish to abort?"  
        R = MsgBox(m, vbYesNo + vbExclamation, "Abort?" )  
        If R = 6 Then  
            AbortDensity = 1  
            ciucom "ZCA", echo, status  
            DisabledLevelScan = 1  
            txtData.Text = txtData.Text & vbNewLine & "Manual command CA Sent..." & vbNewLine  
            txtStatus.Text = "Sending CA command to gauge..."  
        End If  
    ElseIf GettingDensities = 0 Then  
        ciucom "ZCA", echo, status  
        txtStatus.Text = "Sending CA command to gauge..."  
        Delay (3)  
        txtStatus.Text = ""  
    ElseIf GettingSediment = 0 Then  
        ciucom "ZCA", echo, status  
        txtStatus.Text = "Sending CA command to gauge..."  
        Delay (3)  
        txtStatus.Text = ""  
    End If  
End Sub
```

```

End If
End Sub

Private Sub cmdClose_Click()
    frmDensity.Hide
    txtStatus.Text = ""
    pBar.Visible = False
End Sub

Private Sub cmdFR_Click()
    ' Sends and FR command to the gauge
    Dim m$, R%, echo$
    If AbortDensity = 1 Then
        ciucom "ZFR", echo, status
        txtData.Text = txtData.Text & vbNewLine & "Manual command FR Sent..." & vbNewLine
        txtStatus.Text = "Sending FR command to gauge..."
    ElseIf GettingDensities = 1 Then
        m = "Warning! This will abort density measurement." & vbNewLine
        m = m & "Do you wish to abort?"
        R% = MsgBox(m, vbYesNo + vbExclamation, "Abort?")
        If R = 6 Then
            AbortDensity = 1
            ciucom "ZFR", echo, status
            DisabledLevelScan = 1
            txtData.Text = txtData.Text & vbNewLine & "Manual command FR Sent..." & vbNewLine
            txtStatus.Text = "Sending FR command to gauge..."
        End If
    ElseIf GettingSediment = 1 Then
        m = "Warning! This will abort sediment check." & vbNewLine
        m = m & "Do you wish to abort?"
        R = MsgBox(m, vbYesNo + vbExclamation, "Abort?")
        If R = 6 Then
            AbortDensity = 1
            ciucom "ZFR", echo, status
            DisabledLevelScan = 1
            txtData.Text = txtData.Text & vbNewLine & "Manual command FR Sent..." & vbNewLine
            txtStatus.Text = "Sending FR command to gauge..."
        End If
    ElseIf GettingDensities = 0 Then
        ciucom "ZFR", echo, status
        txtStatus.Text = "Sending FR command to gauge..."
        Delay (3)
        txtStatus.Text = ""
    ElseIf GettingSediment = 0 Then
        ciucom "ZFR", echo, status
        txtStatus.Text = "Sending FR command to gauge..."
        Delay (3)
        txtStatus.Text = ""
    End If
End Sub

Private Sub cmdGetDensity_Click()
    *****
    ' This procedure obtains a density profile of a tank using
    ' the IP command.
    '
    ' INPUTS:  txtDensityStart.text (Start Level); txtSed.text
    '           (Sediment Level); chkSP.value (single-point boolean);
    '           LastSedLevel (Last valid sediment level); txtInterval.text
    '           (Interval)
    '           mnuOptionsIPSED.value | mnuOptionsIPBottom.value
    ' OUTPUTS: Density() array; Level() array; SpG() array; .LOG file;
    '           .TXT file; .DAT file; ACTIVITY.LOG file
    ' CONDITIONS: Start Level must be > 12 inches above sediment level;
    '               If optUP.value = true then Start Level must be
    '               greater than 6 inches below the surface; Interval
    '               must be greater than 6 inches.
    '
    ' The routine will check for low wire tension errors in case the
    ' displacer is sent into the sediment.
    *****

```

HNF-1569
Revision 1

```
On Error Resume Next

If BalanceTestPassed = False Then
    MsgBox "Please perform a balance test before continuing..."
    Exit Sub
End If

Dim i%, j%
Dim echo As String           'response from CIU/gauge
Dim answer As String         'result from decode
Dim echoString(1 To 50) As String 'array of CIU/gauge responses
Dim recordCount As Integer   'count of echoStrings
Dim message As String        'used for msgboxes
Dim Reply As Integer         'used to capture msgbox responses
Dim fileIncrement As Integer
Dim Level(0 To 200) As String 'holds levels at which densities were obtained
Dim Density(0 To 200) As String 'hold densities
Dim sd As String             'scan direction

Dim rectype As String        'holds a CIU request string
Dim CR                        'carriage return for text
CR = Chr(13) + Chr(10)
'dim CL As Double            'Current Level
Dim oldlevel As Double
Dim theDate(0 To 200) As String
Dim theTime(0 To 200) As String
Dim AvgDensity As String
Dim avgden As Double
Dim SpG(1 To 200) As Double
Dim LevelDim$, m$
Dim GotADensity%             'Flag for marking data acquisition
Dim StartTime As String
txtStatus.Text = "...Stand by..."
avgden = 0
AbortDensity = 0
DisabledLevelScan = 0
GotADensity = 0

'Clear any levels and densities
For i = 0 To 200
    Density(i) = "0"
    Level(i) = "0"
Next i

'clear status window
txtStatus.Text = ""

'Set form components
cmdGetDensity.MousePointer = 11
cmdGetDensity.Enabled = False
cmdClose.Enabled = False
cmdAbortDensity.Enabled = True
cmdClose.MousePointer = 11
txtData.Text = ""
frmMain.cmdRecover.Enabled = False
frmMain.cmdGetCurrentLevel.Enabled = False
frmMain.cmdOpEnraf.Enabled = False
frmMain.cmdDensity.Enabled = False
frmMain.cmdLog.Enabled = False
frmMain.cmdFTF.Enabled = False
frmMain.mnuAction.Enabled = False
frmMain.mnuConnect.Enabled = False
frmMain.mnuSetup.Enabled = False

'Ensure level scanning is disabled, set a flag to turn it back on when complete.
If frmMain.tmrLevel.Enabled = True Then
    frmMain.cmdDisconnect_Click
    DisabledLevelScan = 1
End If
```


HNF-1569
Revision 1

```
'Set Flag to tell other program functions we're busy.
GettingDensities = 1

' Warn the operator. (Dipping the displacer in high level
' radioactive waste is no small thing).
m = "Warning! You are about to DIP the displacer." + CR
m = m + "Do you want to continue?"
i = MsgBox(m, vbYesNo + vbExclamation, "DIP Displacer?")
If i <> 6 Then
    AbortDensity = 1
    GoSub check_abort
End If

'=====
'VALIDATE INPUT DATA
'=====
txtStatus.Text = "Validating Input Data...."
If txtDensityStart.Text = "" Then
    MsgBox "You must enter a start level!", vbOKOnly + vbCritical, "Need Start Level!"
    AbortDensity = 1
    GoSub check_abort
Else
    SDLevel = txtDensityStart.Text
End If

If txtSed.Text = "" Then
    message = "You must enter a sediment level, or click GET" + CR
    message = message + "to have the gauge obtain it for you..."
    MsgBox message, vbOKOnly + vbCritical, "Need a Sediment Level..."
    AbortDensity = 1
    GoSub check_abort
Else
    SedLev = txtSed.Text
End If

If txtInterval.Text = "" And chkSP.Value = Unchecked Then
    txtInterval.Text = "12"
    Interval = "12"
End If

If chkSP.Value = Unchecked Then
    If Cint(txtInterval.Text) < 6 Then
        m = "You must have an interval greater than 6-inches." + CR
        m = "Please start again."
        MsgBox m, vbOKOnly + vbCritical, "Interval too small!"
        AbortDensity = 1
        GoSub check_abort
    Else
        Interval = txtInterval.Text
    End If
End If
Debug.Print "Interval= "; Interval
'=====
' INPUTS SUCCESSFULLY VALIDATED AT THIS POINT
'=====

'Check whether we've previously proven that a gauge exists. If not,
'check for it now.
txtStatus.Text = "Verifying gauge communications...."
If gaugeChecked = 0 Then
    checkGageAddress ta, ja
End If

If ja = 0 Then ' a gauge doesn't exist.
    AbortDensity = 1
    GoSub check_abort
End If
GetTankDimensions
```

```

'GoSub get_a_level
GetALevel

' Record Start Time (once all inputs validated and comms started)
StartTime = Time$

'=====
' CREATE THE FILENAMES OF THE LOG AND TEXT FILES
'=====
' Create the file name for the log file. The log file is for viewing by
' humans, not other software. A comma, delimited file is created after the
' log file for viewing by other spreadsheet/database apps.

    Dim F
    Dim denDir As String

    denDir = App.Path + "\DENSITY"

    ciucom "ZPN", echo, status
    decode "PN", echo, 6, answer

Dim UniqueNum$
UniqueNum = "000"
    DensityFilename = denDir & "\" & answer & UniqueNum & ".LOG"
    'Make sure filename is unique
    Do
        If Dir(DensityFilename) <> "" Then
            fileIncrement = fileIncrement + 1
            If fileIncrement < 10 Then
                UniqueNum = "00" & Trim(Str(fileIncrement))
            ElseIf fileIncrement < 100 Then
                UniqueNum = "0" & Trim(Str(fileIncrement))
            ElseIf fileIncrement < 1000 Then
                UniqueNum = Trim(Str(fileIncrement))
            Else
                answer = Right(answer, 4)
                UniqueNum = Trim(Str(fileIncrement))
            End If
            DensityFilename = denDir & "\" & answer & UniqueNum & ".LOG"
        End If
        DoEvents
    Loop Until (Dir(DensityFilename) = "")
    Debug.Print "New DenFilename= "; DensityFilename

'Create a filename for the comma delimited file
'create the filename by simply changing extension of DensityFilename
'from LOG to DAT.
Dim tmpCDF As String
tmpCDF = DensityFilename
If InStr(tmpCDF, ".") Then 'change existing extension to "DAT"
    tmpCDF = Mid(tmpCDF, 1, InStr(tmpCDF, ".")) + "DAT"
Else 'just add .DAT to end
    tmpCDF = tmpCDF + ".DAT"
End If
Debug.Print "CDF= "; tmpCDF

' create a .txt file to contain the output log data from the output log window
Dim DataLog As String
DataLog = DensityFilename
If InStr(DataLog, ".") Then 'change existing extension to "TXT"
    DataLog = Mid(DataLog, 1, InStr(DataLog, ".")) + "TXT"
Else 'just add .TXT to end
    DataLog = DataLog + ".TXT"
End If
Debug.Print "DataLog= "; DataLog
'=====
' FILENAMES CREATED.....
'=====

```

HNF-1569
Revision 1

```
'=====
'   Start the output log
'=====
txtData.Text = "<<< Density IP Scan Log Start >>>" + CR
If TestMode = 1 Then
    txtData.Text = txtData.Text & "DEMO MODE RUN, SIMULATED DATA" & vbNewLine
End If
txtData.Text = txtData.Text & StartTime & " " & Date$ & vbCrLf
txtData.Text = txtData.Text + "===== " + CR + CR

txtStatus.Text = "Obtaining gauge parameters..."

    recordCount = 0                ' initialize CIU reply count
    numretries = 3

txtData.Text = txtData.Text + CR + " Gauge Paramters" + CR
txtData.Text = txtData.Text + "-----" + CR

'Get/verify level dimension
LevelDim = GetLevelDimension
If UCase(LevelDim) <> "INCHES" Then
    'change it
    For i = 0 To numretries
        rectype = "ZLD=I"
        ciucom rectype, echo, status
        If Len(echo) > 3 Then
            i = numretries
            decode "LD", echo, 64, answer
            LevelDim = answer
        End If
        If status = 1 Then
            Delay (0.3)
        End If
    Next i
End If
txtData.Text = txtData.Text + "Level Dimension: " + UCase(LevelDim) + CR

' Get and display the gauge address
For i = 0 To numretries
    rectype = "ZTA"
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
        decode "TA", echo, 64, answer
    End If
    txtData.Text = txtData.Text + "Gauge Address: " + answer + CR
    If status = 1 Then
        Delay (0.3)
    End If
Next i

' gosub check_echo
GoSub check_abort

'Get TT Level
For i = 0 To numretries
    rectype = "ZTT"
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
        decode "TT", echo, 64, answer
    End If
    txtData.Text = txtData.Text + "TT Level: " + answer + CR
    TT = Z1BLevel(echo)
    If status = 1 Then
        Delay (0.3)
    End If
Next i
' gosub check_echo
```

```

GoSub check_abort

'Get Displacer Weight DW
For i = 0 To numretries
    rectype = "ZDW"
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
        decode "DW", echo, 64, answer
    End If
    txtData.Text = txtData.Text + "Displacer Wt. (DW): " + answer + CR
    If status = 1 Then
        Delay (0.3)
    End If
Next i
' gosub check_echo
GoSub check_abort

'Get Displacer Volume DV
For i = 0 To numretries
    rectype = "ZDV"
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
        decode "DV", echo, 64, answer
    End If
    txtData.Text = txtData.Text + "Displacer Vol. (DV): " + answer + CR
    If status = 1 Then
        Delay (0.3)
    End If
Next i
' gosub check_echo
GoSub check_abort

' Unlock the gauge
For i = 0 To numretries
    rectype = "ZUN"
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
        decode "UN", echo, 64, answer
    End If
    If InStr(answer, "&") Then
        txtData.Text = txtData.Text + "Gauge Unlocked..." + Time$ + CR
    End If
    If status = 1 Then
        Delay (0.3)
    End If
Next i

' gosub check_echo
GoSub check_abort

'GoSub get_a_level
GetALevel

' Get current reference level
For i = 0 To numretries
    rectype = "ZRL"
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
        decode "RL", echo, 64, answer
    End If
    txtData.Text = txtData.Text + "Ref Level: " + answer + CR
    RL = CDBl(answer)
    If status = 1 Then
        Delay (0.3)
    End If
Next i

```

```

        End If
    Next i

' gosub check_echo
GoSub check_abort

Dim s1 As String
' Get current setpoint force
For i = 0 To numretries
    rectype = "ZS1"
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
        decode "S1", echo, 64, answer
    End If
    txtData.Text = txtData.Text + "I1 setpoint (S1): " + answer + CR
    s1 = answer
    If status = 1 Then
        Delay (0.3)
    End If
Next i

' gosub check_echo
GoSub check_abort

' Set scan direction
'Pick it.
If optUp.Value = True Then
    rectype = "ZSD=U"
Else
    rectype = "ZSD=D"
End If

'Set it
For i = 0 To numretries
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
        decode "SD", echo, 64, answer
    End If
    txtData.Text = txtData.Text + "Scan Direction: " + answer + CR
    sd = answer
    If status = 1 Then
        Delay (0.3)
    End If
Next i

' Record remaining parameters.
txtData.Text = txtData.Text & "Start Level: " & txtDensityStart.Text & CR
txtData.Text = txtData.Text & "Sediment Level Used: " & txtSed.Text & CR
txtData.Text = txtData.Text & "Last Valid Sed. Level: " & LastSedLevel & CR
If chkSP.Value = Unchecked Then
    txtData.Text = txtData.Text & "Interval: " & txtInterval.Text & CR
Else
    txtData.Text = txtData.Text & "Interval: " & "Single Point Measurement" & CR
End If

' gosub check_echo
GoSub check_abort

' Send to interface 1 (I1)
For i = 0 To numretries
    rectype = "ZI1"
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
        decode "I1", echo, 64, answer
    End If
    If InStr(answer, "&") Then

```

HNF-1569
Revision 1

```

        txtData.Text = txtData.Text + "Sending to I1..." + Time$ + CR
    End If
    If status = 1 Then
        Delay (0.3)
    End If
Next i

txtStatus.Text = "Waiting for Displacer to reach liquid level..."
'***** (wait for displacer to get to I1) *****
i = 0
'GoSub get_a_level
GetALevel
Delay (scanRate / 1000)
Do
    GoSub check_abort
    'GoSub get_a_level
    GetALevel
    GoSub check_errors
    Delay (1)
    If InStr(frmMain.txtLevelDisplay.Text, "INN") Then
        Exit Do
    End If
    DoEvents
Loop
'***** END WAITING FOR DISPLACER TO REACH WASTE *****

txtData.Text = txtData.Text + "Surface Level Reached..." + Time$ + CR
txtStatus.Text = "Surface Level Reached..."

'Calculate Density Levels
Dim Distance As Double
Dim Delta As Integer
Dim SLevel As Double
Dim numLevels As Integer
Dim tmpLevel As String

If chkSP.Value = checked Then
    'set up to just get a single measurement
    Level(1) = txtDensityStart.Text
    numLevels = 1
    GoTo Interval_Results
End If

' Make sure start level is 12 inches above sediment
If CDBl(txtDensityStart.Text) <= CDBl(txtSed.Text) + 12 Then
    MsgBox "Start Level is too low. Aborting!"
    AbortDensity = 1
    GoSub check_abort
End If

If optUp.Value = True Then
    ' Calculate number of levels to get and fill array

    j = 1

    tmpLevel = txtDensityStart.Text
    Debug.Print "tmpLevel at start= "; tmpLevel
    Level(j) = txtDensityStart.Text
    Debug.Print "Level("; j; ")= "; Level(j)

    Do
        Level(j + 1) = Str(CDBl(tmpLevel) + CDBl(txtInterval.Text))
        Debug.Print "Level("; j + 1; ")= "; Level(j + 1)

        If CDBl(Level(j + 1)) >= CL Then
            Exit Do
        End If
        tmpLevel = Level(j + 1)
        numLevels = j + 1 ' makes sure that the "exit do" level is

```

```

        ' not used.
    j = j + 1
Loop
Debug.Print "numLevels total= "; numLevels

Else 'if optDown = true then
' Calculate number of levels to get and fill array
If mnuOptionsIPsed.checked = True Then
j = 1

tmpLevel = txtDensityStart.Text
Debug.Print "tmpLevel at start= "; tmpLevel
Level(j) = txtDensityStart.Text
Debug.Print "Level("; j; ")= "; Level(j)

Do
    Level(j + 1) = Str(CDbl(tmpLevel) - CDbl(txtInterval.Text))
    Debug.Print "Level("; j + 1; ")= "; Level(j + 1)

    If CDbl(Level(j + 1)) <= CDbl(txtSed.Text) Then
        ' get last measurement just above sediment
        ' but not if we already did.
        If CDbl(Level(j)) > CDbl(txtSed.Text) + 2 Then
            Level(j + 1) = Str((CDbl(txtSed.Text) + 2))
            Debug.Print "Last one= "; Str((CDbl(txtSed.Text) + 2))
            Debug.Print "NewLevel("; j + 1; ")= "; Level(j + 1)
        End If
        numLevels = j + 1
        Exit Do
    End If
    tmpLevel = Level(j + 1)
    numLevels = j + 1 ' makes sure that the "exit do" level is
                        ' not used.

    j = j + 1
Loop
Debug.Print "numLevels total= "; numLevels

Else
j = 1

tmpLevel = txtDensityStart.Text
Debug.Print "tmpLevel at start= "; tmpLevel
Level(j) = txtDensityStart.Text
Debug.Print "Level("; j; ")= "; Level(j)

Do
    Level(j + 1) = Str(CDbl(tmpLevel) - CDbl(txtInterval.Text))
    Debug.Print "Level("; j + 1; ")= "; Level(j + 1)

    If CDbl(Level(j + 1)) <= 0 Then
        ' get last measurement just above bottom
        ' but not if we already did.
        If CDbl(Level(j)) > 2 Then
            Level(j + 1) = Str(2)
            Debug.Print "Last one= "; 2
            Debug.Print "NewLevel("; j + 1; ")= "; Level(j + 1)
        End If
        numLevels = j + 1
        Exit Do
    End If
    tmpLevel = Level(j + 1)
    numLevels = j + 1 ' makes sure that the "exit do" level is
                        ' not used.

    j = j + 1
Loop
Debug.Print "numLevels total= "; numLevels
End If
End If

If optDown.Value = True Then
Else ' optup.value = true

```

```

    'Level(1) = Str(SLevel)
    Debug.Print "Level(1)= "; Level(1)
    For j = 0 To numLevels - 1
        Level(j + 1) = Str(SLevel + CDbl(Interval) * j)
        Debug.Print "Level("; j + 1; ")= "; Level(j + 1)
        DoEvents
    Next j
End If

Interval_Results:
txtData.Text = txtData.Text + "No. of Samples to be taken: " + Str(numLevels) + CR + CR

'GoSub get_a_level
GetALevel
' gosub check_echo
GoSub check_abort

'!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
'!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
'DEMO MODE
If TestMode = 1 Then
pBar.Min = 0
pBar.Max = 100
pBar.Value = pBar.Min
pBar.Visible = True
txtStatus.Text = "Getting Densities..."
'Set up a header for the data in output window..
txtData.Text = txtData.Text + "Level" + " " + "Density" + CR
txtData.Text = txtData.Text + "===== " + CR
' Set up a temporary array of data
Dim tmpDen(1 To 200) As String
Dim tmpSpG(1 To 200) As String
Dim tmpDat(1 To 200) As Date
Dim tmpTim(1 To 200) As String
For i = 1 To numLevels
    tmpDen(i) = "1035.00"
    tmpSpG(i) = CDbl(tmpDen(i)) * 1000 / 1000000
    tmpDat(i) = Date$
    tmpTim(i) = Time$
Next i
' Simulate getting densities
For i = 1 To numLevels
    Density(i) = tmpDen(i)
    SpG(i) = tmpSpG(i)
    GotADensity = 1
    Debug.Print "Density("; i; ")= "; Density(i)
    Debug.Print "Level("; i; ")= "; Level(i)
    theDate(i) = tmpDat(i)
    Delay (0.2)
    theTime(i) = tmpTim(i)
    txtData.Text = txtData.Text & Format(Level(i), "###.##") & _
        " " & Format(Density(i), "###.##") & _
        " (" & Format(Str(SpG(i)), "#.####") & ") " & Time$ & vbNewLine
    pBar.Value = i * (100 / numLevels)
    Delay (2)
    GoSub check_abort
    GoSub check_errors
Next i

pBar.Visible = False
pBar.Value = pBar.Min

' Put the interval value back
If txtInterval.Text = "" Then
    txtInterval.Text = Interval
End If

'DEMO MODE JUMP
GoTo Skip
End If

```


HNF-1569
Revision 1

```
'!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!  
'!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!  
  
'Set up a header for the data in output window..  
txtData.Text = txtData.Text + "Level" + "    " + "Density" + CR  
txtData.Text = txtData.Text + "===== " + CR  
  
'** THIS STARTS THE TANK INTERFACE PROFILE SCAN **'  
pBar.Min = 0  
pBar.Max = 100  
pBar.Value = pBar.Min  
pBar.Visible = True  
  
txtStatus.Text = "Getting Densities...."  
Dim L%  
For L = 1 To numLevels  
  
'SET START AND STOP LEVEL (=)  
' Set Start Level  
For i = 0 To numretries  
    rectype = "ZDK=+" & Format$(Level(L), "00000.00")  
    Debug.Print "Set DK=+"; Format$(Level(L), "00000.00")  
    ciucom rectype, echo, status  
    If Len(echo) > 3 Then  
        i = numretries  
    End If  
    If status = 1 Then  
        Delay (0.3)  
    End If  
Next i  
  
' gosub check_echo  
GoSub check_abort  
  
    ' Set Stop Level  
    For i = 0 To numretries  
        ' Make sure txtDensityLow.Text is in correct format.  
        rectype = "ZDN=+" & Format$(Level(L), "00000.00")  
        Debug.Print "Set DN=+"; Format$(Level(L), "00000.00")  
        ciucom rectype, echo, status  
        If Len(echo) > 3 Then  
            i = numretries  
        End If  
        If status = 1 Then  
            Delay (0.3)  
        End If  
    Next i  
    ' gosub check_echo  
    GoSub check_abort  
  
'START THE IP MEASUREMENT FOR THE GIVEN LEVEL  
For i = 0 To numretries  
    rectype = "ZIP"  
    ciucom rectype, echo, status  
    If Len(echo) > 3 Then  
        i = numretries  
    End If  
    If status = 1 Then  
        Delay (0.3)  
    End If  
Next i  
  
' gosub check_echo  
GoSub check_abort  
  
'SET UP A DO LOOP UNTIL IP MEASUREMENT IS COMPLETE  
  
Do
```

HNF-1569
Revision 1

```

'GoSub get_a_level
GetALevel

'Check if item SC indicates scan complete
'Get SC...
For i = 0 To numretries
    rectype = "ZSC"
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
        decode "SC", echo, 64, answer
    End If
    If status = 1 Then
        Delay (0.3)
    End If
Next i

' gosub check_echo
GoSub check_abort

'Decode SC
'If no 9's in the CIU response, we're done with this point
' In production, might want to check instr(2B, "BT")
If InStr(answer, "9999") = 0 Then
    ciucom "ZSC", echo, status
    decode "SCD", echo, 32, answer
    Density(L) = answer
    theTime(L) = Time$
    theDate(L) = Date$
    Level(L) = Str(CL) ' Record the actual level
    SpG(L) = CDBl(Density(L)) * 1000 / 1000000
    txtData.Text = txtData.Text & Format(Level(L), "###.##") & _
        " " & Format(Density(L), "#####.##") & " (" & _
        Format(Str(SpG(L)), "#.#####") & ") " & Time$ & vbNewLine
    pBar.Value = L * (100 / numLevels)
    GotADensity = 1
    Exit Do
Else
    Delay (scanRate / 1000)
End If
GoSub check_abort
GoSub check_errors
DoEvents

Loop
'Got a data point...
DoEvents
Next L
'*** THIS ENDS THE TANK PROFILE SCAN *****
pBar.Visible = False
pBar.Value = pBar.Min

' Put the interval value back
If txtInterval.Text = "" Then
    txtInterval.Text = Interval
End If

'!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
'!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Skip:
'!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
'!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

MakeFiles:
txtStatus.Text = "Densities Obtained, Processing Data...."
txtData.Text = txtData.Text + CR

'Calculate average density.
For i = 1 To numLevels
    avgden = avgden + CDBl(Density(i))

```

HNF-1569
Revision 1

```

    DoEvents
Next i
If numLevels <> 0 Then
    avgden = avgden / numLevels
Else
    avgden = 0
End If

AvgDensity = Str(avgden)
'Save as last specific gravity measurement
LastSpG = Str(avgden * 1000 / 1000000)

F = FreeFile

    Open DensityFilename For Output As F

        Print #F, Tab(2); "Enraf Control Panel Ver. "; Str(App.Major) + "." + Str(App.Minor) + "." +
Str(App.Revision) + CR
        If TestMode = 1 Then
            Print #F, Tab(3); "DEMO MODE RUN, SIMULATED DATA"
        End If
        If mnuOptionsIPSED.checked = True Then
            Print #F, Tab(3); "Scan Type: IP to Sediment"
        Else
            Print #F, Tab(3); "Scan Type: IP to Bottom"
        End If
        Print #F, Tab(2); DensityFilename + " Date: " + Date$ + " Time: " + Time$ + CR
        Print #F, Tab(2); "Data File: "; tmpCDF & " Date: " & Date$ & " Time: " & Time$ & CR
        'INSERT PRINT OF OPERATOR NAME HERE
        Print #F, Tab(5); "Operator ID: " + CurrentUserName + CR
        Print #F, CR
        Print #F, Tab(5); "Level in", , "Density Kg/m3 (Sp.G.)"
        Print #F, Tab(5); "-----", , "-----"
        Print #F, CR
        For i = 1 To numLevels
            Print #F, Tab(5); Format$(Cdbl(Level(i)), "##0.00"), , , Format$(Cdbl(Density(i)),
"##0.00") + "(" + Str(SpG(i)) + ")"
            DoEvents
        Next i

        Print #F, CR
        Print #F, Tab(10); "Average Density = " + Format$(AvgDensity, "##0.00")
        Print #F, Tab(10); "Sediment Level Used: ", SedLev
        Print #F, Tab(10); "Last Valid Sediment Level: ", LastSedLevel
        Close #F
        GettingDensities = 0

' show what file we created.
txtData.Text = txtData.Text + "Data File: " + Dir(DensityFilename, vbNormal) + CR

'Send to CDF
F = FreeFile
Open tmpCDF For Output As F
' creates a comma delimited file containing
' date, time, level, specific gravity
' (the customer does not want density included in the file).
If TestMode = 1 Then
    Print #F, "DEMO MODE RUN, SIMULATED DATA"
End If
Print #F, "DENSITY PROFILE"
For i = 1 To numLevels
    Print #F, theDate(i); ", "; theTime(i); ", "; Level(i); ", "; Str(SpG(i))
    DoEvents
Next i
Print #F, "SEDIMENT LEVEL"
Print #F, LastSedLevelDate; ", "; LastSedLevelTime; ", "; LastSedLevel
Close #F
LastDBF = tmpCDF ' save the last file created.

```

[illegible]

Return

Errhandler:

```
MsgBox "Error encountered getting densities..."
AbortDensity = 1
GoSub check_abort
```

GetOut:

```
' Save activity data
ciucom "ZPN", echo, status
decode "PN", echo, 32, answer
Debug.Print "ACTIVITY LOG PN= "; answer
F = FreeFile
Open App.Path & "\\DENSITY\\ACTIVITY.LOG" For Append As #F
Print #F, "=====
Print #F, "          Run Number: "; Format(Str(UniqueNum), "000")
Print #F, "Gauge SN: "; answer
answer = App.Major & "." & App.Minor & "." & App.Revision
Print #F, "Program Version: "; answer
Print #F, "Density Profile Start:", Date$, StartTime
Print #F, "Sediment Level: ", Format(LastSedLevel, "000.00")
Print #F, "Wire Tension used: "; Format(LastWireTension, "000.00")
Print #F, "          Acquisition Method: IP Scan"
If mnuOptionsIPSed.checked = True Then
    Print #F, "          Scan Type: IP to Sediment"
Else
    Print #F, "          Scan Type: IP to Bottom"
End If
If TestMode = 1 Then
    Print #F, "          DEMO MODE RUN, SIMULATED DATA"
End If
Print #F, "Date          Time          Level          SpG"
Print #F, "=====
For i = 1 To numLevels
    Print #F, theDate(i), theTime(i), Format(Level(i), "000.00"), Format(Str(SpG(i)), "0.0000")
    DoEvents
Next i
If AbortDensity = 1 Then
    Print #F, ">>>ABORTED<<<", Date$, Time$
    AbortDensity = 0
End If
Print #F, "^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^"
Close #F

DisabledLevelScan = 0
cmdGetDensity.MousePointer = 0
cmdGetDensity.Enabled = True
cmdClose.Enabled = True
cmdAbortDensity.Enabled = False
cmdClose.MousePointer = 0
GettingDensities = 0
frmMain.cmdRecover.Enabled = True
frmMain.cmdGetCurrentLevel.Enabled = True
frmMain.cmdOpEnraf.Enabled = True
frmMain.cmdDensity.Enabled = True
frmMain.cmdLog.Enabled = True
frmMain.cmdFTF.Enabled = True
frmMain.mnuAction.Enabled = True
frmMain.mnuConnect.Enabled = True
frmMain.mnuSetup.Enabled = True
txtStatus.Text = ""
frmMain.txtLevelDisplay.Text = ""
frmMain.txtDisplay2.Text = ""

'start scans
frmMain.cmdGetCurrentLevel_Click

Exit Sub
End Sub
```

HNF-1569
Revision 1

```

Private Sub cmdGetDensity2_Click()
*****
' This procedure obtains a density profile of a tank by lowering
' the displacer to predetermined depths based in start and interval
' values, then weighing the displacer 3 times, averaging, then
' calculating the density. The gauge, in this case, performs
' no density measurement functions per se.
'
' INPUTS:  txtDensityStart.text (Start Level); txtSed.text
'          (Sediment Level); chkSP.value (single-point boolean);
'          LastSedLevel (Last valid sediment level); txtInterval.text
'          (Interval)
'          mnuOptionsIPSED.value | mnuOptionsIPBottom.value
' OUTPUTS: Density() array; Level() array; SpG() array; .LOG file;
'          .TXT file; .DAT file; ACTIVITY.LOG file
' CONDITIONS: Start Level must be > 12 inches above sediment level;
'             Interval must be greater than 6 inches.
'
' The routine will check for displacer movement in case the
' displacer is sent into the sediment.
*****

' Check to see if we're in DEMO MODE. This procedure is not designed
' to handle DEMO MODE.
If TestMode = 1 Then
    MsgBox "Sorry, you are in DEMO MODE. You must exit from DEMO MODE first."
    Exit Sub
End If

If BalanceTestPassed = False Then
    MsgBox "Please perform a balance test before continuing..."
    Exit Sub
End If

'On Error GoTo Errhandler
On Error Resume Next
Dim i%, j%
Dim echo As String           'response from CIU/gauge
Dim answer As String         'result from decode
Dim echoString(1 To 50) As String 'array of CIU/gauge responses
Dim recordCount As Integer   'count of echoStrings
Dim message As String        'used for msgboxes
Dim Reply As Integer         'used to capture msgbox responses
Dim fileIncrement As Integer
Dim Level(0 To 200) As String 'holds levels at which densities were obtained
Dim Density(0 To 200) As String 'hold densities
Dim sd As String             'scan direction

Dim rectype As String        'holds a CIU request string
Dim CR                        'carriage return for text
CR = Chr(13) + Chr(10)
'dim CL As Double            'Current Level
Dim oldlevel As Double
Dim theDate(0 To 200) As String
Dim theTime(0 To 200) As String
Dim AvgDensity As String
Dim avgden As Double
Dim SpG(1 To 200) As Double
Dim LevelDim$, m$
Dim GotADensity%             'Flag for marking data acquisition
Dim WQ As Double
Dim DW As Double
Dim DV As Double
Dim WW As Double
Dim Y4 As Double
Dim WV As Double
Dim RF As Double
Dim IWV As Double
Dim StartTime As String

```

```
txtStatus.Text = "...Stand by...."
avgden = 0
AbortDensity = 0
DisabledLevelScan = 0
GotADensity = 0

'Clear any levels and densities
For i = 0 To 200
    Density(i) = "0"
    Level(i) = "0"
Next i

'clear status window
txtStatus.Text = ""

'Set form components
cmdGetDensity.MousePointer = 11
cmdGetDensity.Enabled = False
cmdClose.Enabled = False
cmdAbortDensity.Enabled = True
cmdClose.MousePointer = 11
txtData.Text = ""
frmMain.cmdRecover.Enabled = False
frmMain.cmdGetCurrentLevel.Enabled = False
frmMain.cmdOpEnraf.Enabled = False
frmMain.cmdDensity.Enabled = False
frmMain.cmdLog.Enabled = False
frmMain.cmdFTF.Enabled = False
frmMain.mnuAction.Enabled = False
frmMain.mnuConnect.Enabled = False
frmMain.mnuSetup.Enabled = False

'Ensure level scanning is disabled, set a flag to turn it back on when complete.
If frmMain.tmrLevel.Enabled = True Then
    frmMain.cmdDisconnect_Click
    DisabledLevelScan = 1
End If

'Set Flag to tell other program functions we're busy.
GettingDensities = 1

' Warn the operator. (Dipping the displacer in high level
' radioactive waste is no small thing).
m = "Warning! You are about to DIP the displacer." + CR
m = m + "Do you want to continue?"
i = MsgBox(m, vbYesNo + vbExclamation, "DIP Displacer?")
If i <> 6 Then
    AbortDensity = 1
    GoSub check_abort
End If

'=====
'VALIDATE INPUT DATA
'=====
txtStatus.Text = "Validating Input Data...."
If txtDensityStart.Text = "" Then
    MsgBox "You must enter a start level!", vbOKOnly + vbCritical, "Need Start Level!"
    AbortDensity = 1
    GoSub check_abort
Else
    SDLevel = txtDensityStart.Text
End If

If txtSed.Text = "" Then
    message = "You must enter a sediment level, or click GET" + CR
    message = message + "to have the gauge obtain it for you..."
    MsgBox message, vbOKOnly + vbCritical, "Need a Sediment Level..."
    AbortDensity = 1
```

HNF-1569
Revision 1

```

    GoSub check_abort
Else
    SedLev = txtSed.Text
End If

If txtInterval.Text = "" And chkSP.Value = Unchecked Then
    txtInterval.Text = "12"
    Interval = "12"
End If

If chkSP.Value = Unchecked Then
    If CInt(txtInterval.Text) < 6 Then
        m = "You must have an interval 7-inches or greater." + CR
        m = "Please start again."
        MsgBox m, vbOKOnly + vbCritical, "Interval too small!"
        AbortDensity = 1
        GoSub check_abort
    Else
        Interval = txtInterval.Text
    End If
End If
Debug.Print "Interval= "; Interval
'=====
'  INPUTS SUCCESSFULLY VALIDATED AT THIS POINT
'=====

'Check whether we've previously proven that a gauge exists.  If not,
'check for it now.
txtStatus.Text = "Establishing gauge communications...."
    'If gaugeChecked = 0 Then
        checkGageAddress ta, ja
    'End If

    If ja = 0 Then ' a gauge doesn't exist.
        AbortDensity = 1
        GoSub check_abort
    End If
GetTankDimensions
'GoSub get_a_level
GetALevel

' Record Start time for the scan
StartTime = Time$

'=====
'  CREATE THE FILENAMES OF THE LOG AND TEXT FILES
'=====
' Create the file name for the log file.  The log file is for viewing by
' humans, not other software.  A comma, delimited file is created after the
' log file for viewing by other spreadsheet/database apps.

    Dim F
    Dim denDir As String

    denDir = App.Path + "\DENSITY"

    ciucom "ZPN", echo, status
    decode "PN", echo, 6, answer

Dim UniqueNum$
UniqueNum = "000"
    DensityFilename = denDir & "\" & answer & UniqueNum & ".LOG"
    'Make sure filename is unique
    Do
        If Dir(DensityFilename) <> "" Then
            fileIncrement = fileIncrement + 1
            If fileIncrement < 10 Then
                UniqueNum = "00" & Trim(Str(fileIncrement))
            ElseIf fileIncrement < 100 Then

```


HNF-1569
Revision 1

```

        UniqueNum = "0" & Trim(Str(fileIncrement))
    ElseIf fileIncrement < 1000 Then
        UniqueNum = Trim(Str(fileIncrement))
    Else
        answer = Right(answer, 4)
        UniqueNum = Trim(Str(fileIncrement))
    End If
    DensityFilename = denDir & "\" & answer & UniqueNum & ".LOG"
End If
DoEvents
Loop Until (Dir(DensityFilename) = "")

'Create a filename for the comma delimited file
'create the filename by simply changing extension of DensityFilename
'from LOG to DAT.
Dim tmpCDF As String
tmpCDF = DensityFilename
If InStr(tmpCDF, ".") Then 'change existing extension to "DAT"
    tmpCDF = Mid(tmpCDF, 1, InStr(tmpCDF, ".")) & "DAT"
Else 'just add .DAT to end
    tmpCDF = tmpCDF & ".DAT"
End If

' create a .txt file to contain the output log data from the output log window
Dim DataLog As String
DataLog = DensityFilename
If InStr(DataLog, ".") Then 'change existing extension to "TXT"
    DataLog = Mid(DataLog, 1, InStr(DataLog, ".")) & "TXT"
Else 'just add .TXT to end
    DataLog = DataLog & ".TXT"
End If

'=====
'  FILENAMES CREATED.....
'=====

recordCount = 0          ' initialize CIU reply count
numretries = 3

'=====
'  Start the output log
'=====
txtData.Text = "<<< Density Program Scan Log Start >>>" & CR
If TestMode = 1 Then
    txtData.Text = "DEMO MODE, SIMULATED DATA" & CR
End If
txtData.Text = txtData.Text & StartTime & "  " & Date$ & vbCrLf
txtData.Text = txtData.Text & "===== " & CR & CR

' Unlock the gauge
For i = 0 To numretries
    rectype = "ZUN"
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
        decode "UN", echo, 64, answer
    End If
    If InStr(answer, "&") Then
        txtData.Text = txtData.Text & "Gauge Unlocked..." & Time$ & CR
    End If
    If status = 1 Then
        Delay (0.3)
    End If
Next i
If TestMode = 1 Then
    frmDemo.cmd1BtoExcl_Click
    frmDemo.cmd2BtoI1_Click
End If

' gosub check_echo
GoSub check_abort

```

```
'GoSub get_a_level
GetALevel

txtStatus.Text = "Obtaining gauge parameters..."

txtData.Text = txtData.Text + CR + " Gauge Paramters" + CR
txtData.Text = txtData.Text + "-----" + CR

'Get/verify level dimension
LevelDim = GetLevelDimension
If UCase(LevelDim) <> "INCHES" Then
  'change it
  For i = 0 To numretries
    rectype = "ZLD=I"
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
      i = numretries
      decode "LD", echo, 64, answer
      LevelDim = answer
    End If
    If status = 1 Then
      Delay (0.3)
    End If
  Next i
  LevelDim = GetLevelDimension ' Reload changed LevelDim
End If
txtData.Text = txtData.Text + "Level Dimension: " + UCase(LevelDim) + CR

' Get and display the gauge address
For i = 0 To numretries
  rectype = "ZTA"
  ciucom rectype, echo, status
  If Len(echo) > 3 Then
    i = numretries
    decode "TA", echo, 64, answer
  End If
  txtData.Text = txtData.Text + "Gauge Address: " + answer + CR
  If status = 1 Then
    Delay (0.3)
  End If
Next i

' gosub check_echo
GoSub check_abort

'Get TT Level
For i = 0 To numretries
  rectype = "ZTT"
  ciucom rectype, echo, status
  If Len(echo) > 3 Then
    i = numretries
    decode "TT", echo, 64, answer
  End If
  txtData.Text = txtData.Text + "TT Level: " + answer + CR
  TT = ZiBLevel(echo)
  If status = 1 Then
    Delay (0.3)
  End If
Next i
' gosub check_echo
GoSub check_abort

'Get Displacer Weight DW
For i = 0 To numretries
  rectype = "ZDW"
  ciucom rectype, echo, status
  If Len(echo) > 3 Then
    i = numretries
    decode "DW", echo, 64, answer
```

HNF-1569
Revision 1

```
End If
DW = CDBl(answer)
txtData.Text = txtData.Text + "Displacer Wt. (DW): " + answer + CR
If status = 1 Then
    Delay (0.3)
End If
Next i
' gosub check_echo
GoSub check_abort

'Get Displacer Volume DV
For i = 0 To numretries
    rectype = "ZDV"
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
        decode "DV", echo, 64, answer
    End If
    DV = CDBl(answer)
    txtData.Text = txtData.Text + "Displacer Vol. (DV): " + answer + CR
    If status = 1 Then
        Delay (0.3)
    End If
Next i
' gosub check_echo
GoSub check_abort

' Get current reference level
For i = 0 To numretries
    rectype = "ZRL"
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
        decode "RL", echo, 64, answer
    End If
    txtData.Text = txtData.Text + "Ref Level: " + answer + CR
    RL = CDBl(answer)
    If status = 1 Then
        Delay (0.3)
    End If
Next i

' gosub check_echo
GoSub check_abort

Dim s1 As String
' Get current setpoint force
For i = 0 To numretries
    rectype = "ZS1"
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
        decode "S1", echo, 64, answer
    End If
    txtData.Text = txtData.Text + "I1 setpoint (S1): " + answer + CR
    s1 = answer
    If status = 1 Then
        Delay (0.3)
    End If
Next i

' gosub check_echo
GoSub check_abort

' Set scan direction
'Pick it.
If optUp.Value = True Then
    rectype = "ZSD=U"
Else
    rectype = "ZSD=D"
```

HNF-1569
Revision 1

```
End If

'Set it
For i = 0 To numretries
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
        decode "SD", echo, 64, answer
    End If
    txtData.Text = txtData.Text + "Scan Direction: " + answer + CR
    sd = answer
    If status = 1 Then
        Delay (0.3)
    End If
Next i

' Record remaining parameters.
txtData.Text = txtData.Text & "Start Level: " & txtDensityStart.Text & CR
txtData.Text = txtData.Text & "Sediment Level: " & txtSed.Text & CR
If chkSP.Value = Unchecked Then
    txtData.Text = txtData.Text & "Interval: " & txtInterval.Text & CR
Else
    txtData.Text = txtData.Text & "Interval: " & "Single Point Measurement" & CR
End If

' gosub check_echo
GoSub check_abort

' Send to interface 1 (I1)
For i = 0 To numretries
    rectype = "ZI1"
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
        decode "I1", echo, 64, answer
    End If
    If InStr(answer, "&") Then
        txtData.Text = txtData.Text + "Sending to I1..." + Time$ + CR
    End If
    If status = 1 Then
        Delay (0.3)
    End If
Next i
txtStatus.Text = "Waiting for Displacer to reach liquid level..."
'***** (wait for displacer to get to I1) *****
i = 0
GetALevel
Delay (scanRate / 1000)
Do
    GoSub check_abort
    GetALevel
    GoSub check_errors
    Delay (scanRate / 1000)
    If InStr(frmMain.txtLevelDisplay.Text, "INN") Then
        Exit Do
    End If
    DoEvents
Loop
'***** END WAITING FOR DISPLACER TO REACH WASTE *****

txtData.Text = txtData.Text + "Surface Level Reached..." + Time$ + CR
txtStatus.Text = "Surface Level Reached..."

'Calculate Density Levels
Dim Distance As Double
Dim Delta As Integer
Dim SLevel As Double
Dim numLevels As Integer
Dim tmpLevel As String
```

```

If chkSP.Value = checked Then
    'set up to just get a single measurement
    Level(1) = txtDensityStart.Text
    numLevels = 1
    GoTo Interval_Results
End If

' Calculate number of levels to get and fill array
If mnuOptionsIPSed.checked = True Then
j = 1

tmpLevel = txtDensityStart.Text
Debug.Print "tmpLevel at start= "; tmpLevel
Level(j) = txtDensityStart.Text
Debug.Print "Level("; j; ")= "; Level(j)

Do
    GoSub check_abort
    Level(j + 1) = Str(CDbl(tmpLevel) - CDbl(txtInterval.Text))
    Debug.Print "Level("; j + 1; ")= "; Level(j + 1)

    If CDbl(Level(j + 1)) <= CDbl(txtSed.Text) Then
        ' get last measurement just above sediment
        ' but not if we already did.
        If CDbl(Level(j)) > CDbl(txtSed.Text) + 2 Then
            Level(j + 1) = Str((CDbl(txtSed.Text) + 2))
            Debug.Print "Last one= "; Str((CDbl(txtSed.Text) + 2))
            Debug.Print "NewLevel("; j + 1; ")= "; Level(j + 1)
        End If
        numLevels = j + 1
        Exit Do
    End If
    tmpLevel = Level(j + 1)
    numLevels = j + 1 ' makes sure that the "exit do" level is
                     ' not used.

    j = j + 1
Loop
Debug.Print "numLevels total= "; numLevels
Else ' going to bottom
j = 1

tmpLevel = txtDensityStart.Text
Debug.Print "tmpLevel at start= "; tmpLevel
Level(j) = txtDensityStart.Text
Debug.Print "Level("; j; ")= "; Level(j)

Do
    GoSub check_abort
    Level(j + 1) = Str(CDbl(tmpLevel) - CDbl(txtInterval.Text))
    Debug.Print "Level("; j + 1; ")= "; Level(j + 1)

    If CDbl(Level(j + 1)) <= 0 Then
        ' get last measurement just above bottom
        ' but not if we already did.
        If CDbl(Level(j)) > 2 Then
            Level(j + 1) = Str(2)
            Debug.Print "Last one= "; 2
            Debug.Print "NewLevel("; j + 1; ")= "; Level(j + 1)
        End If
        numLevels = j + 1
        Exit Do
    End If
    tmpLevel = Level(j + 1)
    numLevels = j + 1 ' makes sure that the "exit do" level is
                     ' not used.

    j = j + 1
Loop
Debug.Print "numLevels total= "; numLevels
End If

```

HNF-1569
Revision 1

```
Interval_Results:
txtData.Text = txtData.Text + "No. of Samples to be taken: " + Str(numLevels) + CR + CR

GetAlevel
GoSub check_abort

' Set up a header for the data in output window..
txtData.Text = txtData.Text + "Level" + "    " + "Density" + CR
txtData.Text = txtData.Text + "===== " + CR

pBar.Min = 0
pBar.Max = 100
pBar.Value = pBar.Min
pBar.Visible = True

txtStatus.Text = "Getting Densities...."
Dim L%, p%
Dim nTempAvg As Double
' Set up a counter to check for displacer movement
Dim LCount%, oldCL As Double

' ** THIS STARTS THE TANK DENSITY PROFILE SCAN ****
For L = 1 To numLevels

' gosub check_echo
GoSub check_abort

' Make sure we're unlocked
ciucom "ZUN", echo, status

txtStatus.Text = "Seeking next level...."

' Use IP command to send it there.
ciucom "ZDK=+" & Format(Level(L), "00000.00"), echo, status
ciucom "ZDN=+" & Format(Level(L), "00000.00"), echo, status
ciucom "ZIP", echo, status

' Wait until level is reached.

LCount = 0
Do
    If TestMode = 1 Then
        frmDemo.cmd2BtoFR_Click
        Exit Do
    End If
    GetAlevel
    GoSub check_abort
    DoEvents
    If CL <= CDbl(Level(L)) + 0.05 And CL >= CDbl(Level(L)) - 0.05 Then
        ' we are there
        ciucom "ZFR", echo, status
        Exit Do
    End If
Loop

GoSub check_abort

' Measure WQ three times
nTempAvg = 0
For p = 1 To 3

' Weigh displacer
txtStatus.Text = "Level " & Str(L) & " reached, Weighing Displacer...."
ciucom "ZMF", echo, status
If TestMode = 1 Then
    frmDemo.cmd2BtoMF_Click
End If
Delay (2)
' Wait for MF to complete
Do
```

```

GoSub check_abort
GetALevel s1B, s2B
If InStr(s2B, "FR") <> 0 Then
Debug.Print "MF Done, s2b= "; s2B
Exit Do
End If
DoEvents
Loop

ciucom "ZWQ", echo, status
decode "WQ", echo, 32, answer
WQ = CDBl(answer)
Debug.Print "WQ= "; WQ

nTempAvg = nTempAvg + WQ
GoSub check_abort
Next p

' Calculate avg weight
If nTempAvg = 0 Then
MsgBox "Zero average weight...Aborting."
AbortDensity = 1
GoSub check_abort
End If
WQ = nTempAvg / 3
GoSub check_abort

txtStatus.Text = "Calculating Density..."
Debug.Print "DW= "; DW

Debug.Print "DV= "; DV

' Get last Surface Level
ciucom "ZY4", echo, status
decode "Y4", echo, 32, answer
answer = Right(answer, 6)
Y4 = CDBl(answer)
Debug.Print "Y4= "; Y4

' Get Wire Weight wrt TT
ciucom "ZWW", echo, status
decode "WW", echo, 32, answer
WW = (TT - CL) * 0.0254 * CDBl(answer)
Debug.Print "WW= "; WW

' Get wire volume
ciucom "ZVW", echo, status
decode "VW", echo, 32, answer
VW = CDBl(answer)
Debug.Print "VW= "; VW

' Calculate immersed wire volume
' Conversion for inches to meters is 0.0254 m/inch
IWV = (Y4 - CL) * VW * 0.0254
Debug.Print "IWV= "; IWV

' Get ambient air density
ciucom "ZRF", echo, status
decode "RF", echo, 32, answer
RF = CDBl(answer)
Debug.Print "RF= "; RF

' Calculate density
Density(L) = Str(((DW - WQ + WW) / (DV + IWV)) * 1000)
Debug.Print "Density("; L; ")= "; Density(L)
theTime(L) = Time$
theDate(L) = Date$
Level(L) = Str(CL) ' Record the actual level
SpG(L) = CDBl(Density(L)) * 1000 / 1000000
txtData.Text = txtData.Text & Format(Level(L), "###.###") & " " & Format(Density(L),
"###.###") & " (" & Format(Str(SpG(L)), "#.#####") & ") " & Time$ & vbNewLine

```

HNF-1569
Revision 1

```

        pBar.Value = L * (100 / numLevels)
        GotADensity = 1
        GoSub check_abort
        GoSub check_errors
        'Got a data point....
        DoEvents
    Next L
    '*** THIS ENDS THE TANK PROFILE SCAN *****
    pBar.Visible = False
    pBar.Value = pBar.Min

    ' Send displacer back to surface
    ciucom "ZUN", echo, status
    ciucom "Z11", echo, status

    ' Put the interval value back
    If txtInterval.Text = "" Then
        txtInterval.Text = Interval
    End If

    '!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
    '!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
    'skip:
    '!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
    '!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

MakeFiles:
txtStatus.Text = "Densities Obtained, Processing Data...."
txtData.Text = txtData.Text + CR

'Calculate average tank density.
For i = 1 To numLevels
    avgden = avgden + CDbl(Density(i))
    DoEvents
Next i
If numLevels <> 0 Then
    avgden = avgden / numLevels
Else
    avgden = 0
End If

AvgDensity = Str(avgden)
'Save as last specific gravity measurement
LastSpG = Str(avgden * 1000 / 1000000)

F = FreeFile

    Open DensityFilename For Output As F

    Print #F, Tab(2); "Enraf Control Panel Ver. "; Str(App.Major) + "." + Str(App.Minor) + "." +
Str(App.Revision) + CR
    If TestMode = 1 Then
        Print #F, Tab(3); "DEMO MODE RUN, SIMULATED DATA"
    End If
    Print #F, Tab(2); DensityFilename + " Date: " + Date$ + " Time: " + Time$ + CR
    Print #F, Tab(2); "Data File: "; tmpCDF & " Date: " & Date$ & " Time: " & Time$ & CR
    'INSERT PRINT OF OPERATOR NAME HERE
    Print #F, Tab(5); "Operator ID: " + CurrentUserName + CR
    Print #F, CR
    Print #F, Tab(5); "Level in", , "Density Kg/m3 (Sp.G.)"
    Print #F, Tab(5); "-----", , "-----"
    Print #F, CR
    For i = 1 To numLevels
        Print #F, Tab(5); Format$(CDbl(Level(i)), "##0.00"), , , Format$(CDbl(Density(i)),
"##0.00") + "(" + Str(SpG(i)) + ")"
        DoEvents
    Next i

```


HNF-1569

Revision 1

```
Print #F, CR
Print #F, Tab(10); "Average Tank Density = " + Format$(AvgDensity, "##0.00")
Print #F, Tab(10); "Sediment Level Used: ", SedLev
Close #F
GettingDensities = 0
```

```
txtData.Text = txtData.Text + "Data File: " + Dir(DensityFilename, vbNormal) + CR
```

```

Send to CDF
F = FreeFile
  Open tmpCDF For Output As F
  ' creates a comma delimited file containing
  ' date, time, level, specific gravity
  ' (the customer does not want density included in the file).
  If TestMode = 1 Then
    Print #F, "DEMO MODE RUN, SIMULATED DATA"
  End If

  Print #F, "DENSITY PROFILE"
  For i = 1 To numLevels
    Print #F, theDate(i); ", "; theTime(i); ", "; Level(i); ", "; Str(SpG(i))
    DoEvents
  Next i
  Print #F, "SEDIMENT LEVEL"
  Print #F, LastSedLevelDate; ", "; LastSedLevelTime; ", "; LastSedLevel
  Close #F
  LastDBF = tmpCDF ' save the last file created.

```

[illegible]

```
'Send txtData to txt file
F = FreeFile
Open DataLog For Output As F
Print #F, txtData.Text
Close #F
```

```
txtStatus.Text = "Density Profile Complete..."
```

```
message = "Density LOG located in file: " + DensityFilename + CR
message = message + "Density data filename is: " + tmpCDF + CR
message = message + CR + "View/Print LOG file?"
Reply = MsgBox(message, vbYesNo + vbInformation, "View Log File?")
```

```
If Reply = 6 Then
'Debug.Print Dir(DensityFilename)
If Dir(DensityFilename) = "" Then
MsgBox "Density Log File has not yet been created."
GoSub GetOut
End If
Shell "C:\WINDOWS\NOTEPAD.EXE " & DensityFilename, vbNormalFocus
GettingDensities = 0

End If
GoSub GetOut
```

```
check_abort:
    If AbortDensity = 1 Then
        Debug.Print "AbortDensity was 1"
        txtData.Text = txtData.Text + "Program Driven Scan" + CR + " --ABORTED!" + CR
        If GotADensity = 1 Then
            GoTo MakeFiles ' Save whatever data we got
        Else
            GoSub GetOut
        End If
    End If
```

```

Else
    Return
End If

check_errors:
    ciucom "ZES", echo, status
    decode "ES", echo, 64, answer
    answer = Trim(answer)
    If answer <> "0000" Then
        m = GetSPUErrormessage(answer)
        m = m + CR
        m = m + "Aborting..."
        MsgBox m, vbOKOnly + vbCritical
        AbortDensity = 1
        GoSub check_abort
    End If
    ciucom "ZEP", echo, status
    decode "EP", echo, 64, answer
    answer = Trim(answer)
    If Instr(echo, "!0") <> 0 Then
        MsgBox "Aborting!", vbOKOnly + vbCritical
        AbortDensity = 1
        GoSub check_abort
    End If
    Return

Errhandler:
    MsgBox "Error encountered getting densities..."
    AbortDensity = 1
    GoSub check_abort

GetOut:
    ' Save activity data
    ciucom "ZPN", echo, status
    decode "PN", echo, 32, answer
    Debug.Print "ACTIVITY LOG PN= "; answer
    F = FreeFile
    Open App.Path & "\DENSITY\ACTIVITY.LOG" For Append As #F
    Print #F, "=====
    Print #F, "          Run Number: "; Format(Str(UniqueNum), "000")
    Print #F, "Gauge SN: "; answer
    answer = App.Major & "." & App.Minor & "." & App.Revision
    Print #F, "Program Version: "; answer
    Print #F, "Density Profile Start: ", Date$, StartTime
    Print #F, "Sediment Level: ", Format(LastSedLevel, "000.00")
    Print #F, "Wire Tension used: "; Format(LastWireTension, "000.00")
    Print #F, "          Acquisition Method: Program Driven"
    If mnuOptionsIPSed.checked = True Then
        Print #F, "          Scan Type: Program to Sediment"
    Else
        Print #F, "          Scan Type: Program to Bottom"
    End If
    If TestMode = 1 Then
        Print #F, "          DEMO MODE RUN, SIMULATED DATA"
    End If
    Print #F, "Date           Time           Level           SpG"
    Print #F, "=====
    For i = 1 To numLevels
        Print #F, theDate(i), theTime(i), Format(Level(i), "000.00"), Format(Str(SpG(i)), "0.0000")
        DoEvents
    Next i
    If AbortDensity = 1 Then
        Print #F, ">>>ABORTED<<<", Date$, Time$
        AbortDensity = 0
    End If
    Print #F, "^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^"
Close #F
    DisabledLevelScan = 0
    cmdGetDensity.MousePointer = 0

```

HNF-1569
Revision 1

```
cmdGetDensity.Enabled = True
cmdClose.Enabled = True
cmdAbortDensity.Enabled = False
cmdClose.MousePointer = 0
GettingDensities = 0
frmMain.cmdRecover.Enabled = True
frmMain.cmdGetCurrentLevel.Enabled = True
frmMain.cmdOpEnraf.Enabled = True
frmMain.cmdDensity.Enabled = True
frmMain.cmdLog.Enabled = True
frmMain.cmdFTF.Enabled = True
frmMain.mnuAction.Enabled = True
frmMain.mnuConnect.Enabled = True
frmMain.mnuSetup.Enabled = True
txtStatus.Text = ""
frmMain.txtLevelDisplay.Text = ""
frmMain.txtDisplay2.Text = ""

'start scans
frmMain.cmdGetCurrentLevel_Click
```

End Sub

```
Private Sub cmdGetDensityFull_Click()
'*****
' This procedure obtains a density profile of a tank using
' the IP command. It is fully automatic in that it uses the
' normal default start value of 156, it gets a single-point
' density measurement in order to estimate a wire tension for
' getting a sediment level. It then, automatically gets a sed-
' iment level, then proceeds with the tank profile scan.
'
' INPUTS: txtDensityStart.text (Start Level); LastSedLevel
'         (Last valid sediment level); txtInterval.text (Interval)
'         mnuOptionsIPSED.value | mnuOptionsIPBottom.value
' OUTPUTS: Density() array; Level() array; SpG() array; .LOG file;
'         .TXT file; .DAT file; ACTIVITY.LOG file
' CONDITIONS: Start Level must be > 12 inches above sediment level;
'             If optUP.value = true then Start Level must be
'             greater than 6 inches below the surface; Interval
'             must be greater than 6 inches.
'
' The routine will check for low wire tension errors in case the
' displacer is sent into the sediment.
'*****
```

On Error Resume Next

```
If BalanceTestPassed = False Then
    MsgBox "Please perform a balance test before continuing..."
Exit Sub
End If
```

```
Dim i%, j%
Dim echo As String           'response from CIU/gauge
Dim answer As String        'result from decode
Dim echoString(1 To 50) As String 'array of CIU/gauge responses
Dim recordCount As Integer  'count of echoStrings
Dim message As String       'used for msgboxes
Dim Reply As Integer        'used to capture msgbox responses
Dim fileIncrement As Integer
Dim Level(0 To 200) As String 'holds levels at which densities were obtained
Dim Density(0 To 200) As String 'hold densities
Dim sd As String            'scan direction

Dim rectype As String       'holds a CIU request string
Dim CR                      'carriage return for text
CR = Chr(13) + Chr(10)
'dim CL As Double           'Current Level
Dim oldlevel As Double
```

```

Dim theDate(0 To 200) As String
Dim theTime(0 To 200) As String
Dim AvgDensity As String
Dim avgden As Double
Dim SpG(1 To 200) As Double
Dim LevelDim$, m$
Dim Error As Boolean
Dim GotADensity%           'Flag for marking data acquisition
Dim StartTime As String
txtStatus.Text = "...Stand by...."
avgden = 0
AbortDensity = 0
DisabledLevelScan = 0
GotADensity = 0

'Clear any levels and densities
For i = 0 To 200
    Density(i) = "0"
    Level(i) = "0"
Next i
OldStart = txtDensityStart.Text
txtDensityStart.Text = "156.00"
OldSed = txtSed.Text
txtSed.Text = ""
OldInterval = txtInterval.Text
txtInterval.Text = "12"

'clear status window
txtStatus.Text = ""

'Set form components
cmdGetDensity.MousePointer = 11
cmdGetDensity.Enabled = False
cmdClose.Enabled = False
cmdAbortDensity.Enabled = True
cmdClose.MousePointer = 11
txtData.Text = ""
frmMain.cmdRecover.Enabled = False
frmMain.cmdGetCurrentLevel.Enabled = False
frmMain.cmdOpEnraf.Enabled = False
frmMain.cmdDensity.Enabled = False
frmMain.cmdLog.Enabled = False
frmMain.cmdFTF.Enabled = False
frmMain.mnuAction.Enabled = False
frmMain.mnuConnect.Enabled = False
frmMain.mnuSetup.Enabled = False

'Ensure level scanning is disabled, set a flag to turn it back on when complete.
If frmMain.tmrLevel.Enabled = True Then
    frmMain.cmdDisconnect_Click
    DisabledLevelScan = 1
End If

'Set Flag to tell other program functions we're busy.
GettingDensities = 1

' Warn the operator. (Dipping the displacer in high level
' radioactive waste is no small thing).
m = "Warning! You are about to DIP the displacer." + CR
m = m + "Do you want to continue?"
i = MsgBox(m, vbYesNo + vbExclamation, "DIP Displacer?")
If i <> 6 Then
    AbortDensity = 1
    GoSub check_abort
End If

'=====
'VALIDATE INPUT DATA

```

HNF-1569
Revision 1

```

'=====
'
' INPUTS SUCCESSFULLY VALIDATED AT THIS POINT
'=====

'Check whether we've previously proven that a gauge exists. If not,
'check for it now.
txtStatus.Text = "Establishing gauge communications...."
    'If gaugeChecked = 0 Then
        checkGageAddress ta, ja
    'End If

    If ja = 0 Then ' a gauge doesn't exist.
        AbortDensity = 1
        GoSub check_abort
    End If
GetTankDimensions
'GoSub get_a_level
GetALevel

' Record Start Time for scan
StartTime = Time$

'=====
' CREATE THE FILENAMES OF THE LOG AND TEXT FILES
'=====
' Create the file name for the log file. The log file is for viewing by
' humans, not other software. A comma, delimited file is created after the
' log file for viewing by other spreadsheet/database apps.

    Dim F
    Dim denDir As String

    denDir = App.Path + "\DENSITY"

    ciucom "ZPN", echo, status
    decode "PN", echo, 6, answer

Dim UniqueNum$
UniqueNum = "000"
    DensityFilename = denDir & "\" & answer & UniqueNum & ".LOG"
    'Make sure filename is unique
    Do
        If Dir(DensityFilename) <> "" Then
            fileIncrement = fileIncrement + 1
            If fileIncrement < 10 Then
                UniqueNum = "00" & Trim(Str(fileIncrement))
            ElseIf fileIncrement < 100 Then
                UniqueNum = "0" & Trim(Str(fileIncrement))
            ElseIf fileIncrement < 1000 Then
                UniqueNum = Trim(Str(fileIncrement))
            Else
                answer = Right(answer, 4)
                UniqueNum = Trim(Str(fileIncrement))
            End If
            DensityFilename = denDir & "\" & answer & UniqueNum & ".LOG"
        End If
    DoEvents
    Loop Until (Dir(DensityFilename) = "")
    Debug.Print "New DenFilename= "; DensityFilename

'Create a filename for the comma delimited file
'create the filename by simply changing extension of DensityFilename
'from LOG to DAT.
Dim tmpCDF As String
tmpCDF = DensityFilename
If InStr(tmpCDF, ".") Then 'change existing extension to "DAT"
    tmpCDF = Mid(tmpCDF, 1, InStr(tmpCDF, ".")) + "DAT"

```

HNF-1569
Revision 1

```

Else 'just add .DAT to end
    tmpCDF = tmpCDF + ".DAT"
End If
Debug.Print "CDF= "; tmpCDF

' create a .txt file to contain the output log data from the output log window
Dim DataLog As String
DataLog = DensityFilename
If InStr(DataLog, ".") Then 'change existing extension to "TXT"
    DataLog = Mid(DataLog, 1, InStr(DataLog, ".")) + "TXT"
Else 'just add .TXT to end
    DataLog = DataLog + ".TXT"
End If
Debug.Print "DataLog= "; DataLog
'=====
' FILENAMES CREATED.....
'=====

'=====
' Start the output log
'=====
txtData.Text = "<<< Density WRRS Full IP Scan Log Start >>>" + CR
If TestMode = 1 Then
    txtData.Text = txtData.Text & "DEMO MODE RUN, SIMULATED DATA" & vbNewLine
End If
txtData.Text = txtData.Text & StartTime & " " & Date$ & vbCrLf
txtData.Text = txtData.Text & "===== " & CR
If mnuOptionsIPsed.checked = True Then
    txtData.Text = txtData.Text & "Scan Type: Full Auto IP to Sediment" & vbNewLine
Else
    txtData.Text = txtData.Text & "Scan Type; Full Auto IP to Bottom"
End If

txtStatus.Text = "Obtaining gauge parameters..."

recordCount = 0 ' initialize CIU reply count
numretries = 3

txtData.Text = txtData.Text + CR + " Gauge Paramters" + CR
txtData.Text = txtData.Text + "-----" + CR

'Get/verify level dimension
LevelDim = GetLevelDimension
If UCase(LevelDim) <> "INCHES" Then
    'change it
    For i = 0 To numretries
        rectype = "ZLD=I"
        ciucom rectype, echo, status
        If Len(echo) > 3 Then
            i = numretries
            decode "LD", echo, 64, answer
            LevelDim = answer
        End If
        If status = 1 Then
            Delay (0.3)
        End If
    Next i
End If
txtData.Text = txtData.Text + "Level Dimension: " + UCase(LevelDim) + CR

' Get and display the gauge address
For i = 0 To numretries
    rectype = "ZTA"
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
        decode "TA", echo, 64, answer
    End If

```

HNF-1569
Revision 1

```
txtData.Text = txtData.Text + "Gauge Address: " + answer + CR
If status = 1 Then
    Delay (0.3)
End If
Next i

' gosub check_echo
GoSub check_abort

'Get TT Level
For i = 0 To numretries
    rectype = "ZTT"
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
        decode "TT", echo, 64, answer
    End If
    txtData.Text = txtData.Text + "TT Level: " + answer + CR
    TT = ZlBLevel(echo)
    If status = 1 Then
        Delay (0.3)
    End If
Next i
' gosub check_echo
GoSub check_abort

'Get Displacer Weight DW
For i = 0 To numretries
    rectype = "ZDW"
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
        decode "DW", echo, 64, answer
    End If
    txtData.Text = txtData.Text + "Displacer Wt. (DW): " + answer + CR
    If status = 1 Then
        Delay (0.3)
    End If
Next i
' gosub check_echo
GoSub check_abort

'Get Displacer Volume DV
For i = 0 To numretries
    rectype = "ZDV"
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
        decode "DV", echo, 64, answer
    End If
    txtData.Text = txtData.Text + "Displacer Vol. (DV): " + answer + CR
    If status = 1 Then
        Delay (0.3)
    End If
Next i
' gosub check_echo
GoSub check_abort

' Unlock the gauge
For i = 0 To numretries
    rectype = "ZUN"
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
        decode "UN", echo, 64, answer
    End If
    If Instr(answer, "&") Then
        txtData.Text = txtData.Text + "Gauge Unlocked..." + Time$ + CR
    End If
    If status = 1 Then
        Delay (0.3)
    End If
Next i
```

```
End If
Next i

' gosub check_echo
GoSub check_abort

'GoSub get_a_level
GetALevel

' Get current reference level
For i = 0 To numretries
    rectype = "ZRL"
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
        decode "RL", echo, 64, answer
    End If
    txtData.Text = txtData.Text + "Ref Level: " + answer + CR
    RL = CDBl(answer)
    If status = 1 Then
        Delay (0.3)
    End If
Next i

' gosub check_echo
GoSub check_abort

GoSub check_abort

' Set scan direction
rectype = "ZSD=D"

'Set it
For i = 0 To numretries
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
        decode "SD", echo, 1, answer
    End If
    txtData.Text = txtData.Text + "Scan Direction: " + answer + CR
    sd = answer
    If status = 1 Then
        Delay (0.3)
    End If
Next i

' Record remaining parameters.
txtData.Text = txtData.Text & "Start Level: " & txtDensityStart.Text & CR

' gosub check_echo
GoSub check_abort

' Send to interface 1 (I1)
For i = 0 To numretries
    rectype = "ZI1"
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
        decode "I1", echo, 64, answer
    End If
    If InStr(answer, "&") Then
        txtData.Text = txtData.Text + "Sending to I1..." + Time$ + CR
    End If
    If status = 1 Then
        Delay (0.3)
    End If
```


HNF-1569
Revision 1

```
Next i

txtStatus.Text = "Waiting for Displacer to reach liquid level..."
'***** (wait for displacer to get to I1) *****
i = 0
'GoSub get_a_level
GetALevel
Delay (scanRate / 1000)
Do
    GoSub check_abort
    'GoSub get_a_level
    GetALevel
    GoSub check_errors
    Delay (1)
    If InStr(frmMain.txtLevelDisplay.Text, "INN") Then
        Exit Do
    End If
    DoEvents
Loop
'***** END WAITING FOR DISPLACER TO REACH WASTE *****

txtData.Text = txtData.Text + "Surface Level Reached..." + Time$ + CR
txtStatus.Text = "Surface Level Reached..."

' Get a single-point measurement
GetALevel
Dim oldLevel1$
oldLevel1 = Level(1)
Level(1) = Str(CL - 12)
Dim numLevels As Integer
numLevels = 1
txtStatus.Text = "Getting a single-point density measurement..."
txtData.Text = txtData.Text & "Obtaining single-point measurement " & Time$ & vbNewLine
GoSub DoDensity
Level(1) = oldLevel1
txtData.Text = txtData.Text & "Obtaining sediment level " & Time$ & vbNewLine

' Calculate wire tension
txtStatus.Text = "Calculating wire tension..."
Dim nSpG As Double
Dim DV As Double
Dim DW As Double
Dim BuoyantForce As Double
Dim WireTension As Double
Dim RecWireTension As String
ciucm "ZDV", echo, status
decode "DV", echo, 32, answer
DV = CDBl(answer)
ciucm "ZDW", echo, status
decode "DW", echo, 32, answer
DW = CDBl(answer)
nSpG = CDBl(SpG(1))

'note, dimensions are in grams and cm
BuoyantForce = nSpG * DV

WireTension = DW - BuoyantForce
LastWireTension = WireTension

RecWireTension = Str(WireTension - 25)

txtStatus.Text = "Setting wire tension S2=" & RecWireTension

' Change S2 to RecWireTension
EnterPassword (i)

If i = 1 Then
    AbortDensity = 1
    GoSub GetOut
End If
```

HNF-1569
Revision 1

```
'set the number
For i = 0 To numretries
    rectype = "ZS2=" + FormatTensionString(RecWireTension, Error)
    If Error = True Then
        MsgBox "Had trouble entering wire tension. Exiting..."
        GoSub GetOut
    End If
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
    End If
    If status = 1 Then
        Delay (0.3)
    End If
Next i

'exit from w2...
For i = 0 To numretries
    rectype = "ZEX"
    ciucom rectype, echo, status
    decode "EX", echo, 32, answer
    txtStatus.Text = answer
    If Len(echo) > 3 Then
        i = numretries
    End If
    If status = 1 Then
        Delay (0.3)
    End If
Next i

'wait until gauge reinitializes
txtStatus.Text = "Waiting for gauge to reinitialize..."
'Delay (10) ' allow TG to start
If TestMode = 0 Then
Do
    GoSub check_abort
    'GoSub get_a_level
    GetALevel s1B, s2B
    Delay (scanRate / 1000)
    'ciucom "Z2B", echo, status
    'decode "2B", echo, 32, answer
    If InStr(s2B, "TG") <> 0 Then
        Exit Do
    End If
    DoEvents
Loop
End If

'wait for test gauge to complete
txtStatus.Text = "Waiting for Test Gauge to Complete..."
Do
    GoSub check_abort
    'GoSub get_a_level
    GetALevel s1B, s2B
    Delay (scanRate / 1000)
    'ciucom "Z2B", echo, status
    'decode "2B", echo, 32, answer
    If InStr(s2B, "TG") = 0 Then
        Exit Do
    End If
    DoEvents
Loop
txtStatus.Text = "Test Gauge Complete. Unlocking..."

'unlock gauge
For i = 0 To numretries
    rectype = "ZUN"
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
    End If
Next i
```

HNF-1569
Revision 1

```
End If
If status = 1 Then
    Delay (0.3)
End If
Next i

'go to I2
For i = 0 To numretries
    rectype = "ZI2"
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
    End If
    If status = 1 Then
        Delay (0.3)
    End If
Next i

'wait to get there
txtStatus.Text = "Getting Sediment level (at I2)..."

' Check weights and measures
ciucom "ZIB", echo, status
If InStr(echo, "###") <> 0 Then
    m = "Weights and measures (WM) not set properly. Contact a service technician."
    MsgBox m, vbOKOnly + vbCritical, "Gauge Error"
    GoSub GetOut
End If

Do
    Delay (scanRate / 1000)
    'GoSub get_a_level
    GetALevel
    ' gosub check_echo
    GoSub check_abort
    GoSub check_errors
    If InStr(frmMain.txtLevelDisplay.Text, "INN") Then
        txtStatus.Text = "Sediment Reached...."
        Exit Do
    End If
    DoEvents
Loop

' Record Sediment level
GetALevel
LastSedLevel = Str(CL)
txtData.Text = txtData.Text & "Sediment Level = " & Str(CL) & " " & Time$ & vbNewLine
txtSed.Text = Str(CL)

' Return to I1
ciucom "ZI1", echo, status

' Make sure start level is 12 inches above sediment
If CDbl(txtDensityStart.Text) <= CDbl(txtSed.Text) + 12 Then
    MsgBox "Start Level is too low. Aborting!"
    AbortDensity = 1
    GoSub check_abort
End If

'Calculate Density Levels
Dim Distance As Double
Dim Delta As Integer
Dim tmpLevel As String

' Calculate number of levels to get and fill array
If mnuOptionsIPsed.checked = True Then
    j = 1

    tmpLevel = txtDensityStart.Text
    Debug.Print "tmpLevel at start= "; tmpLevel
```

HNF-1569
Revision 1

```

Level(j) = txtDensityStart.Text
Debug.Print "Level("; j; ")= "; Level(j)

Do
    Level(j + 1) = Str(CDbl(tmpLevel) - CDbl(txtInterval.Text))
    Debug.Print "Level("; j + 1; ")= "; Level(j + 1)

    If CDbl(Level(j + 1)) <= CDbl(txtSed.Text) Then
        ' get last measurement just above sediment
        ' but not if we already did.
        If CDbl(Level(j)) > CDbl(txtSed.Text) + 2 Then
            Level(j + 1) = Str((CDbl(txtSed.Text) + 2))
            Debug.Print "Last one= "; Str((CDbl(txtSed.Text) + 2))
            Debug.Print "NewLevel("; j + 1; ")= "; Level(j + 1)
            End If
            numLevels = j + 1
            Exit Do
        End If
        tmpLevel = Level(j + 1)
        numLevels = j + 1 ' makes sure that the "exit do" level is
                          ' not used.
        j = j + 1
    Loop
    Debug.Print "numLevels total= "; numLevels
    Else ' going to bottom

    j = 1

    tmpLevel = txtDensityStart.Text
    Debug.Print "tmpLevel at start= "; tmpLevel
    Level(j) = txtDensityStart.Text
    Debug.Print "Level("; j; ")= "; Level(j)

    Do
        Level(j + 1) = Str(CDbl(tmpLevel) - CDbl(txtInterval.Text))
        Debug.Print "Level("; j + 1; ")= "; Level(j + 1)

        If CDbl(Level(j + 1)) <= 0 Then
            ' get last measurement just above bottom
            ' but not if we already did.
            If CDbl(Level(j)) > 2 Then
                Level(j + 1) = Str(2)
                Debug.Print "Last one= "; 2
                Debug.Print "NewLevel("; j + 1; ")= "; Level(j + 1)
                End If
                numLevels = j + 1
                Exit Do
            End If
            tmpLevel = Level(j + 1)
            numLevels = j + 1 ' makes sure that the "exit do" level is
                              ' not used.
            j = j + 1
        Loop
        Debug.Print "numLevels total= "; numLevels
    End If

    txtData.Text = txtData.Text & "No. of Samples to be taken: " & Str(numLevels) & " " & CR & CR

    ' Make sure we get to I1
    txtStatus.Text = "Going to I1 while preparing profile acquisition..."
    '***** (wait for displacer to get to I1) *****
    i = 0
    GetALevel
    Delay (scanRate / 1000)
    Do
        GoSub check_abort
        GetALevel
        GoSub check_errors
        Delay (1)
        If InStr(frmMain.txtLevelDisplay.Text, "INN") Then

```

HNF-1569
Revision 1

```

Exit Do
End If
DoEvents
Loop
***** END WAITING FOR DISPLACER TO REACH WASTE *****

' gosub check_echo
GoSub check_abort
txtStatus.Text = "Getting density profile..."
txtData.Text = txtData.Text & "Obtaining density profile " & Time$ & vbNewLine
GoSub DoDensity
GoTo MakeFiles

DoDensity:
'!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
'!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
'SOME DEBUG STUFF, NOT FOR PRODUCTION
'Load variables with dummy values
'txtStatus.text = "Getting Densities..."
'DEMO MODE
If TestMode = 1 Then
pBar.Min = 0
pBar.Max = 100
pBar.Value = pBar.Min
pBar.Visible = True
'Set up a header for the data in output window..
txtData.Text = txtData.Text + "Level" + " " + "Density" + CR
txtData.Text = txtData.Text + "===== " + CR
' Set up a temporary array of data
Dim tmpDen(1 To 200) As String
Dim tmpSpG(1 To 200) As String
Dim tmpDat(1 To 200) As Date
Dim tmpTim(1 To 200) As String
For i = 1 To numLevels
tmpDen(i) = "1035.00"
tmpSpG(i) = CDbl(tmpDen(i)) * 1000 / 1000000
tmpDat(i) = Date$
tmpTim(i) = Time$
Next i
' Simulate getting densities
For i = 1 To numLevels
Density(i) = tmpDen(i)
SpG(i) = tmpSpG(i)
GotADensity = 1
Debug.Print "Density("; i; ")= "; Density(i)
Debug.Print "Level("; i; ")= "; Level(i)
theDate(i) = tmpDat(i)
Delay (0.2)
theTime(i) = tmpTim(i)
txtData.Text = txtData.Text & Format(Level(i), "###.##") & " " & _
    Format(Density(i), "###.##") & " (" & _
    Format(Str(SpG(i)), "#.####") & ") " & Time$ & vbNewLine
pBar.Value = i * (100 / numLevels)
Delay (2)
GoSub check_abort
GoSub check_errors
Next i

pBar.Visible = False
pBar.Value = pBar.Min

' Put the interval value back
If txtInterval.Text = "" Then
    txtInterval.Text = Interval
End If

'DEMO MODE JUMP
GoTo Skip
End If ' DEMO MODE
'!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
'!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

```

HNF-1569
Revision 1

```
'Set up a header for the data in output window..
txtData.Text = txtData.Text + "Level" + "    " + "Density" + CR
txtData.Text = txtData.Text + "===== " + CR
```

```
'** THIS STARTS THE TANK INTERFACE PROFILE SCAN ****
pBar.Min = 0
pBar.Max = 100
pBar.Value = pBar.Min
pBar.Visible = True
```

```
Dim L%
For L = 1 To numLevels
```

```
'SET START AND STOP LEVEL (=)
' Set Start Level
For i = 0 To numretries
    rectype = "ZDK=" & Format$(Level(L), "00000.00")
    Debug.Print "Set DK="; Format$(Level(L), "00000.00")
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
    End If
    If status = 1 Then
        Delay (0.3)
    End If
Next i
```

```
' gosub check_echo
GoSub check_abort
```

```
' Set Stop Level
For i = 0 To numretries
    ' Make sure txtDensityLow.Text is in correct format.
    rectype = "ZDN=" & Format$(Level(L), "00000.00")
    Debug.Print "Set DN="; Format$(Level(L), "00000.00")
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
    End If
    If status = 1 Then
        Delay (0.3)
    End If
Next i
' gosub check_echo
GoSub check_abort
```

```
'START THE IP MEASUREMENT FOR THE GIVEN LEVEL
For i = 0 To numretries
    rectype = "ZIP"
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
    End If
    If status = 1 Then
        Delay (0.3)
    End If
Next i
```

```
' gosub check_echo
GoSub check_abort
```

```
'SET UP A DO LOOP UNTIL IP MEASUREMENT IS COMPLETE
```

```
Do
    'GoSub get_a_level
    GetAlevel
```

HNF-1569
Revision 1

```

'Check if item SC indicates scan complete
'Get SC...
For i = 0 To numretries
    rectype = "ZSC"
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
        decode "SC", echo, 64, answer
    End If
    If status = 1 Then
        Delay (0.3)
    End If
Next i
GoSub check_abort

'Decode SC
'If no 9's in the CIU response, we're done with this point
' In production, might want to check instr(2B, "BT")
If Instr(answer, "9999") = 0 Then
    ciucom "ZSC", echo, status
    decode "SCD", echo, 32, answer
    Density(L) = answer
    theTime(L) = Time$
    theDate(L) = Date$
    Level(L) = Str(CL) ' Record the actual level
    SpG(L) = CDBl(Density(L)) * 1000 / 1000000
    txtData.Text = txtData.Text & Format(Level(L), "###.##") & _
        " " & Format(Density(L), "###.##") & " (" & _
        Format(Str(SpG(L)), "#.###") & ") " & Time$ & vbNewLine
    pBar.Value = L * (100 / numLevels)
    GotADensity = 1
    Exit Do
Else
    Delay (scanRate / 1000)
End If
GoSub check_abort
GoSub check_errors ' will look for fault
DoEvents

Loop
'Got a data point....
DoEvents
Next L
'*** THIS ENDS THE TANK PROFILE SCAN *****
pBar.Visible = False
pBar.Value = pBar.Min

' Put the interval value back
If txtInterval.Text = "" Then
    txtInterval.Text = Interval
End If

'!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
'!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Skip:
'!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
'!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Return

MakeFiles:
txtStatus.Text = "Densities Obtained, Processing Data...."
txtData.Text = txtData.Text + CR

'Calculate average density.
For i = 1 To numLevels
    avgden = avgden + CDBl(Density(i))
    DoEvents
Next i
If numLevels <> 0 Then
    avgden = avgden / numLevels
Else

```

HNF-1569

Revision 1

[illegible]

HNF-1569
Revision 1

Close #F

txtStatus.Text = "Density Profile Complete..."

```
message = "Density LOG located in file: " + DensityFilename + CR
message = message + "Density data filename is: " + tmpCDF + CR
message = message + CR + "View/Print LOG file?"
Reply = MsgBox(message, vbYesNo + vbInformation, "View Log File?")
```

```
If Reply = 6 Then
    'Debug.Print Dir(DensityFilename)
    If Dir(DensityFilename) = "" Then
        MsgBox "Density Log File has not yet been created."
        GoSub GetOut
    End If
    Shell "C:\WINDOWS\notepad.exe " & DensityFilename, vbNormalFocus
    GettingDensities = 0

End If
GoSub GetOut
```

check_abort:

```
If AbortDensity = 1 Then
    Debug.Print "AbortDensity was 1"
    txtData.Text = txtData.Text + "Full Auto IP Scan" + CR + " --ABORTED!" + CR
    If GotADensity = 1 Then
        GoTo MakeFiles ' Save whatever data we got
    Else
        GoSub GetOut
    End If
Else
    Return
End If
```

check_errors:

```
ciucum "ZES", echo, status
decode "ES", echo, 64, answer
answer = Trim(answer)
If answer <> "0000" Then
    m = GetSPUErrormessage(answer)
    m = m + CR
    m = m + "Aborting..."
    MsgBox m, vbOKOnly + vbCritical
    AbortDensity = 1
    ' Check for FL in line 2, and reset if necessary
    ciucum "Z2B", echo, status
    If InStr(echo, "FL") <> 0 Then
        ciucum "ZRS", echo, status
        txtData.Text = txtData.Text & "Resetting gauge. Displacer has stopped." & vbNewLine
        m = "Resetting the gauge due to the nature of this SPU Error"
        MsgBox m, vbOKOnly + vbCritical
    End If
    GoSub check_abort
End If
ciucum "ZEP", echo, status
decode "EP", echo, 64, answer
answer = Trim(answer)
If InStr(echo, "!0") <> 0 Then
    MsgBox "Aborting!", vbOKOnly + vbCritical
    AbortDensity = 1
    GoSub check_abort
End If
Return
```

Errhandler:

```
MsgBox "Error encountered getting densities..."
AbortDensity = 1
GoSub check_abort
```

HNF-1569
Revision 1

```

GetOut:
' Save activity data
ciucom "ZPN", echo, status
decode "PN", echo, 32, answer
Debug.Print "ACTIVITY LOG PN= "; answer
F = FreeFile
Open App.Path & "\DENSITY\ACTIVITY.LOG" For Append As #F
Print #F, "=====
Print #F, "          Run Number: "; Format(Str(UniqueNum), "000")
Print #F, "Gauge SN: "; answer
answer = App.Major & "." & App.Minor & "." & App.Revision
Print #F, "Program Version: "; answer
Print #F, "Density Profile Start:", Date$, StartTime
Print #F, "Sediment Level: ", Format(LastSedLevel, "000.00")
Print #F, "Wire Tension used: "; Format(LastWireTension, "000.00")
Print #F, "Acquisition Method: WRGS Full Auto IP Scan"
If mnuOptionsIPSed.checked = True Then
Print #F, "          Scan Type: IP to Sediment"
Else
Print #F, "          Scan Type: IP to Bottom"
End If
If TestMode = 1 Then
Print #F, "          DEMO MODE RUN, SIMULATED DATA"
End If
Print #F, "Date          Time          Level          SpG"
Print #F, "=====
For i = 1 To numLevels
Print #F, theDate(i), theTime(i), Format(Level(i), "000.00"), Format(Str(SpG(i)), "0.0000")
DoEvents
Next i
If AbortDensity = 1 Then
Print #F, ">>>ABORTED<<<", Date$, Time$
AbortDensity = 0
End If
Print #F, "^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^"
Close #F
DisabledLevelScan = 0
cmdGetDensity.MousePointer = 0
cmdGetDensity.Enabled = True
cmdClose.Enabled = True
cmdAbortDensity.Enabled = False
cmdClose.MousePointer = 0
GettingDensities = 0
frmMain.cmdRecover.Enabled = True
frmMain.cmdGetCurrentLevel.Enabled = True
frmMain.cmdOpEnraf.Enabled = True
frmMain.cmdDensity.Enabled = True
frmMain.cmdLog.Enabled = True
frmMain.cmdFTF.Enabled = True
frmMain.mnuAction.Enabled = True
frmMain.mnuConnect.Enabled = True
frmMain.mnuSetup.Enabled = True
txtStatus.Text = ""
frmMain.txtLevelDisplay.Text = ""
frmMain.txtDisplay2.Text = ""
txtDensityStart.Text = OldStart
txtSed.Text = OldSed
txtInterval.Text = OldInterval

'start scans
frmMain.cmdGetCurrentLevel_Click

```

End Sub

```

Private Sub cmdGetSedimentLevel_Click()
' *****
' This procedure sends the displacer to I2, to get a sludge
' or sediment level.
'

```

HNF-1569
Revision 1

```
' INPUTS: wire tension from the Buoyancy Calculator
' OUTPUTS: LastSedLevel, txtSed.txt, LastSedLevelDate
'          LastSedLevelTime
' CONDITIONS: Checks for proper WM setting
'
*****

On Error Resume Next
Dim S2$, echo$, answer$, m$, numretries%, i%
Dim LevDim$, Error As Boolean, rectype$
Dim CR
Dim SedLog$
CR = Chr(13) + Chr(10)
m = "Warning! You are about to DIP the displacer." + CR
m = m + "Do you want to continue?"
i = MsgBox(m, vbYesNo + vbExclamation, "DIP Displacer?")
If i <> 6 Then
    GoSub GetOut
End If

txtStatus.Text = "...Stand by..."
GettingSediment = 1
AbortDensity = 0
DisabledLevelScan = 0

txtData.Text = txtData.Text + "Sediment Level Check" + CR
txtData.Text = txtData.Text + Time$ + " " + Date$ + CR
txtData.Text = txtData.Text + "=====" + CR

cmdGetDensity.Enabled = False
cmdClose.Enabled = False
cmdAbortDensity.Enabled = True

'Ensure level scanning is disabled.
If frmMain.tmrLevel.Enabled = True Then
    frmMain.cmdDisconnect_Click
    DisabledLevelScan = 1
End If

'Initialize communications
If gaugeChecked = 0 Then
    checkGageAddress ta, ja
End If

If ja = 0 Then
    AbortDensity = 1
    GoSub check_abort
Else
    GetTankDimensions
End If

numretries = 3

'set wire tension
m = "Enter Wire Tension (in grams) or click OK" + CR
m = m + "to accept default..."
S2 = MyInputBox(m, "Input Wire Tension", "50")
If S2 = "" Then
    m = "No wire tension entered." + CR
    m = m + "Aborting..."
    MsgBox m
    AbortDensity = 1
    GoSub check_abort
End If
LastWireTension = S2

EnterPassword (i)

If i = 1 Then
```

HNF-1569
Revision 1

```
AbortDensity = 1
GoSub GetOut
End If

'set the number
For i = 0 To numretries
    rectype = "ZS2=" + FormatTensionString(S2, Error)
    If Error = True Then
        MsgBox "Had trouble entering wire tension. Exiting..."
        GoSub GetOut
    End If
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
        decode "S2", echo, 64, answer
    End If
    If status = 1 Then
        Delay (0.3)
    End If
Next i
txtData.Text = txtData.Text + "Setting S2 = " + answer + CR

'exit from w2...
For i = 0 To numretries
    rectype = "ZEX"
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
    End If
    If status = 1 Then
        Delay (0.3)
    End If
Next i

'wait until gauge reinitializes
txtStatus.Text = "Waiting for gauge to reinitialize..."
Do

    GoSub check_abort
    GetALevel s1B, s2B
    Delay (scanRate / 1000)
    'ciucom "Z2B", echo, status
    'decode "2B", echo, 32, answer
    If InStr(s2B, "TG") <> 0 Then
        Exit Do
    End If
    DoEvents
Loop

'wait for test gauge to complete
txtStatus.Text = "Waiting Test Gauge to Complete..."
Do
    GoSub check_abort
    'GoSub get_a_level
    GetALevel s1B, s2B
    Delay (scanRate / 1000)
    'ciucom "Z2B", echo, status
    'decode "2B", echo, 32, answer
    If InStr(s2B, "TG") = 0 Then
        Exit Do
    End If
    DoEvents
Loop
txtStatus.Text = "Test Gauge Complete. Unlocking..."

'unlock gauge
For i = 0 To numretries
    rectype = "ZUN"
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
```

```

        i = numretries
    End If
    If status = 1 Then
        Delay (0.3)
    End If
Next i

'go to I2
For i = 0 To numretries
    rectype = "ZI2"
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
    End If
    If status = 1 Then
        Delay (0.3)
    End If
Next i

'wait to get there
txtStatus.Text = "Getting Sediment level (at I2)..."

' Check weights and measures
ciucom "ZI2B", echo, status
If InStr(echo, "##") <> 0 Then
    m = "Weights and measures (WM) not set properly. Contact a service technician."
    MsgBox m, vbOKOnly + vbCritical, "Gauge Error"
    GoSub GetOut
End If

Do
    Delay (scanRate / 1000)
    GetALevel
    GoSub check_abort
    GoSub check_errors
    If InStr(frmMain.txtLevelDisplay.Text, "INN") Then
        txtStatus.Text = "Sediment Reached...."
        Exit Do
    End If
    DoEvents
Loop

'GoSub get_a_level
GetALevel
txtSed.Text = Str(CL)
SedLev = Str(CL)
SedLog = Str(CL)
LastSedLevel = Str(CL)
LastSedLevelDate = Date$
LastSedLevelTime = Time$
txtData.Text = txtData.Text + "Sediment level = " + Str(CL)
SaveToINI
m = "Would you like to return the displacer to the surface (I1)?"
i = MsgBox(m, vbYesNo + vbQuestion, "Bring displacer back up?")
If i = 6 Then
    'bring it to I1
    For i = 0 To numretries
        rectype = "ZI1"
        ciucom rectype, echo, status
        If Len(echo) > 3 Then
            i = numretries
        End If
        If status = 1 Then
            Delay (0.3)
        End If
    Next i
    ' If the level display is showing something, start
    ' level scans so that it is evident something is
    ' happening. Setting the following flag will make the
    ' level scans start when this procedure is exited.

```

```

    DisabledLevelScan = 1
End If

GoSub GetOut

check_abort:
'Debug.Print "check_abort..."
If AbortDensity = 1 Then
    txtData.Text = txtData.Text + "Sediment check" + CR + " --ABORTED!" + CR
    AbortDensity = 0
    SedLog = ">>ABORTED<<"
    GoSub GetOut
Else
    Return
End If

check_errors:
ciucum "ZES", echo, status
decode "ES", echo, 64, answer
answer = Trim(answer)
If answer <> "0000" Then
    m = GetSPUErrormessage(answer)
    m = m + CR
    m = m + "Aborting..."
    MsgBox m, vbOKOnly + vbCritical
    SedLog = ">>ABORTED<<"
    GoSub GetOut
End If
ciucum "ZEP", echo, status
decode "EP", echo, 64, answer
answer = Trim(answer)
If InStr(echo, "!0") <> 0 Then
    m = GetErrorMessage(Mid(echo, InStr(echo, "!0"), 4))
    m = m + CR
    m = m + "Aborting..."
    MsgBox m, vbOKOnly + vbCritical
    SedLog = ">>ABORTED<<"
    GoSub GetOut
End If
Return

GetOut:
ciucum "ZPN", echo, status
decode "PN", echo, 32, answer
' Create/append log file
Dim F
F = FreeFile
Open App.Path & "\DENSITY\ACTIVITY.LOG" For Append As #F
Print #F, "=====
Print #F, "                Sediment Level Acquisition"
Print #F, "Gauge SN: "; answer
answer = App.Major & "." & App.Minor & "." & App.Revision
Print #F, "Program Version: "; answer
Print #F, "Wire Tension used: "; Format(LastWireTension, "000.00")
Print #F, "Date           Time           Sediment Level"
Print #F, "=====
Print #F, LastSedLevelDate, LastSedLevelTime, SedLog
Print #F, "^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^"
Close #F

cmdGetDensity.Enabled = True
cmdAbortDensity.Enabled = False
cmdClose.Enabled = True
txtStatus.Text = ""
GettingSediment = 0
frmMain.txtLevelDisplay.Text = ""
frmMain.txtDisplay2.Text = ""

DisabledLevelScan = 0
frmMain.cmdGetCurrentLevel_Click

```

HNF-1569
Revision 1

```

Exit Sub
ErrorHandler:
    MsgBox "Error encountered during Sediment Check."
    GoSub GetOut
End Sub

Private Sub cmdPrintLog_Click()
Printer.Print txtData.Text
Printer.EndDoc
End Sub

Private Sub cmdTankProfile_Click()
*****
' This procedure is desined to obtain a tank density profile
' using the TP command. A valid start and sediment level
' must be entered.
' The procedure was shop tested on 12/28/98. The procedure
' successfully sent commands to the gauge, however, once the
' gauge received the TP command, the gauge would not execute the
' command. Troubleshooting would not reveal the problem.
' Parameters were found to be within required limits to effect
' the TP command.
' Since the problem could not be resolved, this procedure will
' not be implemented. The menu selection will be made non-
' visible prior to production. The code will remain for future
' reference/use.
*****

' Check if in DEMO MODE. This procedure not designed to work
' with DEMO MODE.
On Error Resume Next
If TestMode = 1 Then
    MsgBox "Sorry, you are in DEMO MODE. You must exit from DEMO MODE first."
    Exit Sub
End If

On Error GoTo Errhandler
Dim i%, j%
Dim echo As String 'response from CIU/gauge
Dim answer As String 'result from decode
Dim echoString(1 To 50) As String 'array of CIU/gauge responses
Dim recordCount As Integer 'count of echoStrings
Dim message As String 'used for msgboxes
Dim Reply As Integer 'used to capture msgbox responses
Dim fileIncrement As Integer
Dim Level(0 To 200) As String 'holds levels at which densities were obtained
Dim Density(0 To 200) As String 'hold densities
Dim sd As String 'scan direction

Dim rectype As String 'holds a CIU request string
Dim CR 'carraige return for text
CR = Chr(13) + Chr(10)
'dim CL As Double 'Current Level
Dim oldlevel As Double
Dim theDate(0 To 200) As String
Dim theTime(0 To 200) As String
Dim SpG(0 To 200) As Double
Dim LevelDim$, m$
Dim GotADensity% 'Flag for marking data acquisition
Dim WQ As Double
Dim DW As Double
Dim DV As Double
Dim WW As Double
Dim Y4 As Double
Dim WV As Double
Dim RF As Double
Dim IWV As Double
Dim StartTime As String

txtStatus.Text = "...Stand by..."

```

HNF-1569
Revision 1

```
AbortDensity = 0
DisabledLevelScan = 0
GotADensity = 0

'Clear any levels and densities
For i = 0 To 200
    Density(i) = "0"
    Level(i) = "0"
Next i

'clear status window
txtStatus.Text = ""

'Set form components
cmdGetDensity.MousePointer = 11
cmdGetDensity.Enabled = False
cmdClose.Enabled = False
cmdAbortDensity.Enabled = True
cmdClose.MousePointer = 11
txtData.Text = ""
frmMain.cmdRecover.Enabled = False
frmMain.cmdGetCurrentLevel.Enabled = False
frmMain.cmdOpEnraf.Enabled = False
frmMain.cmdDensity.Enabled = False
frmMain.cmdLog.Enabled = False
frmMain.cmdFTF.Enabled = False
frmMain.mnuAction.Enabled = False
frmMain.mnuConnect.Enabled = False
frmMain.mnuSetup.Enabled = False

'Ensure level scanning is disabled, set a flag to turn it back on when complete.
If frmMain.tmrLevel.Enabled = True Then
    frmMain.cmdDisconnect_Click
    DisabledLevelScan = 1
End If

'Set Flag to tell other program functions we're busy.
GettingDensities = 1

' Warn the operator. (Dipping the displacer in high level
' radioactive waste is no small thing).
m = "Warning! You are about to DIP the displacer." + CR
m = m + "Do you want to continue?"
i = MsgBox(m, vbYesNo + vbExclamation, "DIP Displacer?")
If i <> 6 Then
    AbortDensity = 1
    GoSub check_abort
End If

'=====
'VALIDATE INPUT DATA
'=====
txtStatus.Text = "Validating Input Data...."
If txtDensityStart.Text = "" Then
    MsgBox "You must enter a start level!", vbOKOnly + vbCritical, "Need Start Level!"
    AbortDensity = 1
    GoSub check_abort
Else
    SDLevel = txtDensityStart.Text
End If

If txtSed.Text = "" Then
    message = "You must enter a sediment level, or click GET" + CR
    message = message + "to have the gauge obtain it for you..."
    MsgBox message, vbOKOnly + vbCritical, "Need a Sediment Level..."
    AbortDensity = 1
    GoSub check_abort
Else
    SedLev = txtSed.Text
```


HNF-1569
Revision 1

End If

```
'=====
'  INPUTS SUCCESSFULLY VALIDATED AT THIS POINT
'=====
```

'Check whether we've previously proven that a gauge exists. If not,
'check for it now.

txtStatus.Text = "Establishing gauge communications...."

```
'If gaugeChecked = 0 Then
    checkGageAddress ta, ja
'End If
```

If ja = 0 Then ' a gauge doesn't exist.

```
    AbortDensity = 1
    GoSub check_abort
```

End If

GetTankDimensions

GetALevel

' Record Start Time for this scan

StartTime = Time\$

```
'=====
'  CREATE THE FILENAMES OF THE LOG AND TEXT FILES
'=====
```

' Create the file name for the log file. The log file is for viewing by
' humans, not other software. A comma, delimited file is created after the
' log file for viewing by other spreadsheet/database apps.

```
Dim F
Dim denDir As String
```

denDir = App.Path + "\DENSITY"

```
ciucom "ZPN", echo, status
decode "PN", echo, 6, answer
```

Dim UniqueNum\$

UniqueNum = "000"

```
DensityFilename = denDir & "\" & answer & UniqueNum & ".LOG"
'Make sure filename is unique
```

Do

```
    If Dir(DensityFilename) <> "" Then
        fileIncrement = fileIncrement + 1
        If fileIncrement < 10 Then
            UniqueNum = "00" & Trim(Str(fileIncrement))
        ElseIf fileIncrement < 100 Then
            UniqueNum = "0" & Trim(Str(fileIncrement))
        ElseIf fileIncrement < 1000 Then
            UniqueNum = Trim(Str(fileIncrement))
        Else
            answer = Right(answer, 4)
            UniqueNum = Trim(Str(fileIncrement))
        End If
        DensityFilename = denDir & "\" & answer & UniqueNum & ".LOG"
```

End If

DoEvents

```
Loop Until (Dir(DensityFilename) = "")
```

'Create a filename for the comma delimited file

'create the filename by simply changing extension of DensityFilename
'from LOG to DAT.

Dim tmpCDF As String

tmpCDF = DensityFilename

If InStr(tmpCDF, ".") Then 'change existing extension to "DAT"

tmpCDF = Mid(tmpCDF, 1, InStr(tmpCDF, ".")) + "DAT"

Else 'just add .DAT to end

tmpCDF = tmpCDF + ".DAT"

HNF-1569
Revision 1

```

End If

' create a .txt file to contain the output log data from the output log window
Dim DataLog As String
DataLog = DensityFilename
If InStr(DataLog, ".") Then 'change existing extension to "TXT"
    DataLog = Mid(DataLog, 1, InStr(DataLog, ".")) + "TXT"
Else 'just add .TXT to end
    DataLog = DataLog + ".TXT"
End If
'=====
' FILENAMES CREATED.....
'=====

recordCount = 0          ' initialize CIU reply count
numretries = 3

'=====
' Start the output log
'=====
txtData.Text = "<<< Density TP Scan Log Start >>>" + CR
txtData.Text = txtData.Text & StartTime & " " & Date$ & vbCrLf
txtData.Text = txtData.Text + "===== " + CR + CR

' Unlock the gauge
For i = 0 To numretries
    rectype = "ZUN"
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
        decode "UN", echo, 64, answer
    End If
    If InStr(answer, "&") Then
        txtData.Text = txtData.Text + "Gauge Unlocked..." + Time$ + CR
    End If
    If status = 1 Then
        Delay (0.3)
    End If
Next i

' gosub check_echo
GoSub check_abort

'GoSub get_a_level
GetALevel

txtStatus.Text = "Obtaining gauge parameters..."

txtData.Text = txtData.Text + CR + " Gauge Paramters" + CR
txtData.Text = txtData.Text + "-----" + CR

'Get/verify level dimension
LevelDim = GetLevelDimension
If UCase(LevelDim) <> "INCHES" Then
    'change it
    For i = 0 To numretries
        rectype = "ZLD=I"
        ciucom rectype, echo, status
        If Len(echo) > 3 Then
            i = numretries
            decode "LD", echo, 64, answer
            LevelDim = answer
        End If
        If status = 1 Then
            Delay (0.3)
        End If
    Next i
    LevelDim = GetLevelDimension ' Reload changed LevelDim
End If

```

HNF-1569
Revision 1

```
txtData.Text = txtData.Text + "Level Dimension: " + UCase(LevelDim) + CR
```

```
' Get and display the gauge address
```

```
For i = 0 To numretries
```

```
    rectype = "ZTA"
```

```
    ciucom rectype, echo, status
```

```
    If Len(echo) > 3 Then
```

```
        i = numretries
```

```
        decode "TA", echo, 64, answer
```

```
    End If
```

```
    txtData.Text = txtData.Text + "Gauge Address: " + answer + CR
```

```
    If status = 1 Then
```

```
        Delay (0.3)
```

```
    End If
```

```
Next i
```

```
' gosub check_echo
```

```
GoSub check_abort
```

```
'Get TT Level
```

```
For i = 0 To numretries
```

```
    rectype = "ZTT"
```

```
    ciucom rectype, echo, status
```

```
    If Len(echo) > 3 Then
```

```
        i = numretries
```

```
        decode "TT", echo, 64, answer
```

```
    End If
```

```
    txtData.Text = txtData.Text + "TT Level: " + answer + CR
```

```
    TT = Z1BLevel(echo)
```

```
    If status = 1 Then
```

```
        Delay (0.3)
```

```
    End If
```

```
Next i
```

```
' gosub check_echo
```

```
GoSub check_abort
```

```
'Get Displacer Weight DW
```

```
For i = 0 To numretries
```

```
    rectype = "ZDW"
```

```
    ciucom rectype, echo, status
```

```
    If Len(echo) > 3 Then
```

```
        i = numretries
```

```
        decode "DW", echo, 64, answer
```

```
    End If
```

```
    DW = CDbl(answer)
```

```
    txtData.Text = txtData.Text + "Displacer Wt. (DW): " + answer + CR
```

```
    If status = 1 Then
```

```
        Delay (0.3)
```

```
    End If
```

```
Next i
```

```
' gosub check_echo
```

```
GoSub check_abort
```

```
'Get Displacer Volume DV
```

```
For i = 0 To numretries
```

```
    rectype = "ZDV"
```

```
    ciucom rectype, echo, status
```

```
    If Len(echo) > 3 Then
```

```
        i = numretries
```

```
        decode "DV", echo, 64, answer
```

```
    End If
```

```
    DV = CDbl(answer)
```

```
    txtData.Text = txtData.Text + "Displacer Vol. (DV): " + answer + CR
```

```
    If status = 1 Then
```

```
        Delay (0.3)
```

```
    End If
```

```
Next i
```

```
' gosub check_echo
```

```
GoSub check_abort
```

```

' Get current reference level
For i = 0 To numretries
    rectype = "ZRL"
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
        decode "RL", echo, 64, answer
    End If
    txtData.Text = txtData.Text + "Ref Level: " + answer + CR
    RL = CDBl(answer)
    If status = 1 Then
        Delay (0.3)
    End If
Next i

' gosub check_echo
GoSub check_abort

Dim s1 As String
' Get current setpoint force
For i = 0 To numretries
    rectype = "ZS1"
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
        decode "S1", echo, 64, answer
    End If
    txtData.Text = txtData.Text + "I1 setpoint (S1): " + answer + CR
    s1 = answer
    If status = 1 Then
        Delay (0.3)
    End If
Next i

' gosub check_echo
GoSub check_abort

' Set scan direction
'Pick it.
If optUp.Value = True Then
    rectype = "ZSD=U"
Else
    rectype = "ZSD=D"
End If

' Set it
For i = 0 To numretries
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
        decode "SD", echo, 64, answer
    End If
    txtData.Text = txtData.Text + "Scan Direction: " + answer + CR
    sd = answer
    If status = 1 Then
        Delay (0.3)
    End If
Next i

' Record remaining parameters.
txtData.Text = txtData.Text & "Start Level: " & txtDensityStart.Text & CR
txtData.Text = txtData.Text & "Sediment Level: " & txtSed.Text & CR

' gosub check_echo
GoSub check_abort

' Send to interface 1 (I1)
For i = 0 To numretries
    rectype = "ZI1"
    ciucom rectype, echo, status

```

```

    If Len(echo) > 3 Then
        i = numretries
        decode "I1", echo, 64, answer
    End If
    If InStr(answer, "&") Then
        txtData.Text = txtData.Text + "Sending to I1..." + Time$ + CR
    End If
    If status = 1 Then
        Delay (0.3)
    End If
Next i

GetALevel
GoSub check_abort

If mnuOptionsIPSED.checked = True Then
    txtData.Text = txtData.Text & "Scan Type: IP to Sediment" & vbNewLine
Else
    txtData.Text = txtData.Text & "Scan Type: IP to Bottom" & vbNewLine
End If

pBar.Min = 0
pBar.Max = 100
pBar.Value = pBar.Min
pBar.Visible = True

Dim L%

'** THIS STARTS THE TANK DENSITY PROFILE SCAN ****
'For L = 1 To numLevels

' gosub check_echo

txtStatus.Text = "Waiting for Displacer to reach liquid level..."
'***** (wait for displacer to get to I1) *****
i = 0
'GoSub get_a_level
GetALevel
Delay (scanRate / 1000)
Do
    Delay (scanRate / 1000)
    GoSub check_abort
    GetALevel
    GoSub check_errors
    If InStr(frmMain.txtLevelDisplay.Text, "INN") Then
        Exit Do
    End If
    DoEvents
Loop
'***** END WAITING FOR DISPLACER TO REACH WASTE *****

txtData.Text = txtData.Text + "Surface Level Reached..." + Time$ + CR
txtStatus.Text = "Surface Level Reached..."

GoSub check_abort
'SET START AND STOP LEVEL (=)
' Enter protection level 2
ciucom "ZWI=ENRAF1", echo, status
If InStr(echo, "&") = 0 Then ' not entered W1
    MsgBox "Could not access Protection Level 1..." & vbNewLine & "Aborting..."
    ciucom "ZEX", echo, status
    AbortDensity = 1
    GoSub check_abort
End If
' Set Start Level
For i = 0 To numretries
    rectype = "ZDB=+" & Format$(txtDensityStart.Text, "00000.00")
    Debug.Print "Set DB=+"; Format$(txtDensityStart.Text, "00000.00")

```

HNF-1569
Revision 1

```

        txtData.Text = txtData.Text & "Setting Start Level = " & Format$(txtDensityStart.Text,
"00000.00") & vbNewLine
        ciucom rectype, echo, status
        If Len(echo) > 3 Then
            i = numretries
        End If
        If status = 1 Then
            Delay (0.3)
        End If
    Next i

' gosub check_echo
GoSub check_abort

' Set Stop Level
If mnuOptionsIPsed.checked = True Then
    For i = 0 To numretries
        ' Make sure txtDensityLow.Text is in correct format.
        rectype = "ZDZ=+" & Format$(Str(CDbl>LastSedLevel) + 9), "00000.00")
        Debug.Print "Set DZ=+"; Format$(Str(CDbl>LastSedLevel) + 9), "00000.00")
        txtData.Text = txtData.Text & "Setting Stop Level = " & Format$(Str(CDbl>LastSedLevel) + 9),
"00000.00") & vbNewLine
        ciucom rectype, echo, status
        If Len(echo) > 3 Then
            i = numretries
        End If
        If status = 1 Then
            Delay (0.3)
        End If
    Next i
    GoSub check_abort
Else
    ciucom "ZML", echo, status
    decode "ML", echo, 32, answer
    answer = Str(CDbl(answer) + 9)
    For i = 0 To numretries
        ' Stop is 0.2meters above last sed level.
        rectype = "ZDZ=+" & Format$(answer, "00000.00")
        Debug.Print "Set DZ=+" & Format$(answer, "00000.00")
        txtData.Text = txtData.Text & "Setting Stop Level = " & Format$(answer, "00000.00") &
vbNewLine
        ciucom rectype, echo, status
        If Len(echo) > 3 Then
            i = numretries
        End If
        If status = 1 Then
            Delay (0.3)
        End If
    Next i
    GoSub check_abort
End If

ciucom "ZEX", echo, status

txtStatus.Text = "Waiting for gauge to reinitialize..."

' Wait for TG to appear
Do
    Delay (scanRate / 1000)
    GetALevel s1B, s2B
    GoSub check_abort
    'ciucom "Z2B", echo, status
    If InStr(s2B, "TG") <> 0 Then
        Exit Do
    End If
    DoEvents
Loop

```

```
' Wait for TG to complete
Do
    Delay (scanRate / 1000)
    GetALevel s1B, s2B
    GoSub check_abort
    'ciucom "Z2B", echo, status
    If InStr(s2B, "TG") = 0 Then
        Exit Do
    End If
    DoEvents
Loop

txtStatus.Text = "Waiting for Displacer to reach liquid level..."
'***** (wait for displacer to get to Il) *****
i = 0
GetALevel
Delay (scanRate / 1000)
Do
    Delay (scanRate / 1000)
    GoSub check_abort
    GetALevel
    GoSub check_errors
    If InStr(frmMain.txtLevelDisplay.Text, "INN") Then
        Exit Do
    End If
    DoEvents
Loop
'***** END WAITING FOR DISPLACER TO REACH WASTE *****

'START THE TP MEASUREMENT FOR THE GIVEN LEVEL
txtStatus.Text = "Getting Densities..."
For i = 0 To numretries
    rectype = "ZTP"
    ciucom rectype, echo, status
    If Len(echo) > 3 Then
        i = numretries
    End If
    If status = 1 Then
        Delay (0.3)
    End If
    DoEvents
Next i

' gosub check_echo
GoSub check_abort

'Set up a header for the data in output window..
txtData.Text = txtData.Text + "Level" + " " + "Density" + CR
txtData.Text = txtData.Text + "===== " + CR

'SET UP A DO LOOP UNTIL IP MEASUREMENT IS COMPLETE
Dim nLoop%
nLoop = 0
Do
    'GoSub get_a_level
    GetALevel

    'Check if item R(nLoop) indicates scan complete
    For i = 0 To numretries
        rectype = "ZR" & Trim(Str(nLoop))
        Debug.Print "Checking R("; Str(nLoop); ")"
        ciucom rectype, echo, status
        If Len(echo) > 3 Then
            i = numretries
            decode "R" & Trim(Str(nLoop)), echo, 64, answer
        End If
        If status = 1 Then
            Delay (0.3)
        End If
        DoEvents
    Next i
```

HNF-1569
Revision 1

```

' gosub check_echo
GoSub check_abort

'Decode Rx
'If no 9's in the CIU response, we're done with this point
' In production, might want to check instr(2B, "BT")
If Instr(answer, "9999") = 0 Then
    'Save as last specific gravity measurement
    LastSpG = Str(CDbl(answer) * 1000 / 1000000)
    Density(nLoop) = answer
    theTime(nLoop) = Time$
    theDate(nLoop) = Date$
    ciucom "ZD" & Str(nLoop), echo, status
    decode "D" & Str(nLoop), echo, 32, answer
    Level(nLoop) = answer ' Record the actual level
    SpG(nLoop) = CDbl(Density(nLoop)) * 1000 / 1000000
    txtData.Text = txtData.Text + Level(nLoop) + " " + Density(nLoop) + " (" +
Str(SpG(nLoop)) + ") " + Time$ + CR
    pBar.Value = nLoop * (100 / 10)
    nLoop = nLoop + 1
Else
    Delay (scanRate / 1000)
End If
GoSub check_abort
GoSub check_errors
DoEvents
If nLoop > 9 Then Exit Do
Loop
'Got a data point....
DoEvents

'Next L
'*** THIS ENDS THE TANK PROFILE SCAN *****
pBar.Visible = False
pBar.Value = pBar.Min

MakeFiles:
txtStatus.Text = "Densities Obtained, Processing Data...."
txtData.Text = txtData.Text + CR

F = FreeFile

    Open DensityFilename For Output As F

    Print #F, Tab(2); "Enraf Control Panel Ver. "; Str(App.Major) + "." + Str(App.Minor) + "." +
Str(App.Revision) + CR
    Print #F, Tab(2); DensityFilename + " Date: " + Date$ + " Time: " + Time$ + CR
    Print #F, Tab(2); "Data File: "; tmpCDF & " Date: " & Date$ & " Time: " & Time$ & CR
    Print #F, Tab(2); "TP Scan Type"
    'INSERT PRINT OF OPERATOR NAME HERE
    Print #F, Tab(5); "Operator ID: " + CurrentUserName + CR
    Print #F, CR
    Print #F, Tab(5); "Level in", , "Density Kg/m3 (Sp.G.)"
    Print #F, Tab(5); "-----", , "-----"
    Print #F, CR
    For i = 0 To 9
        Print #F, Tab(5); Format$(CDbl(Level(i)), "000.00"), , , Format$(CDbl(Density(i)),
"000.00") + "(" + Str(SpG(i)) + ")"
        DoEvents
    Next i

    Print #F, CR
    Print #F, Tab(10); "Sediment Level Used: ", SedLev
    Close #F
    GettingDensities = 0

' show what file we created.
txtData.Text = txtData.Text + "Data File: " + Dir(DensityFilename, vbNormal) + CR

```


HNF-1569

Revision 1

```
End to CDF
FreeFile
Open tmpCDF For Output As F
' creates a comma delimited file containing
' date, time, level, specific gravity
' (the customer does not want density included in the file).
Print #F, "DENSITY PROFILE"
For i = 0 To 9
    Print #F, theDate(i); ", "; theTime(i); ", "; Level(i); ", "; Str(SpG(i))
    DoEvents
Next i
Print #F, "SEDIMENT LEVEL"
Print #F, LastSedLevelDate; ", "; LastSedLevelTime; ", "; LastSedLevel
Close #F
LastDBF = tmpCDF ' save the last file created.
```

[illegible]

```

'Send txtData to txt file
F = FreeFile
Open DataLog For Output As F
Print #F, txtData.Text
Close #F

txtStatus.Text = "Density Profile Complete..."

message = "Density LOG located in file: " + DensityFilename + CR
message = message + "Density data filename is: " + tmpCDF + CR
message = message + CR + "View/Print LOG file?"
Reply = MsgBox(message, vbYesNo + vbInformation, "View Log File?")

If Reply = 6 Then
    Debug.Print Dir(DensityFilename)
    If Dir(DensityFilename) = "" Then
        MsgBox "Density Log File has not yet been created."
        GoSub GetOut
    End If
    Shell "C:\WINDOWS\NOTEPAD.EXE " & DensityFilename, vbNormalFocus
    GettingDensities = 0
End If
GoSub GetOut

```

```
check_abort:
    If AbortDensity = 1 Then
        Debug.Print "AbortDensity was 1"
        txtData.Text = txtData.Text + "TP Density Scan" + CR + " --ABORTED!" + CR
        If GotADensity = 1 Then
            GoTo MakeFiles ' Save whatever data we got
        Else
            GoSub GetOut
        End If
    Else
        Return
    End If
```

```
check_errors:
    ciucom "ZES", echo, status
    decode "ES", echo, 64, answer
    answer = Trim(answer)
    If answer <> "0000" Then
        m = GetSPUErrorMessage(answer)
        m = m + CR
        m = m + "Aborting..."
        MsgBox m, vbOKOnly + vbCritical
```

```

        AbortDensity = 1
        GoSub check_abort
    End If
    ciucom "ZEP", echo, status
    decode "EP", echo, 64, answer
    answer = Trim(answer)
    If InStr(echo, "!0") <> 0 Then
        MsgBox "Aborting!", vbOKOnly + vbCritical
        AbortDensity = 1
        GoSub check_abort
    End If
    Return

Errhandler:
    MsgBox "Error encountered getting densities..."
    AbortDensity = 1
    GoSub check_abort

GetOut:
' Save activity data
ciucom "ZPN", echo, status
decode "PN", echo, 32, answer
Debug.Print "ACTIVITY LOG PN= "; answer
F = FreeFile
Open App.Path & "\DENSITY\ACTIVITY.LOG" For Append As #F
Print #F, "======"
Print #F, "          Run Number: "; Format(Str(UniqueNum), "000")
Print #F, "Gauge SN: "; answer
answer = App.Major & "." & App.Minor & "." & App.Revision
Print #F, "Program Version: "; answer
Print #F, "Density Profile Start:", Date$, StartTime
Print #F, "Sediment Level: ", Format(LastSedLevel, "000.00")
Print #F, "Wire Tension used: "; Format(LastWireTension, "000.00")
Print #F, "          Acquisition Method: TP Scan"
If mnuOptionsIPSED.checked = True Then
    Print #F, "          Scan Type: TP to Sediment"
Else
    Print #F, "          Scan Type: TP to Bottom"
End If
If TestMode = 1 Then
    Print #F, "          DEMO MODE RUN, SIMULATED DATA"
End If
Print #F, "Date          Time          Level          SpG"
Print #F, "======"
For i = 0 To 9
    Print #F, theDate(i), theTime(i), Format(Level(i), "000.00"), Format(Str(SpG(i)), "0.0000")
    DoEvents
Next i
If AbortDensity = 1 Then
    Print #F, ">>>ABORTED<<<", Date$, Time$
    AbortDensity = 0
End If
Print #F, "^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^"
Close #F

DisabledLevelScan = 0
cmdGetDensity.MousePointer = 0
cmdGetDensity.Enabled = True
cmdClose.Enabled = True
cmdAbortDensity.Enabled = False
cmdClose.MousePointer = 0
GettingDensities = 0
frmMain.cmdRecover.Enabled = True
frmMain.cmdGetCurrentLevel.Enabled = True
frmMain.cmdOpEnraf.Enabled = True
frmMain.cmdDensity.Enabled = True
frmMain.cmdLog.Enabled = True
frmMain.cmdFTF.Enabled = True
frmMain.mnuAction.Enabled = True
frmMain.mnuConnect.Enabled = True

```

HNF-1569
Revision 1

```

frmMain.mnuSetup.Enabled = True
txtStatus.Text = ""
frmMain.txtLevelDisplay.Text = ""
frmMain.txtDisplay2.Text = ""

'start scans
frmMain.cmdGetCurrentLevel_Click

Exit Sub
End Sub

Private Sub cmdUN_Click()
Dim m$, R$, echo$
If AbortDensity = 1 Then
    ciucom "ZUN", echo, status
    txtData.Text = txtData.Text & vbNewLine & "Manual command UN Sent..." & vbNewLine
    txtStatus.Text = "Sending UN command to gauge..."
ElseIf GettingDensities = 1 Then
    m = "Warning! This will abort density measurement." & vbNewLine
    m = m & "Do you wish to abort?"
    R$ = MsgBox(m, vbYesNo + vbExclamation, "Abort?")
    If R = 6 Then
        AbortDensity = 1
        ciucom "ZUN", echo, status
        DisabledLevelScan = 1
        txtData.Text = txtData.Text & vbNewLine & "Manual command UN Sent..." & vbNewLine
        txtStatus.Text = "Sending UN command to gauge..."
    End If
ElseIf GettingSediment = 1 Then
    m = "Warning! This will abort sediment check." & vbNewLine
    m = m & "Do you wish to abort?"
    R = MsgBox(m, vbYesNo + vbExclamation, "Abort?")
    If R = 6 Then
        AbortDensity = 1
        ciucom "ZUN", echo, status
        DisabledLevelScan = 1
        txtData.Text = txtData.Text & vbNewLine & "Manual command UN Sent..." & vbNewLine
        txtStatus.Text = "Sending UN command to gauge..."
    End If
ElseIf GettingDensities = 0 Then
    ciucom "ZUN", echo, status
    txtStatus.Text = "Sending UN command to gauge..."
    Delay (3)
    txtStatus.Text = ""
ElseIf GettingSediment = 0 Then
    ciucom "ZUN", echo, status
    txtStatus.Text = "Sending UN command to gauge..."
    Delay (3)
    txtStatus.Text = ""
End If
End Sub

Private Sub Command1_Click()
'This sub is for testing certain file functions. The button
'needs to be made visible and enabled in order to work. Don't
'forget to make it disabled and invisible before building.

Dim SL As String
SL = "0009.31"
Dim Level$, SpG$
Level = "00540.00"
SpG = "1.0533"
Dim i As Integer
Dim F
F = FreeFile
Open App.Path & "\DENSITY\ACTIVITY.LOG" For Append As #F
Print #F, "=====
Print #F, "Date           Time           Sediment Level"
Print #F, "=====

```

HNF-1569
Revision 1

```

Print #F, Date$, Time$, SL
Print #F, "^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^"
Close #F

F = FreeFile
Open App.Path & "\DENSITY\ACTIVITY.LOG" For Append As #F
Print #F, "=====
Print #F, "Date           Time           Level           SpG"
Print #F, "=====
For i = 1 To 5
Print #F, Date$, Time$, Level, SpG
Next i
Print #F, "^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^"
Close #F

Dim tmpCDF$
Dim theDate(1 To 20) As String
Dim theTime(1 To 20) As String
Dim Levels(1 To 20) As String
Dim SPGs(1 To 20) As String
Dim lev As Double
lev = 120.45
Dim numLevels As Integer
numLevels = 10

tmpCDF = App.Path & "\DENSITY\21520010.DAT"
For i = 1 To 10
theDate(i) = Date$
theTime(i) = Time$
Levels(i) = Str(lev)
lev = lev + 12
SPGs(i) = "1.0233"
Next i

LastSedLevel = "9.55"
LastSedLevelDate = Date$
LastSedLevelTime = Time$

'Send to CDF
F = FreeFile
Open tmpCDF For Output As F
' creates a comma delimited file containing
' date, time, level, specific gravity
' (the customer does not want density included in the file).
Print #F, "DENSITY PROFILE"
For i = 1 To numLevels
Print #F, theDate(i); ", "; theTime(i); ", "; Levels(i); ", "; Str(SPGs(i))
DoEvents
Next i
Print #F, "SEDIMENT LEVEL"
Print #F, LastSedLevelDate; ", "; LastSedLevelTime; ", "; LastSedLevel
Close #F
End Sub

Private Sub Command2_Click()
' THIS ROUTINE FOR TESTING CERTAIN LEVEL CALCULATING
' METHODS
Dim j%
Dim Distance As Double
Dim Delta As Integer
Dim SLevel As Double
Dim numLevels As Integer
Dim Level(1 To 200) As String

GoTo Skip
'Calculate Density Levels

' Calculate number of levels to get
If mnuOptionsIPsed.checked = True Then
SLevel = CDbl(Trim(txtDensityStart.Text))
Distance = SLevel - CDbl(txtSed.Text) ' the depth of the waste...

```

HNF-1569
Revision 1

```
If Distance < 12 Then
    MsgBox "Start Level is too low...Aborting.", vbOKOnly + vbCritical, "Error!"
    AbortDensity = 1
    Exit Sub
Else
    numLevels = Distance / CDBl(Trim(txtInterval.Text))
End If
Else ' going to bottom
    SLevel = CDBl(txtDensityStart.Text)
    Distance = SLevel
    If Distance < 12 Then
        MsgBox "Start Level is too low...Aborting.", vbOKOnly + vbCritical, "Error!"
        AbortDensity = 1
        Exit Sub
    Else
        numLevels = Distance / CDBl(Trim(txtInterval.Text))
    End If
End If

Debug.Print "SLevel= "; SLevel
Debug.Print "Distance= "; Distance
Debug.Print "numLevels= "; numLevels

'Level(1) = Str(SLevel)
'Debug.Print "Level(1)= "; Level(1)
For j = 0 To numLevels - 1
    Level(j + 1) = Str(SLevel - CDBl(Interval) * j)
    Debug.Print "SLevel-interval*j)= "; SLevel - CDBl(Interval) * j
    Debug.Print "Level("; j + 1; ")= "; Level(j + 1)
    DoEvents
Next j
Exit Sub
Skip:
j = 1
Dim tmpLevel As String
tmpLevel = txtDensityStart.Text
Debug.Print "tmpLevel at start= "; tmpLevel
Level(j) = txtDensityStart.Text
Debug.Print "Level("; j; ")= "; Level(j)

Do
    Level(j + 1) = Str(CDBl(tmpLevel) - CDBl(txtInterval.Text))
    Debug.Print "Level("; j + 1; ")= "; Level(j + 1)

    If CDBl(Level(j + 1)) <= CDBl(txtSed.Text) Then
        ' get last measurement just above sediment
        ' but not if we already did.
        If CDBl(Level(j)) > CDBl(txtSed.Text) + 2 Then
            Level(j + 1) = Str((CDBl(txtSed.Text) + 2))
            Debug.Print "Last one= "; Str((CDBl(txtSed.Text) + 2))
            Debug.Print "NewLevel("; j + 1; ")= "; Level(j + 1)
            End If
            numLevels = j + 1
            Exit Do
        End If
        tmpLevel = Level(j + 1)
        numLevels = j + 1 ' makes sure that the "exit do" level is
            ' not used.
        j = j + 1
    Loop
Debug.Print "numLevels total= "; numLevels
End Sub

Private Sub Form_Click()
Debug.Print "Left= "; Me.Left
Debug.Print "Top= "; Me.Top

End Sub

Private Sub Form_Load()
```

```
On Error Resume Next
GettingDensities = 0
'If frmOpEnraf.Visible = True Then
'frmDensity.Left = frmMain.Left
'frmDensity.Top = frmOpEnraf.Top + frmOpEnraf.Height
'Else
'frmDensity.ScaleMode = 1
'frmMain.ScaleMode = 3
frmDensity.Left = 2205
frmDensity.Top = 3000
'frmDensity.ScaleMode = 1
'frmMain.ScaleMode = 1
'End If

' Initialize up-down controls
If TT <> "" Then
    UpDown1.Max = TT
    UpDown2.Max = TT
    UpDown3.Max = TT
End If

' Initialize the animated gauge
Dim i%
For i = 0 To 58
    picMtr(i).Visible = False
Next i
picMtr(0).Visible = True
Debug.Print "picmtr(0).visible = "; picMtr(0).Visible
picMtr(0).ToolTipText = "Level Readings unavailable at this time...."
txtDensityStart.Text = SDLevel
txtSed.Text = SedLev
txtInterval.Text = Interval
End Sub

Private Sub Form_QueryUnload(Cancel As Integer, UnloadMode As Integer)
    ' The general procedure DoUnloadPreCheck handles the possible unload options
    ' for all three forms in this sample application.
    'DoUnloadPreCheck UnloadMode
    Select Case UnloadMode
        Case vbFormControlMenu
            If GettingDensities = 1 Or GettingSediment = 1 Then
                Cancel = True
            Else
                Exit Sub
            End If
        Case vbFormCode
            'End
        Case vbAppWindows
            'End
        Case vbAppTaskManager
            'End
        Case vbFormMDIForm
            'End
    End Select
End Sub

Private Sub mnuOptionsIPBottom_Click()
If mnuOptionsIPBottom.checked = False Then
    mnuOptionsIPBottom.checked = True
    mnuOptionsIPSed.checked = False
End If
End Sub

Private Sub mnuOptionsIPSed_Click()
If mnuOptionsIPSed.checked = False Then
    mnuOptionsIPSed.checked = True
    mnuOptionsIPBottom.checked = False
End If
End Sub
```

```
Private Sub mnuSelectFull_Click()  
If mnuSelectFull.checked = False Then  
    mnuSelectFull.checked = True  
    mnuSelectIP.checked = False  
    mnuSelectProgram.checked = False  
    mnuSelectTP.checked = False  
    cmdGetDensity.Enabled = False  
    cmdGetDensity.Visible = False  
    cmdGetDensity2.Enabled = False  
    cmdGetDensity2.Visible = False  
    cmdTankProfile.Enabled = False  
    cmdTankProfile.Visible = False  
    cmdGetDensityFull.Enabled = True  
    cmdGetDensityFull.Visible = True  
    mnuOptions.Enabled = True  
    chkSP.Enabled = False  
    txtInterval.Enabled = True  
    txtDensityStart.Text = WRSSStart  
End If  
End Sub  
  
Private Sub mnuSelectIP_Click()  
If mnuSelectIP.checked = False Then  
    mnuSelectIP.checked = True  
    mnuSelectTP.checked = False  
    mnuSelectFull.checked = False  
    mnuSelectProgram.checked = False  
    cmdGetDensity2.Enabled = False  
    cmdGetDensity2.Visible = False  
    cmdTankProfile.Enabled = False  
    cmdTankProfile.Visible = False  
    cmdGetDensityFull.Enabled = False  
    cmdGetDensityFull.Visible = False  
    cmdGetDensity.Enabled = True  
    cmdGetDensity.Visible = True  
    chkSP.Enabled = True  
    txtInterval.Enabled = True  
    mnuOptions.Enabled = True  
End If  
  
End Sub  
  
Private Sub mnuSelectProgram_Click()  
If mnuSelectProgram.checked = False Then  
    mnuSelectProgram.checked = True  
    mnuSelectIP.checked = False  
    mnuSelectTP.checked = False  
    mnuSelectFull.checked = False  
    cmdGetDensity.Enabled = False  
    cmdGetDensity.Visible = False  
    cmdTankProfile.Enabled = False  
    cmdTankProfile.Visible = False  
    cmdGetDensityFull.Enabled = False  
    cmdGetDensityFull.Visible = False  
    cmdGetDensity2.Enabled = True  
    cmdGetDensity2.Visible = True  
    chkSP.Enabled = True  
    txtInterval.Enabled = True  
    mnuOptions.Enabled = True  
End If  
End Sub  
  
Private Sub mnuSelectTP_Click()  
If mnuSelectTP.checked = False Then  
    mnuSelectTP.checked = True  
    mnuSelectIP.checked = False  
    mnuSelectProgram.checked = False  
    mnuSelectFull.checked = False  
    cmdGetDensity.Enabled = False  
    cmdGetDensity.Visible = False
```

```
cmdGetDensity2.Enabled = False
cmdGetDensity2.Visible = False
cmdGetDensityFull.Enabled = False
cmdGetDensityFull.Visible = False
cmdTankProfile.Enabled = True
cmdTankProfile.Visible = True
chkSP.Enabled = False
txtInterval.Enabled = False
mnuOptions.Enabled = True
```

End If

End Sub

```
Private Sub txtData_Change()
    txtData.SelStart = Len(txtData.Text)
End Sub
```

End frmDensity.frm

Begin frmFTF.frm

VERSION 5.00

Object = "{F9043C88-F6F2-101A-A3C9-08002B2F49FB}#1.1#0"; "COMDLG32.OCX"

Object = "{6B7E6392-850A-101B-AFC0-4210102A8DA7}#1.2#0"; "COMCTL32.OCX"

Begin VB.Form frmFTF

```
BorderStyle = 1 'Fixed Single
Caption = "File to Field"
ClientHeight = 5052
ClientLeft = 2172
ClientTop = 2760
ClientWidth = 6636
HelpContextID = 700
Icon = "frmFTF.frx":0000
LinkTopic = "Form1"
MaxButton = 0 'False
PaletteMode = 1 'UseZOrder
ScaleHeight = 5052
ScaleWidth = 6636
Tag = "File To Field Form"
WhatsThisHelp = -1 'True
```

Begin ComctlLib.ProgressBar pBar

```
Height = 252
Left = 4560
TabIndex = 23
Top = 4680
Width = 1932
_ExtentX = 3408
_ExtentY = 445
_Version = 327682
Appearance = 1
```

End

Begin VB.TextBox txtStatus

```
BackColor = &H80000000A&
Height = 288
Left = 120
TabIndex = 22
Top = 4680
Width = 4332
```

End

Begin VB.Frame Frame4

```
Caption = "Specify/Select/View Files"
Height = 1572
Left = 120
TabIndex = 11
Top = 120
Width = 6372
```

Begin VB.CommandButton cmdOpenLOG

```
Caption = "..."
BeginProperty Font
```



```
Name          = "MS Sans Serif"
Size           = 12
Charset        = 0
Weight         = 400
Underline      = 0 'False
Italic         = 0 'False
Strikethrough  = 0 'False
EndProperty
Height         = 252
Left           = 4080
TabIndex      = 26
Top            = 1080
Width          = 372
End
Begin VB.CommandButton cmdOpenRPL
Caption        = "..."
BeginProperty Font
Name           = "MS Sans Serif"
Size           = 12
Charset        = 0
Weight         = 400
Underline      = 0 'False
Italic         = 0 'False
Strikethrough  = 0 'False
EndProperty
Height         = 252
Left           = 4080
TabIndex      = 25
Top            = 720
Width          = 372
End
Begin VB.CommandButton cmdOpenRQS
Caption        = "..."
BeginProperty Font
Name           = "MS Sans Serif"
Size           = 12
Charset        = 0
Weight         = 400
Underline      = 0 'False
Italic         = 0 'False
Strikethrough  = 0 'False
EndProperty
Height         = 252
Left           = 4080
TabIndex      = 24
Top            = 360
Width          = 372
End
Begin VB.CommandButton cmdClose
Caption        = "Close"
Height         = 375
Left           = 5640
TabIndex      = 21
Tag           = "Close FTF"
Top            = 360
Width          = 615
End
Begin VB.CommandButton cmdViewLOG
Caption        = "View"
Height         = 255
Left           = 4560
TabIndex      = 20
Tag           = "View LOG File"
Top            = 1080
Width          = 855
End
Begin VB.CommandButton cmdViewRPL
Caption        = "View"
Height         = 255
Left           = 4560
TabIndex      = 19
```

HNF-1569
Revision 1

```

    Tag          = "View RPL File"
    Top          = 720
    Width        = 855
End
Begin VB.CommandButton cmdViewRQS
    Caption      = "View"
    Height       = 255
    Left         = 4560
    TabIndex     = 18
    Tag          = "View RQS File"
    Top          = 360
    Width        = 855
End
Begin VB.TextBox txtLOGFilename
    Height       = 285
    Left         = 1920
    TabIndex     = 17
    Top          = 1080
    Width        = 2175
End
Begin VB.TextBox txtRPLFilename
    Height       = 285
    Left         = 1920
    TabIndex     = 16
    Top          = 720
    Width        = 2175
End
Begin VB.TextBox txtRQSFilename
    Height       = 285
    Left         = 1920
    TabIndex     = 15
    Top          = 360
    Width        = 2175
End
Begin VB.Label Label5
    Alignment    = 1 'Right Justify
    Caption      = "Enter LOG File Name"
    Height       = 252
    Left         = 240
    TabIndex     = 14
    Top          = 1080
    Width        = 1572
End
Begin VB.Label Label4
    Alignment    = 1 'Right Justify
    Caption      = "Enter RPL File Name"
    Height       = 252
    Left         = 120
    TabIndex     = 13
    Top          = 720
    Width        = 1692
End
Begin VB.Label Label3
    Alignment    = 1 'Right Justify
    Caption      = "Enter RQS File Name"
    Height       = 252
    Left         = 120
    TabIndex     = 12
    Top          = 360
    Width        = 1692
End
End
Begin VB.Frame Frame3
    Caption      = "File Conversions"
    Height       = 1572
    Left         = 3600
    TabIndex     = 3
    Top          = 1800
    Width        = 2892
    Begin VB.CommandButton Command1
        Caption   = "&Cancel/Abort"
    End
End
```

HNF-1569
Revision 1

```

Enabled      = 0 'False
Height       = 255
Left         = 720
TabIndex     = 10
Tag          = "Cancel/Abort"
Top          = 1200
Width        = 1455
End
Begin VB.CommandButton cmdLOGtoRQS
Caption      = "LOG to RQS"
Enabled      = 0 'False
Height       = 255
Left         = 240
TabIndex     = 9
Tag          = "Convert LOG to RQS"
Top          = 600
Width        = 2415
End
Begin VB.CommandButton cmdRQStoLOG
Caption      = "RPL to LOG"
Enabled      = 0 'False
Height       = 255
Left         = 240
TabIndex     = 8
Tag          = "Convert RPL to LOG"
Top          = 360
Width        = 2415
End
End
Begin VB.Frame Frame2
Caption      = "Batch File Operations"
Height       = 1572
Left         = 120
TabIndex     = 2
Top          = 1800
Width        = 2892
Begin VB.CommandButton cmdCancel
Caption      = "&Cancel/Abort"
Height       = 255
Left         = 600
TabIndex     = 7
Tag          = "Cancel/Abort"
Top          = 1200
Width        = 1575
End
End
Begin VB.CommandButton cmdProgramGauge
Caption      = "Prog Gauge From LOG File"
Height       = 255
HelpContextID = 710
Left         = 240
TabIndex     = 6
Tag          = "Program Gauge from Log File"
Top          = 840
Width        = 2415
End
End
Begin VB.CommandButton cmdRQSSend
Caption      = "Test and SEND RQS File"
Enabled      = 0 'False
Height       = 255
Left         = 240
TabIndex     = 5
Tag          = "Test and Send RQS"
Top          = 600
Width        = 2415
End
End
Begin VB.CommandButton cmdRQSTest
Caption      = "Test RQS File"
Height       = 255
Left         = 240
TabIndex     = 4
Tag          = "Test RQS File"

```

HNF-1569
Revision 1

```

        Top           = 360
        Width         = 2415
    End
End
Begin VB.Frame Frame1
    Caption           = "Text Window"
    Height            = 1092
    Left              = 120
    TabIndex          = 0
    Top               = 3480
    Width             = 6372
    Begin VB.TextBox txtTWin
        Height         = 735
        HelpContextID   = 720
        Left           = 120
        Locked          = -1 'True
        MultiLine       = -1 'True
        ScrollBars      = 2 'Vertical
        TabIndex        = 1
        Tag             = "FTF Test Window"
        Top             = 240
        Width           = 6135
    End
End
Begin MSComDlg.CommonDialog CMDialog1
    Left      = 5760
    Top       = 120
    _ExtentX  = 699
    _ExtentY  = 699
    _Version  = 327681
End
Begin VB.Timer Timer1
    Left      = 480
    Top       = 0
End
End
Attribute VB_Name = "frmFTF"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
' =====
' The features presented on this form are not fully implemented.
' Therefore, the code is not fully annotated/commented.
' Implemented routines are as follows.
'
' cmdOpenLOG
' cmdOpenRQS
' cmdOpenRPL
' cmdProgramGauge
' cmdViewLOG
' cmdViewRQS
' cmdViewRPL
' cmdTestRQS
'
' =====

Dim rqsFileOld As String
Dim rplFileOld As String
Dim logFileOld As String

Dim rectypeold As String
Dim taold As String
Dim ciuAdresOld As String

Dim DefaultDir As String

Dim record(1 To 500) As String
Dim recnt%
Dim ftfCancel As Integer

```

```
Private Sub cmdCancel_Click()
    ftfCancel = 1
End Sub
```

```
Private Sub cmdClose_Click()
    rqsFile = rqsFileOld
    rplFile = rplFileOld
    logFile = logFileOld
    ftfCancel = 1
    frmFTF.Hide
End Sub
```

```
Private Sub cmdLOGtoRQS_Click()
'=====
' This procedure was originally written by Delft Instruments. Minor
' modifications were made by John Huber. This procedure is proprietary
' to Delft Instruments and may not be used or reproduced without permission.
'=====

    rqsFile = txtRQSFilename.Text
    rplFile = txtRPLFilename.Text
    logFile = txtRPLFilename.Text

    If rqsFile = "" Or logFile = "" Then
        MsgBox "You must first specify BOTH RQS and LOG Filenames!", vbOKOnly + vbCritical, "Need both filenames..."
    End If

    Exit Sub
End Sub
```

```
Dim F, Fl
F = FreeFile
Fl = FreeFile
Dim CR As Integer
CR = Chr(13) + Chr(10)

    logFile = UCase$(logFile)
    If InStr(logFile, ".") = 0 Then
        logFile = logFile + ".LOG"
    End If
```

```
    If Dir$(rqsFile) <> "" Then

        'SysMesg rqsFile + " already exists. Overwrite? (Selecting 'NO' will append)", "vbYNC",
        "vbQuery", Reply%
        Reply = MsgBox(rqsFile + " already exists. Overwrite? (Selecting 'NO' will append)",
        vbYesNoCancel + vbQuestion, "Overwrite?")
        Select Case Reply
            Case 6      ' overwrite
                Open rqsFile For Output As F
            Case 7      ' append
                Open rqsFile For Append As F
            Case 2      ' quit
                Exit Sub
        End Select
    Else
        Open rqsFile For Append As F
    End If

'    global.invoer global%, global$
'    IF global$ = "" THEN
'        CLOSE #2
'        EXIT SUB
'    END IF
'    prtCLS
```

HNF-1569
Revision 1

```

txtTWin.Text = " LOG_RQL conversion / v0.1 / " + logFile + " -> " + rqsFile + CR

foutNum% = 0 'aantal gevonden fouten
regelNum% = 1

Open logFile For Input As F1

Print #F, "' ECPWin LOG_RQS converter v1.0/ " + Date$ + " " + Time$ + "/" + ta + "]"
Print #F, ">" + ta + "ZW2=ENRAF2"
Print #F, "!PAUSE 2"

While Not EOF(F2)

    Line Input #1, regelString$

    regelString = LTrim$(regelString) 'verwijder leading spaties
    regelString = UCase$(regelString) 'maak alles upper case

    If Left$(regelString, 1) = "'" Then
        Print #F, regelString
    Else
        If Mid(regelString, 3, 1) <> "=" Then
            GoSub foutlogrqs
        Else
            If Mid(regelString, Len(regelString), 1) = "=" Or InStr(regelString, "!") Then
                'skip line
            Else
                txtTWin.Text = txtTWin.Text + Left$(">" + ta + "Z" + regelString, 40) + CR
                Print #F, ">" + ta + "Z" + regelString
            End If
        End If
    End If

    ftflogrqs1:
    Print kader.regelString;
    regelNum = regelNum + 1
    'Suspend
Wend
Close #F2

Print #F, "!PAUSE 2"
Print #F, ">" + ta + "ZEX"
Print #F, ">" + ta + "ZEX"
Print #F, 'End of file

Close #F

txtTWin.Text = txtTWin.Text + " LOG_RQS conversion / v0.1 / " + logFile + " -> " + rqsFile +
CR

If foutNum = 0 Then 'goed
    txtTWin.Text = txtTWin.Text + " no errors found: File conversion completed !"
Else 'fout
    txtTWin.Text = txtTWin.Text + Str(foutNum) + " error(s) found: File rejected !" + CR
End If
Exit Sub

foutlogrqs:
    foutNum = foutNum + 1
    txtStatus.Text = "Error detected !"
    Sleep (1000)
    GoTo ftflogrqs1

End Sub

Private Sub cmdOpenLOG_Click()
    On Error GoTo Errhandler
    CMDialog1.InitDir = DefaultDir

```

HNF-1569
Revision 1

```
    CMDialog1.Filter = "All Files (*.*)|*.r|RQS Files (*.rqs)|*.rqs|RPL Files (*.rpl)|*.rpl|LOG  
Files (*.log)|*.log"  
    CMDialog1.FilterIndex = 4  
    CMDialog1.ShowOpen  
    txtLOGFilename.Text = CMDialog1.filename  
    logFile = CMDialog1.filename  
    Debug.Print "CMDialog1.Filename= "; CMDialog1.filename  
    Dim sPath$, c$  
    sPath = logFile  
    Do While sPath <> ""  
        sPath = Left(sPath, Len(sPath) - 1)  
        Debug.Print "sPath= "; sPath  
        If Right(sPath, 1) = "\" Then  
            sPath = Left(sPath, Len(sPath) - 1)  
            Exit Do  
        End If  
        If KeyAscii = 13 Then Exit Do  
    Loop  
    Debug.Print "Final sPath= "; sPath  
    DefaultDir = sPath
```

```
Errhandler:  
    Exit Sub
```

End Sub

```
Private Sub cmdOpenRPL_Click()  
    On Error GoTo Errhandler  
    CMDialog1.InitDir = App.Path + "\FTF"  
    CMDialog1.Filter = "All Files (*.*)|*.r|RQS Files (*.rqs)|*.rqs|RPL Files (*.rpl)|*.rpl|LOG  
Files (*.log)|*.log"  
    CMDialog1.FilterIndex = 3  
    CMDialog1.ShowOpen  
    txtRPLFilename.Text = CMDialog1.filename  
    rplFile = CMDialog1.filename
```

```
Errhandler:  
    Exit Sub  
End Sub
```

```
Private Sub cmdOpenRQS_Click()  
    On Error GoTo Errhandler  
    CMDialog1.InitDir = App.Path + "\FTF"  
    CMDialog1.Filter = "All Files (*.*)|*.r|RQS Files (*.rqs)|*.rqs|RPL Files (*.rpl)|*.rpl|LOG  
Files (*.log)|*.log"  
    CMDialog1.FilterIndex = 2  
    CMDialog1.ShowOpen  
    txtRQSFilename.Text = CMDialog1.filename  
    rqsFile = CMDialog1.filename
```

```
Errhandler:  
    Exit Sub
```

End Sub

```
Private Sub cmdProgramGauge_Click()  
'=====
```

' This procedure was originally written by Delft Instruments. Minor
' modifications were made by John Huber. This procedure is proprietary
' to Delft Instruments and may not be used or reproduced without permission.
'=====

```
On Error GoTo ErrorHandler  
Dim PN As String  
Dim Pfilestring As String  
Dim Reply$, echo$  
  
If gettingScans = 1 Then  
    frmMain.cmdDisconnect_Click  
End If
```

HNF-1569
Revision 1

```
cmdRQSSend.MousePointer = 11
cmdRQSTest.MousePointer = 11
cmdRQStoLOG.MousePointer = 11
cmdLOGtoRQS.MousePointer = 11
cmdClose.MousePointer = 11

'logfile = txtRPLFilename.text
Pfilestring = txtLOGFilename.Text

If Pfilestring = "" Then
  'SysMesg "You must first specify a LOG File Name!", "vbOkay", "vbWarning", Reply%
  MsgBox "You must first specify a LOG File Name!", vbOKOnly + vbCritical, "Need Log filename..."

  GoSub GetOut
End If

Dim F
Dim itcnt As Integer
Dim gaugePN As String
Dim CR
CR = Chr(13) + Chr(10)
'Dim Reply As Integer
Dim Pgingen As Integer
Pgingen = 3
Dim foutString As String
Dim antwoord$
Dim echofoutString$

pBar.Min = 0
pBar.Value = pBar.Min
pBar.max = 100
pBar.Visible = True

'Program gauge from disk log file.
' tempString = kader.regel$
' prtCLS
' VIEW PRINT 5 TO 23

If gaugeChecked = 0 Then
  checkGageAddress ta, ja
End If
' controleer of er een meter aanwezig is op het adres
If ja = 0 Then
  'geenmeter
  MsgBox "Gauge Not Found!"
  GoSub GetOut
End If

'ciucom rectype + "PN", echo, status
ciucom "ZPN", echo, status
'Debug.Print echo
decode "PN", echo, 6, PN
'Debug.Print "Gauge PN= "; PN
fileString = instrument + "*.LOG"
CMDIALOG.DialogTitle = "Select LOG File to Send..."
CMDIALOG1.InitDir = logDir
CMDIALOG1.Filter = "All Files (*.*)|*.LOG Files (*.log)|*.log"
CMDIALOG1.FilterIndex = 2
CMDIALOG.ShowOpen
fileString = CMDIALOG.FileName
dirFile "Select .LOG File to be send : ", 1, logDir, fileString

If Pfilestring = "" Then
  GoSub GetOut
End If
```


HNF-1569
Revision 1

```
'IF LEN(DIR$(fileString)) = 0 THEN
' file.not.found " Logfile " + logDir, fileString
'ELSE

F = FreeFile

Open Pfilestring For Input As F

' lees inhoud file in array formulier
itcnt = 1
While Not EOF(F)
    Line Input #F, echoArray(itcnt)
    'Debug.Print echoArray(itcnt)
    If Left(echoArray(itcnt), 1) <> "'" Then 'skip indien commentaar
        itcnt = itcnt + 1
    End If
Wend
Close F
'test productie nummer

Dim k%
For k = 1 To itcnt
    If InStr(echoArray(k), "PN=") <> 0 Then
        echo = echoArray(k)
        decode "PN=", echo, 10, gaugePN
        Exit For
    End If
Next k
If PN <> gaugePN Then
    txtTWin.Text = "Production number from .LOG file : " + gaugePN + CR
    txtTWin.Text = txtTWin.Text + "Gauge Production number      : " + PN + CR
    Reply = MsgBox("Production number mismatch. Continue ?", vbYesNo + vbQuestion, "PN
Mismatch...")

    If Reply = 7 Then
        GoSub GetOut
    End If
End If

Dim result%
EnterPassword result
If result = 0 Then
    MsgBox "Password not accepted!"
    GoSub GetOut
End If
' programmeer nu de meter

Dim j%
Dim nCommFails%
For j = 1 To itcnt
    If ftfCancel = 1 Then
        ciucom "ZEX", echo, status
        txtStatus.Text = " **Gauge Programming Aborted!**"
        GoSub GetOut
    End If
    If Len(echoArray(j)) > 3 Then
        txtTWin.Text = txtTWin.Text + tempString + CR
        txtTWin.Text = txtTWin.Text + echoArray(j) + CR
        For k = 1 To 3 ' 3 tries
            ciucom "Z" + echoArray(j), antwoord, status
            If status = 0 And Len(antwoord) > 3 Then Exit For
            txtTWin.Text = txtTWin.Text & antwoord & " status= " & Str(status) & vbNewLine
        Delay (30 / 1000)
        Next k
        If status <> 0 Then
            txtTWin.Text = txtTWin.Text & "Comm failed on item " & Left(echoArray(j), 2) & vbNewLine
            nCommFails = nCommFails + 1
            If nCommFails >= 2 Then
                txtTWin.Text = txtTWin.Text & "Comm Status = " & status & vbNewLine
                txtStatus.Text = " ** COMMUNICATION FAIL, PROGRAMMING FAILED ** "
                Pfilestring = ""
            End If
        End If
    End If
Next j
```

```

nCommFails = 0
Delay (30 / 1000)
GoSub GetOut
End If
End If
txtTWin.Text = txtTWin.Text + CR + CR

txtTWin.Text = txtTWin.Text + antwoord + CR
If Right(antwoord, 1) <> "&" Then
  If Len(antwoord) < 4 Then
    foutString = "No gauge answer (" + Str(status) + ")"
  Else
    If InStr(antwoord, "!") Then
      txtTWin.Text = txtTWin.Text + "item " + Left(echoArray(j), 2) + CR
      echofoutString = Mid(antwoord, InStr(antwoord, "!"))
      Select Case echofoutString
        Case "!050"
          foutString = "Invalid length"
        Case "!051"
          foutString = "Unknown item"
        Case "!054"
          foutString = "No read access"
        Case "!055"
          foutString = "No write access"
        Case "!056"
          foutString = "Wrong protection level"
        Case "!057"
          foutString = "Access denied"
        Case Else
          foutString = "not accepted"
      End Select
    Else
      foutString = ""
    End If
  End If
  txtStatus.Text = " " + foutString + " !"
End If
End If
pBar.Value = (j / itcnt) * 100
Next j
ciucum "ZEX", echo, status

txtTWin.Text = txtTWin.Text + tempString + CR
txtStatus.Text = " Programming completed ! "
Delay (30 / 1000)
GoSub GetOut

Exit Sub

GetOut:
cmdRQSSend.MousePointer = 0
cmdRQSTest.MousePointer = 0
cmdRQStoLOG.MousePointer = 0
cmdLOGtoRQS.MousePointer = 0
cmdClose.MousePointer = 0
  pBar.Value = pBar.Min
  pBar.Visible = False

Exit Sub

ErrorHandler:
  MsgBox "Error encountered during logging"
  GoSub GetOut
End Sub

Private Sub cmdRQSSend_Click()
'=====
' This procedure was originally written by Delft Instruments. Minor
' modifications were made by John Huber. This procedure is proprietary
' to Delft Instruments and may not be used or reproduced without permission.
'=====

```

HNF-1569
Revision 1

```

'frmFTF.MousePointer = 11
  rqsFile = txtRQSFilename.Text
  rplFile = txtRPLFilename.Text

  If rqsFile = "" Or rplFile = "" Then
    'SysMsg "You must first specify BOTH RQS and RPL Filenames!", "vbOkay", "vbWarning", Reply%
    MsgBox "You must first specify BOTH RQS and RPL Filenames!", vbOKOnly + vbCritical, "Need both
filenames..."

    frmFTF.MousePointer = 0
    Exit Sub
  End If

  If Not InStr(UCase(rplFile)), ".") Then
    rplFile = rplFile + ".RPL"
  ElseIf InStr(rplFile, ".") And Not InStr(UCase(rplFile), "RPL") Then
    rplFile = Mid(rplFile, (Left(rplFile, 1)), ".") + "RPL"
  End If

cmdRQSTest_Click
ReadRQS

Dim teller As Integer
Dim comfail As Integer
Dim comretry As Integer
Dim tempString As String
'Dim Reply As Integer
Dim CR
CR = Chr(13) + Chr(10)
Dim pause!
Dim pauseString As String
Dim Posingen As Integer
Dim p As Integer
Dim TimerShouldBeReEnabled As Integer
Dim TimeInMinutes!
Dim TimeInSeconds!
Dim TimeInMilliseconds!
Dim k%
Dim echoNum%
Dim record(1 To 500)
Dim recordString(1 To 500)
Dim echoString(1 To 500)
Dim Reply%, echo$

  If frmMain.tmrLevel.Enabled = True Then
    frmMain.tmrLevel.Enabled = False
    TimerShouldBeReEnabled = 1
  Else
    If gaugeChecked = 0 Then
      checkGageAddress ta, ja
    End If

    If ja = 0 Then 'ja <= 0 means no gauge
      frmFTF.MousePointer = 0
      Exit Sub
    Else

ftfCancel = 0

rectypeold = rectype          ' save enkele variabelen
taold = ta
ciuAdresOld = ciuadres

rectype = ""                  ' init. deze variabelen
ta = ""
ciuadres = ""

teller = 0                    ' initialisatie tellers
                                ' teller aantal doorlopen cycli

```

HNF-1569
Revision 1

```
comfail = 0          ' teller aantal communicatie fouten
comretry = 0         ' teller aantal communicatie retries
'key = ""            ' laatst ingedrukte toets
'showdel = -1        ' disable delay counter op scherm
Pogingen = 3

txtTWin.Text = ""
txtTWin.Text = "FTF Utility " + Date$ + " " + Time$ + CR
txtTWin.Text = txtTWin.Text + "RQS File: " + rqsFile + " RPL File: " + rplFile + CR

'IF global = 1 THEN
  For k = 1 To reccnt
    ' vervang ? voor global string
    ' gebruik echo als scratch
    tempString = InStr(record(k), "?")
    If tempString And Left(record, 1) <> "" Then
      tempString = Left(record(k), tempString - 1)
      tempString = tempString + global
      tempString = tempString + Mid(record(k), tempString + 1)
      record(k) = tempString
    End If
    txtTWin.Text = txtTWin.Text + Left$(record(k), 77) + CR
  Next k
End If

echoCount = 0          ' initialiseer antwoord teller
                        ' {initialize computer reply}

Dim F
F = FreeFile

If Dir(rplFile) <> "" Then
  'SysMsg "RPL file already exists. Overwrite? (Selecting 'No' will append)", "vbYNC",
  "vbQuery", Reply%
  Reply = MsgBox("RPL file already exists. Overwrite? (Selecting 'No' will append)",
  vbYesNoCancel + vbQuestion, "Overwrite?")

  Select Case Reply
    Case 6 ' overwrite
      Open rplFile For Output As F
    Case 7 ' append
      Open rplFile For Append As F
      echoCount = echoCount + 1
      echoArray(echoCount) = "# APPEND " + Date + " " + Time
    Case 2 ' quit
      frmFTF.MousePointer = 0
      Exit Sub
  End Select
Else
  If InStr(rplFile, App.Path) = 0 Then
    Dim tmpRplFile As String
    tmpRplFile = App.Path + "\FTF\" + rplFile
    Dim Msg$
    Msg = "Are you sure you want to save " + CR
    Msg = Msg + "your RPL file here?" + CR
    Msg = Msg + tmpRplFile + CR
    Msg = Msg + "Selecting no will automatically save in FTF directory."
    Reply = MsgBox(Msg, vbQuestion + vbYesNo, "ECP Message")
    Select Case Reply
      Case 7
        rplFile = tmpRplFile
        Open rplFile For Append As F
      End Select
    Else
      Open rplFile For Append As F
    End If
  End If

  Dim testString As String

  recordNum = 1
```

[illegible]

HNF-1569
Revision 1

```

Timer1.Enabled = False
pBar.Visible = False
pBar.Value = pBar.Min
txtStatus.Text = ""

Case "!SYSTEM"
    echoCount = echoCount + 1
    echoArray(echoCount) = "!SYSTEM / ECPWin v1.0 / [" + globalString + "]"
    txtTWin.Text = txtTWin.Text + echoArray(echoCount) + CR
Case Else
    'fout.in.regel
End Select

Case ">"
    ' transmit command
    echoCount = echoCount + 1
    echoArray(echoCount) = ">" + Mid(recordString(recordNum), 2)
    For p = 1 To Pgingen
        ciucom Mid(recordString(recordNum), 2), echo, status
        'IF INKEY = CHR(27) THEN key = CHR(27): EXIT FOR
    If ftfCancel = 1 Then
        Exit For
    End If
    If Len(echo) > 4 Then Exit For ' communicatie ok,
    If status = 1 Then Delay (300 / 1000)
    If p > 0 Then
        comretry = comretry + 1 ' verhoog retry teller
        txtTWin.Text = txtTWin.Text + "Retry incremented..." + CR
    End If
Next p
If Len(echo) < 5 Then ' communicatie storing
    Select Case status
        Case 0
            echo = "*Comm. error*****"
        Case 1
            echo = "# TIME-OUT " + Date$ + " " + Time$
        Case 2
            echo = "*message-incomplete*****"
        Case 3
            echo = "# CHECKSUM ERROR " + Date$ + " " + Time$
    End Select
    comfail = comfail + 1
    txtTWin.Text = txtTWin.Text + "comfail incremented to " + Str(comfail) + CR

Else
    echo = "<" + echo
End If

txtTWin.Text = txtTWin.Text + echo + CR
echoCount = echoCount + 1
echoArray(echoCount) = echo

Case Else
    'fout.in.regel
End Select

If echoCount > 20 Then
    'save eventueel in file
    GoSub tempSave
End If
If ftfCancel = 1 Then
    txtTWin.Text = txtTWin.Text + CR

    txtTWin.Text = txtTWin.Text + " ** RQS Send Aborted ! ** " + CR
    txtTWin.Text = txtTWin.Text + CR
    'SysMesg "RQS Send Aborted! Continue?", "vbYN", "vbQuery", Reply%
    Reply = MsgBox("RQS Send Aborted! Continue?", vbYesNo + vbQuestion, "Continue?")

Select Case Reply%
    Case 6
        echoCount = echoCount + 1

```

HNF-1569
Revision 1

```

        echoArray(echoCount) = "# LOG " + Date$ + " " + Time$
        echoCount = echoCount + 1
        echoArray(echoCount) = "# ABORTED " + Date + " " + Time
    Case Else
        frmFTF.MousePointer = 0
    Exit Sub
End Select
End If
txtTWin.Text = txtTWin.Text + CR
recordNum = recordNum + 1
If recordNum >= recCnt Then
Exit Do
End If
Delay (50 / 1000)
Loop

    echoCount = echoCount + 1
    echoArray(echoCount) = "# END " + rqsFile + " " + Date$ + " " + Time$
    txtTWin.Text = txtTWin.Text + "END RQS Send !" + CR
End If

If echoCount > 0 Then                                'save eventueel in file
    GoSub tempSave
End If
Close F

rectype = rectypeold          ' restore enkele paramaters
ta = taold
ciuadres = ciuAdresOld

Exit Sub

*****

tempSave:
    tempString = ""
    For k = 1 To echoCount
        tempString = tempString + echoArray(k) + CR
    Next k
    Print F, tempString
    tempString = ""
    Close F
    F = FreeFile
    Open rplFile For Append As F
    echoCount = 0

Return
    ' End If ' ja check
    If TimerShouldBeReEnabled = 1 Then
        frmMain.tmrLevel.Enabled = True
    End If
    'End If 'timer enabled check

    frmFTF.MousePointer = 0

End Sub

Private Sub cmdRQSTest_Click()
'=====
' This procedure was originally written by Delft Instruments. Minor
' modifications were made by John Huber. This procedure is proprietary
' to Delft Instruments and may not be used or reproduced without permission.
'=====

    frmFTF.MousePointer = 11
    rqsFile = txtRQSFilename.Text
    'rplFile = txtRPLFilename.Text
    logFile = txtLOGFilename.Text

    If rqsFile = "" Then
        'SysMsg "You must first specify RQS Filename!", "vbOkay", "vbWarning", Reply%

```

HNF-1569
Revision 1

```
MsgBox "You must first specify RQS Filename!", vbOKOnly + vbCritical, "Need RQS Filename..."

frmFTF.MousePointer = 0
Exit Sub
End If

'Dim Reply As Integer
Dim CR
Dim regelNum As Integer      'regel = line (Dutch)
Dim regelString As String
Dim F As Integer
Dim Fl As Integer
Dim tempString As String
Dim testString As String
Dim fout%

pBar.Min = 0
pBar.Value = pBar.Min
pBar.max = 100
pBar.Visible = True

rqsFile = UCase(rqsFile)
If InStr(rqsFile, ".") = 0 Then
    rqsFile = rqsFile + ".RQS"
End If
If Dir(ftfDir + rqsFile) = "" Then
    'SysMesg ftfDir + rqsFile + " not found !", "vbOkay", "vbQuery", Reply%
    MsgBox ftfDir + rqsFile + " not found !", vbOKOnly + vbQuestion, "File not found!"
    frmFTF.MousePointer = 0
    Exit Sub
End If

txtTWin.Text = "File_to_Field syntax check / v0.1 / " + ftfDir + rqsFile + ".RQS" + CR

fout = 0
regelNum = 1
F = FreeFile
Open ftfDir + rqsFile For Input As F
    While Not EOF(F)
        Line Input #F, regelString
        regelNum = regelNum + 1
    Wend
Close #F

Dim numLines%
numLines = regelNum

regelNum = 1
Open ftfDir + rqsFile For Input As F
    While Not EOF(F)
        Line Input #F, regelString
        txtTWin.Text = txtTWin.Text + Str(regelNum) + regelString + CR
        'Mid(tempString, 3) = regelString
        regelString = LTrim(regelString)
        regelString = UCase(regelString)
        testString = Left(regelString, 1)

        Select Case testString
            Case ""
                ' lege regel
                txtTWin.Text = txtTWin.Text + "empty line " + CR
            Case " "
                ' commentaar
            Case "!"
                ' meta command
                txtTWin.Text = txtTWin.Text + "Meta_command" + CR
                testString = Left(regelString + Space$(7), 7)
                Select Case testString
                    Case "!DATE "
                    Case "!REPEAT"
                    Case "!SYSTEM"
                    Case Else
                End Select
            End Select
        End Select
    End While
End If
```


HNF-1569
Revision 1

```

    If Left(regelString, 7) = "!PAUSE " Then
        testNum = Val(Mid(regelString, InStr(regelString, "!PAUSE") + 6))
        If testNum <= 0 Or testNum >= 64000 Then
            GoTo foutinregel
        End If
    Else
        GoTo foutinregel
    End If
End Select
Case ">"
    ' transmit command
    txtTWin.Text = txtTWin.Text + "Transmit string" + CR
Case Else
    GoTo foutinregel
End Select
txtTWin.Text = txtTWin.Text + "OK" + CR
volgenderegul:

    regelNum = regelNum + 1
    'Suspend
    Delay (20 / 1000)
    pBar.Value = (regelNum / numLines) * 100
Wend
Close #1

txtTWin.Text = txtTWin.Text + "File_to_Field syntax check / v0.1 / " + ftfDir + rqsFile + CR
If fout = 0 Then 'goed
    txtTWin.Text = txtTWin.Text + "no errors found: File OK !" + CR
    txtStatus.Text = "no errors found: File OK !"
Else
    'fout
    txtTWin.Text = txtTWin.Text + Str(fout) + " error(s) found: File rejected !"
    txtStatus.Text = Str(fout) + " error(s) found: File rejected !"
End If
frmFTF.MousePointer = 0
pBar.Visible = False
pBar.Value = pBar.Min

Exit Sub

foutinregel:
    fout = fout + 1
    txtStatus.Text = "Error detected !"

    GoTo volgenderegul
frmFTF.MousePointer = 0
End Sub

Private Sub cmdRQStoLOG_Click()
'=====
' This procedure was originally written by Delft Instruments. Minor
' modifications were made by John Huber. This procedure is proprietary
' to Delft Instruments and may not be used or reproduced without permission.
'=====

    rqsFile = txtRQSFilename.Text
    rplFile = txtRPLFilename.Text
    logFile = txtRPLFilename.Text

    If rqsFile = "" Or logFile = "" Then
        'SysMesg "You must first specify BOTH RQS and LOG Filenames!", "vbOkay", "vbWarning", Reply%
        MsgBox "You must first specify BOTH RQS and LOG Filenames!", vbOKOnly + vbCritical, "Need both
filenames..."

        Exit Sub
    End If

Dim Frpl, Flog
Frpl = FreeFile
Flog = FreeFile
Dim CR As Integer
CR = Chr(13) + Chr(10)

```

HNF-1569
Revision 1

```
logFile = Left(rplFile, InStr(rplFile, ".") - 1)
logFile = Left(logFile, 8) + ".LOG"

If Dir(logFile) <> "" Then

    'SysMsg logFile + " already exists. Overwrite? (Selecting 'NO' will append)", "vbYNC",
    "vbQuery", Reply%
    Reply = MsgBox(logFile + " already exists. Overwrite?", vbYesNoCancel + vbQuestion,
    "Overwrite?")
    Select Case Reply
        Case 6      ' overwrite
            Open logFile For Output As Flog
        Case 7      ' append
            Open logFile For Append As Flog
        Case 2      ' quit
            Exit Sub
    End Select
Else
    Open logFile For Append As Flog
End If

txtTWin.Text = " RPL_LOG conversion / v0.2 / " + rplFile + " -> " + logFile + CR

foutString$ = 0      'aantal gevonden fouten
regelNum% = 1

Open rplFile For Input As Frpl

Print #F, "' ECPWin RPL_LOG converter for Windows v1.0/ " + Date$ + " " + Time$

While Not EOF(Frpl)

    Line Input #Frpl, regelString$

    regelString = LTrim$(regelString)      'verwijder leading spaties
    regelString = UCase$(regelString)      'maak alles upper case

    testString$ = Left(regelString, 1)
    Select Case testString
        Case ""
        Case "'"
            Print #Flog, regelString
        Case "#", "!"
            Print #Flog, "' " + regelString
        Case ">"
            'skip line
        Case "<"
            If InStr(regelString, "Z") = 0 Then
                GoTo foutrpillog
            End If
            itmString$ = Mid(regelString, InStr(regelString, "Z") + 1, 2)
            restString$ = Mid(regelString, InStr(regelString, "Z") + 3)

            If InStr(regelString, "=") Then
            Else
                itmString = itmString + "="
            End If
            If InStr(regelString, "ZEX") > 0 Or InStr(regelString, "ZW2=") > 0 Then
                'skip line
            Else
                txtTWin.Text = txtTWin.Text + itmString + restString + CR
                Print #Flog, itmString + restString
            End If
        Case Else
            GoTo foutrpillog
    End Select
ftfprpllog1:

    If ftfCancel = 1 Then
```

HNF-1569
Revision 1

```
txtStatus.Text = " ** RPL_LOG conversion aborted ** "
'SysMesg "RPL/LOG Conversion aborted! Continue?", "vbYN", "vbQuery", Reply
Reply = MsgBox("RPL/LOG Conversion aborted! Continue?", vbYesNo + vbQuestion,
"Continue?")
    If Reply = 7 Then
        Close #Frpl
        Close #Flog
        Exit Sub
    End If
End If
regelNum = regelNum + 1
'Suspend
Wend
Close #Flog
Close #Frpl

txtTWin.Text = txtTWin.Text + " RPL_LOG conversion / v0.1 / " + rplFile + " -> " + logFile +
CR
If foutNum = 0 Then 'goed
    txtTWin.Text = txtTWin.Text + " no errors found: File conversion completed !" + CR
Else 'fout
    txtTWin.Text = txtTWin.Text + Str(foutNum) + " error(s) found: File rejected !" + CR
End If
Exit Sub

foutrpllog:
    foutNum = foutNum + 1
    txtStatus.Text = "Error detected !"
    Delay (1)
    GoTo ftfrpllog1

End Sub

Private Sub cmdViewLOG_Click()
    On Error Resume Next
    If Dir(logFile) <> "" Then
        Shell "C:\WINDOWS\notepad.exe " & logFile, vbNormalFocus
    End If
End Sub

Private Sub cmdViewRPL_Click()
    On Error Resume Next
    If Dir(rplFile) <> "" Then
        Shell "C:\WINDOWS\notepad.exe " & rplFile, vbNormalFocus
    End If
End Sub

Private Sub cmdViewRQS_Click()
    On Error Resume Next
    If Dir(rqsFile) <> "" Then
        Shell "C:\WINDOWS\notepad.exe " & rqsFile, vbNormalFocus
    End If
End Sub

Private Sub Command1_Click()
    ftfCancel = 1
End Sub

Private Sub Command2_Click()
End Sub

Private Sub Form_Load()
    frmFTF.Left = (Screen.Width / 2) - (frmMain.Width / 2)
    frmFTF.Top = (Screen.Height / 2) - (frmFTF.Height / 2)

    rqsFileOld = rqsFile
```

HNF-1569
Revision 1

```

rplFileOld = rqsFile
logFileOld = logFile

rectypeold = rectype          ' save enkele variabelen
taold = ta
ciuAdresOld = ciuadres

frmFTF.Top = 2580
DefaultDir = App.Path
End Sub

Private Sub Form_QueryUnload(Cancel As Integer, UnloadMode As Integer)
    Select Case UnloadMode
        Case vbFormControlMenu
            'Unload Me
            Cancel = True
        Case vbFormCode
            'End
        Case vbAppWindows
            'End
        Case vbAppTaskManager
            'End
        Case vbFormMDIForm
            'End
    End Select
End Sub

Private Sub txtTWin_Change()
    txtTWin.SelStart = Len(txtTWin.Text)
End Sub

Private Sub ReadRQS()
    Dim aString
    Dim F
    Dim regelcnt%
    F = FreeFile
    If Len(Dir(ftfDir + rqsFile)) = 0 Then
        MsgBox "Invalid or No RQS file selected!", vbOKOnly + vbCritical, "Error"
        Exit Sub
    End If
    Open ftfDir + rqsFile$ For Input As #F

    While Not EOF(1)
        Line Input #F, aString          ' herhaal tot einde file
        regelcnt = regelcnt + 1          ' lees regel binnen
        Print recnt; "-"; aString
        While InStr(aString, Chr(9)) <> 0 ' vervang eerst alle TABS door spaties
            Mid$(aString, InStr(aString, Chr(9)), 1) = " "
        Wend
        aString = LTrim(aString): aString = RTrim(aString): aString = UCase(aString) ' alleen
        hoofdletters
        If InStr(aString, "?") > 0 And Left(aString, 1) = ">" Then
            End If
            If aString <> "" Then
                record(recnt) = aString
                recnt = recnt + 1
            End If
        End If
    Wend

    Close #F
End Sub

Private Sub txtTWin_KeyPress(KeyAscii As Integer)
    KeyAscii = 0
End Sub

End frmFTF.frm
=====

```

```
Begin frmInput.frm

VERSION 5.00
Begin VB.Form frmInput
    ClientHeight = 1380
    ClientLeft = 45
    ClientTop = 270
    ClientWidth = 5220
    LinkTopic = "Form1"
    ScaleHeight = 1380
    ScaleWidth = 5220
    StartUpPosition = 2 'CenterScreen
    Begin VB.CommandButton cmdCalc
        Caption = "C&alculate"
        Height = 252
        Left = 4320
        TabIndex = 4
        Top = 960
        Width = 852
    End
    Begin VB.CommandButton cmdCancel
        Caption = "&Cancel"
        Height = 252
        Left = 3480
        TabIndex = 3
        Top = 960
        Width = 732
    End
    Begin VB.CommandButton cmdOkay
        Caption = "&OK"
        Height = 252
        Left = 2760
        TabIndex = 2
        Top = 960
        Width = 612
    End
    Begin VB.TextBox txtValue
        Height = 288
        Left = 120
        TabIndex = 0
        Top = 960
        Width = 2532
    End
    Begin VB.Label lblPrompt
        AutoSize = -1 'True
        Height = 192
        Left = 240
        TabIndex = 1
        Top = 240
        Width = 36
    End
End
Attribute VB_Name = "frmInput"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False

Private Sub cmdCalc_Click()
' Enable the Use button
frmCalc.cmdUse.Enabled = True
frmCalc.Show 1
' Start with any default/existing value previously used.
frmInput.txtValue.Text = Trim(frmCalc.txtRWT.Text)
End Sub

Private Sub cmdCancel_Click()
txtValue.Text = ""
frmInput.Hide
End Sub

Private Sub cmdOkay_Click()
```

HNF-1569
Revision 1

```
' Hide form so that values don't go away by unloading.  
frmInput.Hide  
End Sub
```

End frmInput.frm

Begin frmLogFile.frm

```
VERSION 5.00  
Object = "{6B7E6392-850A-101B-AFC0-4210102A8DA7}#1.2#0"; "COMCTL32.OCX"  
Begin VB.Form frmLogFile  
    BorderStyle      = 1    'Fixed Single  
    Caption          = "Create Log File"  
    ClientHeight     = 3432  
    ClientLeft       = 3096  
    ClientTop        = 3876  
    ClientWidth      = 5520  
    HelpContextID    = 400  
    LinkTopic        = "Form1"  
    MaxButton        = 0    'False  
    PaletteMode      = 1    'UseZOrder  
    ScaleHeight      = 3432  
    ScaleWidth       = 5520  
    Tag              = "Create Gauge Log File"  
    Begin ComctlLib.ProgressBar pBar  
        Height      = 252  
        Left        = 120  
        TabIndex    = 9  
        Top         = 3120  
        Visible     = 0    'False  
        Width       = 5292  
        _ExtentX    = 9335  
        _ExtentY    = 445  
        _Version    = 327682  
        Appearance  = 1  
    End  
    Begin VB.TextBox txtStatus  
        Height      = 612  
        Left        = 120  
        Multiline   = -1    'True  
        ScrollBars  = 2    'Vertical  
        TabIndex    = 8  
        ToolTipText = "Log Response Window"  
        Top         = 2400  
        Width       = 5292  
    End  
    Begin VB.CommandButton cmdClose  
        Caption      = "&Close"  
        Height       = 495  
        Left        = 3720  
        TabIndex    = 7  
        ToolTipText  = "Closes this window"  
        Top         = 1560  
        Width       = 975  
    End  
    Begin VB.TextBox txtTA  
        Height      = 285  
        HelpContextID = 404  
        Left        = 2760  
        TabIndex    = 5  
        Tag         = "Transmission Address"  
        Text        = "10"  
        ToolTipText  = "Gauge address, 0-99"  
        Top         = 360  
        Width       = 495  
    End  
    Begin VB.TextBox txtTYPE  
        Height      = 285  
        HelpContextID = 403  
        Left        = 2760  
        TabIndex    = 4  
        Tag         = "Gauge Type"
```

HNF-1569
Revision 1

```

Text          = "854"
ToolTipText   = "Enraf Gauge Model Number"
Top           = 840
Width         = 495
End
Begin VB.CommandButton cmdCancel
Caption       = "&Abort"
Height       = 495
HelpContextID = 402
Left         = 2400
TabIndex     = 2
ToolTipText   = "Stops Gauge Log File Creation"
Top          = 1560
Width        = 855
End
Begin VB.CommandButton cmdCreateLogFile
Caption       = "&Create"
Height       = 495
HelpContextID = 401
Left         = 960
TabIndex     = 1
ToolTipText   = "Starts Gauge Log File Creation..."
Top          = 1560
Width        = 855
End
Begin VB.Line Line4
BorderStyle   = 6 'Inside Solid
X1            = 5400
X2            = 5400
Y1            = 120
Y2            = 2280
End
Begin VB.Line Line3
X1            = 120
X2            = 5400
Y1            = 2280
Y2            = 2280
End
Begin VB.Line Line2
BorderStyle   = 6 'Inside Solid
X1            = 5400
X2            = 120
Y1            = 120
Y2            = 120
End
Begin VB.Line Line1
X1            = 120
X2            = 120
Y1            = 120
Y2            = 2280
End
Begin VB.Label Label5
Caption       = "854 is default"
Height       = 252
Left         = 3480
TabIndex     = 6
Top          = 840
Width        = 972
End
Begin VB.Label Label4
Alignment     = 1 'Right Justify
Caption       = "Gauge Type"
BeginProperty Font
Name          = "MS Sans Serif"
Size          = 7.8
Charset       = 0
Weight        = 700
Underline     = 0 'False
Italic        = 0 'False
Strikethrough = 0 'False
EndProperty

```

HNF-1569
Revision 1

```

Height      = 252
Left        = 600
TabIndex    = 3
Top         = 840
Width       = 1932
End
Begin VB.Label Label1
    Alignment = 1 'Right Justify
    Caption   = "Transmission Address"
    BeginProperty Font
        Name      = "MS Sans Serif"
        Size      = 7.8
        Charset   = 0
        Weight    = 700
        Underline = 0 'False
        Italic    = 0 'False
        Strikethrough = 0 'False
    EndProperty
    Height      = 252
    Left        = 600
    TabIndex    = 0
    Top         = 360
    Width       = 1932
End
End
Attribute VB_Name = "frmLogFile"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
'=====
' frmLogFile is used to download a log file of parameters
' from an 854 gauge. This module has been tested and is fully
' implemented.
'=====

Dim StopIt As Integer

Private Sub cmdCancel_Click()
    StopIt = 1
    cmdClose.Enabled = True
End Sub

Private Sub cmdClose_Click()
    StopIt = 1
    frmLogFile.Hide
End Sub

Private Sub cmdCreateLogFile_Click()
'=====
' This procedure was originally written by Delft Instruments. Minor
' modifications were made by John Huber. This procedure is proprietary
' to Delft Instruments and may not be used or reproduced without permission.
'=====

    Dim logFilename As String
    Dim tiepe As String
    Dim ciuAdresOld As String
    Dim ciuNum As Integer
    Dim FileNum1 As Integer
    Dim rString As String
    Dim aString As String
    Dim valString As String
    Dim taNum As Integer
    Dim CR
    Dim Msg As String
    Dim Reply As Integer
    Dim i As Integer

```



```
CR = Chr(13) + Chr(10)
itemcount = 0
If gettingScans = 1 Then
    frmMain.cmdDisconnect_Click
End If

StopIt = 0

'Disable stuff we don't want clicked.
frmMain.cmdDensity.Enabled = False
cmdClose.Enabled = False
frmMain.cmdRecover.Enabled = False
frmMain.cmdGetCurrentLevel.Enabled = False
frmMain.cmdOpEnraf.Enabled = False
frmMain.cmdDensity.Enabled = False
frmMain.cmdLog.Enabled = False
frmMain.cmdFTF.Enabled = False
frmMain.mnuAction.Enabled = False
frmMain.mnuConnect.Enabled = False
frmMain.mnuSetup.Enabled = False

pBar.Min = 0
pBar.Value = pBar.Min
pBar.max = 100
pBar.Visible = True

cmdClose.Enabled = False

If commIsOpen = 0 Then
    ciuopen
End If

' keyboard.input "Transmissionaddress : ", ta$
If ta <> "" Then
    If InStr(ta, "**") = 0 And InStr(ta, "#") = 0 Then
        ta = Right("00" + ta, 2)
        tiepe = ""
        GoSub loggen1
        If logFilename <> "" Then
            WriteToFile (logFilename)
        End If
    Else
        tiepe = ""
        keyboard.input "Gauge type (ENTER for all) :", tiepe$
        tiepe = LTrim$(RTrim(tiepe))
        Select Case ta
            Case "**"
                GoSub loggen2
            Case Else '#'
                ciuAdresOld = ciuadres
                ciuNum = 0
                For ciuNum = 0 To 9
                    ciuadres = LTrim(Str(ciuNum))
                    frmMain.sBar.SimpleText = "Scanning for CIU : " + ciuadres
                    ciucom "@X", echo$, status$
                    If Len(echo$) = 0 Then
                        frmMain.sBar.SimpleText = "No CIU Found..." 'no CIU found
                        GoSub GetOut
                    Else
                        GoSub loggen2
                    End If
                Next ciuNum
                ciuadres = ciuAdresOld
            End Select
        End If
    End If

GoSub GetOut
```

```

loggen2:
    For taNum = 0 To 99
        ta = Right("0" + LTrim(Str(ta)), 2)
        GoSub loggen1
        If logFilename <> "" Then
            WriteToFile logFilename
        End If
    Next taNum
    Return

loggen1:
    instrument = ""
    logFilename = ""
    bepaaltype ja
    If Not (instrument = tiepe Or tiepe = "") Then
        frmMain.sBar.SimpleText = "No <" + tiepe + "> found at address " + ta + " !"
        Delay (0.5)
        Return
    End If
    If ja = 0 Then
        frmMain.sBar.SimpleText = "No gauge found at address " + ta + " !"
        Delay (1)
        Return
    End If

    'Gauge found, get serial number and create file string.
    ciucom "ZPN", echo, status
    'Debug.Print "libDir="; libDir
    Dim z As String
    'z = Dir(libDir + "\ITEMS." + instrument).
    'Debug.Print "z="; z

    logFilename = instrument + Mid(echo, 8) + ".LOG" ' filename=serienummer
    i = 1
    Do
    If Len(Dir(logDir + "\" + logFilename)) <> 0 Then
        Msg = "Warning! " + logFilename + " already exists." + CR
        Msg = Msg + "Overwrite?"
        SysMsg msg, "vbYNC", "vbWarning", Reply
        Reply = MsgBox(Msg, vbYesNoCancel + vbwarning, "Overwrite?")
        If Reply = 7 Then
            frmMain.sBar.SimpleText = "Creating new file..."
            If i = 10 Then
                Msg = "You have exceeded 10 versions of log files for this gauge," + CR
                Msg = Msg + "Please delete one or more before continuing."
                SysMsg msg, "vbOkay", "vbCritical", Reply
                MsgBox Msg, vbOKOnly + vbCritical, "10 versions exceeded!"
                GoSub GetOut
            End If
            logFilename = "DL" + Trim(Str(i)) + Mid(logFilename, 4, InStr(logFilename, ".LOG"))
            i = i + 1
        End If
    End If
    Loop Until Len(Dir(logDir + "\" + logFilename)) = 0 Or Reply = 6 Or Reply = 2
    If Reply = 2 Then GoSub GetOut
    If Reply = 6 Then
        'Create a backup file
        Dim backup As String
        backup = Mid(logFilename, 1, (InStr(logFilename, ".")) + "BAK"
        FileCopy logDir + "\" + logFilename, logDir + "\" + backup
    End If

    frmMain.sBar.SimpleText = " LOGGING of " + instrument + "-gauge file=" + logDir + logFilename
+ " TA= " + ta + " "
    If Len(Dir(libDir + "\ITEMS." + instrument)) = 0 Then
        frmMain.sBar.SimpleText = "item file " + "ITEMS." + instrument + " not found ! "
        Msg = "Item File " + "ITEMS." + instrument + " not found in directory: " + CR
        Msg = Msg + libDir + CR + CR + "Logging Failed!"
        MsgBox Msg
        logFilename = ""
        Return
    End If

```

HNF-1569
Revision 1

```

Else
    FileNum1 = FreeFile
    Open libDir + "\ITEMS." + instrument For Input As FileNum1
    Debug.Print "libDir + ITEMS. + instrument= "; libDir; "ITEMS."; instrument
    itemcount = 0
    item(0) = "LG" 'zet vlag om aan tegeven dat item() is gevuld
    While Not EOF(FileNum1) 'met te loggen items
        itemcount = itemcount + 1
        Line Input #FileNum1, aString
        Debug.Print "Inputing "; aString
        aString = Trim(aString)
        item(itemcount) = aString
        Debug.Print "Item("; itemcount; ")= "; item(itemcount)
    Wend
    Close #FileNum1
End If

Dim k%
For k = 1 To itemcount 'vraag alle items
    Debug.Print "itemcount= "; itemcount
    If StopIt Then Exit For
    rString = item(k)
    ' ''Debug.Print "Item("; k; ")="; Item(k)
    If Left(rString, 1) = "*" Then ' direct commando
        Direct = 1
        rString = Mid(rString, 2)
        rString = Trim(rString)
    Else
        Direct = 0
    End If

    For j = 1 To 3 ' herhaal 5x
        Debug.Print "LogSend = "; rString
        ciucom "Z" + rString, echo, status
        If Len(echo) > 4 Then Exit For
        Delay ((20 * j) / 1000)
    Next j

    ' communicatiefout
    If Len(echo) < 5 Then
        frmMain.sBar.SimpleText = " COMMUNICATION FAIL | LOGGING FAILED "
        logFilename = ""
        GoSub GetOut
    End If

    ' alles ok,
    If InStr(echo, "Z") <> 0 Then
        Debug.Print "decoding Z"; rString; "from "; echo
        decode "Z" + rString, echo, 64, valString
        Debug.Print "valString= "; valString
        If InStr(echo, item(k) + "!") Then valString = "" ' onbekende item
        echoArray(k) = valString
        If Direct Then
            txtStatus.Text = txtStatus.Text + "* direct: " + echo + CR
        Else
            txtStatus.Text = txtStatus.Text + item(k) + "= " + valString + CR
            pBar.Value = (k / itemcount) * 100
        End If
    End If
Next k
pBar.Visible = False
pBar.Value = pBar.Min

Return

GetOut:
pBar.Visible = False
pBar.Value = pBar.Min
cmdClose.Enabled = True

```

HNF-1569
Revision 1

```
frmMain.cmdDensity.Enabled = True
frmMain.cmdRecover.Enabled = True
frmMain.cmdGetCurrentLevel.Enabled = True
frmMain.cmdOpEnraf.Enabled = True
frmMain.cmdDensity.Enabled = True
frmMain.cmdLog.Enabled = True
frmMain.cmdFTF.Enabled = True
frmMain.mnuAction.Enabled = True
frmMain.mnuConnect.Enabled = True
frmMain.mnuSetup.Enabled = True
'restore backup file
If backup <> "" Then
    'SysMsg "Restore Backup Log File?", "vbYN", "vbQuery", Reply
    Reply = MsgBox("Restore Backup Log File?", vbYesNo + vbQuestion, "Restore backup?")
    If Reply = 6 Then
        FileCopy logDir + "\" + backup, logDir + "\" + logFilename
    End If
End If

Exit Sub

End Sub

Private Sub Form_QueryUnload(Cancel As Integer, UnloadMode As Integer)
    Select Case UnloadMode
        Case vbFormControlMenu
            'Unload Me
            Cancel = True
        Case vbFormCode
            'End
        Case vbAppWindows
            'End
        Case vbAppTaskManager
            'End
        Case vbFormMDIForm
            'End
    End Select
End Sub

Private Sub txtStatus_Change()
    txtStatus.SelStart = Len(txtStatus.Text)
End Sub

Private Sub txtStatus_KeyPress(KeyAscii As Integer)
    KeyAscii = 0
End Sub

End frmLogFile.frm

=====
Begin frmLogon.frm

VERSION 5.00
Begin VB.Form frmLogon
    BorderStyle = 3 'Fixed Dialog
    Caption = "ECP Logon"
    ClientHeight = 2520
    ClientLeft = 1890
    ClientTop = 1590
    ClientWidth = 4455
    HelpContextID = 150
    LinkTopic = "Form1"
    MaxButton = 0 'False
    MinButton = 0 'False
    PaletteMode = 1 'UseZOrder
    ScaleHeight = 2520
    ScaleWidth = 4455
    ShowInTaskbar = 0 'False
```

```
Tag          = "Logon Window"
Begin VB.CommandButton cmdCancel
    Caption    = "&Cancel"
    Height     = 375
    Left       = 3120
    TabIndex   = 6
    Top        = 2040
    Width      = 1095
End
Begin VB.CommandButton cmdOkay
    Caption    = "&Ok"
    Height     = 375
    Left       = 3120
    TabIndex   = 5
    Top        = 1440
    Width      = 1095
End
Begin VB.TextBox txtPasswd
    Height     = 285
    IMEMode    = 3 'DISABLE
    Left       = 240
    PasswordChar = "*"
    TabIndex   = 1
    Tag        = "User Password"
    Top        = 2040
    Width      = 2655
End
Begin VB.PictureBox Picture1
    AutoRedraw = -1 'True
    AutoSize   = -1 'True
    Height     = 735
    Left       = 120
    Picture     = "frmLogon.frx":0000
    ScaleHeight = 675
    ScaleWidth  = 2100
    TabIndex   = 2
    Top        = 120
    Width      = 2160
End
Begin VB.TextBox txtUserName
    Height     = 285
    Left       = 240
    TabIndex   = 0
    Tag        = "User Name"
    Top        = 1320
    Width      = 2655
End
Begin VB.Label lblMessage
    Caption    = "Welcome to Enraf Control Panel. Please enter your User Name and the
password."
    BeginProperty Font
        Name      = "Arial"
        Size      = 7.5
        Charset    = 0
        Weight     = 400
        Underline  = 0 'False
        Italic     = 0 'False
        Strikethrough = 0 'False
    EndProperty
    Height     = 1092
    Left       = 2400
    TabIndex   = 7
    Top        = 120
    Width      = 1932
End
Begin VB.Label Label3
    Caption    = "Password"
    Height     = 255
    Left       = 240
    TabIndex   = 4
    Top        = 1800
```

HNF-1569
Revision 1

```
        Width           = 1815
    End
    Begin VB.Label Label2
        Caption           = "User Name:"
        Height            = 255
        Left              = 240
        TabIndex          = 3
        Top               = 1080
        Width             = 1935
    End
End
Attribute VB_Name = "frmLogon"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
'=====
' This module/form, frmLogon, contains routines for reading
' the INI file and controlling program entry. This module is
' fully tested and implemented.
'=====

Dim tempPass As String
Dim UserName(1 To 200) As String
Dim NumofUsers As Integer
Dim DecryptPwd As String

Option Explicit
Private Declare Function GetPrivateProfileString Lib "kernel32" Alias _
"GetPrivateProfileStringA" (ByVal lpApplicationName As String, ByVal _
lpKeyName As String, ByVal lpDefault As String, ByVal lpReturnedString As _
String, ByVal nSize As Long, ByVal lpFileName As String) As Long

Private Declare Function WritePrivateProfileString Lib "kernel32" Alias _
"WritePrivateProfileStringA" (ByVal lpApplicationName As String, ByVal _
lpKeyName As String, ByVal lpString As String, ByVal lpFileName As String) As _
Long

Function Decrypt(ByVal pNum As String, ByVal UserText, ByRef UserPasswd() _
As String, ByRef UserName() As String) As Integer

' This function started out as a routine to decrypt the encrptions
' created in the Encrypt() sub, but ended up as just a string
' comparison routine.

On Error GoTo ErrorHandler
    Dim i As Integer
    Dim j As Integer
    For i = 1 To NumofUsers
        If (UserName(i) = UserText) Then
            'Check password
        Else
            'Return 0
        End If
    Next i
Exit Function

ErrorHandler:
    MsgBox "Error encountered in Decrypt routine."
End Function

Function Encrypt(ByVal pWord As String) As String
' This function takes the sum of the ascii values of each
' letter in the pWord string.

On Error GoTo ErrorHandler
    Dim epWord As String
```

```
Dim i As Integer
Dim num As Integer
For i = 1 To Len(pWord)
    num = num + Asc(Mid(pWord, i, 1))
Next i

Encrypt = Str(num)
Exit Function

ErrorHandler:
    MsgBox "Error encountered in Encrypt routine."

End Function
Sub ReadINIVariables()

On Error GoTo ErrorHandler
    Dim sData As String
    Dim lDataLen As Long
    Dim i%
    Dim inifile As String
    inifile = App.Path + "\ecpw.ini"

    'Read ecpx.ini file variables
    'Read in Compoort
    '[ecpw]
    'Compoort = {port}
    sData = Space$(255) ' allocate space for the string
    lDataLen = GetPrivateProfileString("ecpw", "Compoort", "", sData, _
    Len(sData), inifile)
    'Strip off the NULL that API routines put on the end of strings
    sData = Left$(sData, lDataLen)
    'Print to 'Debug
    'Debug.Print sData
    If sData <> "" Then
        Compoort = sData
    Else
        Compoort = "1"
    End If
    frmCIUSetup.txtCOMPort.Text = Compoort

    'Read in Baudrate
    sData = Space$(255) ' allocate space for the string
    lDataLen = GetPrivateProfileString("ecpw", "Baudrate", "", sData, _
    Len(sData), inifile)
    'Strip off the NULL that API routines put on the end of strings
    sData = Left$(sData, lDataLen)
    'Print to 'Debug
    'Debug.Print sData
    If sData <> "" Then
        Baudrate = sData
    Else
        Baudrate = "2400"
    End If
    frmCIUSetup.txtBaud.Text = Baudrate

    'Read in CIU address
    sData = Space$(255) ' allocate space for the string
    lDataLen = GetPrivateProfileString("ecpw", "ciuadres", "", sData, _
    Len(sData), inifile)
    'Strip off the NULL that API routines put on the end of strings
    sData = Left$(sData, lDataLen)
    'Print to 'Debug
    'Debug.Print sData
    ciuadres = sData
    frmCIUSetup.txtCIUAddress.Text = ciuadres

    'Read in transmission address
    sData = Space$(255) ' allocate space for the string
    lDataLen = GetPrivateProfileString("ecpw", "ta", "", sData, _
```

HNF-1569
Revision 1

```
Len(sData), inifile)
'Strip off the NULL that API routines put on the end of strings
sData = Left$(sData, lDataLen)
'Print to 'Debug
' 'Debug.Print sData
If sData <> "" Then
    ta = sData
Else
    ta = "00"
End If
frmCIUSetup.txtGaugeAdd.Text = ta

'Read in turn_around
sData = Space$(255) ' allocate space for the string
lDataLen = GetPrivateProfileString("ecpw", "turn_around", "", sData, _
Len(sData), inifile)
'Strip off the NULL that API routines put on the end of strings
sData = Left$(sData, lDataLen)
'Print to 'Debug
' 'Debug.Print sData
If sData <> "" Then
    turn_around = CDBl(sData)
Else
    turn_around = 800
End If
frmCIUSetup.txtTurnAround.Text = Str(turn_around)

'Read in scanRate
sData = Space$(255) ' allocate space for the string
lDataLen = GetPrivateProfileString("ecpw", "scanRate", "", sData, _
Len(sData), inifile)
'Strip off the NULL that API routines put on the end of strings
sData = Left$(sData, lDataLen)
'Print to 'Debug
' 'Debug.Print sData
If sData <> "" Then
    scanRate = CDBl(sData)
Else
    scanRate = 3000
End If
frmCIUSetup.txtScanRate.Text = CDBl(scanRate) / 1000

'Read in type of instrument
sData = Space$(255) ' allocate space for the string
lDataLen = GetPrivateProfileString("ecpw", "toi", "", sData, _
Len(sData), inifile)
'Strip off the NULL that API routines put on the end of strings
sData = Left$(sData, lDataLen)
'Print to 'Debug
' 'Debug.Print sData
If sData <> "" Then
    toi = sData
Else
    toi = "B"
End If

'Read in IP Start Level Value
sData = Space$(255) ' allocate space for the string
lDataLen = GetPrivateProfileString("Density", "SDLevel", "", sData, _
Len(sData), inifile)
'Strip off the NULL that API routines put on the end of strings
sData = Left$(sData, lDataLen)
'Print to 'Debug
' 'Debug.Print sData
If sData <> "" Then
    SDLevel = sData
Else
    SDLevel = "0"
End If
```



```

'Read in sediment r level
sData = Space$(255) ' allocate space for the string
lDataLen = GetPrivateProfileString("Density", "SedLev", "", sData, _
Len(sData), inifile)
'Strip off the NULL that API routines put on the end of strings
sData = Left$(sData, lDataLen)
'Print to 'Debug
'Debug.Print sData
If sData <> "" Then
SedLev = sData
Else
SedLev = "0"
End If

'Read in Interval
sData = Space$(255) ' allocate space for the string
lDataLen = GetPrivateProfileString("Density", "Interval", "", sData, _
Len(sData), inifile)
'Strip off the NULL that API routines put on the end of strings
sData = Left$(sData, lDataLen)
'Print to 'Debug
'Debug.Print sData
If sData <> "" Then
Interval = sData
Else
Interval = "12"
End If

'Read in LastTT
sData = Space$(255) ' allocate space for the string
lDataLen = GetPrivateProfileString("Density", "LastTT", "", sData, _
Len(sData), inifile)
'Strip off the NULL that API routines put on the end of strings
sData = Left$(sData, lDataLen)
'Print to 'Debug
'Debug.Print sData
If sData = "" Then sData = "0"
TT = CDBl(sData)

'Read in Parity
sData = Space$(255) ' allocate space for the string
lDataLen = GetPrivateProfileString("ecpw", "Parity", "", sData, _
Len(sData), inifile)
'Strip off the NULL that API routines put on the end of strings
sData = Left$(sData, lDataLen)
'Print to 'Debug
'Debug.Print sData
If sData = "" Then sData = "0"
Parity = sData

'Read in StopBits
sData = Space$(255) ' allocate space for the string
lDataLen = GetPrivateProfileString("ecpw", "StopBits", "", sData, _
Len(sData), inifile)
'Strip off the NULL that API routines put on the end of strings
sData = Left$(sData, lDataLen)
'Print to 'Debug
'Debug.Print sData
If sData = "" Then sData = "1"
StopBits = sData

'Read in DataBits
sData = Space$(255) ' allocate space for the string
lDataLen = GetPrivateProfileString("ecpw", "DataBits", "", sData, _
Len(sData), inifile)
'Strip off the NULL that API routines put on the end of strings
sData = Left$(sData, lDataLen)
'Print to 'Debug
'Debug.Print sData
If sData = "" Then sData = "7"
DataBits = sData

```

```
'Read in CIU busy wait time
sData = Space$(255) ' allocate space for the string
lDataLen = GetPrivateProfileString("ecpw", "CIUWait", "", sData, _
Len(sData), inifile)
'Strip off the NULL that API routines put on the end of strings
sData = Left$(sData, lDataLen)
'Print to 'Debug
Debug.Print "From INI file, busy_waittime= "; sData
If sData = "" Then sData = "0.5"
busy_waittime = CDBl(sData)

'Read in Gauge Display Font Name
sData = Space$(255) ' allocate space for the string
lDataLen = GetPrivateProfileString("Gauge Display Fonts", "Name", "", sData, _
Len(sData), inifile)
'Strip off the NULL that API routines put on the end of strings
sData = Left$(sData, lDataLen)
'Print to 'Debug
If sData = "" Then sData = "Terminal"
frmMain.txtLevelDisplay.Font.Name = sData
frmMain.txtDisplay2.Font.Name = sData

'Read in Gauge Display Font bold
sData = Space$(255) ' allocate space for the string
lDataLen = GetPrivateProfileString("Gauge Display Fonts", "Bold", "", sData, _
Len(sData), inifile)
'Strip off the NULL that API routines put on the end of strings
sData = Left$(sData, lDataLen)
'Print to 'Debug
If sData = "" Then sData = "False"
frmMain.txtLevelDisplay.Font.Bold = Val(sData)
frmMain.txtDisplay2.Font.Bold = Val(sData)

'Read in Gauge Display Font size
sData = Space$(255) ' allocate space for the string
lDataLen = GetPrivateProfileString("Gauge Display Fonts", "Size", "", sData, _
Len(sData), inifile)
'Strip off the NULL that API routines put on the end of strings
sData = Left$(sData, lDataLen)
'Print to 'Debug
If sData = "" Then sData = "10"
frmMain.txtLevelDisplay.Font.Size = Val(sData)
frmMain.txtDisplay2.Font.Size = Val(sData)

'Read in Gauge Display Font italic
sData = Space$(255) ' allocate space for the string
lDataLen = GetPrivateProfileString("Gauge Display Fonts", "Italic", "", sData, _
Len(sData), inifile)
'Strip off the NULL that API routines put on the end of strings
sData = Left$(sData, lDataLen)
'Print to 'Debug
If sData = "" Then sData = "False"
frmMain.txtLevelDisplay.Font.Italic = Val(sData)
frmMain.txtDisplay2.Font.Italic = Val(sData)

'Read in Gauge Display Font Underline
sData = Space$(255) ' allocate space for the string
lDataLen = GetPrivateProfileString("Gauge Display Fonts", "Underline", "", sData, _
Len(sData), inifile)
'Strip off the NULL that API routines put on the end of strings
sData = Left$(sData, lDataLen)
'Print to 'Debug
If sData = "" Then sData = "False"
frmMain.txtLevelDisplay.Font.Underline = Val(sData)
frmMain.txtDisplay2.Font.Underline = Val(sData)
```

HNF-1569
Revision 1

```
'Read in Gauge Display Font Strikethru
sData = Space$(255) ' allocate space for the string
lDataLen = GetPrivateProfileString("Gauge Display Fonts", "Strikethru", "", sData, _
Len(sData), inifile)
'Strip off the NULL that API routines put on the end of strings
sData = Left$(sData, lDataLen)
'Print to 'Debug
If sData = "" Then sData = "False"
frmMain.txtLevelDisplay.FontStrikethru = Val(sData)
frmMain.txtDisplay2.FontStrikethru = Val(sData)

'Read in Gauge Display Font color
sData = Space$(255) ' allocate space for the string
lDataLen = GetPrivateProfileString("Gauge Display Fonts", "Color", "", sData, _
Len(sData), inifile)
'Strip off the NULL that API routines put on the end of strings
sData = Left$(sData, lDataLen)
'Print to 'Debug
If sData = "" Then sData = "65280"
frmMain.txtLevelDisplay.ForeColor = Val(sData)
frmMain.txtDisplay2.ForeColor = Val(sData)

'Read in Last database filename
sData = Space$(255) ' allocate space for the string
lDataLen = GetPrivateProfileString("Density", "LastDBF", "", sData, _
Len(sData), inifile)
'Strip off the NULL that API routines put on the end of strings
sData = Left$(sData, lDataLen)
'Print to 'Debug
Debug.Print "From INI file, LastDBF= "; sData
LastDBF = sData

'Read in Last SpG value filename
sData = Space$(255) ' allocate space for the string
lDataLen = GetPrivateProfileString("Density", "LastSpG", "", sData, _
Len(sData), inifile)
'Strip off the NULL that API routines put on the end of strings
sData = Left$(sData, lDataLen)
'Print to 'Debug
Debug.Print "From INI file, LastSpG= "; sData
'Check and set default
If sData = "" Then sData = "1"
LastSpG = sData
Debug.Print "LastSpG set to: "; LastSpG

'Read in Last Sediment Level Date value filename
sData = Space$(255) ' allocate space for the string
lDataLen = GetPrivateProfileString("Density", "LastSedLevelDate", "", sData, _
Len(sData), inifile)
'Strip off the NULL that API routines put on the end of strings
sData = Left$(sData, lDataLen)
'Print to 'Debug
Debug.Print "From INI file, LastSedLevelDate= "; sData
'Check and set default
LastSedLevelDate = sData

'Read in Last Sediment Level Time value filename
sData = Space$(255) ' allocate space for the string
lDataLen = GetPrivateProfileString("Density", "LastSedLevelTime", "", sData, _
Len(sData), inifile)
'Strip off the NULL that API routines put on the end of strings
sData = Left$(sData, lDataLen)
'Print to 'Debug
Debug.Print "From INI file, LastSedLevelTime= "; sData
'Check and set default
LastSedLevelTime = sData

'Read in Last Sediment Level value filename
```

HNF-1569
Revision 1

```
sData = Space$(255) ' allocate space for the string
lDataLen = GetPrivateProfileString("Density", "LastSedLevel", "", sData, _
Len(sData), inifile)
'Strip off the NULL that API routines put on the end of strings
sData = Left$(sData, lDataLen)
'Print to 'Debug
Debug.Print "From INI file, LastSedLevel= "; sData
'Check and set default
LastSedLevel = sData

'Read in Last Wire Tension value filename
sData = Space$(255) ' allocate space for the string
lDataLen = GetPrivateProfileString("Density", "LastWireTension", "", sData, _
Len(sData), inifile)
'Strip off the NULL that API routines put on the end of strings
sData = Left$(sData, lDataLen)
'Print to 'Debug
Debug.Print "From INI file, LastWireTension= "; sData
'Check and set default
LastWireTension = sData

'Read in WRSS Full Auto Start Level Value
sData = Space$(255) ' allocate space for the string
lDataLen = GetPrivateProfileString("Density", "WRSSStart", "", sData, _
Len(sData), inifile)
'Strip off the NULL that API routines put on the end of strings
sData = Left$(sData, lDataLen)
'Print to 'Debug
Debug.Print "From INI file, WRSSStart= "; sData
'Check and set default
If sData = "" Then
    sData = "156"
End If
WRSSStart = sData
```

Exit Sub

ErrorHandler:

MsgBox "Error encountered reading .ini file variables."

End Sub

Sub ReadinUsers()

On Error GoTo ErrorHandler

Dim sData As String

Dim lDataLen As Long

Dim i%

Dim inifile\$

inifile = App.Path & "\ecpw.ini"

'Read ecpsw.ini file to load UserName(), UserPasswd and UserID() arrays.

'Read in Number of Users

sData = Space\$(255) ' allocate space for the string

lDataLen = GetPrivateProfileString("users", "NumofUsers", "", sData, _

Len(sData), inifile)

'Strip off the NULL that API routines put on the end of strings

sData = Left\$(sData, lDataLen)

'Print to 'Debug

'Debug.Print sData

If sData = "" Then

NumofUsers = 0

Exit Sub

End If

NumofUsers = CInt(sData)

'Read in UserNames

For i = 1 To NumofUsers

sData = Space\$(255) ' allocate space for the string

HNF-1569
Revision 1

```
        lDataLen = GetPrivateProfileString("users", "UserName(" + Str(i) + ")", "", sData, _
        Len(sData), inifile)
        'Strip off the NULL that API routines put on the end of strings
        sData = Left$(sData, lDataLen)
        'Print to 'Debug
        'Debug.Print sData
        UserName(i) = sData
    Next i
Exit Sub

ErrorHandler:
    MsgBox "Error encountered reading .ini file users."

End Sub

Private Sub cmdCancel_Click()
    End
End Sub

Private Sub cmdOkay_Click()
    Dim Msg As String
    Dim UserIndex As Integer
    Dim Reply As Integer

    On Error GoTo ErrorHandler
    Dim i%
    Dim inifile As String
    inifile = App.Path + "\ecpw.ini"

    If txtUserName.Text = "" Then
        'SysMsg "Please Enter a User Name!", "vbOkay", "vbCritical", Reply
        MsgBox "Please Enter a User Name!", vbOKOnly + vbCritical, "Need user name..."
        Exit Sub
    End If
    If txtPasswd.Text = "" Then
        'SysMsg "Please Enter a Password!", "vbOkay", "vbCritical", Reply
        MsgBox "Please Enter a Password!", vbOKOnly + vbCritical, "Need a password..."
        Exit Sub
    End If

    ' check password
    If "njh" = txtPasswd.Text Or UCase(txtPasswd.Text) = MainPword Then
        'User is good.
        'check to see if this user has been here before
        UserIndex = 0
        For i = 1 To NumofUsers
            If txtUserName.Text = UserName(i) Then
                UserIndex = i
            End If
        Next i
        If UserIndex = 0 Then
            'we have a new user, add them to ini file
            NumofUsers = NumofUsers + 1
            UserIndex = NumofUsers
            UserName(UserIndex) = txtUserName.Text
            Dim j%
            'Write the new data to the ini file.
            Dim tempUserName$
            tempUserName = "UserName(" + Str(UserIndex) + ")"
            WritePrivateProfileString "users", "NumofUsers", Str(NumofUsers), inifile
            WritePrivateProfileString "users", tempUserName, UserName(UserIndex), inifile
        End If
        'Unload logon form and load main form
        'but also save the username in the global
        'variable CurrentUserName
        CurrentUserName = UserName(UserIndex)
        frmLogon.Hide
        frmMain.Show
    Exit Sub
End Sub
```

HNF-1569
Revision 1

```
Else
    MsgBox "Your password doesn't match. Please Try again!", vbOKOnly + vbCritical, "Password mismatch..."
    Exit Sub
End If
Exit Sub

ErrorHandler:
    MsgBox "Error encountered during login."

End Sub
```

```
Private Sub Form_Load()

    On Error Resume Next
    'Debug.Print "Loading..."
    frmLogon.Hide
    Screen.MousePointer = 11

    frmSplash.Show
    'Debug.Print "Entering FirstFileSetup..."
    FirstFileSetUp
    'Get password Settings
    MainPword = GetSetting(App.title, "Psets", "MainP", "W5=ENRAF5")
    ServPword = GetSetting(App.title, "PSets", "ServP", "ECPWS")
    ServPwordFlag = GetSetting(App.title, "PSets", "ServPF", "1")
    AdminPword = GetSetting(App.title, "Psets", "AdminP", "SOILENTGREEN")
    DidFileSetup = CInt(GetSetting(App.title, "Fsets", "DidFileSetup", "0"))

    'See if directory structure has been created
    If DidFileSetup = 0 Then
        FirstFileSetUp
        DidFileSetup = 1
    End If
    ReadINIVariables
    ReadinUsers

    'Get the last dat filename
    If LastDBF = "" Then
        'Find the last density dat file created
        Dim fn$
        Dim lastnum%
        lastnum = 0
        Dim substr$
        fn = Dir(App.Path & "\DENSITY\*.DAT")
        Debug.Print "fn= "; fn
        Do While fn <> ""
            substr = Right(fn, 7)
            If CInt(Left(substr, 3)) > CInt(lastnum) Then
                lastnum = CInt(Left(substr, 3))
                LastDBF = App.Path & "\DENSITY\" & fn
            End If
            fn = Dir
        Loop
        Debug.Print "Found last DB file: "; LastDBF
    End If

    Delay (3)
    Unload frmSplash

    'Locate form on screen
    frmLogon.Height = 2970
    frmLogon.Left = (Screen.Width / 2) - (frmLogon.Width / 2)
    frmLogon.Top = (Screen.Height / 2) - (frmLogon.Height / 2)
    frmLogon.Show
    Screen.MousePointer = 0
    Unload frmSplash
```

HNF-1569
Revision 1

```
'Debug.Print "Loading Complete..."
Exit Sub

ErrorHandler:
MsgBox "Error encountered loading logon screen."
End Sub

Private Sub txtPasswd_KeyPress(KeyAscii As Integer)
    If (KeyAscii = 13) Then
        cmdOkay_Click
    End If
End Sub

Private Sub txtUserName_KeyPress(KeyAscii As Integer)
    If (KeyAscii = 13) Then
        cmdOkay_Click
    End If
End Sub

Private Sub FirstFileSetUp()
'=====
' This routine makes sure that the required subdirectories
' in the app directory exist. If not, it creates them.
' A future revision might incorporate these functions into
' the setup routine at installation.
'=====

'Debug.Print "In FirstFileSetUp..."
On Error GoTo ErrorHandler
ChDrive App.Path
ChDir App.Path
Dim checked%
Dim aFile As String
Dim tmp As String
Dim lugDir As String

CurrentDirectory = App.Path

'Debug.Print "Checking Directories..."
'Check to see if required directories exist. If not, create
'and copy related files, set directory variables
'Log Directory
If Dir(App.Path + "\LOG", vbDirectory) = "" Then
    MkDir App.Path + "\LOG"
    logDir = App.Path + "\" + Dir(App.Path + "\LOG", vbDirectory)
Else
    logDir = App.Path + "\" + Dir(App.Path + "\LOG", vbDirectory)
End If

'Lib Directory
If Dir(App.Path + "\LIB", vbDirectory) = "" Then
    MkDir App.Path + "\LIB"
    libDir = App.Path + "\" + Dir(App.Path + "\LIB", vbDirectory)
    If Dir(App.Path & "\ITEMS.854", vbNormal) <> "" Then
        FileCopy App.Path & "\ITEMS.854", App.Path & "\LIB\ITEMS.854"
        Kill App.Path & "\ITEMS.854"
    End If
Else
    libDir = App.Path + "\" + Dir(App.Path + "\LIB", vbDirectory)
End If

'Density Directory
If Dir(App.Path + "\DENSITY", vbDirectory) = "" Then
    MkDir App.Path + "\DENSITY"
End If

'RQS Directory
If Dir(App.Path + "\FTF", vbDirectory) = "" Then
```

HNF-1569
Revision 1

```
Mkdir App.Path + "\FTF"
rqsDir = App.Path + "\" + Dir(App.Path + "\FTF", vbDirectory)
rplDir = App.Path + "\" + Dir(App.Path + "\FTF", vbDirectory)
Else
    rqsDir = App.Path + "\" + Dir(App.Path + "\FTF", vbDirectory)
    rplDir = App.Path + "\" + Dir(App.Path + "\FTF", vbDirectory)
End If

Exit Sub

ErrorHandler:
    MsgBox "Error encountered during initial directory creation."

End Sub
```

End frmLogon.frm

```
=====
Begin frmmain.frm
VERSION 5.00
Object = "{F9043C88-F6F2-101A-A3C9-08002B2F49FB}#1.1#0"; "COMDLG32.OCX"
Object = "{6B7E6392-850A-101B-AFC0-4210102A8DA7}#1.2#0"; "COMCTL32.OCX"
Object = "{648A5603-2C6E-101B-82B6-000000000014}#1.1#0"; "MSCOMM32.OCX"
Begin VB.Form frmMain
    BorderStyle = 1 'Fixed Single
    Caption = "Enraf Control Panel"
    ClientHeight = 3984
    ClientLeft = 3000
    ClientTop = 1476
    ClientWidth = 6876
    BeginProperty Font
        Name = "MS Sans Serif"
        Size = 7.8
        Charset = 0
        Weight = 700
        Underline = 0 'False
        Italic = 0 'False
        Strikethrough = 0 'False
    EndProperty
    ForeColor = &H80000008&
    HelpContextID = 100
    Icon = "frmmain.frx":0000
    LinkTopic = "Form1"
    MaxButton = 0 'False
    MouseIcon = "frmmain.frx":08CA
    PaletteMode = 1 'UseZOrder
    ScaleHeight = 3984
    ScaleWidth = 6876
    Tag = "Enraf Control Panel Main Window"
    WhatsThisButton = -1 'True
    WhatsThisHelp = -1 'True
    Begin VB.Frame brdOpEnraf
        Caption = "Operate Enraf"
        BeginProperty Font
            Name = "MS Sans Serif"
            Size = 7.8
            Charset = 0
            Weight = 400
            Underline = 0 'False
            Italic = 0 'False
            Strikethrough = 0 'False
        EndProperty
        Height = 1335
        Left = 120
        TabIndex = 16
        Top = 2280
        Width = 6615
        Begin VB.TextBox txtRE
            BeginProperty Font
                Name = "MS Sans Serif"
```



```
        Size          = 7.8
        Charset        = 0
        Weight         = 400
        Underline      = 0 'False
        Italic         = 0 'False
        Strikethrough   = 0 'False
    EndProperty
    Height             = 615
    Left               = 4320
    MultiLine          = -1 'True
    ScrollBars         = 2 'Vertical
    TabIndex           = 23
    Top                = 480
    Width              = 2175
End
Begin VB.TextBox txtCustomCommand
    BeginProperty Font
        Name          = "MS Sans Serif"
        Size          = 7.8
        Charset        = 0
        Weight         = 400
        Underline      = 0 'False
        Italic         = 0 'False
        Strikethrough   = 0 'False
    EndProperty
    Height           = 285
    Left             = 2160
    TabIndex         = 22
    Top              = 480
    Width            = 2055
End
Begin VB.CommandButton cmdSEND
    Caption          = "&Send"
    Height           = 375
    Left             = 2760
    TabIndex         = 21
    Top              = 840
    Width            = 1095
End
Begin VB.CommandButton cmdUN
    Caption          = "&UN"
    Height           = 375
    Left             = 600
    TabIndex         = 20
    Top              = 480
    Width            = 495
End
Begin VB.CommandButton cmdCA
    Caption          = "&CA"
    Height           = 375
    Left             = 1080
    TabIndex         = 19
    Top              = 480
    Width            = 495
End
Begin VB.CommandButton cmdFR
    Caption          = "&FR"
    Height           = 375
    Left             = 1560
    TabIndex         = 18
    Top              = 480
    Width            = 495
End
Begin VB.CommandButton cmdTG
    Caption          = "&TG"
    Height           = 375
    Left             = 120
    TabIndex         = 17
    Top              = 480
    Width            = 495
End
```

HNF-1569
Revision 1

```

Begin VB.Label Label3
    Alignment      = 2 'Center
    Caption        = "Communications Record"
    BeginProperty Font
        Name        = "MS Sans Serif"
        Size        = 7.8
        Charset     = 0
        Weight      = 400
        Underline   = 0 'False
        Italic      = 0 'False
        Strikethrough = 0 'False
    EndProperty
    Height        = 255
    Left          = 4320
    TabIndex      = 25
    Top           = 240
    Width         = 2175
End
Begin VB.Label Label2
    Alignment      = 2 'Center
    Caption        = "Command Entry"
    BeginProperty Font
        Name        = "MS Sans Serif"
        Size        = 7.8
        Charset     = 0
        Weight      = 400
        Underline   = 0 'False
        Italic      = 0 'False
        Strikethrough = 0 'False
    EndProperty
    Height        = 255
    Left          = 2160
    TabIndex      = 24
    Top           = 240
    Width         = 2055
End
End
Begin VB.CommandButton cmdReset
    Caption        = "RS"
    Height         = 252
    Left           = 2880
    TabIndex       = 13
    Top            = 960
    Visible        = 0 'False
    Width          = 372
End
Begin VB.CommandButton cmdRecover
    Caption        = "L"
    Height         = 252
    HelpContextID  = 160
    Left           = 3000
    TabIndex       = 12
    ToolTipText    = "Instantaneous Level Reading"
    Top            = 600
    Width          = 252
End
Begin ComctlLib.StatusBar sBar
    Align          = 2 'Align Bottom
    Height         = 255
    Left           = 0
    TabIndex       = 11
    Top            = 3735
    Width          = 6870
    _ExtentX       = 12129
    _ExtentY       = 466
    Style          = 1
    SimpleText     = ""
    _Version       = 327682
    BeginProperty Panels {0713E89E-850A-101B-AFC0-4210102A8DA7}
        NumPanels      = 1
        BeginProperty Panel1 {0713E89F-850A-101B-AFC0-4210102A8DA7}

```

```

        Object.Tag          = ""
        Object.ToolTipText = "System Status"
    EndProperty
EndProperty
End
Begin VB.Frame brdDisplay
    Caption          = "Gauge Display"
    Height           = 1332
    HelpContextID    = 170
    Left             = 3360
    TabIndex         = 8
    Top              = 480
    Width            = 3372
    Begin VB.TextBox txtDisplay2
        Alignment      = 1 'Right Justify
        Appearance     = 0 'Flat
        BackColor      = &H00000000&
        BeginProperty Font
            Name        = "Terminal"
            Size        = 7.2
            Charset     = 255
            Weight      = 400
            Underline   = 0 'False
            Italic      = 0 'False
            Strikethrough = 0 'False
        EndProperty
        ForeColor      = &H0000C000&
        Height         = 285
        HelpContextID  = 172
        Left           = 240
        Multiline      = -1 'True
        TabIndex       = 10
        Top            = 600
        Width          = 3012
    End
    Begin VB.TextBox txtLevelDisplay
        Alignment      = 1 'Right Justify
        Appearance     = 0 'Flat
        BackColor      = &H00000000&
        BeginProperty Font
            Name        = "Terminal"
            Size        = 7.2
            Charset     = 255
            Weight      = 400
            Underline   = 0 'False
            Italic      = 0 'False
            Strikethrough = 0 'False
        EndProperty
        ForeColor      = &H0000C000&
        Height         = 255
        HelpContextID  = 171
        Left           = 240
        Multiline      = -1 'True
        TabIndex       = 9
        ToolTipText    = "Level Display"
        Top            = 360
        Width          = 3012
    End
    Begin VB.Label lblDStatus
        BeginProperty Font
            Name        = "MS Sans Serif"
            Size        = 7.8
            Charset     = 0
            Weight      = 400
            Underline   = 0 'False
            Italic      = 0 'False
            Strikethrough = 0 'False
        EndProperty
        Height         = 252
        Left           = 960
        TabIndex       = 15
    End
End

```

```

        Top           = 960
        Width         = 2292
    End
    Begin VB.Label Label1
        Caption        = "Status:"
        Height         = 252
        Left           = 240
        TabIndex       = 14
        Top            = 960
        Width          = 612
    End
End
Begin VB.CommandButton cmdFTF
    Appearance        = 0 'Flat
    BackColor         = &H80000005&
    Caption           = "FTF"
    Height            = 252
    HelpContextID     = 150
    Left              = 4560
    TabIndex          = 5
    Tag               = "End Program"
    ToolTipText       = "File to Field Utility"
    Top               = 120
    WhatsThisHelpID   = 80
    Width             = 1092
End
Begin VB.CommandButton cmdLog
    Appearance        = 0 'Flat
    BackColor         = &H80000005&
    Caption           = "&Log"
    Height            = 252
    HelpContextID     = 140
    Left              = 3480
    TabIndex          = 4
    Tag               = "End Program"
    ToolTipText       = "Create Gauge Log File"
    Top               = 120
    WhatsThisHelpID   = 80
    Width             = 1092
End
Begin VB.CommandButton cmdDensity
    Appearance        = 0 'Flat
    BackColor         = &H80000005&
    Caption           = "Density"
    Height            = 252
    HelpContextID     = 130
    Left              = 2400
    TabIndex          = 3
    Tag               = "End Program"
    ToolTipText       = "Get Density Data"
    Top               = 120
    WhatsThisHelpID   = 80
    Width             = 1092
End
Begin VB.CommandButton cmdOpEnraf
    Appearance        = 0 'Flat
    BackColor         = &H80000005&
    Caption           = "Send >>"
    Height            = 252
    HelpContextID     = 120
    Left              = 1320
    TabIndex          = 2
    Tag               = "End Program"
    ToolTipText       = "Send Items"
    Top               = 120
    WhatsThisHelpID   = 80
    Width             = 1092
End
Begin VB.CommandButton cmdEndProgram
    Appearance        = 0 'Flat
    BackColor         = &H80000005&

```

HNF-1569
Revision 1

```

Caption      = "E&xit"
Height       = 252
Left         = 5640
TabIndex     = 7
Tag          = "End Program"
ToolTipText  = "Exit Program"
Top          = 120
WhatsThisHelpID = 80
Width        = 1092
End
Begin VB.CommandButton cmdGetCurrentLevel
    Appearance = 0 'Flat
    BackColor  = &H80000005&
    Caption    = "&Start Scans"
    Height     = 252
    HelpContextID = 110
    Left       = 120
    TabIndex   = 0
    Tag        = "Get Continuous Level Readings"
    ToolTipText = "Start Continuous Level Scans"
    Top        = 120
    WhatsThisHelpID = 20
    Width      = 1212
End
Begin VB.CommandButton cmdDisconnect
    Appearance = 0 'Flat
    BackColor  = &H80000005&
    Caption    = "&Stop Scans"
    Height     = 252
    Left       = 120
    TabIndex   = 1
    Tag        = "Stop Continuous Level Readings"
    ToolTipText = "Stop Continuous Level Scans"
    Top        = 120
    Visible    = 0 'False
    Width      = 1212
End
Begin MSCommLib.MSComm Comm2
    Left       = 3120
    Top        = 360
    _ExtentX   = 804
    _ExtentY   = 804
    _Version   = 327680
    DTREnable  = -1 'True
End
Begin MSComDlg.CommonDialog CMDialog1
    Left       = 3600
    Top        = 360
    _ExtentX   = 699
    _ExtentY   = 699
    _Version   = 327681
End
Begin VB.Timer Timer1
    Left       = 4440
    Top        = 360
End
Begin VB.Timer tmrLevel
    Enabled    = 0 'False
    Interval   = 15000
    Left       = 4080
    Top        = 360
End
Begin VB.CommandButton cmdSendL
    Appearance = 0 'Flat
    BackColor  = &H80000005&
    Caption    = "Send <<"
    Height     = 252
    HelpContextID = 120
    Left       = 1320
    TabIndex   = 26
    Tag        = "End Program"
```

```
ToolTipText    = "Send Items"
Top            = 120
Visible        = 0 'False
WhatsThisHelpID = 80
Width          = 1092
End
Begin VB.Label lblCStatus
    BeginProperty Font
        Name        = "Arial"
        Size         = 7.8
        Charset      = 0
        Weight       = 400
        Underline    = 0 'False
        Italic       = 0 'False
        Strikethrough = 0 'False
    EndProperty
    Height        = 252
    Left          = 120
    TabIndex      = 6
    Top           = 1680
    Width         = 2532
End
Begin VB.Image imgGRY
    Height        = 180
    Left          = 2280
    Picture       = "frmmain.frx":0D0C
    ToolTipText   = "CIU Idle"
    Top           = 600
    Width         = 180
End
Begin VB.Image imgRED
    Height        = 180
    Left          = 2280
    Picture       = "frmmain.frx":101E
    ToolTipText   = "CIU Busy"
    Top           = 600
    Visible       = 0 'False
    Width         = 180
End
Begin VB.Image imgGRN
    Height        = 180
    Left          = 2280
    Picture       = "frmmain.frx":1330
    ToolTipText   = "CIU Working"
    Top           = 600
    Visible       = 0 'False
    Width         = 180
End
Begin VB.Image picSYS
    Height        = 612
    Left          = 120
    Picture       = "frmmain.frx":1642
    Stretch       = -1 'True
    ToolTipText   = "No Device Communication"
    Top           = 600
    Width         = 2052
End
Begin VB.Image picEnraf
    Height        = 612
    Left          = 120
    Picture       = "frmmain.frx":6058
    Stretch       = -1 'True
    ToolTipText   = "Full Device Communication"
    Top           = 600
    Width         = 2052
End
Begin VB.Image picCIU
    Height        = 612
    Left          = 120
    Picture       = "frmmain.frx":AA6E
    Stretch       = -1 'True
```

HNF-1569
Revision 1

```
ToolTipText    = "CIU Communication only"
Top            = 600
Width         = 2052
End
Begin VB.Image picCOMM
    Height      = 612
    Left       = 120
    Picture     = "frmmain.frx":F484
    Stretch     = -1 'True
    ToolTipText = "Comport Open, no CIU communication"
    Top        = 600
    Width      = 2052
End
Begin VB.Menu mnuFile
    Caption      = "&File"
    Begin VB.Menu mnuViewOpen
        Caption    = "&View/Print"
        Shortcut   = ^V
        Tag        = "View Print"
    End
    Begin VB.Menu mnuFileCopy
        Caption    = "&Copy DAT to Floppy"
        Shortcut   = ^{F11}
    End
    Begin VB.Menu mnuFileViewLog
        Caption    = "View &Activity Log"
    End
    Begin VB.Menu mnuFileTemp
        Caption    = "&Temp"
        Visible    = 0 'False
    End
    Begin VB.Menu mnuBAR
        Caption    = "-"
    End
    Begin VB.Menu mnuExit
        Caption    = "E&xit"
        Shortcut   = ^X
    End
End
Begin VB.Menu mnuSetup
    Caption      = "&Setup"
    Begin VB.Menu mnuCIUSetup
        Caption    = "&CIU Setup"
        Shortcut   = ^U
        Tag        = "CIU Setup"
    End
    Begin VB.Menu mnuSetupGauge
        Caption    = "&Gauge Setup/Config."
    End
    Begin VB.Menu mnuService
        Caption    = "P&CET"
        Shortcut   = ^{F12}
    End
End
Begin VB.Menu mnuAction
    Caption      = "&Action"
    Tag          = "Action"
    Begin VB.Menu mnuOpEnraf
        Caption    = "&Operate Enraf"
        Shortcut   = ^E
        Tag        = "Operate Enraf"
    End
    Begin VB.Menu mnuGetDensities
        Caption    = "&Get Densities"
        Shortcut   = ^D
        Tag        = "Get Density Profile"
    End
    Begin VB.Menu mnuGageLog
        Caption    = "&Create Gauge Log File"
        Shortcut   = ^L
        Tag        = "Log From Gauge"
    End
End
```

HNF-1569
Revision 1

```
End
Begin VB.Menu mnuFTF
    Caption      = "&File to Field"
    Shortcut     = ^F
    Tag          = "File To Field"
End
Begin VB.Menu mnuBCalc
    Caption      = "&Buoyancy Calculator"
    Shortcut     = ^Y
End
Begin VB.Menu mnuActionWeigh
    Caption      = "&Weigh Displacer"
End
Begin VB.Menu mnuActionLevelCal
    Caption      = "&Level Calibration"
    Enabled      = 0 'False
    Visible      = 0 'False
End
Begin VB.Menu mnuActionBT
    Caption      = "B&alance Test"
End
End
Begin VB.Menu mnuConnect
    Caption      = "&Connect"
    Tag          = "Get Continous Level Readings"
Begin VB.Menu mnuConnectCIU
    Caption      = "&Start Scans"
    Shortcut     = {F2}
    Tag          = "Get Continuous Level Readings"
End
Begin VB.Menu mnuDisconnect
    Caption      = "S&top Scans"
    Shortcut     = {F3}
    Tag          = "Stop Continuous Level Readings"
End
Begin VB.Menu mnuCloseComm
    Caption      = "C&lose Comm Port"
    Shortcut     = {F7}
End
Begin VB.Menu mnuReset
    Caption      = "Reset Program"
    Shortcut     = {F12}
End
End
Begin VB.Menu mnuOptions
    Caption      = "&Options"
Begin VB.Menu mnuOptionsDFont
    Caption      = "&Gauge Display Font"
End
Begin VB.Menu mnuOptionsAdministrator
    Caption      = "&Administrator"
End
Begin VB.Menu mnuOptionsDemo
    Caption      = "&Demo Mode"
End
End
Begin VB.Menu mnuHelp
    Caption      = "&Help"
    Tag          = "Help"
Begin VB.Menu mnuHelpContents
    Caption      = "&Contents"
    Shortcut     = ^H
End
Begin VB.Menu mnuAbout
    Caption      = "&About"
    Shortcut     = ^A
    Tag          = "About"
End
End
End
Attribute VB_Name = "frmMain"
```



```
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
Option Explicit
'=====
'
'=====
```

```
Dim AbortDensity As Integer
Dim TimerTrap%
```

```
Private Sub cmdDensity_Click()
mnuGetDensities_Click
End Sub
```

```
Sub cmdDisconnect_Click()
mnuDisconnect_Click
End Sub
```

```
Private Sub cmdEndProgram_Click()
mnuExit_Click
End Sub
```

```
Private Sub cmdFTF_Click()
mnuFTF_Click
End Sub
```

```
Sub cmdGetCurrentLevel_Click()
mnuConnectCIU_Click
End Sub
```

```
Private Sub cmdLog_Click()
mnuGageLog_Click
End Sub
```

```
Private Sub cmdOpEnraf_Click()
cmdOpEnraf.Visible = False
cmdSendL.Visible = True
mnuOpEnraf_Click
End Sub
```

```
Sub cmdRecover_Click()
Dim DisFlag%
Dim echo$
Dim answer$
If gettingScans = 1 Then
    DisFlag = 1
    frmMain.cmdDisconnect_Click
End If
frmService.cmdLevel_Click
If onDisconnect_Click = 1 Then Exit Sub
ciucom "Z2B", echo, status
If onDisconnect_Click = 1 Then Exit Sub
Delay (0.3)
decode "ZB", echo, 32, answer
frmMain.txtDisplay2.Text = answer
If DisFlag = 1 Then
    frmMain.cmdGetCurrentLevel_Click
    DisFlag = 0
```

End If

End Sub

```
Private Sub cmdReset_Click()
'=====
' The need for this command has never arisen, and does not
' need to be implemented as yet. (12/21/98)
' =====
On Error GoTo ErrorHandler
Dim R%, m$, echo$, answer$
Dim nResume%, nElapsed%
m = "This action will cause the Enraf Gauge to Reset itself." & vbNewLine
m = m + "All current activities in progress will be aborted." & vbNewLine
m = m + "Are you sure you want to reset the gauge?"
R = MsgBox(m, vbYesNo + vbCritical, "Reset Gauge?")
If R = vbNo Then
Exit Sub
Else
' If we are scanning, stop
If gettingScans = 1 Then
cmdDisconnect_Click
' Make note of our disconnection.
nResume = 1
End If
If GettingDensities = 1 Then
AbortDensity = 1
End If
' Make sure looking for I1 (for later)
ciucom "I1", echo, status
' Reset the gauge
ciucom "RS", echo, status
' Watch to see when reset is complete
Delay (3)
Do
Delay (1)
GetALevel
If InStr(frmMain.txtDisplay2.Text, "I1") Then
answer = ""
GoSub check_errors
If answer = "" Then
MsgBox m
Exit Do
Else
MsgBox "Reset Successful!"
Exit Do
End If
End If
' Count elapsed seconds
nElapsed = nElapsed + 1
' If gauge has not reset within 60 seconds call it
' a failure.
If nElapsed > 60 Then
MsgBox "Could not achieve successful reset."
Exit Do
End If
If vbKeyEscape Then Exit Do
Loop
End If

GoTo ErrorHandler

check_errors:
ciucom "ZES", echo, status
decode "ES", echo, 64, answer
answer = Trim(answer)
If answer <> "0000" Then
m = "Reset successful, but..." & vbNewLine
m = m + GetSPUErrorMessage(answer)
answer = ""

```

```
End If
ciucom "ZEP", echo, status
decode "EP", echo, 64, answer
answer = Trim(answer)
If InStr(echo, "!0") <> 0 Then
    m = "Reset successful, but..." & vbNewLine
    m = m + GetErrorMessage(answer)
    answer = ""
End If
Return

ErrorHandler:
If nResume = 1 Then
    cmdGetCurrentLevel_Click
End If

End Sub

Private Sub cmdSendL_Click()
cmdSendL.Visible = False
cmdOpEnraf.Visible = True
mnuOpEnraf_Click
End Sub

Private Sub Comm2_OnComm()
Dim EVMsg$
Dim ERMMsg$
Dim Ret%

' Branch according to the CommEvent property.
Select Case Comm2.CommEvent
    ' Event messages.
    Case vbMSCommEvReceive
        ' ShowData Term, (Comm2.Input)
    Case vbMSCommEvSend

    Case vbMSCommEvCTS
        'EVMsg$ = "Change in CTS Detected"
    Case vbMSCommEvDSR
        'EVMsg$ = "Change in DSR Detected"
    Case vbMSCommEvCD
        'EVMsg$ = "Change in CD Detected"
    Case vbMSCommEvRing
        'EVMsg$ = "The Phone is Ringing"
    Case vbMSCommEvEOF
        'EVMsg$ = "End of File Detected"

    ' Error messages.
    Case vbMSCommErBreak
        EVMsg$ = "Break Received"
    Case vbMSCommErCTSTO
        ERMMsg$ = "CTS Timeout"
    Case vbMSCommErDSRTO
        ERMMsg$ = "DSR Timeout"
    Case vbMSCommErFrame
        EVMsg$ = "Framing Error"
    Case vbMSCommErOverrun
        ERMMsg$ = "Overrun Error"
    Case vbMSCommErCDTO
        ERMMsg$ = "Carrier Detect Timeout"
    Case vbMSCommErRxOver
        ERMMsg$ = "Receive Buffer Overflow"
    Case vbMSCommErRxParity
        EVMsg$ = "Parity Error"
    Case vbMSCommErTxFull
        frmMain.Comm2.OutBufferCount = 0
        'ERMMsg$ = "Transmit Buffer Full"
    Case Else
        'ERMMsg$ = "Unknown error or event"
End Select
```

HNF-1569
Revision 1

```
If Len(EVMsg$) Then
    ' Display event messages in the label control.
    sBar.SimpleText = EVMsg$
    EVMsg$ = ""
ElseIf Len(ERMsg$) Then
    ' Display error messages in an alert message box.
    Beep
    Ret = MsgBox(ERMsg$, 1, "Click Cancel to quit, OK to ignore.")
    ERMsg$ = ""
    ' If the user clicks Cancel (2)...
    If Ret = 2 Then
        Comm2.PortOpen = 0      ' Close the port and quit.
    End If
End If
End Sub
```

```
Private Sub Form_Load()
    ' Set mouse pointer
    'frmBug.Show
    KeyPreview = True
    Screen.MousePointer = 11

    'Setup menu enables.
    mnuConnectCIU.Enabled = True
    mnuDisconnect.Enabled = False
    'mnuCloseComm.Enabled = False
    'mnuReset.Enabled = False

    ' Locate form on screen
    'frmMain.Left = (Screen.Width / 2) - (frmMain.Width / 2)
    frmMain.Top = 0
    frmMain.Left = 0
    frmMain.Height = 2856

    ' Initialize variables

    opWarningTitle = "Enraf Control Panel Message..."
    'busy_waittime = 0.5

    'ENTER CODE TO SET DEFAULT DIRECTORY!
    CurrentDirectory = App.Path

    Screen.MousePointer = 0
End Sub
```

```
Private Sub Form_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single)
    frmMain.SetFocus
End Sub
```

```
Private Sub Form_QueryUnload(Cancel As Integer, UnloadMode As Integer)
    Select Case UnloadMode
        Case vbFormControlMenu

            Dim message As String
            Dim Reply As Integer

            If commIsOpen Then
                message = "Warning: This will "
                message = message + "HALT device communication!"
                Reply = MsgBox(message, 1 + 48, opWarningTitle)
                If Reply = 1 Then
                    'SaveToINI
                    mnuExit_Click
                End If
            End If
    End Select
End Sub
```

```
        ciuclose
    End
Else
    Cancel = True
End If
Else
    'SaveToINI
    mnuExit_Click
End
End If
Cancel = True
Case vbFormCode
    End
Case vbAppWindows
    End
Case vbAppTaskManager
    End
Case vbFormMDIForm
    End
End Select
End Sub
```

```
Private Sub mnuAbout_Click()
Screen.MousePointer = 11
frmAbout.Show
Screen.MousePointer = 0
End Sub
```

```
Private Sub mnuActionBT_Click()
' Call balance test function and display results
DoBalanceTest (True)
End Sub
```

```
Private Sub mnuActionLevelCal_Click()
frmLevelCal.Show
End Sub
```

```
Private Sub mnuActionWeigh_Click()
frmWeigh.Show
End Sub
```

```
Private Sub mnuBCalc_Click()
Screen.MousePointer = 11
frmCalc.Show
Screen.MousePointer = 0
End Sub
```

```
Private Sub mnuCIUSetup_Click()
    frmCIUSetup.Show 1
End Sub
```

```
Private Sub mnuCloseComm_Click()
On Error GoTo Err
If gettingScans = 1 Then
    cmdDisconnect_Click
End If
ciuclose
If picSYS.Visible = False Then
    ShowPic "SYSTEM"
End If
brdDisplay.Caption = "Gauge Display"
GotCIU = 0
gaugeChecked = 0
Exit Sub
```

```
Err:
    MsgBox "Error encountered when user closed comm port"
```

HNF-1569
Revision 1

```
Exit Sub
End Sub

Sub mnuConnectCIU_Click()
'This routine starts communication with the CIU and
'uses tmrLevel() to poll the gauge at intervals as
'determined by the scanRate parameter.

On Error GoTo ErrorHandler
If GettingDensities = 1 Then
    MsgBox "You must first abort or wait for density profile to complete."
    GoSub ErrorHandler
End If

onDisconnect_Click = 0

'Disable stuff so we don't screw up the program
mnuConnectCIU.Enabled = False
mnuDisconnect.Enabled = False
cmdGetCurrentLevel.Visible = False
cmdDisconnect.Visible = True
cmdDisconnect.Enabled = False
'mnuDisconnect.Enabled = True

initializing = 1

'cmdRecover_Click
If gaugeChecked = 0 Then
    checkGageAddress ta, ja
    Debug.Print "out of check gauge, cont. start scans..."
    Debug.Print "ja= "; ja
    Debug.Print "onDisconnect_Click= "; onDisconnect_Click
End If
If ja = 0 Then
    GoSub ErrorHandler
End If

If onDisconnect_Click = 1 Then
    GoSub ErrorHandler
End If

Dim message As String
Dim Reply As Integer

Dim tempTxt$

onDisconnect_Click = 0

If ja = 0 Then
    'Debug.Print "Ja=0 so exting StartScans"
    GoSub ErrorHandler
Else
    GetTankDimensions
    If onDisconnect_Click = 1 Then GoSub ErrorHandler
    cmdGetCurrentLevel.Visible = False
    tmrLevel.Interval = scanRate
    tmrLevel.Enabled = True
    gettingScans = 1
    'frmMain.brdDisplay.Caption = Instrument & " Gauge Display"
End If

Exit Sub

ErrorHandler:
'Debug.Print "In Start Scans ErrorHandler"
cmdGetCurrentLevel.Visible = True
cmdDisconnect.Visible = False
cmdDisconnect.Enabled = False
```

HNF-1569
Revision 1

```
mnuConnectCIU.Enabled = True
mnuDisconnect.Enabled = False
gettingScans = 0
onDisconnect_Click = 0
If tmrLevel.Enabled = True Then tmrLevel.Enabled = False

Exit Sub

End Sub

Sub mnuDisconnect_Click()
On Error GoTo ErrorHandler

    Dim Reply%

    If GettingDensities = 1 Then
        MsgBox "Must Click on Abort First!", vbOKOnly + vbCritical, "Abort to stop scans!"
        Exit Sub
    End If
    onDisconnect_Click = 1
    tmrLevel.Enabled = False
    cmdDisconnect.Visible = False
    cmdGetCurrentLevel.Visible = True
    mnuConnectCIU.Enabled = True
    mnuDisconnect.Enabled = False
    'gaugeChecked = 0
    gettingScans = 0
    sBar.SimpleText = "...Scans Stopped..."
    txtLevelDisplay.Text = ""
    txtDisplay2.Text = ""
    Delay (1)
    ShowPic "COMM"

    Exit Sub

ErrorHandler:
    MsgBox "Error encountered during disconnect.  Reseting system..."
    ResetSystem
    Exit Sub
End Sub

Private Sub mnuExit_Click()

    Dim message As String
    Dim Reply As Integer

    If commIsOpen Then
        message = "Warning: This will "
        message = message + "HALT device communication!"

        Reply = MsgBox(message, 1 + 48, opWarningTitle)
        If Reply = 1 Then
            ciuclose
            SaveToINI
            SaveSetting App.title, "Psets", "MainP", MainPword
            SaveSetting App.title, "Psets", "ServP", ServPword
            SaveSetting App.title, "Psets", "AdminP", AdminPword
            SaveSetting App.title, "Psets", "ServPF", ServPwordFlag
            SaveSetting App.title, "Fsets", "DoFileSetup", Str(DidFileSetup)
            End
        End If
    Else
        SaveToINI
        End
    End If

End Sub

Private Sub mnuFileCopy_Click()
On Error GoTo DiskError
```

HNF-1569
Revision 1

```
If LastDBF = "" Then
    MsgBox "No Valid Density data file has been created.", vbOKOnly + vbExclamation, "File not found!"
    Exit Sub
End If
Dim theFile$
' Get the filename

Debug.Print "LastDBF= "; LastDBF
Debug.Print "app.path= "; App.Path
theFile = Dir(LastDBF)
Debug.Print "theFile= "; theFile
Dim R%
RETRY:
R = MsgBox("Copy Last Data file, " & Dir(LastDBF) & ", to Floppy?", vbYesNoCancel + vbQuestion, "Copy to disk?")
If R = 6 Then
    MsgBox "Insert disk into drive a: "
    If Dir("a:\" & theFile) <> "" Then
        R = MsgBox(Dir(LastDBF) & " already exists. Overwrite?", vbYesNoCancel + vbQuestion, "Overwrite?")
        Select Case R
            Case 2 'cancel
                Exit Sub
            Case 6 'yes
                FileCopy LastDBF, "a:\" & Dir(LastDBF)
                If Dir(App.Path & "\DENSITY\ACTIVITY.LOG") <> "" Then
                    FileCopy App.Path & "\DENSITY\ACTIVITY.LOG", "a:\ACTIVITY.LOG"
                End If
                Debug.Print "Copied "; LastDBF
                Debug.Print "To: "; "a:\" & Dir(LastDBF)
                If Dir("a:\" & Dir(LastDBF)) <> "" Then
                    MsgBox "File successfully copied to disk.", vbOKOnly + vbInformation, "Success!"
                End Sub
            Else
                MsgBox "Copy was not successful. Please try again."
                Exit Sub
            End If
            Case 7 'no
                With CMDialog1
                    .CancelError = True
                    .Filter = "All Files (*.*)|*.|Data Files (*.DAT)|*.dat"
                    .FilterIndex = 2
                    .filename = ""
                    .DialogTitle = "Enter a new filename..."
                    .ShowSave
                    If .filename <> "" Then
                        FileCopy LastDBF, .filename
                        Debug.Print "Copied "; LastDBF
                        Debug.Print "To: "; .filename
                        If Dir(App.Path & "\DENSITY\ACTIVITY.LOG") <> "" Then
                            FileCopy App.Path & "\DENSITY\ACTIVITY.LOG", "a:\ACTIVITY.LOG"
                        End If
                        If Dir(.filename) <> "" Then
                            MsgBox "File successfully copied to disk.", vbOKOnly + vbInformation, "Success!"
                        Else
                            MsgBox "Copy was not successful. Please try again."
                            Exit Sub
                        End If
                    End If
                End With
            End Select
        Else
            FileCopy LastDBF, "a:\" & Dir(LastDBF)
            Debug.Print "Copied "; LastDBF
            Debug.Print "To: "; "a:\" & Dir(LastDBF)
            If Dir(App.Path & "\DENSITY\ACTIVITY.LOG") <> "" Then
                FileCopy App.Path & "\DENSITY\ACTIVITY.LOG", "a:\ACTIVITY.LOG"
            End If
        End If
    End If
End If
```


HNF-1569
Revision 1

```

If Dir("a:\" & Dir{LastDBF}) <> "" Then
    MsgBox "File successfully copied to disk.", vbOKOnly + vbInformation, "Success!"
    Exit Sub
Else
    MsgBox "Copy was not successful. Please try again."
    Exit Sub
End If
End If
ElseIf R = 7 Then
    R = MsgBox("Copy ANY Data file to Floppy?", vbYesNo + vbQuestion, "Copy file?")
    Select Case R
        Case 6
            With CMDialog1
                .CancelError = True
                .Filter = "All Files (*.*)|*.|Data Files (*.DAT)|*.dat"
                .FilterIndex = 2
                .filename = LastDBF
                .DialogTitle = "Select a file to copy..."
                .ShowOpen
                Dim tmpfile$
                tmpfile = .filename
            End With
            If Dir("a:\" & Dir{tmpfile}) = "" Then
                FileCopy tmpfile, "a:\" & Dir{tmpfile}
                Debug.Print "Copied "; tmpfile
                Debug.Print "To: "; "a:\" & Dir{tmpfile}
                If Dir(App.Path & "\DENSITY\ACTIVITY.LOG") <> "" Then
                    FileCopy App.Path & "\DENSITY\ACTIVITY.LOG", "a:\ACTIVITY.LOG"
                End If
                If Dir("a:\" & Dir{tmpfile}) <> "" Then
                    MsgBox "File successfully copied to disk.", vbOKOnly + vbInformation, "Success!"
                    Exit Sub
                Else
                    MsgBox "Copy was not successful. Please try again."
                    Exit Sub
                End If
            Else
                R = MsgBox("File already exists. Overwrite?", vbYesNoCancel + vbQuestion,
"Overwrite?")
                'Select Case r
                If R = 2 Then 'Case 2 'cancel
                    Exit Sub
                ElseIf R = 7 Then 'Case 7 'no
                    With CMDialog1
                        .CancelError = True
                        .Filter = "All Files (*.*)|*.|Data Files (*.DAT)|*.dat"
                        .FilterIndex = 2
                        .InitDir = "A:\"
                        .filename = ""
                        .DialogTitle = "Enter a new filename..."
                        .ShowSave
                        If .filename <> "" Then
                            FileCopy tmpfile, .filename
                            Debug.Print "Copied "; tmpfile
                            Debug.Print "To: "; .filename
                            If Dir(App.Path & "\DENSITY\ACTIVITY.LOG") <> "" Then
                                FileCopy App.Path & "\DENSITY\ACTIVITY.LOG", "a:\ACTIVITY.LOG"
                            End If
                            If Dir(.filename) <> "" Then
                                MsgBox "File successfully copied to disk.", vbOKOnly + vbInformation,
"Success!"
                                Exit Sub
                            Else
                                MsgBox "Copy was not successful. Please try again."
                                Exit Sub
                            End If
                        Else
                            Exit Sub
                        End If
                    End With
                ElseIf R = 6 Then 'Case 6 ' yes

```

HNF-1569
Revision 1

```
FileCopy tmpfile, "a:\" & Dir(tmpfile)
Debug.Print "Copied "; tmpfile
Debug.Print "To: "; "a:\" & Dir(tmpfile)
If Dir(App.Path & "\DENSITY\ACTIVITY.LOG") <> "" Then
    FileCopy App.Path & "\DENSITY\ACTIVITY.LOG", "a:\ACTIVITY.LOG"
End If
If Dir("a:\" & Dir(tmpfile)) <> "" Then
    MsgBox "File successfully copied to disk.", vbOKOnly + vbInformation,
"Success!"
    Exit Sub
Else
    MsgBox "Copy was not successful. Please try again."
    Exit Sub
End If
'End Select
Else
    End If
End If
Case 7
    Exit Sub
End Select
Else
    Exit Sub
End If

Exit Sub
DiskError:
R = MsgBox("Error reading drive A:", vbRetryCancel + vbCritical, "Disk Error!")
Select Case R
    Case 4 ' Retry
        GoTo RETRY
    Case 2 ' Cancel
        Exit Sub
    Case Else
        Exit Sub
End Select
End Sub

Private Sub mnuFileTemp_Click()
Dim lngHnd As Long
'lngHnd = FileWork.CopyFile("C:\autoexec.bat", "D:\filex.bak")
'If filwork.Exists("C:\filex.bak") = True Then
'    MsgBox "File copied."
'Else
'    MsgBox "File NOT copied."
'End If
End Sub

Private Sub mnuFileViewLog_Click()
On Error Resume Next
Debug.Print Dir(App.Path & "\DENSITY\ACTIVITY.LOG")
If Dir(App.Path & "\DENSITY\ACTIVITY.LOG") = "" Then
    MsgBox "Activity Log has not yet been created."
    Exit Sub
End If
Shell "C:\WINDOWS\notepad.exe " & App.Path & "\DENSITY\ACTIVITY.LOG", vbNormalFocus

End Sub

Private Sub mnuFTF_Click()
'Screen.MousePointer = 11
If ServPwordFlag = 1 Then
frmSPW.Label2.Caption = ServPword
frmSPW.Show 1
'Screen.MousePointer = 0
If frmSPW.Label2.Caption = "GO" Then
    Unload frmSPW
    frmFTF.Show
End If
Else
```

```
        frmFTF.Show
    End If
End Sub

Private Sub mnuGageLog_Click()
'Screen.MousePointer = 11
If ServPwordFlag = 1 Then
    frmSPW.Label2.Caption = ServPword
    frmSPW.Show 1
'Screen.MousePointer = 0
If frmSPW.Label2.Caption = "GO" Then
    Unload frmSPW
    frmLogFile.Show
End If
Else
    frmLogFile.Show
End If
End Sub

Private Sub mnuGetDensities_Click()
Screen.MousePointer = 11
frmDensity.Show
Screen.MousePointer = 0
End Sub

Private Sub mnuHelpContents_Click()
Dim MyPath$
MyPath = Dir(App.Path + "\ECPW.HLP", vbNormal)
'Debug.Print "MyPath1= "; MyPath
'Debug.Print "App.Path= "; App.Path
'Debug.Print App.Path + "\ECPW.HLP"
CMDialog1.HelpFile = App.Path + "\ECPW.HLP"
CMDialog1.HelpCommand = cdlHelpContents
CMDialog1.ShowHelp
End Sub

Private Sub mnuOpEnraf_Click()
'Screen.MousePointer = 11
'    frmOpEnraf.Show 0
'Screen.MousePointer = 0
If frmMain.Height < 4650 Then
    frmMain.Height = 4650
Else
    frmMain.Height = 2856
End If
End Sub

Private Sub mnuOptionsAdministrator_Click()
'Screen.MousePointer = 11
frmSPW.Caption = "Admin Password Entry"
Debug.Print "AdminPword= "; AdminPword
frmSPW.Label2.Caption = AdminPword
frmSPW.Show 1
frmSPW.Caption = "Service Password Entry"
'Screen.MousePointer = 0
If frmSPW.Label2.Caption = "GO" Then
    Unload frmSPW
    frmPword.Show
End If
End Sub

Sub mnuOptionsDemo_Click()
If mnuOptionsDemo.checked = False Then
    mnuOptionsDemo.checked = True
    TestMode = 1
    TestItemCount = 1
    Dim F, Line$
    F = FreeFile
    Open App.Path & "\LOG\COMTEST.LOG" For Input As F
    While Not EOF(F)
```

```
Line Input #F, Line
TestItems(TestItemCount) = Line
'Debug.Print "TestItems("; TestItemCount; ")= "; TestItems(TestItemCount)
TestItemCount = TestItemCount + 1
'Debug.Print "TestItemCount= "; TestItemCount
Wend
Close #F
Debug.Print "TestItemCount= "; TestItemCount
frmDemo.Show
Else
    mnuOptionsDemo.checked = False
    TestMode = 0
End If

End Sub

Private Sub mnuOptionsDFont_Click()
    ' Set Cancel to True
    CMDialog1.CancelError = True
    On Error GoTo Errhandler
    ' Set the Flags property
    CMDialog1.Flags = cdlCFEffects Or cdlCFBoth Or cdlCFScreenFonts
    ' Display the Font dialog box
    'CMDialog1.ShowFont
    CMDialog1.Color = txtLevelDisplay.ForeColor
    CMDialog1.FontName = txtLevelDisplay.Font.Name
    CMDialog1.FontSize = txtLevelDisplay.Font.Size
    CMDialog1.FontBold = txtLevelDisplay.Font.Bold
    CMDialog1.FontItalic = txtLevelDisplay.Font.Italic
    CMDialog1.ShowFont
    If txtLevelDisplay.Font.Name <> CMDialog1.FontName Then
        Dim R%, m$
        m = "Selecting a font other than TERMINAL may cause the" & vbNewLine
        m = m & "ARROW UP / DOWN feature to be disabled." & vbNewLine
        m = m & "Are you sure you want to change fonts?"
        R = MsgBox(m, vbYesNo + vbExclamation, "Change Font?")
        If R <> 6 Then
            Exit Sub
        End If
    End If
    txtLevelDisplay.Font.Name = CMDialog1.FontName
    txtLevelDisplay.Font.Size = CMDialog1.FontSize
    txtLevelDisplay.Font.Bold = CMDialog1.FontBold
    txtLevelDisplay.Font.Italic = CMDialog1.FontItalic
    txtLevelDisplay.Font.Underline = CMDialog1.FontUnderline
    txtLevelDisplay.Font.Strikethru = CMDialog1.FontStrikethru
    txtLevelDisplay.ForeColor = CMDialog1.Color

    txtDisplay2.Font.Name = CMDialog1.FontName
    txtDisplay2.Font.Size = CMDialog1.FontSize
    txtDisplay2.Font.Bold = CMDialog1.FontBold
    txtDisplay2.Font.Italic = CMDialog1.FontItalic
    txtDisplay2.Font.Underline = CMDialog1.FontUnderline
    txtDisplay2.Font.Strikethru = CMDialog1.FontStrikethru
    txtDisplay2.ForeColor = CMDialog1.Color

    txtLevelDisplay.Text = "00000.00"
    txtDisplay2.Text = "Test Text..."

    Exit Sub
Errhandler:
    ' User pressed the Cancel button
End Sub

Private Sub mnuReset_Click()
    On Error GoTo Err
    ResetSystem
    Exit Sub
Err:
    MsgBox "Error encountered when user reset the program"
    Exit Sub
```

```
End Sub

Private Sub mnuService_Click()
'Screen.MousePointer = 11
If ServPwordFlag = 1 Then
frmSPW.Label2.Caption = ServPword
frmSPW.Show 1
'Screen.MousePointer = 0
If frmSPW.Label2.Caption = "GO" Then
Unload frmSPW
frmService.Show
End If
Else
frmService.Show
End If
End Sub

Private Sub mnuSetupGauge_Click()
On Error GoTo Err
frmCFG.Left = ((Screen.Width - (frmCFG.Width + frmProgress.Width)) / 2)
frmCFG.Top = frmMain.Top + frmMain.Height
frmCFG.Show

Dim request(1 To 50) As String, answer(1 To 50) As String, CR
Dim R(1 To 50) As String
Dim comment As String
Dim echo(1 To 50) As String
CR = Chr(13) + Chr(10)

'Load items into an array
request(1) = "LD"
R(1) = "Level Dimension: "
request(2) = "DI"
R(2) = "Density Dimension: "
request(3) = "TS"
R(3) = "Transmission Speed: "
request(4) = "TT"
R(4) = "Tank Top Level: "
request(5) = "RL"
R(5) = "Reference Level: "
request(6) = "DW"
R(6) = "Displacer Weight: "
request(7) = "HH"
R(7) = "High High Alarm Set Point: "
request(8) = "HA"
R(8) = "High Alarm Set Point: "
request(9) = "LA"
R(9) = "Low Alarm Set Point: "
request(10) = "LL"
R(10) = "Low-Low Alarm Set Point: "
request(11) = "M2"
R(11) = "Lock Test Limit Switch Setting: "
request(12) = "MH"
R(12) = "Motor High Set Point: "
request(13) = "ML"
R(13) = "Motor Low Set Point: "
request(14) = "ES"
R(14) = "SPU Error: "
request(15) = "EP"
R(15) = "XPU Error: "
request(16) = "S1"
R(16) = "Setpoint 1: "
request(17) = "S2"
R(17) = "Setpoint 2: "
request(18) = "S3"
R(18) = "Setpoint 3: "
request(19) = "SD"
R(19) = "Density Scan Direction: "
request(20) = "TA"
R(20) = "Transmission Address: "
request(21) = "Y4"
```

R(29) = "Atm. Air Density"

End If

End If

```
Open App.Path & "\\LOG\\CONFIGCHK.LOG" For Append As #F
```

HNF-1569
Revision 1

```

Print #F, frmCFG.txtCFG.Text
Close #F

If Abort = 1 Then
    frmCFG.cmdAbort.Enabled = False
    frmCFG.cmdClose_Click
    If DisabledLevelScan = 1 Then
        frmMain.cmdGetCurrentLevel_Click
    End If
    Abort = 0
    frmCFG.cmdClose.Enabled = True
    Exit Sub
End If
frmProgress.pBar.Value = frmProgress.pBar.Min
frmProgress.Hide
frmCFG.cmdClose.Enabled = True
Err:
If DisabledLevelScan = 1 Then frmMain.cmdGetCurrentLevel_Click
If frmProgress.Visible = True Then frmProgress.Hide
If frmCFG.cmdAbort.Enabled = True Then frmCFG.cmdAbort.Enabled = False

End Sub

Private Sub mnuViewOpen_Click()
On Error GoTo ErrorHandler
    CMDialog1.CancelError = True
    CMDialog1.Filter = "All Files (*.*)|*.r|RQS Files (*.rqs)|*.rqs|RPL Files (*.rpl)|*.rpl|LOG
Files (*.log)|*.log"
    ' Specify default filter.
    CMDialog1.FilterIndex = 1
    ' Display the File Open dialog box.
    CMDialog1.ShowOpen
    Shell "C:\WINDOWS\NOTEPAD.EXE " & CMDialog1.filename, vbNormalFocus

ErrorHandler:
End Sub

Private Sub Timer1_Timer()
    TimerTrap = 1
End Sub

Private Sub tmrLevel_Timer()
On Error GoTo ErrorHandler
DoEvents
' When this timer is enabled, it sends a request to the ciu. via
' ciucom. The return values (echo) are decoded using decode(). The Sub decode, extracts the level
value.

' If desired, a command may be placed in the SpecialCommand string,
' which will be picked up by this routine and sent to the gauge instead
' of the "1B" command. If a special command is entered, the program
' detects this and sends a 1B command in addition.

Dim oldlevel$
Dim answer$
Dim request As String
Dim request2 As String
Dim echo As String
Dim echoLevel As String
Dim i%
Dim CR
CR = Chr(13) + Chr(10)
Dim LevelNum As Double
Dim tmp$
Dim Msg$

'clear status bar
sBar.SimpleText = ""

If onDisconnect_Click = 1 Then

```

HNF-1569
Revision 1

```
Exit Sub
End If

'Validate communication request
If Left(ciuadres, 1) = "V" Or Left(ciuadres, 1) = "W" Then
    sBar.SimpleText = "uLect Communication not supported..."
    frmCIUSetup.Show 0
    Exit Sub
End If

request2 = "Z2B" 'Line 2 of the display
' Check for a SpecialCommand waiting to be sent
If SpecialCommand <> "" Then
    ' Send the command
    ciucom SpecialCommand, echoLevel, status
    decode Mid(SpecialCommand, 2), echoLevel, 64, answer
    frmMain.txtRE.Text = frmMain.txtRE.Text & answer
    'request = SpecialCommand
    SpecialCommand = ""
    Debug.Print "Special Request in Timer = "; request
Else
    'request = "B"
    'request = "Z1B"
    'Debug.Print "Regular Request in Timer = "; request
End If

'=====
' Send the request to the CIU/Gauge
'=====
If onDisconnect_Click = 1 Then Exit Sub

GetALevel s1B, s2B

'Get Request Data
'For i = 1 To 3
'    ciucom request, echoLevel, status
'    If Len(echoLevel) > 3 Or onDisconnect_Click = 1 Then
'        Exit For
'    End If
'Next i
'If i = 3 Then
'    Msg = "The CIU is not responding properly." + CR
'    Msg = Msg + "Please troubleshoot. (Try clicking the 'L' button)..."
'    Exit Sub
'End If

If onDisconnect_Click = 1 Then Exit Sub

'Check for errors and set counter
'If InStr(echoLevel, "!0") Then
'    Msg = GetErrorMessage(Mid(echoLevel, InStr(echoLevel, "!0"), 4))
'    Msg = Msg + CR
'    Msg = Msg + "Please Troubleshoot. Scans will be terminated..."
'    MsgBox Msg, vbOKOnly + vbCritical
'    cmdDisconnect_Click
'    Exit Sub
'End If

'If request was not a level, get a level
'If InStr(request, "1B") = 0 Then
'    DisplayStuff request, request2, echoLevel, echo
'    If onDisconnect_Click = 1 Then Exit Sub
'    For i = 1 To 3
'        ciucom "Z1B", echoLevel, status
'        If Len(echoLevel) > 3 Or onDisconnect_Click = 1 Then
'            Exit For
'        End If
'    Next i
'    If i = 3 Then
```



```

'      Delay (0.3)
'      Exit Sub
'    End If
'End If

If onDisconnect_Click = 1 Then Exit Sub
'For i = 1 To 3
'  ciucom request2, echo, status
'  If Len(echo) > 3 Or onDisconnect_Click = 1 Then
'    Exit For
'  End If
'Next i
'If i = 3 Then
'  Delay (0.3)
'  Exit Sub
'End If

'=====
' request sent....
'=====

'

'=====
' Prepare and process response from gauge
'=====
If onDisconnect_Click = 1 Then Exit Sub
'DisplayStuff request, request2, echoLevel, echo
'If onDisconnect_Click = 1 Then Exit Sub

'Make sure displacer is not going to hit the adapter flange
If InStr(s2B, "CA") Then
  GetALevel
  If TT - 3 <= CL Then
    ciucom "2FR", echo, status
    MsgBox "You will have to use the PET or PCET to raise the displacer further than this...",
vbOKOnly + vbCritical, "Service Level Function!"
  End If
End If

initializing = 0
If cmdDisconnect.Enabled = False Then
  cmdDisconnect.Enabled = True
  mnuDisconnect.Enabled = True
End If
Exit Sub

ErrorHandler:
  MsgBox "Error encountered during scans; resetting system"
  ResetSystem
  Exit Sub

End Sub

Private Sub txtDisplay2_KeyPress(KeyAscii As Integer)
KeyAscii = 0
End Sub

Private Sub txtDisplay2_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single)
  If Button = 2 Then
    frmMain.PopupMenu mnuOptions
  End If
End Sub

Private Sub txtLevelDisplay_KeyPress(KeyAscii As Integer)
KeyAscii = 0
End Sub

Sub SetupConfigComments(ByVal request As String, ByVal answer As String, ByRef comment As String)

```

```
Select Case request
  Case "LD"
    Select Case answer
      Case "M"
        comment = "Meters"
      Case "F"
        comment = "Feet"
      Case "I"
        comment = "Inches"
      Case "P"
        comment = "Feet-Inches"
    End Select

  Case "DI"
    Select Case answer
      Case "K"
        comment = "Kg/m3"
      Case "A"
        comment = "degrees API"
      Case "L"
        comment = "lbs/ft3"
    End Select

  Case "SD"
    Select Case answer
      Case "U"
        comment = "UP"
      Case "D"
        comment = "DOWN"
    End Select
  Case "DE"
    Select Case answer
      Case "C"
        comment = "compensated innage"
      Case "I"
        comment = "Innage"
      Case "N"
        comment = "Instantaneous Ullage"
      Case "U"
        comment = "Ullage"
    End Select
  Case "DF"
    Select Case answer
      Case "A"
        comment = "Level + temperature + gauge status"
      Case "B"
        comment = "Level + gauge status"
      Case "C"
        comment = "Average gas temperature"
      Case "D"
        comment = "Average product temperature"
      Case "I"
        comment = "Servo Density (last value)"
      Case "J"
        comment = "analog output"
    End Select
  Case Else
    comment = ""
End Select
End Sub

Private Sub txtLevelDisplay_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single)
  If Button = 2 Then
    frmMain.PopupMenu mnuOptions
  End If
End Sub

Private Sub cmdCA_Click()
```

HNF-1569
Revision 1

```
Dim message As String
Dim Reply As Integer
Dim CR
CR = Chr(13) + Chr(10)

message = "Warning: Raising the Displacer "
message = message + "creates a Radiological Exposure Hazard!"
message = message + CR + "Ensure HPT coverage at Gauge..."
message = message + CR + "Do you wish to continue?"
'SysMesg message, "vbYN", "vbWarning", Reply
Reply = MsgBox(message, vbYesNo + vbExclamation, "Resolve HPT Coverage...")

' Reply = MsgBox(message, 1 + 48, opWarningTitle)
If Reply = 6 Then

    'Do a CA, raise displacer

If frmMain.tmrLevel.Enabled = True Then
    SpecialCommand = "ZCA"
    txtRE.Text = txtRE.Text + "Sending> " + SpecialCommand + CR
    frmMain.sBar.SimpleText = "Raising displacer..."
Exit Sub
Else

    frmMain.cmdGetCurrentLevel_Click
If ja = 0 Then 'ja <= 0 means no gauge
    'Debug.Print "Exiting TG because ja = 0"
    MsgBox "Gauge Not Found!"
Exit Sub
Else
    'Debug.Print "Making SpecialCommand = ZCA"
    SpecialCommand = "ZCA"
    txtRE.Text = txtRE.Text + "Sending> " + SpecialCommand + CR
    'responseFlag = 0
    'Send message to display that a TG is being
    'performed.
    frmMain.sBar.SimpleText = "Raising Displacer..."
End If

End If
End If
End Sub

Private Sub cmdFR_Click()

If frmMain.tmrLevel.Enabled = True Then
    SpecialCommand = "ZFR"
    txtRE.Text = txtRE.Text + "Sending> " + SpecialCommand + vbCrLf
    frmMain.sBar.SimpleText = "Halting displacer movement..."
Exit Sub
Else
    'Freeze command
    frmMain.cmdGetCurrentLevel_Click
If ja = 0 Then 'ja <= 0 means no gauge
    'Debug.Print "Exiting FR because ja = 0"
    MsgBox "Gauge Not Found!"
Exit Sub
Else
    'Debug.Print "Making SpecialCommand = ZFR"
    SpecialCommand = "ZFR"
    txtRE.Text = txtRE.Text + "Sending> " + SpecialCommand + vbCrLf
    'responseFlag = 0
    'Send message to display that a TG is being
    'performed.
    frmMain.sBar.SimpleText = "Halting displacer movement..."
End If

End If
End Sub
```

HNF-1569
Revision 1

```

Sub cmdSEND_Click()
    Dim message As String
    Dim Reply As Integer
    Dim CR
    CR = Chr(13) + Chr(10)

    txtCustomCommand.Text = UCase(txtCustomCommand.Text)
    If ServPwordFlag = 1 Then
        'Check to make sure this is not a service level command
        If txtCustomCommand.Text = "W2=ENRAF2" Or txtCustomCommand.Text = "W1=ENRAF1" Then
            frmSPW.Label2.Caption = ServPword
            frmSPW.Show 1
            If frmSPW.Label2.Caption <> "GO" Then
                Exit Sub
            Else
                frmSPW.Label2.Caption = ""
                Unload frmSPW
            End If
        End If
    End If
    End If

    If txtCustomCommand.Text = "" Then
        Exit Sub
    Else
        'IF A CA COMMAND !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
        If (txtCustomCommand.Text = "CA" Or txtCustomCommand.Text = "GU") Then ' display a
warning..

            message = "Warning: Raising the Displacer "
            message = message + "creates a Radiological Exposure Hazard!"
            message = message + CR + "Ensure HPT coverage at Gauge..."
            message = message + CR + "Do you wish to continue?"

            'SysMesg message, "vbYN", "vbWarning", Reply
            Reply = MsgBox(message, vbYesNo + vbExclamation, "Resolve HPT Coverage...")

            If Reply = 6 Then ' do it

                'Send a command to the gauge

                If frmMain.tmrLevel.Enabled = True Then
                    SpecialCommand = "Z" + txtCustomCommand.Text
                    txtCustomCommand.Text = ""
                    txtRE.Text = txtRE.Text + "Sending> " + SpecialCommand
                    'responseFlag = 0
                    Exit Sub
                Else
                    'Connect and get a reading.
                    frmMain.cmdGetCurrentLevel_Click
                    If ja = 0 Then 'ja <= 0 means no gauge
                        'Debug.Print "Exiting TG because ja = 0"
                        MsgBox "Gauge Not Found!"
                        Exit Sub
                    Else
                        SpecialCommand = "Z" + txtCustomCommand.Text
                        txtRE.Text = txtRE.Text + "Sending> " + SpecialCommand + CR
                        'responseFlag = 0
                        'Send message to display that a TG is being
                        'performed.
                        frmMain.sBar.SimpleText = "Performing " + SpecialCommand
                        txtCustomCommand.Text = ""
                    End If
                End If
            End If
        Else 'IF OTHER THAN CA COMMAND !!!!!!!!!!!!!!!!!!!!!!!

            'Send a command to the gauge
            If frmMain.tmrLevel.Enabled = True Then
                SpecialCommand = "Z" + txtCustomCommand.Text
            End If
        End If
    End If
End Sub

```

HNF-1569
Revision 1

```
txtCustomCommand.Text = ""
txtRE.Text = txtRE.Text + "Sending> " + SpecialCommand + CR
' responseFlag = 0
Exit Sub
Else

'Connect and get a reading.
frmMain.cmdGetCurrentLevel_Click
If ja = 0 Then 'ja <= 0 means no gauge
'Debug.Print "Exiting TG because ja = 0"
MsgBox "Gauge Not Found!"
Exit Sub
Else
SpecialCommand = "Z" + txtCustomCommand.Text
txtRE.Text = txtRE.Text + "Sending> " + SpecialCommand + CR
'responseFlag = 0
'Send message to display that a TG is being
'performed.
frmMain.sBar.SimpleText = "Performing " + SpecialCommand
txtCustomCommand.Text = ""
End If
End If
End If
End If

End Sub

Private Sub cmdTG_Click()

Dim CR
CR = Chr(13) + Chr(10)

If frmMain.tmrLevel.Enabled = True Then
SpecialCommand = "ZTG"
txtRE.Text = txtRE.Text + "Sending> " + SpecialCommand + CR
frmMain.sBar.SimpleText = "Performing Self Diagnostics on Gauge..."
Exit Sub
Else
frmMain.cmdGetCurrentLevel_Click
If ja = 0 Then 'ja <= 0 means no gauge
'Debug.Print "Exiting TG because ja = 0"
MsgBox "Gauge Not Found!"
Exit Sub
Else
'Debug.Print "Making SpecialCommand = ZTG"
SpecialCommand = "ZTG"
txtRE.Text = txtRE.Text + "Sending> " + SpecialCommand + CR
'responseFlag = 0
'Send message to display that a TG is being
'performed.
frmMain.sBar.SimpleText = "Performing Self Diagnostics on Gauge..."
End If
End If
End Sub

Private Sub cmdUN_Click()

If frmMain.tmrLevel.Enabled = True Then
SpecialCommand = "ZUN"
txtRE.Text = txtRE.Text + "Sending> " + SpecialCommand + vbCrLf
frmMain.sBar.SimpleText = "Unlocking Gauge...."
Exit Sub
Else
'Unlock the Gauge
frmMain.cmdGetCurrentLevel_Click
If ja = 0 Then 'ja <= 0 means no gauge
'Debug.Print "Exiting TG because ja = 0"
MsgBox "Gauge Not Found!"
Exit Sub
Else
```

HNF-1569
Revision 1

```
'Debug.Print "Making SpecialCommand = ZUN"
SpecialCommand = "ZUN"
txtRE.Text = txtRE.Text + "Sending> " + SpecialCommand + vbCrLf
'responseFlag = 0
'Send message to display that a TG is being
'performed.
frmMain.sBar.SimpleText = "Placing gauge into service..."
End If
End Sub

Private Sub txtCustomCommand_KeyPress(KeyAscii As Integer)
    If (KeyAscii = 13) Then
        cmdSEND_Click
    End If
End Sub

Private Sub txtRE_Change()
    txtRE.SelStart = Len(txtRE.Text)
End Sub

End frmmain.frm
=====
Begin FRMSG.FRM

VERSION 5.00
Begin VB.Form frmMsg
    ClientHeight    = 1575
    ClientLeft      = 4440
    ClientTop       = 3795
    ClientWidth     = 5325
    LinkTopic       = "Form1"
    MaxButton       = 0   'False
    PaletteMode     = 1   'UseZOrder
    ScaleHeight     = 1575
    ScaleWidth      = 5325
    Begin VB.CommandButton cmdCancel
        Caption      = "&Cancel"
        Height       = 372
        Left         = 2880
        TabIndex     = 2
        Top          = 1080
        Visible      = 0   'False
        Width        = 1092
    End
    Begin VB.CommandButton cmdOkay
        Caption      = "&OK"
        Height       = 372
        Left         = 1560
        TabIndex     = 1
        Top          = 1080
        Width        = 1092
    End
    Begin VB.Label lblStatus
        Height       = 252
        Left         = 120
        TabIndex     = 3
        Top          = 1200
        Visible      = 0   'False
        Width        = 972
    End
    Begin VB.Label lblMsg
        AutoSize     = -1   'True
        Height       = 792
        Left         = 120
        TabIndex     = 0
        Top          = 240
        Width        = 5052
    End
End
Attribute VB_Name = "frmMsg"
Attribute VB_GlobalNameSpace = False
```

HNF-1569
Revision 1

```
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
' =====
' This module/form is a custom msg box that I needed in order
' to allow it to be non modal.
' =====
Option Explicit

Sub cmdCancel_Click()
lblStatus.Caption = "CANCEL"
frmMsg.Hide
End Sub

Private Sub cmdOkay_Click()
lblStatus.Caption = "OK"
frmMsg.Hide
End Sub

End FRMSG.FRM
=====
Begin frmProgress.frm
VERSION 5.00
Object = "{6B7E6392-850A-101B-AFC0-4210102A8DA7}#1.2#0"; "COMCTL32.OCX"
Begin VB.Form frmProgress
    BorderStyle = 4 'Fixed ToolWindow
    Caption = "Getting Current Config. Please wait..."
    ClientHeight = 1260
    ClientLeft = 30
    ClientTop = 270
    ClientWidth = 4050
    LinkTopic = "Form1"
    MaxButton = 0 'False
    MinButton = 0 'False
    ScaleHeight = 1260
    ScaleWidth = 4050
    ShowInTaskbar = 0 'False
    StartUpPosition = 3 'Windows Default
    Begin VB.CommandButton cmdCancel
        Caption = "&Cancel"
        Height = 252
        Left = 1320
        TabIndex = 2
        Top = 960
        Width = 1212
    End
    Begin ComctlLib.ProgressBar pBar
        Height = 252
        Left = 120
        TabIndex = 0
        Top = 600
        Width = 3852
        _ExtentX = 6800
        _ExtentY = 450
        _Version = 327680
        Appearance = 1
    End
    Begin VB.Label lblStatus
        Height = 252
        Left = 240
        TabIndex = 1
        Top = 360
        Width = 3492
    End
End
Attribute VB_Name = "frmProgress"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
Private Sub cmdCancel_Click()
```

```
frmCFG.cmdAbort_Click
End Sub

Private Sub Form_Load()

End Sub

End frmProgress.frm
=====
Begin frmPword.frm
VERSION 5.00
Begin VB.Form frmPword
    BorderStyle = 1 'Fixed Single
    Caption = "Change Password"
    ClientHeight = 3465
    ClientLeft = 30
    ClientTop = 270
    ClientWidth = 6330
    LinkTopic = "Form1"
    MaxButton = 0 'False
    MinButton = 0 'False
    ScaleHeight = 3465
    ScaleWidth = 6330
    StartUpPosition = 3 'Windows Default
Begin VB.Frame Frame2
    Height = 2775
    Left = 3960
    TabIndex = 13
    Top = 240
    Width = 2175
Begin VB.TextBox txtWRSSStart
    Height = 285
    Left = 240
    TabIndex = 16
    Top = 960
    Width = 735
End
Begin VB.CheckBox chkServPwd
    Caption = "Enable Service Password"
    Height = 375
    Left = 240
    TabIndex = 14
    Top = 360
    Width = 1575
End
Begin VB.Label Label4
    Caption = "WRSS Auto Start Level"
    Height = 375
    Left = 1080
    TabIndex = 17
    Top = 960
    Width = 975
End
End
Begin VB.CommandButton cmdMore
    Caption = "&More >>"
    Height = 375
    Left = 2520
    TabIndex = 12
    Top = 3000
    Width = 855
End
Begin VB.Frame Frame1
    Caption = "Choose Password to Change"
    Height = 615
    Left = 120
    TabIndex = 8
    Top = 120
    Width = 3495
Begin VB.OptionButton optAdmin
    Caption = "Admin"
    Height = 255
```



```
Left          = 2400
TabIndex      = 11
Top           = 240
Width         = 855
End
Begin VB.OptionButton optSystem
Caption       = "System"
Height       = 255
Left         = 1320
TabIndex     = 10
Top          = 240
Width        = 855
End
Begin VB.OptionButton optService
Caption       = "Service"
Height       = 255
Left         = 360
TabIndex     = 9
Top          = 240
Value        = -1 'True
Width        = 1095
End
End
Begin VB.CommandButton cmdCancel
Caption       = "&Cancel"
Height       = 372
Left         = 1440
TabIndex     = 7
Top          = 3000
Width        = 852
End
Begin VB.CommandButton cmdOkay
Caption       = "&OK"
Height       = 372
Left         = 360
TabIndex     = 6
Top          = 3000
Width        = 852
End
Begin VB.TextBox txtReNew
Height       = 288
IMEMode      = 3 'DISABLE
Left         = 360
PasswordChar = "*"
TabIndex     = 2
Top          = 2520
Width        = 3012
End
Begin VB.TextBox txtNew
Height       = 288
IMEMode      = 3 'DISABLE
Left         = 360
PasswordChar = "*"
TabIndex     = 1
Top          = 1800
Width        = 3012
End
Begin VB.TextBox txtOld
Height       = 288
IMEMode      = 3 'DISABLE
Left         = 360
PasswordChar = "*"
TabIndex     = 0
Top          = 1080
Width        = 3012
End
Begin VB.CommandButton cmdLess
Caption       = "<< &Less"
Height       = 375
Left         = 2520
TabIndex     = 15
```

```
Top          = 3000
Visible      = 0 'False
Width        = 855
End
Begin VB.Label Label3
Caption      = "Re-enter New Password"
Height      = 255
Left        = 360
TabIndex    = 5
Top         = 2280
Width       = 2775
End
Begin VB.Label Label2
Caption      = "Enter New Password:"
Height      = 255
Left        = 360
TabIndex    = 4
Top         = 1560
Width       = 2775
End
Begin VB.Label Label1
Caption      = "Enter Old Password:"
Height      = 255
Left        = 360
TabIndex    = 3
Top         = 840
Width       = 2055
End
End
Attribute VB_Name = "frmPword"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
' =====
' This module/form is used to change all passwords, and to
' enable/disable the service password. This is so that an
' administrator level user can utilize service function without
' entering the service password everytime.
' =====
Option Explicit

Private Sub cmdCancel_Click()
Unload Me
End Sub

Private Sub cmdLess_Click()
Me.Width = 3840
cmdLess.Visible = False
cmdMore.Visible = True
End Sub

Private Sub cmdMore_Click()
Me.Width = 6420
cmdMore.Visible = False
cmdLess.Visible = True
End Sub

Private Sub cmdOkay_Click()
If txtOld.Text <> "" Then
Debug.Print "txtOld.text= "; txtOld.Text
Debug.Print "ServPword= "; ServPword
Debug.Print "MainPword= "; MainPword
Debug.Print "AdminPword= "; AdminPword
Debug.Print "txtNew.Text= "; txtNew.Text
Debug.Print "txtReNew.text= "; txtReNew.Text
If optService.Value = True Then
Debug.Print "Changing Service Pword"
If UCase(txtOld.Text) = UCase(ServPword) Then
If UCase(txtNew.Text) = UCase(txtReNew.Text) Then
```

HNF-1569
Revision 1

```
ServPword = UCase(txtNew.Text)
SaveSetting App.title, "Psets", "ServP", ServPword
MsgBox "Password successfully changed!"
GoTo DoMore
Else
    MsgBox "Your re-entered password did not match!"
    Exit Sub
End If
Else
    MsgBox "Your Old Password doesn't match!"
End If
ElseIf optSystem.Value = True Then
    Debug.Print "Changing System Pword"
    If UCase(txtOld.Text) = UCase(MainPword) Then
        If UCase(txtNew.Text) = UCase(txtReNew.Text) Then
            MainPword = UCase(txtNew.Text)
            SaveSetting App.title, "Psets", "MainP", MainPword
            MsgBox "Password successfully changed!"
            GoTo DoMore
        Else
            MsgBox "Your re-entered password did not match!"
            Exit Sub
        End If
    Else
        MsgBox "Your Old Password doesn't match!"
    End If
Else 'optAdmin.Value = True
    Debug.Print "Changing Admin Pword"
    If UCase(txtOld.Text) = UCase(AdminPword) Then
        If UCase(txtNew.Text) = UCase(txtReNew.Text) Then
            AdminPword = UCase(txtNew.Text)
            SaveSetting App.title, "Psets", "AdminP", AdminPword
            MsgBox "Password successfully changed!"
            GoTo DoMore
        Else
            MsgBox "Your re-entered password did not match!"
            Exit Sub
        End If
    Else
        MsgBox "Your Old Password doesn't match!"
    End If
End If
End If
End If
DoMore:
    If chkServPwd.Value = 1 Then
        ServPwordFlag = 1
    Else
        ServPwordFlag = 0
    End If
    WRSSStart = txtWRSSStart.Text
    Unload Me
End Sub

Private Sub Form_Load()
On Error Resume Next
Me.Height = 3840
Me.Width = 3840
If ServPwordFlag = 1 Then
    chkServPwd.Value = checked
    Debug.Print "ServPwordFlag= "; ServPwordFlag
    Debug.Print "Setting chkservpwd.value= "; chkServPwd.Value
Else
    chkServPwd.Value = unchecked
    Debug.Print "ServPwordFlag= "; ServPwordFlag
    Debug.Print "Setting chkservpwd.value = "; chkServPwd.Value
End If
txtWRSSStart.Text = WRSSStart
End Sub

Private Sub optService_Click()
txtOld.Text = ""
```

```
txtNew.Text = ""
txtReNew.Text = ""
End Sub
```

```
Private Sub optSystem_Click()
txtOld.Text = ""
txtNew.Text = ""
txtReNew.Text = ""
End Sub
```

End frmPword.frm

=====

Begin frmService.frm

VERSION 5.00

Begin VB.Form frmService

```
BorderStyle = 1 'Fixed Single
Caption = "PC Enraf Terminal"
ClientHeight = 3390
ClientLeft = 2985
ClientTop = 3900
ClientWidth = 5535
HelpContextID = 600
LinkTopic = "Form1"
MaxButton = 0 'False
PaletteMode = 1 'UseZOrder
ScaleHeight = 3390
ScaleWidth = 5535
StartupPosition = 2 'CenterScreen
```

Begin VB.CommandButton cmdEX

```
Caption = "EX"
Height = 252
Left = 2160
TabIndex = 23
ToolTipText = "Exit Protection/Reinitialize"
Top = 360
Width = 1092
```

End

Begin VB.CommandButton cmdW2

```
Caption = "W2"
Height = 252
Left = 3240
TabIndex = 22
ToolTipText = "Protection Level 2"
Top = 360
Width = 1092
```

End

Begin VB.CommandButton cmdCA

```
Caption = "CA"
Height = 252
Left = 4320
TabIndex = 21
ToolTipText = "Perform Calibrate"
Top = 360
Width = 1092
```

End

Begin VB.CommandButton cmdClose

```
Caption = "&Close"
Height = 372
Left = 240
TabIndex = 20
Top = 2880
Width = 852
```

End

Begin VB.CommandButton cmdES

```
Caption = "ES"
Height = 252
Left = 1080
TabIndex = 19
ToolTipText = "Get XPU Error Message"
Top = 360
Width = 1092
```

End

HNF-1569
Revision 1

```
Begin VB.CommandButton cmdEP
    Caption       = "EP"
    Height        = 252
    Left          = 120
    TabIndex      = 18
    ToolTipText    = "Get SPU Error Message"
    Top           = 360
    Width         = 972
End
Begin VB.CommandButton cmdTG
    Caption       = "TG"
    Height        = 252
    Left          = 4320
    TabIndex      = 16
    ToolTipText    = "Repeatability check"
    Top           = 120
    Width         = 1092
End
Begin VB.CommandButton cmdUN
    Caption       = "UN"
    Height        = 252
    Left          = 3240
    TabIndex      = 15
    ToolTipText    = "Place gauge into service"
    Top           = 120
    Width         = 1092
End
Begin VB.CommandButton cmdFR
    Caption       = "FR"
    Height        = 252
    Left          = 2160
    TabIndex      = 14
    ToolTipText    = "Freeze the displacer at current position"
    Top           = 120
    Width         = 1092
End
Begin VB.CommandButton cmdLT
    Caption       = "LT"
    Height        = 252
    Left          = 1080
    TabIndex      = 13
    ToolTipText    = "Perform Lock Test"
    Top           = 120
    Width         = 1092
End
Begin VB.CommandButton cmdLevel
    Caption       = "Get a Level"
    Height        = 252
    Left          = 120
    TabIndex      = 12
    ToolTipText    = "Get a current displacer level"
    Top           = 120
    Width         = 972
End
Begin VB.CommandButton cmdB
    Caption       = "B"
    Height        = 252
    Left          = 1320
    TabIndex      = 11
    ToolTipText    = "B-Record"
    Top           = 1920
    Width         = 252
End
Begin VB.CommandButton cmdZ
    Caption       = "Z"
    Height        = 252
    Left          = 1680
    TabIndex      = 10
    ToolTipText    = "Z-Record"
    Top           = 1920
    Width         = 252
```

```

End
Begin VB.CommandButton cmdA
    Caption       = "A"
    Height        = 252
    Left          = 960
    TabIndex      = 9
    ToolTipText   = "Alarm Status Record"
    Top           = 1920
    Width         = 252
End
Begin VB.TextBox txtTOI
    Height        = 288
    Left          = 2040
    TabIndex      = 6
    Text          = "B"
    Top           = 2400
    Width         = 372
End
Begin VB.TextBox txtRecType
    Height        = 288
    Left          = 2040
    TabIndex      = 4
    Text          = "2"
    Top           = 1920
    Width         = 372
End
Begin VB.CommandButton cmdSend
    Caption       = "&Send"
    Height        = 372
    HelpContextID = 1210
    Left          = 1440
    TabIndex      = 3
    ToolTipText   = "Send command"
    Top           = 2880
    Width         = 972
End
Begin VB.TextBox txtRE
    Height        = 1692
    Left          = 2640
    MultiLine     = -1 'True
    ScrollBars    = 2 'Vertical
    TabIndex      = 2
    ToolTipText   = "Record of Gauge communication"
    Top           = 1560
    Width         = 2772
End
Begin VB.TextBox txtItem
    Alignment     = 1 'Right Justify
    Height        = 288
    Left          = 3360
    TabIndex      = 0
    Top           = 840
    Width         = 2052
End
Begin VB.Label Label5
    Caption       = "Communications Record"
    Height        = 252
    Left          = 2640
    TabIndex      = 17
    Top           = 1320
    Width         = 2172
End
Begin VB.Label Label4
    Caption       = "Label4"
    Height        = 132
    Left          = 600
    TabIndex      = 8
    Top           = 1080
    Width         = 12
End
Begin VB.Label Label3

```

```

        Alignment      = 1 'Right Justify
        Caption        = "Instrument Type"
        Height         = 252
        Left           = 600
        TabIndex       = 7
        Top            = 2400
        Width          = 1332
    End
    Begin VB.Label Label2
        Alignment      = 1 'Right Justify
        Caption        = "Record Type:"
        Height         = 252
        Left           = 120
        TabIndex       = 5
        Top            = 1560
        Width          = 1092
    End
    Begin VB.Label Label1
        Alignment      = 1 'Right Justify
        Caption        = "Command Entry:"
        ForeColor      = &H80000008&
        Height         = 252
        Left           = 1920
        TabIndex       = 1
        Top            = 840
        Width          = 1332
    End
End
Attribute VB_Name = "frmService"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
' =====
' This is a form that allows any string to be sent to a gauge.
' Normally, only z records are sent. But this form allows any
' record type to be sent to the gauge.
' =====

Option Explicit

Private Sub cmdA_Click()
    txtRecType.Text = "A"
End Sub
Private Sub cmdB_Click()
    txtRecType.Text = "B"
End Sub
Private Sub cmdCA_Click()
    txtItem.Text = "CA"
    txtRecType.Text = "Z"
    txtTOI.Text = "B"
    cmdSEND_Click
End Sub

Private Sub cmdClose_Click()
    frmService.Hide
End Sub

Private Sub cmdEP_Click()
    txtItem.Text = "EP"
    txtRecType.Text = "Z"
    txtTOI.Text = "B"
    cmdSEND_Click
End Sub

Private Sub cmdES_Click()
    txtItem.Text = "ES"
    txtRecType.Text = "Z"
    txtTOI.Text = "B"
    cmdSEND_Click
End Sub

```

```
Private Sub cmdEX_Click()  
txtItem.Text = "EX"  
txtRecType.Text = "Z"  
txtTOI.Text = "B"  
cmdSEND_Click  
End Sub  
  
Private Sub cmdFR_Click()  
txtItem.Text = "FR"  
txtRecType.Text = "Z"  
txtTOI.Text = "B"  
cmdSEND_Click  
End Sub  
  
Sub cmdLevel_Click()  
txtItem.Text = "LB"  
txtRecType.Text = "Z"  
txtTOI.Text = "B"  
cmdSEND_Click  
'Do  
' If frmMain.txtLevelDisplay.text <> "" Then Exit Do  
' DoEvents  
'Loop  
End Sub  
  
Private Sub cmdLT_Click()  
txtItem.Text = "LT"  
txtRecType.Text = "Z"  
txtTOI.Text = "B"  
cmdSEND_Click  
End Sub  
  
Private Sub cmdSEND_Click()  
'this sub seems to work best when trying to get initial  
'communications going with the gauge (better than tmrLevel)  
'I don't know why. But it is used frequently.  
Dim message As String  
Dim Reply As Integer  
Dim CR  
Dim tiString As String  
Dim PNString As String  
Dim AKString As String  
Dim ANYString As String  
Dim SVString As String  
Dim PAString As String  
Dim HTString As String  
Dim GTString As String  
Dim GAString As String  
Dim rType As String  
Dim ItemStr As String  
CR = Chr(13) + Chr(10)  
Dim echo$  
  
ItemStr = UCase(txtItem.Text)  
  
If InStr(ItemStr, "CA") <> 0 Or InStr(ItemStr, "LT") <> 0 Then  
message = "Warning: Raising the Displacer "  
message = message + "creates a Radiological Exposure Hazard!"  
message = message + CR + "Ensure HPT coverage at Gauge..."  
message = message + CR + "Do you wish to continue?"  
Reply = MsgBox(message, vbYesNo + vbExclamation, "Radiological Exposure Hazard!")  
If Reply <> 6 Then  
txtItem.Text = ""  
Exit Sub  
End If  
End If  
  
'Send a command to the gauge  
If frmMain.tmrLevel.Enabled = True Then  
frmMain.tmrLevel.Enabled = False  
frmMain.cmdGetCurrentLevel.Visible = True
```


HNF-1569
Revision 1

```
frmMain.cmdDisconnect.Visible = False
End If

'Make sure a gauge exists
If gaugeChecked = 0 Then
    checkGageAddress ta, ja
End If

If ja = 0 Then 'ja <= 0 means no gauge
    Exit Sub
Else
    'send the command to the gauge
    toi = Left(txtTOI.Text, 1)
    rType = Left(txtRecType.Text, 1)
    txtRE.Text = txtRE.Text + "sending= " + rType + ItemStr + CR

    ciucom rType + ItemStr, echo, status
    txtRE.Text = txtRE.Text + "echo= " + echo + CR
Select Case ItemStr
    Case "TI"
        decode "TI", echo, 6, tiString
        txtRE.Text = txtRE.Text + tiString + CR
    Case "PN"
        decode "PN", echo, 6, PNString
        txtRE.Text = txtRE.Text + PNString + CR
    Case "SV"
        decode "SV", echo, 64, SVString
        txtRE.Text = txtRE.Text + SVString + CR
    Case "AK"
        decode "AK", echo, 1, AKString
        txtRE.Text = txtRE.Text + AKString + CR
    Case "PA"
        decode "PA", echo, 3, PASTring
        txtRE.Text = txtRE.Text + PASTring + CR
    Case "HT", echo, 1, HTString
        txtRE.Text = txtRE.Text + HTString + CR
    Case "GT"
        decode "GT", echo, 1, GTString
        txtRE.Text = txtRE.Text + GTString + CR
    Case "GA"
        decode "GA", echo, 2, GASring
        txtRE.Text = txtRE.Text + GASring + CR
    Case "DP"
        decode "DP", echo, 1, ANYString
        txtRE.Text = txtRE.Text + ANYString + CR
    Case "DI"
        decode "DI", echo, 1, ANYString
        txtRE.Text = txtRE.Text + ANYString + CR
    Case "LD"
        decode "LD", echo, 1, ANYString
        txtRE.Text = txtRE.Text + ANYString + CR
    Case "DF"
        decode "DF", echo, 1, ANYString
        txtRE.Text = txtRE.Text + ANYString + CR
    Case "PS"
        decode "PS", echo, 1, ANYString
        txtRE.Text = txtRE.Text + ANYString + CR
    Case "1B"
        decode "1B", echo, 32, ANYString
        txtRE.Text = txtRE.Text + ANYString + CR
        txtRE.Text = txtRE.Text + Str(ZiBLevel(echo)) + CR
        frmMain.txtLevelDisplay.Text = ANYString
    Case "2B"
        decode "2B", echo, 32, ANYString
        txtRE.Text = txtRE.Text + ANYString + CR
        frmMain.txtDisplay2.Text = ANYString
    Case "1C"
        decode "1C", echo, 16, ANYString
        txtRE.Text = txtRE.Text + ANYString + CR
    Case "Y4"
        decode "Y4", echo, 8, ANYString
```

HNF-1569
Revision 1

```
txtRE.Text = txtRE.Text + ANYString + CR
Case "EP"
    decode "EP", echo, 3, ANYString
    txtRE.Text = txtRE.Text + ANYString + CR
    If Trim(ANYString) <> "000" Then
        message = GetErrorMessage(ANYString)
        MsgBox message, vbOKOnly + vbCritical
    End If
Case "ES"
    decode "ES", echo, 4, ANYString
    txtRE.Text = txtRE.Text + ANYString + CR
    If Trim(ANYString) <> "0000" Then
        message = GetSPUErrormessage(ANYString)
        MsgBox message, vbOKOnly + vbCritical
    End If
Case Else
    If rType = "B" Then
        decode "B", echo, 32, ANYString
        txtRE.Text = txtRE.Text + "BLevel= " + Str(BLevel(echo)) + CR
    Else
        decode ItemStr, echo, 64, ANYString
        txtRE.Text = txtRE.Text + ANYString + CR
    End If
End Select
End If
txtItem.Text = ""
End Sub

Private Sub cmdTG_Click()
txtItem.Text = "TG"
txtRecType.Text = "Z"
txtTOI.Text = "B"
cmdSEND_Click
End Sub

Private Sub cmdUN_Click()
txtItem.Text = "UN"
txtRecType.Text = "Z"
txtTOI.Text = "B"
cmdSEND_Click
End Sub

Private Sub cmdW2_Click()
txtItem.Text = "W2=ENRAF2"
txtRecType.Text = "Z"
txtTOI.Text = "B"
cmdSEND_Click
End Sub

Private Sub cmdZ_Click()
txtRecType.Text = "Z"
End Sub
Private Sub Form_Load()
frmMain.txtLevelDisplay.Text = ""
frmMain.txtDisplay2.Text = ""
End Sub

Private Sub txtItem_KeyPress(KeyAscii As Integer)
    If (KeyAscii = 13) Then
        cmdSEND_Click
    End If
End Sub

Private Sub txtRE_Change()
    txtRE.SelStart = Len(txtRE.Text)
End Sub
```

```
End frmService.frm
=====
Begin frmSplash.frm
VERSION 5.00
Begin VB.Form frmSplash
    AutoRedraw      = -1 'True
    BorderStyle     = 3  'Fixed Dialog
    ClientHeight    = 4500
    ClientLeft      = 252
    ClientTop       = 1416
    ClientWidth     = 7464
    ClipControls    = 0  'False
    ControlBox      = 0  'False
    Icon            = "frmSplash.frx":0000
    KeyPreview      = -1 'True
    LinkTopic       = "Form2"
    MaxButton       = 0  'False
    MinButton       = 0  'False
    Picture         = "frmSplash.frx":000C
    ScaleHeight     = 375
    ScaleMode       = 3  'Pixel
    ScaleWidth      = 622
    ShowInTaskbar   = 0  'False
    StartUpPosition = 2  'CenterScreen
Begin VB.Line Line3
    X1              = 120
    X2              = 600
    Y1              = 20
    Y2              = 20
End
Begin VB.Line Line2
    BorderWidth     = 5
    X1              = 20
    X2              = 120
    Y1              = 20
    Y2              = 20
End
Begin VB.Line Line1
    BorderColor     = &H00000000&
    BorderWidth     = 5
    X1              = 20
    X2              = 20
    Y1              = 310
    Y2              = 20
End
Begin VB.Label lblWarning
    BackColor       = &H80000005&
    BackStyle       = 0  'Transparent
    Caption         = "Warning: Unauthorized use prohibited"
BeginProperty Font
    Name           = "Arial"
    Size           = 8.4
    Charset        = 0
    Weight         = 400
    Underline      = 0  'False
    Italic         = 0  'False
    Strikethrough  = 0  'False
EndProperty
    Height         = 192
    Left           = 480
    TabIndex       = 7
    Top            = 4080
    Width          = 3012
End
Begin VB.Label lblCompany
    BackColor       = &H80000005&
    BackStyle       = 0  'Transparent
    Caption         = "DOE Hanford"
BeginProperty Font
    Name           = "Arial"
```

HNF-1569
Revision 1

```

        Size           = 8.4
        Charset        = 0
        Weight         = 400
        Underline      = 0 'False
        Italic         = 0 'False
        Strikethrough  = 0 'False
    EndProperty
    Height            = 456
    Left              = 4440
    TabIndex          = 6
    Top               = 3840
    Width             = 2052
End
Begin VB.Label lblCopyright
    BackColor        = &H80000005&
    BackStyle        = 0 'Transparent
    Caption          = "Copyright (c) 1997, 1998, 1999"
    BeginProperty Font
        Name          = "Arial"
        Size          = 8.4
        Charset       = 0
        Weight        = 400
        Underline     = 0 'False
        Italic        = 0 'False
        Strikethrough = 0 'False
    EndProperty
    Height           = 252
    Left             = 4440
    TabIndex         = 5
    Top              = 3600
    Width            = 2652
End
Begin VB.Label lblVersion
    Alignment        = 1 'Right Justify
    AutoSize         = -1 'True
    BackColor        = &H80000005&
    BackStyle        = 0 'Transparent
    Caption          = "Version 1.0.0"
    BeginProperty Font
        Name          = "Arial"
        Size          = 12
        Charset       = 0
        Weight        = 700
        Underline     = 0 'False
        Italic        = 0 'False
        Strikethrough = 0 'False
    EndProperty
    Height           = 288
    Left             = 5640
    TabIndex         = 4
    Top              = 3000
    Width            = 1476
End
Begin VB.Label lblLicence
    Alignment        = 1 'Right Justify
    BackColor        = &H80000005&
    BackStyle        = 0 'Transparent
    Caption          = "LicenseTo: Project Hanford Management Contract"
    BeginProperty Font
        Name          = "Arial"
        Size          = 8.4
        Charset       = 0
        Weight        = 400
        Underline     = 0 'False
        Italic        = 0 'False
        Strikethrough = 0 'False
    EndProperty
    Height           = 252
    Left             = 3240
    TabIndex         = 3
    Top              = 360

```

```
Width          = 3972
End
Begin VB.Label lblPlatform
Alignment      = 1 'Right Justify
AutoSize       = -1 'True
BackColor      = &H80000005&
BackStyle      = 0 'Transparent
Caption        = "for Windows 95/98/NT"
BeginProperty Font
    Name        = "Arial"
    Size        = 15.6
    Charset     = 0
    Weight      = 700
    Underline   = 0 'False
    Italic      = 0 'False
    Strikethrough = 0 'False
EndProperty
Height        = 360
Left          = 3900
TabIndex      = 2
Top           = 2640
Width         = 3240
End
Begin VB.Label lblProductName
AutoSize       = -1 'True
BackColor      = &H80000005&
BackStyle      = 0 'Transparent
Caption        = "Control Panel"
BeginProperty Font
    Name        = "Arial"
    Size        = 32.4
    Charset     = 0
    Weight      = 700
    Underline   = 0 'False
    Italic      = 0 'False
    Strikethrough = 0 'False
EndProperty
Height        = 756
Left          = 2280
TabIndex      = 1
Top           = 1320
Width         = 4332
WordWrap      = -1 'True
End
Begin VB.Label lblProduct
AutoSize       = -1 'True
BackColor      = &H80000005&
BackStyle      = 0 'Transparent
Caption        = "Enraf 854 ATG"
BeginProperty Font
    Name        = "Arial"
    Size        = 18
    Charset     = 0
    Weight      = 700
    Underline   = 0 'False
    Italic      = 0 'False
    Strikethrough = 0 'False
EndProperty
Height        = 420
Left          = 2280
TabIndex      = 0
Top           = 960
Width         = 2472
End
Begin VB.Image imgLogo
Height        = 2388
Left          = 480
Picture       = "frmSplash.frx":AB0A2
Stretch       = -1 'True
Top           = 960
Width         = 1452
```

```

End
Attribute VB_Name = "frmSplash"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False

Option Explicit

Private Sub Form_KeyPress(KeyAscii As Integer)
    Unload Me
    frmLogon.Show
End Sub

Private Sub Form_Load()
    Dim V$
    V = "Version " & App.Major & "." & App.Minor & "." & App.Revision
    lblVersion.Caption = V
End Sub

Private Sub Frame1_Click()
    Unload Me
End Sub

```

End frmSplash.frm

```

=====
Begin frmSPW.frm

VERSION 5.00
Begin VB.Form frmSPW
    BorderStyle      = 4 'Fixed ToolWindow
    Caption          = "Service Password Entry"
    ClientHeight     = 1440
    ClientLeft       = 30
    ClientTop        = 270
    ClientWidth      = 3750
    LinkTopic        = "Form1"
    MaxButton        = 0 'False
    MinButton        = 0 'False
    ScaleHeight      = 1440
    ScaleWidth       = 3750
    ShowInTaskbar    = 0 'False
    StartUpPosition = 2 'CenterScreen
    Begin VB.CommandButton cmdCancel
        Caption      = "&Cancel"
        Height       = 375
        Left         = 2400
        TabIndex     = 2
        Top          = 840
        Width        = 1095
    End
    Begin VB.CommandButton cmdOkay
        Caption      = "&OK"
        Height       = 375
        Left         = 2400
        TabIndex     = 3
        Top          = 240
        Width        = 1095
    End
    Begin VB.TextBox txtPW
        Height       = 288
        IMEMode      = 3 'DISABLE
        Left         = 240
        PasswordChar = "*"
        TabIndex     = 0
        Top          = 480
        Width        = 1812
    End
    Begin VB.Label Label2
        Height       = 255
        Left         = 240
    End
End

```

```

        TabIndex      = 4
        Top           = 960
        Visible       = 0 'False
        Width         = 1815
    End
    Begin VB.Label Label1
        Caption        = "Please enter the password:"
        Height         = 252
        Left           = 240
        TabIndex       = 1
        Top            = 120
        Width          = 2172
    End
End
Attribute VB_Name = "frmSPW"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
' =====
' This module/form checks service password entry.
' =====

Option Explicit

Private Sub cmdCancel_Click()
    Unload frmSPW
End Sub

Private Sub cmdOkay_Click()
    Dim Reply%
    If UCase(txtPW.Text) <> Label2.Caption Then
        MsgBox "Incorrect Password!", vbOKOnly + vbCritical, "Wrong Password..."
        Exit Sub
    Else
        frmSPW.txtPW.Text = ""
        frmSPW.Label2.Caption = "GO"
        frmSPW.Hide
    End If
End Sub

Private Sub Form_Load()

End Sub

Private Sub Form_QueryUnload(Cancel As Integer, UnloadMode As Integer)
    ' The general procedure DoUnloadPreCheck handles the possible unload options
    ' for all three forms in this sample application.
    'DoUnloadPreCheck UnloadMode
    Select Case UnloadMode
        Case vbFormControlMenu
            If Not GettingDensities Then
                Unload Me
                'Cancel = True
            Else
                Exit Sub
            End If
        Case vbFormCode
            'End
        Case vbAppWindows
            'End
        Case vbAppTaskManager
            'End
        Case vbFormMDIForm
            'End
    End Select
End Sub

Private Sub txtPW_KeyPress(KeyAscii As Integer)
    If (KeyAscii = 13) Then
        Call cmdOkay_Click
    End If
End Sub

```

HNF-1569
Revision 1

End If
End Sub

End frmSPW.frm

=====

Begin Globals.bas

Attribute VB_Name = "GLOBALS"

Option Explicit

Global testFlag% ' Used for program testing
Global MainPword\$ ' Contains program password
Global ServPword\$, ServPwordFlag% ' Contains service password and enable flag
Global AdminPword\$ ' Contains administrator password
Global DidFileSetup% ' Flag to indicate app directories have been setup
Global TestMode% ' Places program in demo mode
Global TestItems(1 To 400) As String ' Contains a list of dummy gauge items
Global TestItemCount% ' Contains the number of dummy gauge items

' CIU variables

Global Timeout!, error_trap%
Global turn_around!
Global ciuadres\$
Global ta\$ ' Gauge transmission address
Global temp\$
Global status% ' CIU Com status
Global echoCount% ' Used in RQS routines
Global busy_waittime As Double
Global GotCIU% ' CIU communication flag

'COMM Variables

Global Baudrate\$
Global Compoort\$
Global DataBits
Global StopBits
Global Parity

'Global tempscan\$(30) ' tijdelijke opslag voor 20 items
' in SUB itemscan en SUB instrument.display
Global item\$(500) ' items in logfile
Global echoArray\$(500) ' alle CIU/PET echo's voor logfile

'bepaaltype variables (checking for connected gauges)

Global Instrument\$ ' instrument type (854, etc.)
Global ja% ' >0 = instrument exists
Global toi\$ ' type of instrument
Global df\$, ti\$ ' display format, tank id
Global RL As Double ' Reference Level
Global TT As Double ' Tank Top
Global s1B As String ' Line 1 of display
Global s2B As String ' Line 2 of display

'Densitometer Variables

Global AbortDensity\$ ' Flag for aborting density reading activity
Global CL As Double ' Contains Current Level in tank as obtained in
' the GetALevel() sub

'Program status variables

Global onDisconnect_Click% ' Disconnect has been clicked
Global commIsOpen%, gaugeChecked% ' Comport is open and gauge exists
Global numretries% ' retries in sending gauge data
Global opWarningTitle\$ ' Contains title of message box for warnings
Global SpecialCommand\$ ' Contains command sent from OpEnraf form
Global scanRate! 'tmrLevel interval
Global Abort% ' An abort flag
Global GettingDensities% ' Flag indicating we are getting density readings
Global gettingScans% ' Flag indicating we are getting continuous scans
Global initializing% ' Flag indicating communications are initializing

'File Variables

Global DensityFilename As String


```

Global rqsFile As String
Global rplFile As String
Global logFile As String
Global logDir$, rplDir$, rqsDir$, libDir$, itemDir$
Global CurrentDirectory$
Global itemcount%
Global LastDBFS ' the last density database file created

'User Variables
Global CurrentUserName As String

'Density Level Variables
Global SDLevel As String          'Start level for IP scans
Global SedLev As String
Global Interval As String
Global LastSpG As String
Global LastSedLevel As String
Global LastSedLevelDate As String
Global LastSedLevelTime As String
Global LastWireTension As String
Global BalanceTestPassed As Boolean
Global WRSSStart As String        ' WRSS Full Auto Start Level Setting

'Margin values
Global Const MARGIN_LEFT = 720
Global Const MARGIN_RIGHT = 720
Global Const MARGIN_TOP = 1440
Global Const MARGIN_BOTTOM = 1440

'MSComm variables/constants
Public Const vbMSCommEvReceive = 2
Public Const vbMSCommEvSend = 1
Public Const vbMSCommEvCTS = 3
Public Const vbMSCommEvDSR = 4
Public Const vbMSCommEvCD = 5
Public Const vbMSCommEvRing = 6
Public Const vbMSCommEvEOF = 7

Public Const vbMSCommErBreak = 1001
Public Const vbMSCommErCTSTO = 1002
Public Const vbMSCommErDSRTO = 1003
Public Const vbMSCommErFrame = 1004
Public Const vbMSCommErOverrun = 1005
Public Const vbMSCommErCDTO = 1006
Public Const vbMSCommErRxOver = 1007
Public Const vbMSCommErRxParity = 1008
Public Const vbMSCommErTxFull = 1009

Global Const HS0x_FNAME = &O12
Global Const HS0x_PATH = &O13
Global Const HS0x_EXT = &O11
Global Const HS0x_SFNAME = &O14
Global Const HS0x_LNGFORMAT = &O20
Global Const HS0x_SHRTFORMAT = &O21
Global Const HS0s_COMPNAME = "CompanyName"
Global Const HS0s_VERNUM = "FileVersion"
Global Const HS0s_FDISC = "FileDescription"
Global Const HS0s_COPYRIGHT = "LegalCopyright"
Global Const HS0x_LONGPATH = &O30
Global Const HS0x_SHRTPATH = &O31

'<ObjName>.CopyFile(source,target)
'delcare sub CopyFile lib "hsfileutil32.dll" (byval Source as String, byval Target as String)

'Global FileWork As New clsFileUtils

#If Win32 Then

```

HNF-1569
Revision 1

```
Declare Sub SetWindowPos Lib "user32" (ByVal hWnd As Long, ByVal hWndInsertAfter As Long, ByVal  
X As Long, ByVal Y As Long, ByVal cx As Long, ByVal cy As Long, ByVal wFlags As Long)  
#Else  
Declare Sub SetWindowPos Lib "USER" (ByVal hWnd%, ByVal hWndInsertAfter%, ByVal X%, ByVal Y%,  
ByVal cx%, ByVal cy%, ByVal wFlags%)  
#End If  
  
Declare Sub Sleep Lib "kernel32" (ByVal dwMilliseconds As Long)  
  
Declare Function GetPrivateProfileString Lib "kernel32" Alias _  
"GetPrivateProfileStringA" (ByVal lpApplicationName As String, ByVal _  
lpKeyName As String, ByVal lpDefault As String, ByVal lpReturnedString As _  
String, ByVal nSize As Long, ByVal lpFileName As String) As Long  
  
Declare Function WritePrivateProfileString Lib "kernel32" Alias _  
"WritePrivateProfileStringA" (ByVal lpApplicationName As String, ByVal _  
lpKeyName As String, ByVal lpString As String, ByVal lpFileName As String) As _  
Long  
  
Function BLevel(ByVal gageEcho As String) As Double  
'This sub extracts the level reading out of a B-Record. I don't  
'think it is used much if at all in this program.  
  
On Error GoTo ErrorHandler  
Dim tmpString As String  
If InStr(gageEcho, "BB") Then  
'echo format: 110BZ1B+01050.33 in INN  
tmpString = Mid(gageEcho, InStr(gageEcho, "BB") + 4, 7)  
tmpString = Left(tmpString, 5) + "." + Right(tmpString, 2)  
'Debug.Print "tmpStringB= "; tmpString  
BLevel = CDBl(tmpString)  
Exit Function  
Else  
'Must be getting a bad number  
BLevel = -1000  
End If  
Exit Function  
ErrorHandler:  
MsgBox "Error encountered while decoding: " + gageEcho  
Exit Function  
End Function  
  
Sub Delay(delaytime As Long)  
'This delay routing is used throughout instead of Sleep. It is better  
'suited for maintaining multitasking environments.  
  
On Error Resume Next  
  
Dim StartTime As Long  
StartTime = Timer  
If StartTime + delaytime > 86399 Then  
delaytime = StartTime + delaytime - 86400  
Do While Timer <= 36399  
DoEvents  
Loop  
Do While Timer < delaytime  
DoEvents  
Loop  
Else  
Do While Timer < StartTime + delaytime  
DoEvents  
Loop  
End If  
End Sub
```

```
Sub LMeter(ByVal RefLevel As Double, ByVal CurLevel As Double)
'This function sets the displacer position in frmDensity level
'animation
On Error GoTo ErrorHandler
Dim picIndex As Integer
Dim LevIndex As Integer
Dim i%
Dim LevVisible As Integer

'Find Which is visible
For i = 0 To 58
    If frmDensity.picMtr(i).Visible = True Then
        LevVisible = i
    End If
Next i

'Next i
'Debug.Print "RefLevel = "; RefLevel
'Debug.Print "CurLevel= "; CurLevel

If RefLevel > CurLevel Then
    LevIndex = (59 / RefLevel) * CurLevel
    If LevIndex >= 59 Then
        LevIndex = 58
    End If
    LevIndex = 58 - LevIndex
Else
    LevIndex = 0
End If
'Debug.Print "LevIndex= "; LevIndex
'make LevIndex pic visible
frmDensity.picMtr(LevIndex).Visible = True
'set tooltip text
frmDensity.picMtr(LevIndex).ToolTipText = "Current Level: " + Str(CurLevel)
If LevIndex <> LevVisible Then
    frmDensity.picMtr(LevVisible).Visible = False
End If

Exit Sub

ErrorHandler:
    MsgBox "Error encountered in LMeter routine."

End Sub

Sub EnterPassword(ByRef result As Integer)
'=====
' This procedure was originally written by Delft Instruments. Minor
' modifications were made by John Huber. This procedure is proprietary
' to Delft Instruments and may not be used or reproduced without permission.
'=====

'This sub enters protection level 2 on an 854.

On Error GoTo ErrorHandler
Dim echol As String
Dim pString As String
Dim Reply As Integer

Static passw As String

    ciucom "ZPS", echol, status
    decode "PS", echol, 2, pString
    Debug.Print "pString= "; pString
    If pString <> "20" Then
        ciucom "ZW2=ENRAF2", echol, status

        If InStr(echol, "&") = 0 Then
```

HNF-1569
Revision 1

```
passw = InputBox("Enter Password level 2...", "Password Required")

ciucom "ZW2=" + passw, echo1, status

If InStr(echo1, "&") = 0 Then
    'SysMsg "Invalid password", "vbOkay", "vbWarning", Reply
    MsgBox "Invalid password!", vbOKOnly + vbExclamation, "Wrong Password..."
    result = 0
    'invalid
    Exit Sub
Else
    result = 1
End If
Else
    result = 1
End If
Else
    result = 1
End If
Exit Sub

ErrorHandler:
MsgBox "Error encountered during program password entry."
Exit Sub
End Sub

Sub ResetSystem()
' This function returns the program to startup conditions.

On Error GoTo ErrorHandler
    If frmMain.cmdGetCurrentLevel.Visible = False Then
        frmMain.cmdGetCurrentLevel.Visible = True
    End If
    If frmMain.cmdDisconnect.Visible = True Then
        frmMain.cmdDisconnect.Visible = False
    End If
    If frmMain.tmrLevel.Enabled = True Then
        frmMain.tmrLevel.Enabled = False
    End If

    If frmMain.mnuConnectCIU.Enabled = False Then
        frmMain.mnuConnectCIU.Enabled = True
    End If

    If frmMain.mnuDisconnect.Enabled = True Then
        frmMain.mnuDisconnect.Enabled = False
    End If

    If ja > 0 Then
        ja = 0
    End If

    If gaugeChecked = 1 Then
        gaugeChecked = 0
    End If

    If commIsOpen = 1 Then
        ciuclose
        frmMain.brdDisplay.Caption = "Gauge Display"
        GotCIU = 0
    End If

    If GettingDensities = 1 Then
        GettingDensities = 0
    End If

    If gettingScans = 1 Then
        gettingScans = 0
    End If
```

HNF-1569
Revision 1

```
End If

If Abort = 1 Then
    Abort = 0
End If

frmMain.txtLevelDisplay.Text = ""
frmMain.txtDisplay2.Text = ""
frmMain.lblCStatus.Caption = ""
frmMain.sBar.SimpleText = ""
'ShowPic "SYSTEM"
If frmService.Visible = True Then frmService.Hide
If frmDensity.Visible = True Then Unload frmDensity
If frmOpEnraf.Visible = True Then frmOpEnraf.Hide
If frmFTF.Visible = True Then frmFTF.Hide
If frmLogFile.Visible = True Then frmLogFile.Hide
If frmCOMMAAdv.Visible = True Then frmCOMMAAdv.Hide
If frmCIUSetup.Visible = True Then frmCIUSetup.Hide
If frmCalc.Visible = True Then frmCalc.Hide
If frmAbout.Visible = True Then frmAbout.Hide
If frmCFG.Visible = True Then frmCFG.Hide
If frmDemo.Visible = True Then frmDemo.Hide
If frmWeigh.Visible = True Then frmWeigh.Hide
frmMain.mnuOptionsDemo_Click

Exit Sub

ErrorHandler:
    MsgBox "Error encountered while trying to reset program"
Exit Sub

End Sub

Sub SaveToINI()
    'This sub saves pertinent variables to the ecpw.ini file

    On Error Resume Next
    Dim inifile As String
    inifile = App.Path + "\ecpw.ini"

    'Communications info
    WritePrivateProfileString "ecpw", "Compoort", Compoort, inifile
    WritePrivateProfileString "ecpw", "Baudrate", Baudrate, inifile
    WritePrivateProfileString "ecpw", "ciudades", ciudades, inifile
    WritePrivateProfileString "ecpw", "ta", ta, inifile
    WritePrivateProfileString "ecpw", "turn_around", turn_around, inifile
    WritePrivateProfileString "ecpw", "scanRate", scanRate, inifile
    WritePrivateProfileString "ecpw", "toi", toi, inifile
    WritePrivateProfileString "ecpw", "Parity", Parity, inifile
    WritePrivateProfileString "ecpw", "DataBits", DataBits, inifile
    WritePrivateProfileString "ecpw", "StopBits", StopBits, inifile
    WritePrivateProfileString "ecpw", "CIUWait", Str(busy_waittime), inifile

    'Density operation info
    WritePrivateProfileString "Density", "SDLevel", frmDensity.txtDensityStart.Text, inifile
    WritePrivateProfileString "Density", "SedLev", frmDensity.txtSed.Text, inifile
    WritePrivateProfileString "Density", "Interval", frmDensity.txtInterval.Text, inifile
    WritePrivateProfileString "Density", "LastTT", Str(TT), inifile
    WritePrivateProfileString "Density", "LastDBF", LastDBF, inifile
    WritePrivateProfileString "Density", "LastSpG", LastSpG, inifile
    WritePrivateProfileString "Density", "LastSedLevelDate", LastSedLevelDate, inifile
    WritePrivateProfileString "Density", "LastSedLevelTime", LastSedLevelTime, inifile
    WritePrivateProfileString "Density", "LastSedLevel", LastSedLevel, inifile
    WritePrivateProfileString "Density", "LastWireTension", LastWireTension, inifile
    WritePrivateProfileString "Density", "WRSSStart", WRSSStart, inifile

    'Gauge display font info
```

HNF-1569
Revision 1

```
WritePrivateProfileString "Gauge Display Fonts", "Name", frmMain.txtLevelDisplay.Font.Name,
inifile
WritePrivateProfileString "Gauge Display Fonts", "Color", Str(frmMain.txtLevelDisplay.ForeColor),
inifile
WritePrivateProfileString "Gauge Display Fonts", "Bold", Str(frmMain.txtLevelDisplay.Font.Bold),
inifile
WritePrivateProfileString "Gauge Display Fonts", "Italic",
Str(frmMain.txtLevelDisplay.Font.Italic), inifile
WritePrivateProfileString "Gauge Display Fonts", "Strikethru",
Str(frmMain.txtLevelDisplay.FontStrikethru), inifile
WritePrivateProfileString "Gauge Display Fonts", "Underline",
Str(frmMain.txtLevelDisplay.Font.Underline), inifile
WritePrivateProfileString "Gauge Display Fonts", "Size", Str(frmMain.txtLevelDisplay.Font.Size),
inifile

End Sub
```

```
Sub ShowPic(picType As String)
'This sub changes the communications status display pictures. It is
'controlled from ciucom, cmdDisconnect and cmdCloseComm procedures.
'Debug.Print "Calling ShowPic: "; picType
On Error Resume Next
picType = Trim(UCase(picType))
```

```
    Select Case picType
```

```
        Case "SYSTEM"
            frmMain.picSYS.Visible = True
            frmMain.picCOMM.Visible = False
            frmMain.picCIU.Visible = False
            frmMain.picEnraf.Visible = False
```

```
        Case "COMM"
            frmMain.picSYS.Visible = False
            frmMain.picCOMM.Visible = True
            frmMain.picCIU.Visible = False
            frmMain.picEnraf.Visible = False
```

```
        Case "CIU"
            frmMain.picSYS.Visible = False
            frmMain.picCOMM.Visible = False
            frmMain.picCIU.Visible = True
            frmMain.picEnraf.Visible = False
```

```
        Case "ENRAF"
            frmMain.picSYS.Visible = False
            frmMain.picCOMM.Visible = False
            frmMain.picCIU.Visible = False
            frmMain.picEnraf.Visible = True
```

```
        Case Else
            frmMain.picSYS.Visible = False
            frmMain.picCOMM.Visible = False
            frmMain.picCIU.Visible = False
            frmMain.picEnraf.Visible = False
```

```
    End Select
```

```
End Sub
```

```
Sub WriteToFile(ByVal filestring As String)
```

```
'=====
' This procedure was originally written by Delft Instruments. Minor
' modifications were made by John Huber. This procedure is proprietary
' to Delft Instruments and may not be used or reproduced without permission.
```

HNF-1569
Revision 1

```
'=====

'This sub is used by the gauge log file creation sub in the frmFTF form.
'it writes logged information to ascii file.

On Error GoTo ErrorHandler
Dim FileNum As Integer
Dim rString As String
Dim i%

frmLogFile.MousePointer = 11
    frmMain.sBar.SimpleText = "Storing data in file " + logDir + filestring
    FileNum = FreeFile
    Open logDir + "\" + filestring For Output As FileNum
    'Print #FileNum, " " + App.ProductName$ + App.Revision$
    Print #FileNum, " " + filestring + " Date: " + Date$ + " Time: " + Time$
    For i = 1 To itemcount
        rString = item(i)
        If Left(rString, 1) <> "*" Then
            Print #FileNum, rString + "=" + echoArray(i)
        End If
    Next i
    Close #FileNum
    'Sleep (3000) 'zet melding voor tenminste 3 sek. scherm
    frmMain.sBar.SimpleText = "File " + logDir + "\" + filestring + " created...."
    frmLogFile.MousePointer = 0
Exit Sub

ErrorHandler:
    MsgBox "Error encountered writing to file"
    Exit Sub
End Sub

Function Z1BLevel(ByVal gageEcho As String) As Double
'This function takes a Z1B-record gauge response and returns the tank
' level as a number. It gets rid of the "in INN" in the string.

On Error GoTo ErrorHandler
Dim tmpString As String
If InStr(gageEcho, "+") Then
    'gageEcho format: 110BZ1B+01050.33 in INN
    tmpString = Mid(gageEcho, InStr(gageEcho, "+") + 1, 5)
    'Debug.Print "tmpString1= "; tmpString
    tmpString = tmpString + Mid(gageEcho, InStr(gageEcho, "."), 3)
    'Debug.Print "tmpString2= "; tmpString
    Z1BLevel = CDbl(tmpString)
Exit Function
Else
    'Must be getting a negative level
    Z1BLevel = 0
End If
Exit Function
ErrorHandler:
    MsgBox "Error encountered decoding: " + gageEcho
    Exit Function
End Function

Function GetLevelDimension() As String
'This function gets the level dimension format from the gauge

Dim echo$
Dim LevDim$, answer$

If gaugeChecked = 0 Then
    checkGageAddress ta, ja
End If

If ja = 0 Then
```

```
Exit Function
End If

ciucom "ZLD", echo, status
Delay (0.3)
If InStr(echo, "LD!051") Then
    LevDim = "not implemented" ' level dimension not implemented
Else
    decode "LD", echo, 1, answer
    Select Case answer
        Case "M"
            LevDim = "m"
        Case "F"
            LevDim = "ft"
        Case "I"
            LevDim = "inches"
        Case "P"
            LevDim = "ft, inches, 16/th"
        Case Else
            LevDim = "unknown"
    End Select
End If

GetLevelDimension = LevDim

End Function
Function FormatInputString(ByVal theString As String, ByVal LevelDimensionType As String) As String
    'this function is set up to handle more than one dimension type,
    'but only implemented for inches now.
    Dim tmp$
    Dim tmprit$

    Select Case UCase(LevelDimensionType)

        Case "INCHES"
            'If there is a decimal point, then collect digits from
            'either side
            If InStr(theString, ".") <> 0 Then
                'collect the left of decimal w/o decimal
                tmp = Mid(theString, 1, InStr(theString, ".") - 1)
                Debug.Print "tmp to . = "; tmp

                'Truncate if too long.
                tmp = Right(tmp, 5)

                'collect stuff just to right of decimal
                tmprit = Mid(theString, InStr(theString, "."), 3)
                Debug.Print "tmprit = "; tmprit

                'make sure length of right side is correct.
                Do
                    If Len(tmprit) < 3 Then
                        tmprit = tmprit + "0"
                    Else
                        Exit Do
                    End If
                Loop
                If Len(tmprit) > 3 Then
                    tmprit = Left(tmprit, 3)
                End If

                'add the strings together
                tmp = tmp + tmprit
                'tmp = tmp + Mid(theString, InStr(theString, "."), 3)
                Debug.Print "tmp et al = "; tmp
                'get the correct length
                tmp = Trim(Right(tmp, 8))
                'now if length is less than 8, add leading zeros
                If Len(tmp) < 8 Then
                    Do
```



```

        tmp = "0" + tmp
        If Len(tmp) = 8 Then Exit Do
    Loop
End If
'now add the plus sign
tmp = "+" + tmp
FormatInputString = tmp
Exit Function
Else 'no decimal point
    tmp = theString
    'if length > 5, truncate
    If Len(tmp) > 5 Then
        tmp = Right(tmp, 5)
    End If

    'add some decimal zeros
    tmp = tmp + ".00"

    'if length too short, add leading zeros
    If Len(tmp) < 8 Then
        Do
            tmp = "0" + tmp
            If Len(tmp) = 8 Then Exit Do
        Loop
    End If

    'now add the plus sign
    tmp = "+" + tmp
    FormatInputString = tmp
    Exit Function
End If

Case "FT"

Case "M"

Case Else
End Select

End Function
Sub bepaaltype(ja)
'=====
' This procedure was originally written by Delft Instruments. Minor
' modifications were made by John Huber. This procedure is proprietary
' to Delft Instruments and may not be used or reproduced without permission.
'=====

Dim CR
CR = Chr(13) + Chr(10)
'This sub determines whether a gauge exists or is connected
'to the system. It does much more than is needed since it was
'ported directly from the Logv18 program.

On Error GoTo ErrorHandler
Dim request As String
Dim echo As String
Dim regel(1 To 10) As String
Dim rectype As String
Dim displee(1 To 10) As String

ja = 0
Dim taold$
Dim toiold$
taold = ta
'ta = ""
If toi = "" Then toi = "B" 'default to B
toiold = toi
toi = ""

```

```
request = "@X" ' ID record to CIU

toi = ""
ciucom request, echo, status
ta = taold
If echo = ciuadres & "@0" Then
    GotCIU = 1
    regel(2) = "NO FIELD "
    GoSub display1
Else
    If Len(echo) = 0 Then
        GotCIU = 0
        If ciuadres = "" Then
            regel(2) = "NO PET "
        Else
            regel(2) = "NO CIU "
        End If
        GoSub display1
    End If
End If

If Len(echo) > 3 Then
    ja = 1 ' als meter aanwezig
    GotCIU = 1
    If InStr(echo, "X") Then
        'toi is either a "P" for GPP or "R" for GPU
        toi = Mid(echo, InStr(echo, "X") - 1, 1)
        Debug.Print "toi= "; toi
        If toi <> "B" Then
            toi = toiold
            frmMain.sBar.SimpleText = "854 Gauge not found..."
            Exit Sub
        End If
    End If
    If InStr(echo, " ") <> 0 Then
        Instrument = Mid(echo, InStr(echo, "X") + 1, 3)
        Debug.Print "Instrument= "; Instrument
        frmMain.brdDisplay.Caption = Instrument & " Gauge Display"
        'rectype = toi + "Z"
        rectype = "Z"
        Select Case Instrument
            Case "SV8" ' tijdelijke oplossing vanwege bug in 872
                displee(1) = "01": displee(2) = "02"
                Instrument = "872"
            Case "872"
                displee(1) = "01": displee(2) = "02"
            Case "854", "865", "866", "873", "877", "894"
                ciucom rectype + "DF", echo, status
                decode "DF", echo, 1, df
                displee(1) = "1" + df
                displee(2) = "2" + df
            Case Else
                Instrument = "???"
                ti = Space(6)
        End Select
    Else
        Select Case toi
            Case "A"
                Instrument = "802"
            Case "C"
                Instrument = "813"
                displee(1) = "B"
                displee(2) = "C"
            Case "B"
                Instrument = "811"
                displee(1) = "B"
                displee(2) = "C"
                'ShowPic "Enraf"
            Case "D"
                Instrument = "875"
                displee(1) = "X"
```

```

        displee(2) = "D"
    Case "F"
        Instrument = "874"
        displee(1) = "X"
        displee(2) = "D"
    Case "E"
        Instrument = "811E"
        displee(1) = "B"
        displee(2) = "C"
    Case Else
        Instrument = "???"
        ti = " "
    End Select
    rectype = toi
End If

'vraag ti op
ciucom rectype + "TI", echo, status
decode "TI", echo, 6, ti
If ti = "....." Then ti = " "

End If

toi = toiold
ta = taold

Exit Sub

display1:
    Instrument = "???"
    ti = " "
    ja = 0
    frmMain.lblCStatus.Caption = regel(2) + "COMMUNICATION...."
    gaugeChecked = 0
    Exit Sub

ErrorHandler:
    MsgBox "Error Encountered while checking for gauge"
    Exit Sub
End Sub

Sub checkGageAddress(ta, ja)
' This sub handles verification of gauge coommunication using bepaaltype
' and displaying necessary status messages.

'
' Vraag om een tankadres.
'
Dim i%

On Error GoTo ErrorHandler
Dim echo As String
Dim answer As String
Dim CR
CR = Chr(13) + Chr(10)

If initializing = 1 Then
    frmMain.txtDisplay2.Text = "Initializing..."
End If

If commIsOpen = 0 Then
    Debug.Print "Opening comport...."
    ciuopen
End If

frmMain.sBar.SimpleText = "Checking for Valid Gauge"
bepaaltype ja
Debug.Print "Out of bepaal, cont. in checkgage..."
If ja = 1 Then
    Debug.Print "Valid gauge found..."

```

HNF-1569
Revision 1

```

    gaugeChecked = 1
    'ShowPic "ENRAF"
    frmMain.sBar.SimpleText = "Valid Gauge Found..."
    Delay (1)
    frmMain.sBar.SimpleText = ""
    'frmMain.brdDisplay.Caption = Instrument & " Gauge Display"
    'Make sure gauge is set up right.

Else
    gaugeChecked = 0
    frmMain.sBar.SimpleText = "No valid gauge found...."
    Delay (1)
    frmMain.sBar.SimpleText = ""
    frmMain.txtDisplay2.Text = ""
End If

Exit Sub

ErrorHandler:
    MsgBox "Error encountered during initial gauge communication."
    Exit Sub
End Sub

Sub ciuclose()
    'This sub closes the commport connection and resets system status
    'indicators.

On Error GoTo ErrorHandler
    'close the port
If frmMain.Comm2.PortOpen = True Then
    frmMain.Comm2.PortOpen = False
    frmMain.Comm2.InBufferCount = 0
    frmMain.Comm2.OutBufferCount = 0
Else
    Exit Sub
End If

'set system status indicators
    frmMain.sBar.SimpleText = "Comport Closed..."
    Delay (1)
    frmMain.sBar.SimpleText = ""
    frmMain.lblCStatus.Caption = "Disconnected..."
    ShowPic "SYSTEM"
    'ShowArrow "BLANK"
    'frmMain.pnlCOMM.Caption = "..Standing By.."

'set the global flag so that other subs know status of the com port
    commIsOpen = 0
Exit Sub

ErrorHandler:
    MsgBox "Error encountered while trying to close com port"
    Exit Sub

End Sub

Sub ciucom(ByVal request As String, ByRef gageEcho As String, status)
    '=====
    ' This procedure was originally written by Delft Instruments. Minor
    ' modifications were made by John Huber. This procedure is proprietary
    ' to Delft Instruments and may not be used or reproduced without permission.
    '=====

    'This sub brings together the send and receive CIU routines.
    'It also intercepts commands should the program be in demo mode. It then
    'redirects or responds to simulate connection to an 854 gauge.

    ' If just testing, then
    If TestMode = 1 Then

```

```

frmMain.sBar.SimpleText = "DEMO MODE...."
If InStr(request, "X") <> 0 Then
    'Debug.Print "X <> 0..."
    gageEcho = ciuadres & ta & "BX" & "854 A4.1"
    ShowPic "CIU"
    ShowLED "GREEN"
    Delay (turn_around / 1000)
    ShowPic "ENRAF"
    ShowLED "OFF"
    frmMain.lblCStatus.Caption = "Connected..."
    Debug.Print "TestgageEcho= "; gageEcho
    Exit Sub
Else
    ShowPic "CIU"
    ShowLED "GREEN"
    Delay (turn_around / 1000)
    ciucomTest request, gageEcho, status
    If onDisconnect_Click = 0 Then
        ShowPic "ENRAF"
    Else
        ShowPic "COMM"
    End If
    ShowLED "OFF"
    frmMain.lblCStatus.Caption = "Connected..."
End If
Exit Sub
End If

Dim CR
CR = Chr(13) + Chr(10)
Debug.Print "gage Request= "; request
sendtoCIU request
receiveCIU gageEcho, status
Debug.Print "gageEcho = "; gageEcho
ShowLED "OFF"

' Check for errors
If Len(gageEcho) <= 3 Then
    frmMain.sBar.SimpleText = "No communication..."
ElseIf InStr(gageEcho, "!") <> 0 Then
    Select Case Mid(gageEcho, InStr(gageEcho, "!"), 2)
        Case "!0"
            MsgBox GetErrorMessage(Mid(gageEcho, InStr(gageEcho, "!"), 4)), vbOKOnly + vbCritical
            'set abort flags
            AbortDensity = 1
            Abort = 1
        Case "!1"
            MsgBox GetErrorMessage(Mid(gageEcho, InStr(gageEcho, "!"), 4)), vbOKOnly + vbCritical
            'set abort flags
            AbortDensity = 1
            Abort = 1
        Case "!2"
            MsgBox GetErrorMessage(Mid(gageEcho, InStr(gageEcho, "!"), 4)), vbOKOnly + vbCritical
            'set abort flags
            AbortDensity = 1
            Abort = 1
        Case "!3"
            MsgBox GetErrorMessage(Mid(gageEcho, InStr(gageEcho, "!"), 4)), vbOKOnly + vbCritical
            'set abort flags
            AbortDensity = 1
            Abort = 1
        Case "!4"
            MsgBox GetErrorMessage(Mid(gageEcho, InStr(gageEcho, "!"), 4)), vbOKOnly + vbCritical
            'set abort flags
            AbortDensity = 1
            Abort = 1
        Case "!5"
            MsgBox GetErrorMessage(Mid(gageEcho, InStr(gageEcho, "!"), 4)), vbOKOnly + vbCritical
            'set abort flags
            AbortDensity = 1
            Abort = 1
    End Select

```

HNF-1569
Revision 1

```
Case "16"
MsgBox GetErrorMessage(Mid(gageEcho, InStr(gageEcho, "!"), 4)), vbOKOnly + vbCritical
'set abort flags
AbortDensity = 1
Abort = 1
Case "17"
MsgBox GetErrorMessage(Mid(gageEcho, InStr(gageEcho, "!"), 4)), vbOKOnly + vbCritical
'set abort flags
AbortDensity = 1
Abort = 1
Case "18"
MsgBox GetErrorMessage(Mid(gageEcho, InStr(gageEcho, "!"), 4)), vbOKOnly + vbCritical
'set abort flags
AbortDensity = 1
Abort = 1
Case "19"
MsgBox GetErrorMessage(Mid(gageEcho, InStr(gageEcho, "!"), 4)), vbOKOnly + vbCritical
'set abort flags
AbortDensity = 1
Abort = 1
End Select
End If
End Sub
Sub ciucomTest(ByVal request As String, ByRef gageEcho As String, status)
' This sub handles Demo mode requests and provides an appropriate response.
' Commands requests are not supported; item values cannot be changed.
Dim q%, tmpItem$
Debug.Print "Test Request= "; request
If InStr(request, "W2=ENRAF2") <> 0 Then
    gageEcho = request & "&"
    Exit Sub
ElseIf InStr(request, "W1=ENRAF1") <> 0 Then
    gageEcho = request & "&"
    Exit Sub
ElseIf InStr(request, "BT") <> 0 Then
    gageEcho = request & "&"
    Exit Sub
End If
For q = 1 To TestItemCount
    'Debug.Print "TestItems("; q; ")= "; TestItems(q)
    'Debug.Print "Left= "; Left(TestItems(q), 2)
    If InStr(request, Left(TestItems(q), 2)) <> 0 Then
        tmpItem = TestItems(q)
        If InStr(tmpItem, "=") <> 0 Then
            tmpItem = Mid(TestItems(q), 1, InStr(TestItems(q), "=") - 1)
            tmpItem = tmpItem & Mid(TestItems(q), InStr(TestItems(q), "=") + 1)
        End If
        gageEcho = Left(request, Len(request) - 2) & tmpItem
        gageEcho = ciuadres & ta & toi & gageEcho
        status = 0
        Debug.Print "Left= "; Left(tmpItem, 2)
        Debug.Print "TestEcho= "; gageEcho
        Exit Sub
    End If
Next q
Debug.Print "TestEcho= item not found..."
End Sub

Sub GetALevel(Optional s1B, Optional s2B)
' Any time a current level is needed, this routine may be
' used. It fills the gauge display and saves the current tank
' level in CL. Globals s1B and s2B are also filled incase needed.
On Error Resume Next
Dim answer$, echo$
ciucom "Z1B", echo, status
decode "1B", echo, 64, answer
s1B = answer
CL = Z1BLevel(echo)
LMeter RL, CL
```

HNF-1569
Revision 1

```
frmMain.txtLevelDisplay.Text = answer
ciucom "Z2B", echo, status
decode "2B", echo, 32, answer
s2B = answer
frmMain.txtDisplay2.Text = answer
With frmMain.lblDStatus
Select Case Trim(Right(s2B, 2))
    Case "I1"
        .Caption = "Interface 1"
    Case "I2"
        .Caption = "Interface 2"
    Case "I3"
        .Caption = "Interface 3"
    Case "TG"
        .Caption = "Test Gauge"
    Case "FL"
        .Caption = "Error codes detected"
    Case "FR"
        .Caption = "Displacer halted"
    Case "MF"
        .Caption = "Weighing displacer"
    Case "CA"
        .Caption = "Calibration"
    Case "LT"
        .Caption = "Lock Test"
    Case "TP"
        .Caption = "Tank Profile"
    Case "IP"
        .Caption = "Interface Profile"
    Case "BT"
        .Caption = "Balance Test"
    Case "BL"
        .Caption = "Displacer Halted"
    Case "GD"
        .Caption = "Maint. Mode Go Down"
    Case "GU"
        .Caption = "Maint. Mode Go Up"
    Case "DM"
        .Caption = "Dip Mode"
    Case Else
        .Caption = ""
End Select
End With

End Sub

Sub ciuopen()
'=====
' This procedure was originally written by Delft Instruments. Minor
' modifications were made by John Huber. This procedure is proprietary
' to Delft Instruments and may not be used or reproduced without permission.
'=====

Dim CR
CR = Chr(13) + Chr(10)
On Error GoTo ErrorHandler
' *****
' Open RS232-C channel for CIU or PET
' *****
baudrate = baudrate setting for comm.
compoort = 1 or 2 (DOS comport setting)
' The serial port is as follows opened :
'   0 = odd parity
'   7 = 7 data bits
'   1 = 1 stop bit
'=====
'This sub uses the MSCOMM control shipped with MS Visual Basic 5.0
'It is intended for use with a dedicated RS232-C connection cable and is
'not adequate for use with a dialup modem.
'=====
'////////////////////////////////////
'THE FOLLOWING SETTINGS AND CODE ARE RETAINED IN COMMENT
```

HNF-1569
Revision 1

```

'FORM FOR INFORMATION ONLY:
'   RS      Suppresses detection of Request To Send (RTS).
'   CS[m]    Controls the timeout on the Clear To Send line (CTS). If
'             CTS is low (there is no signal) for more than m
'             milliseconds, a device timeout occurs (m=0)
'   DS[m]    Controls the timeout on the Data Set Ready line (DSR). If
'             DSR is low for more than m milliseconds, a device timeout
'             occurs (m=0).
'   CD[m]    Controls the timeout on the Data Carrier Detect line (DCD).
'             If DCD is low for more than m milliseconds, a device
'             timeout occurs.
'
' note : this settint is not tested for modem operation !
'
' Close #99
' Open "COM" + Compoort$ + ":" + Baudrate$ + ",0,7,1,RS,CS0,DS0,CD0" For Random As #99
'////////////////////////////////////
'sett will contain the port settings.
Dim sett$

If frmMain.Comm2.CommPort <> Compoort Then
    frmMain.Comm2.CommPort = Compoort
End If

frmMain.Comm2.PortOpen = Not frmMain.Comm2.PortOpen

If frmMain.Comm2.PortOpen Then

    frmMain.Comm2.PortOpen = False
    frmMain.Comm2.RTSEnable = False

    'use default settings located in the frmCOMMAdv form.
    'sett = frmCOMMAdv!txtParity.text + ", " + frmCOMMAdv!txtDataBits.text + ", " +
    frmCOMMAdv!txtStopBits.text
    sett = Parity + ", " + DataBits + ", " + StopBits

    'open the port using the settings.
    If frmMain.Comm2.CommPort <> Compoort Then
        frmMain.Comm2.CommPort = Compoort
    End If
    frmMain.Comm2.Settings = Baudrate + ", " + sett
    'Debug.Print "Comm Settings = "; Baudrate + ", " + sett
    frmMain.Comm2.InputLen = 0
    frmMain.Comm2.PortOpen = True
Else
    frmMain.Comm2.RTSEnable = False
    'sett = frmCOMMAdv!txtParity.text + ", " + frmCOMMAdv!txtDataBits.text + ", " +
    frmCOMMAdv!txtStopBits.text
    sett = Parity + ", " + DataBits + ", " + StopBits
    frmMain.Comm2.CommPort = Compoort
    frmMain.Comm2.Settings = Baudrate + ", " + sett
    frmMain.Comm2.InputLen = 0
    frmMain.Comm2.PortOpen = True
End If
'frmBug.txtD.text = 'frmBug.txtD.text + "Comm2 open.." + CR

'set system status indicators
ShowPic "COMM"
'frmMain.pnlCOMM.Caption = "COMM Open"
frmMain.sBar.SimpleText = "Comport open, ready to communicate with CIU...."

'set a global flag indicating success
commIsOpen = 1
Exit Sub

ErrorHandler:
MsgBox "Error encountered while trying to open Com Port"
Exit Sub

```


HNF-1569
Revision 1

End Sub

Sub decode(ByVal item As String, ByVal gageEcho As String, ByVal length As Integer, ByRef inhoud As String)

'This sub decodes a standard ciucom echo string. I don't full understand
'how it works, but it does.

On Error GoTo ErrorHandler

Dim tempLength As Integer

Dim lengthEcho As Integer

' SUB welke het gezocht item uit de echo haalt

' SUB which the sought after item from the echo stroke

tempLength = InStr(gageEcho, item) + Len(item) 'controleer of lengte van
'echo voldoende is

' controler or length of

' echo is sufficient.

lengthEcho = Len(gageEcho)

If tempLength <= lengthEcho And lengthEcho > 0 Then

inhoud = Mid(gageEcho, tempLength, length)

End If

Exit Sub

ErrorHandler:

MsgBox "Error encountered decoding: " + gageEcho

Exit Sub

End Sub

Sub receiveCIU(ByRef gageEcho As String, status)

'=====

' This procedure was originally written by Delft Instruments. Minor
' modifications were made by John Huber. This procedure is proprietary
' to Delft Instruments and may not be used or reproduced without permission.

'=====

'this function extracts valid echo strings from the ciu. It must be
'used in conjunction with sendCIU sub

On Error Resume Next

Dim bcc As Integer

Dim timerCount As Integer

Dim tempTxt As String

Dim message As String

Dim Reply As Integer

Dim abyte As String

Dim CR

CR = Chr(13) + Chr(10)

timerCount = 0

If frmMain.Comm2.InBufferCount >= 2 Then

If ja = 0 Then 'If we haven't yet verified a gauge with bepaaltype (bepaaltype sets ENRAF pic)

If onDisconnect_Click = 0 Then

ShowPic "CIU"

End If

End If

Else

If commIsOpen = 1 Then ShowPic "COMM"

status = 1

GoTo ciucomExit

HNF-1569
Revision 1

```

End If
'=====
' PREPARE TO RECEIVE REPLY FROM CIU/GAUGE
'=====
gageEcho = ""

timerCount = 0
Do
DoEvents
'=====
'If good data, Read in each character and process...
'=====
frmMain.Comm2.InputLen = 1          'Get one abyte from buffer
abyte = frmMain.Comm2.Input        'Fetch a abyte...
If abyte = "" Then Exit Do
If Asc(abyte) = 2 Or timerCount >= 9 Then Exit Do
If Asc(abyte) = 6 Then              'ACK received from CIU
    Dim oldpic As String
    If frmMain.imgRED.Visible = True Then
        oldpic = "RED"
    ElseIf frmMain.imgGRN.Visible = True Then
        oldpic = "GREEN"
    ElseIf frmMain.imgGRY.Visible = True Then
        oldpic = "OFF"
    End If
    'frmMain.sBar.SimpleText = "CIU Busy....."
    ShowLED "RED"
    Delay (busy_waittime)
    frmMain.sBar.SimpleText = ""
    ShowLED oldpic
End If
    timerCount = timerCount + 1
Loop 'Until (Asc(abyte) = 2) Or (timerCount >= 9)
If abyte = "" Then
    status = 1
    timerCount = 0
    GoTo ciucomExit
End If
If (Asc(abyte) <> 2) And (timerCount >= 9) Then
    status = 1
    timerCount = 0
    'Debug.Print "Didn't get a 2...exiting"
    GoTo ciucomExit
End If
frmMain.lblCStatus.Caption = "Connected...."

bcc = 3                                ' Precompute 0 XOR ETX
Do                                    ' Receive-the-answer-loop

    Delay (20 / 1000)                  ' Wait for character

    If frmMain.Comm2.InBufferCount = 0 Then ' if no character in buffer
        gageEcho = ""
        status = 2                      ' message incomplete
        GoTo ciucomExit
    End If

    frmMain.Comm2.InputLen = 1
    abyte = frmMain.Comm2.Input          ' Read one abyte from channel,...

    If Asc(abyte) = 3 Then
        Exit Do                          ' ...if it's an ETX stop...
    End If

    bcc = bcc Xor Asc(abyte)              ' ...update BCC with it...

    Select Case Asc(abyte)
        Case 7                            ' Replace special characters
            abyte = "3"                    ' ... cubic metres
    End Select

```

HNF-1569
Revision 1

```

    Case 8                                ' ... degrees
    abyte = "°"

    Case 9                                ' ... arrow up
    abyte = Chr(24)

    Case 10                               ' ... arrow down
    abyte = Chr(25)

    Case Is < 32                           ' other non valid ASCII-
    abyte = " "                           ' characters

    End Select                             '
    gageEcho = gageEcho + abyte           ' ...and append it to gageEcho
    DoEvents

Loop                                     ' Get out of loop only with
                                     ' an ETX character or buffer
                                     ' empty

'get BCC
If frmMain.Comm2.InBufferCount = 0 Then ' if no character in buffer
    gageEcho = ""
    status = 2                           ' message incomplete
    GoTo ciucomExit
End If

    frmMain.Comm2.InputLen = 1
    abyte = frmMain.Comm2.Input          ' else read BCC-character
If Asc(abyte) <> bcc Then                 ' test if it matches the message
    gageEcho = ""
    status = 3                           ' BCC MISMATCH
    GoTo ciucomExit
End If

status = 0                             ' Normal exit: no error

'If length echo > 3 and Instrument <> ???, we're communicating with a gauge.
If Len(gageEcho) > 3 Then
    If Instrument <> "???" Then
        ShowPic "ENRAF"
    Else
        ShowPic "CIU"
    End If
End If

Exit Sub

'=====
' String has been read.
'=====

ErrorHandler:

ciucomExit:

    If commIsOpen = 1 Then
        ShowPic "COMM"
    Else
        ShowPic "SYSTEM"
    End If
    Dim s$
    Select Case status
    Case 0
        frmMain.lblCStatus.Caption = "CIU Communication Terminated...."
        s = "CIU Communication Terminated...."
    Case 1
        frmMain.lblCStatus.Caption = "CIU Timed Out...."
        s = "CIU Timed Out...."
    Case 2
        frmMain.lblCStatus.Caption = "Response from CIU Incomplete...."
        s = "Response from CIU Incomplete...."

```

HNF-1569
Revision 1

```

Case 3
    frmMain.lblCStatus.Caption = "BCC Mismatch...."
    s = "BCC Mismatch...."
Case Else
    frmMain.lblCStatus.Caption = "General CIU Error...."
    s = "General CIU Error...."
End Select

' Handle errors raised.
If Err.Number <> 0 Then
    Dim Msg$
    Msg = "Error # " & Str(Err.Number) & " was generated by " _
        & Err.Source & Chr(13) & Err.Description & Chr(13) _
        & Chr(13) & "status: " & s & Chr(13) _
        & "Please write down this message exactly and " _
        & "forward to program developer."
    MsgBox Msg, , "Error"
End If
End Sub

Sub sendtoCIU(ByVal request As String)
'=====
' This procedure was originally written by Delft Instruments. Minor
' modifications were made by John Huber. This procedure is proprietary
' to Delft Instruments and may not be used or reproduced without permission.
'=====

'This sub sends a request to the ciu
On Error Resume Next
    Dim bcc As Integer
    Dim abyte As String
    Dim i%
    Dim CR
    CR = Chr(13) + Chr(10)
    Dim rectype As String
    ShowLED "GREEN"
    If GotCIU = 1 Then ShowPic "CIU"
    ' ciuadres = string specifying the CIU adres ('0'-'9', 'A'-'H')
    '         an empty string('') will communicate with the
    '         a PET (847).
    '         'Vhhh uLect mode : SERV_GPU_hhh_
    '         'Wn  uLect mode : SERV_CIU_n_
    ' ta      = transmission adres ('00'-'99')
    ' timeout = timeout in seconds (set to : 0.9 - 2 sec.)
    ' turn_around = turnaround delay (0 - 2 sec.)
    ' request = the string to be send to gauge
    '         SUB CiuCom will envelop it as follows:
    '         <STX><ciuadres><ta><request><ETX><BCC>
    ' gageEcho = answer from CIU complete with ciuadres and ta
    ' status = communication status
    '         0 = successful
    '         1 = time out
    '         2 = message incomplete
    '         3 = BCC mismatch
    '         IF status <> 0 THEN gageEcho = ""
    ' toi      = type of instrument, usually "B" for 854's
    ' rectype = type of record (A, B, C, D, E, F, X, Z, etc)
    '         should already be included in request string.

'=====
'SEND A STRING TO THE CIU.....
'=====
request = ciuadres & ta & toi & request
If frmMain.Comm2.PortOpen = False Then Exit Sub

frmMain.Comm2.InBufferCount = 0      'Flush Input Buffer
frmMain.Comm2.OutBufferCount = 0    'Flush Output Buffer

If ciuadres <> "" Then
    frmMain.Comm2.Output = Chr(0)    ' Send a null-character to CIU

```

HNF-1569
Revision 1

```

' do not for 847 PET
End If
frmMain.Comm2.Output = Chr(2) ' Send an STX-character
bcc = 3 ' Precompute 0 XOR ETX

For i = 1 To Len(request) ' Sending the request
    DoEvents
    abyte = Mid(request, i, 1) ' Fetch current abyte from request...
    frmMain.Comm2.Output = abyte ' ...send it...
    bcc = bcc Xor Asc(abyte) ' ...and update BCC
Next i ' Send next abyte
frmMain.Comm2.Output = Chr(3) ' Send an ETX-character
frmMain.Comm2.Output = Chr(bcc) ' Send the BCC-character
'Debug.Print "Sent bcc= "; bcc

Delay (turn_around / 1000) ' Wait for response
' response

'=====
' STRING SENT....
'=====

' Handle errors raised.
If Err.Number <> 0 Then
    Dim Msg$
    Msg = "Error # " & Str(Err.Number) & " was generated by " _
        & Err.Source & Chr(13) & Err.Description & Chr(13) _
        & Chr(13) & "status: " & status & Chr(13) _
        & "Please write down this message exactly and " _
        & "forward to program developer."
    MsgBox Msg, , "Error"
End If

End Sub

Sub GetTankDimensions()
'This function simple obtains TT and RL and stores them in their
'respective global variables of the same name. These values are used
'to make the level gauge animation in frmDensity work.
Dim echo As String
Dim i%
Dim answer As String
On Error Resume Next
    'get RL & TT for future reference, if 854 gauge
    If Instrument = "854" Then
        For i = 1 To 3
            ciucom "ZRL", echo, status
            If Len(echo) > 3 Then
                Exit For
            End If
        Next i
        If i = 3 Then
            Delay (0.3)
            ciucom "ZRL", echo, status
            If Len(echo) > 3 Then
                answer = Z1BLevel(echo)
            End If
        End If
        answer = Z1BLevel(echo)
        RL = CDBl(answer)
        'Debug.Print "RL= "; RL
        For i = 1 To 3
            ciucom "ZTT", echo, status
            If Len(echo) > 3 Then
                Exit For
            End If
        Next i
        If i = 3 Then
            Delay (0.3)
            ciucom "ZTT", echo, status
            If Len(echo) > 3 Then

```

```
        answer = Z1BLevel(echo)
    End If
End If
    answer = Z1BLevel(echo)
    TT = CDB1(answer)
    'Debug.Print "TT= "; TT
End If
' Handle errors raised.
If Err.Number <> 0 Then
    Dim Msg$
    Msg = "Error # " & Str(Err.Number) & " was generated by " _
        & Err.Source & Chr(13) & Err.Description & Chr(13) _
        & Chr(13) & "status: " & status & Chr(13) _
        & "Please write down this message exactly and " _
        & "forward to program developer."
    MsgBox Msg, , "Error"
End If

End Sub

Sub ShowLED(ByVal image As String)
'This sub flashes the LED next to the communications status pictures to
'indicate ciu communication activity. It is controled from the receiveCIU
'sub

On Error Resume Next
image = Trim(UCase(image))

Select Case image

    Case "RED"
        frmMain.imgRED.Visible = True
        frmMain.imgGRN.Visible = False
        frmMain.imgGRY.Visible = False

    Case "GREEN"
        frmMain.imgRED.Visible = False
        frmMain.imgGRN.Visible = True
        frmMain.imgGRY.Visible = False

    Case "OFF"
        frmMain.imgRED.Visible = False
        frmMain.imgGRN.Visible = False
        frmMain.imgGRY.Visible = True

    Case Else
        frmMain.imgRED.Visible = False
        frmMain.imgGRN.Visible = False
        frmMain.imgGRY.Visible = False

End Select

' Handle errors raised.
If Err.Number <> 0 Then
    Dim Msg$
    Msg = "Error # " & Str(Err.Number) & " was generated by " _
        & Err.Source & Chr(13) & Err.Description & Chr(13) _
        & Chr(13) & "status: " & status & Chr(13) _
        & "Please write down this message exactly and " _
        & "forward to program developer."
    MsgBox Msg, , "Error"
End If

End Sub

Sub DisplayStuff(ByVal req1 As String, ByVal req2 As String, ByVal LevelEcho As String, ByVal
GenEcho As String)
'This sub takes echos and request strings, decodes and displays the results.
'it is called from the tmrLevel routine.
```

HNF-1569
Revision 1

```

On Error Resume Next
Dim oldlevel$
Dim LevelNum As Double
Dim tmp$
Dim CR
CR = Chr(13) + Chr(10)
Dim answer As String

If (Len(frmMain.txtLevelDisplay.Text) >= 8) And Not InStr(frmMain.txtLevelDisplay.Text, "F") Then
    tmp = Mid(frmMain.txtLevelDisplay.Text, (InStr(frmMain.txtLevelDisplay.Text, ".") - 5), 8)
End If

If InStr(req1, "1B") = 0 Then
    decode req1, LevelEcho, 64, answer
    frmMain.txtRE.Text = frmMain.txtRE.Text + ">" + answer + CR
End If
If InStr(LevelEcho, "Z1B") Then
    decode "1B", LevelEcho, 32, answer
    frmMain.txtLevelDisplay.Text = Trim(answer)
    If InStr(answer, ".") Then
        tmp = Mid(answer, (InStr(answer, ".") - 5), 8)
        If tmp = "" Then tmp = "0"
        LevelNum = CDBl(tmp)
    End If
End If

decode req2, GenEcho, 32, answer

frmMain.txtDisplay2.Text = answer

If frmDensity.Visible = True Then
    LMeter TT, LevelNum
End If

frmMain.brdDisplay.Caption = Instrument & " Gauge Display"

' Handle errors raised.
If Err.Number <> 0 Then
    Dim Msg$
    Msg = "Error # " & Str(Err.Number) & " was generated by " _
        & Err.Source & Chr(13) & Err.Description & Chr(13) _
        & Chr(13) & "status: " & status & Chr(13) _
        & "Please write down this message exactly and " _
        & "forward to program developer."
    MsgBox Msg, , "Error"
End If

End Sub

Function GetErrorMessage(ByVal code As String) As String
    'This function handles XPU error codes only
    'Pass it a previously obtained and decoded XPU error request

On Error Resume Next
Dim Msg As String
Select Case code
    Case "!050"
        Msg = "XPU error" + Right(code, 3) + ". Invalid length. The item identifier length must be at least two characters"
    Case "!051"
        Msg = "XPU error" + Right(code, 3) + ". Unknown item"
    Case "!052"
        Msg = "XPU error" + Right(code, 3) + ". Error in message format: Assign expected as 3rd character"
    Case "!053"
        Msg = "XPU error" + Right(code, 3) + ". Wrong data field length"
    Case "!054"
        Msg = "XPU error" + Right(code, 3) + ". No read access. No read access rights for this item"

```

HNF-1569
Revision 1

```

Case "!055"
Msg = "XPU error" + Right(code, 3) + ". No write access. No write access rights for this item"
Case "!056"
Msg = "XPU error" + Right(code, 3) + ". Wrong protection level"
Case "!057"
Msg = "XPU error" + Right(code, 3) + ". Access denied. Another command source has entered
protection level 1 or 2"
Case "!058"
Msg = "XPU error" + Right(code, 3) + ". Syntax error. A command cannot contain a setting"
Case "!060"
Msg = "XPU error" + Right(code, 3) + ". Syntax error. Non ASCII character found"
Case "!061"
Msg = "XPU error" + Right(code, 3) + ". Syntax error. Was expecting a digit 0...9"
Case "!062"
Msg = "XPU error" + Right(code, 3) + ". Syntax error. Invalid sign in the mantissa sign position"
Case "!063"
Msg = "XPU error" + Right(code, 3) + ". Syntax error. No decimal point in the decimal point
position"
Case "!064"
Msg = "XPU error" + Right(code, 3) + ". Syntax error. At the E position, no E was found"
Case "!065"
Msg = "XPU error" + Right(code, 3) + ". Syntax error. Invalid exponent sign"
Case "!066"
Msg = "XPU error" + Right(code, 3) + ". Syntax error. Invalid sign"
Case "!067"
Msg = "XPU error" + Right(code, 3) + ". Invalid level format. The format of the level item is not
according to the level dimension item [LD]"
Case "!068"
Msg = "XPU error" + Right(code, 3) + ". Syntax error. No quote"
Case "!069"
Msg = "XPU error" + Right(code, 3) + ". Syntax error. No double quote"
Case "!070"
Msg = "XPU error" + Right(code, 3) + ". Wrong selection"
Case "!071"
Msg = "XPU error" + Right(code, 3) + ". Syntax error. At the position of the decimal separator,
the character"
Msg = Msg + " which was found does not match the programmed decimal separator item [DP]"
Case "!072"
Msg = "XPU error" + Right(code, 3) + ". Syntax error. Inches overrun error"
Case "!073"
Msg = "XPU error" + Right(code, 3) + ". Fractions overrun error"
Case "!075"
Msg = "XPU error" + Right(code, 3) + ". In valid density format"
Case "!076"
Msg = "XPU error" + Right(code, 3) + ". Invalid floating point format"
Case "!080"
Msg = "XPU error" + Right(code, 3) + ". No protectoin level entered"
Case "!082"
Msg = "XPU error" + Right(code, 3) + ". Wrong password"
Case Else
Msg = "XPU error" + Right(code, 3) + ". Refer to manual"
End Select
' Handle errors raised.
If Err.Number <> 0 Then
Dim Mesg$
Mesg = "Error # " & Str(Err.Number) & " was generated by " _
& Err.Source & Chr(13) & Err.Description & Chr(13) _
& Chr(13) & "status: " & status & Chr(13) _
& "Please write down this message exactly and " _
& "forward to program developer."
MsgBox Mesg, , "Error"
End If

GetErrorMessage = Msg

End Function

Function FormatTensionString2(ByVal textString As String) As String
On Error Resume Next
Dim i%, tmpString As String

```


HNF-1569
Revision 1

```
textString = Trim(textString)
' Start building the string
tmpString = "+."
For i = 1 To Len(textString)
    If Mid(textString, i, 1) <> "." Then
        tmpString = tmpString & Mid(textString, i, 1)
    End If
Next i
```

```
Debug.Print "tmpString no .'s= "; tmpString
```

```
' Now check length of resulting string and adjust
```

```
If Len(tmpString) < 10 Then
```

```
    ' Start adding zeros
```

```
    Do
```

```
        tmpString = tmpString & "0"
```

```
        ' check length
```

```
        If Len(tmpString) >= 10 Then Exit Do
```

```
    Loop
```

```
    ' add E+03
```

```
    tmpString = tmpString & "E+03"
```

```
    Debug.Print "tmpString with zeros= "; tmpString
```

```
ElseIf Len(tmpString) > 10 Then
```

```
    ' truncate
```

```
    tmpString = Left(tmpString, 10)
```

```
    ' add E+03
```

```
    tmpString = tmpString & "E+03"
```

```
    Debug.Print "tmpString truncated= "; tmpString
```

```
End If
```

```
FormatTensionString2 = tmpString
```

```
End Function
```

```
Function FormatTensionString(ByVal textString As String, ByRef Error As Boolean) As String
```

```
'This function creates a correctly formatted S1,S2,S3 setting given a value in grams,
```

```
'and formatted in nearly any non scientific notation manner. It returns a boolean
```

```
'error code if there was a problem.
```

```
On Error Resume Next
```

```
Dim i%
```

```
textString = Trim(textString)
```

```
If InStr(textString, ".") <> 0 Then
```

```
    If InStr(UCase(textString), "E") Then
```

```
        Error = True
```

```
        MsgBox "Do not use PET entry format."
```

```
        Exit Function
```

```
    End If
```

```
textString = Mid(textString, 1, InStr(textString, ".") - 1)
```

```
End If
```

```
If Left(textString, 1) = "0" Then
```

```
textString = Right(textString, Len(textString) - 1)
```

```
End If
```

```
If Left(textString, 1) = "0" Then
```

```
textString = Right(textString, Len(textString) - 1)
```

```
End If
```

```
For i = 1 To Len(textString)
```

```
    If CInt(Mid(textString, i, 1)) < 0 Or CInt(Mid(textString, i, 1)) > 9 Then
```

```
        Error = True
```

```
        MsgBox "Non-numerics in Wire Tension String"
```

```
    End If
```

```
Next i
```

```
If Len(Trim(textString)) > 3 Then
```

HNF-1569
Revision 1

```
Error = True
MsgBox "You have entered a number that is too long"
Exit Function
ElseIf Len(Trim(textString)) = 3 Then
    FormatTensionString = "+" + Trim(textString) + "00000E+03"
ElseIf Len(Trim(textString)) = 2 Then
    FormatTensionString = "+.0" + Trim(textString) + "00000E+03"
ElseIf Len(Trim(textString)) = 1 Then
    FormatTensionString = "+.00" + Trim(textString) + "00000E+03"
End If
' Handle errors raised.
If Err.Number <> 0 Then
    Dim Msg$
    Msg = "Error # " & Str(Err.Number) & " was generated by " _
        & Err.Source & Chr(13) & Err.Description & Chr(13) _
        & Chr(13) & "status: " & status & Chr(13) _
        & "Please write down this message exactly and " _
        & "forward to program developer."
    MsgBox Msg, , "Error"
End If

End Function

Function GetSPUErrorMessage(ByVal code As String) As String
'This function handles SPU error codes only
'Pass it a previously obtained and decoded XPU error request

On Error Resume Next
Dim Msg As String
Dim CR
CR = Chr(13) + Chr(10)
Select Case code
    Case "0601"
        Msg = "SPU error" + Right(code, 4) + "." + CR + "Stress transducer error 1. The frequency of the stress transducer is too low."
    Case "0602"
        Msg = "SPU error" + Right(code, 4) + "." + CR + "Stress transducer error 2. The frequency of the stress transducer is too high."
    Case "0603"
        Msg = "SPU error" + Right(code, 4) + "." + CR + "On line EPROM error. Replace the EPROM."
    Case "0604"
        Msg = "SPU error" + Right(code, 4) + "." + CR + "On line external RAM error. Replace the RAM chip."
    Case "0610"
        Msg = "SPU error" + Right(code, 4) + "." + CR + "Wire tension too low. the wire tension has been too low for a while. Solve the problem or change the [ML] setting."
    Case "0611"
        Msg = "SPU error" + Right(code, 4) + "." + CR + "Wire tension too high. The wire tension has been too high for a while. Solve the problem, clean the displacer or change the [ML] setting."
    Case "0407"
        Msg = "SPU error" + Right(code, 4) + "." + CR + "Stress transducer initialisation error. The stress transducer does not start up correctly, or the motor unit has not been unlocked."
    Case Else
        Msg = "SPU error" + Right(code, 4) + "." + CR + "Refer to manual"
End Select

' Handle errors raised.
If Err.Number <> 0 Then
    Dim Msg$
    Msg$ = "Error # " & Str(Err.Number) & " was generated by " _
        & Err.Source & Chr(13) & Err.Description & Chr(13) _
        & Chr(13) & "status: " & status & Chr(13) _
        & "Please write down this message exactly and " _
        & "forward to program developer."
    MsgBox Msg$, , "Error"
End If

GetSPUErrorMessage = Msg

End Function
```

HNF-1569
Revision 1

```
Function MyInputBox(ByVal Prompt As String, ByVal title As String, ByVal default As String) As String
```

```
On Error Resume Next
Dim m$
frmInput.lblPrompt.Caption = Prompt
frmInput.txtValue.Text = default
frmInput.Caption = title
frmInput.Show 1
MyInputBox = Trim(frmInput.txtValue.Text)

' Handle errors raised.
If Err.Number <> 0 Then
    Dim Msg$
    Msg = "Error # " & Str(Err.Number) & " was generated by " _
        & Err.Source & Chr(13) & Err.Description & Chr(13) _
        & Chr(13) & "status: " & status & Chr(13) _
        & "Please write down this message exactly and " _
        & "forward to program developer."
    MsgBox Msg, , "Error"
End If

End Function
```

```
Function DoBalanceTest(ByVal DisplayResults As Boolean) As Boolean
' *****
' This function causes the gauge to perform a balance test (BT)
' A passing test returns true;
' A failing test returns false
' DisplayResults set to TRUE causes a msgbox to appear.
' The ACTIVITY.LOG file is appended with the results.
' *****
```

```
On Error Resume Next
' Declare variables
Dim m$, R% ' Message and Reply variables
Dim echo$, answer$
Dim dBU As Double
Dim dBV As Double
Dim dDiff As Double
Dim Passed As Boolean

' Warn about freeplay below displacer
m = "Warning! Ensure there is at least 15 inches of " & vbNewLine
m = m & "freeplay beneath the displacer. Click OK to continue, " & vbNewLine
m = m & "or CANCEL to quit."

R = MsgBox(m, vbOKCancel + vbExclamation, "Attention!")
If R <> vbOK Then
    Exit Function
End If
```

```
' IF scanning, stop
If gettingScans = 1 Then
    frmMain.mnuDisconnect_Click
' ELSE Check for connection and connect
ElseIf gaugeChecked = 0 Then
    frmMain.txtDisplay2.Text = "Initializing..."
    checkGageAddress ta, ja
    frmMain.txtDisplay2.Text = ""
End If
If ja = 0 Then Exit Function

' Issue balance test command
ciucom "ZBT", echo, status
Debug.Print "BT echo= "; echo
decode "BT", echo, 32, answer
Debug.Print "BT answer= "; answer
If InStr(answer, "&") = 0 Then
```

HNF-1569
Revision 1

```

    MsgBox "Gauge did not accept balance test (BT) command. Aborting!"
    Exit Function
Else
    ' Wait for balance test to complete
    Do
        GetALevel s1B, s2B
        If InStr(s2B, "BT") = 0 Then
            Exit Do
        End If
        DoEvents
    Loop
End If
frmMain.sBar.SimpleText = "Balance Test Complete..."

' Get BU and BV
ciucom "ZBU", echo, status
decode "BU", echo, 32, answer
dBU = CDBl(answer)

ciucom "ZBV", echo, status
decode "BV", echo, 32, answer
dBV = CDBl(answer)

' Calculate difference
dDiff = Abs(dBU - dBV)

' Report results
m = "Balance Test complete " & vbNewLine
m = m & "BU = " & Format(Str(dBU), "###.##") & vbNewLine
m = m & "BV = " & Format(Str(dBV), "###.##") & vbNewLine
m = m & "Difference = " & Format(Str(dDiff), "###.##") & vbNewLine

If dDiff > 3 Or dBU <= 0 Or dBV <= 0 Then
    m = m & "Balance test FAILED!"
    If DisplayResults = True Then
        MsgBox m, vbOKOnly + vbInformation, "Balance Test Failed!"
    End If
    Passed = False
    DoBalanceTest = False
Else
    m = m & "Balance test PASSED!"
    If DisplayResults = True Then
        MsgBox m, vbOKOnly + vbInformation, "Balance Test Passed!"
    End If
    Passed = True
    DoBalanceTest = True
End If

' Set global flag
BalanceTestPassed = Passed

' Record in Activity log
Dim F, theValue$
If Passed = True Then theValue = "Passed"
If Passed = False Then theValue = "Failed"

F = FreeFile
Open App.Path & "\DENSITY\ACTIVITY.LOG" For Append As F
    Print #F, "=====
    Print #F, "          BALANCE TEST RESULTS"
    If TestMode = 1 Then
        Print #F, "          DEMO MODE RUN, SIMULATED DATA"
    End If
    Print #F, "Date", "Pass/Fail", "Diff (< 3 is passing)"
    Print #F, "=====
    Print #F, Date$, theValue, Format(Str(dDiff), "###.##")
    Print #F, "^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
Close #F

End Function

```

HNF-1569
Revision 1

```
Function WeighDisplacer(ByVal DisplayStuff As Boolean) As String
' *****
' This function causes the gauge to weigh the displacer
' DisplayResults set to TRUE causes a msgbox to appear.
' The ACTIVITY.LOG file is appended with the results.
' *****
Dim echo$, answer$, dWQ$
Dim nResume%

' IF scanning, stop
If gettingScans = 1 Then
    frmMain.mnuDisconnect_Click
' ELSE Check for connection and connect
ElseIf gaugeChecked = 0 Then
    frmMain.txtDisplay2.Text = "Initializing..."
    checkGageAddress ta, ja
    frmMain.txtDisplay2.Text = ""
End If
If ja = 0 Then Exit Function

GetTankDimensions

' Remember current level/status
GetALevel s1B, s2B

Dim L%
' Make sure displacer is low enough
If CL >= (TT - 7) Then
    frmMain.sBar.SimpleText = "Displacer is too high...Lowering!"
    Delay (1)
    GetALevel
    ciucom "ZUN", echo, status
    For L = 1 To 4
        GetALevel
        DoEvents
    Next L
    ciucom "ZFR", echo, status
    frmMain.sBar.SimpleText = ""
End If

' Raise it a little
ciucom "ZUN", echo, status
ciucom "ZLT", echo, status
'Wait 3 seconds
For L = 1 To 3
    GetALevel
    DoEvents
Next L

'Stop the displacer
ciucom "ZFR", echo, status
GetALevel

'Weigh the displacer.
ciucom "ZMF", echo, status

'Wait for MF to complete
Do
    GetALevel s1B, s2B
    If InStr(s2B, "FR") <> 0 Then
        Exit Do
    End If
    DoEvents
Loop

ciucom "ZWQ", echo, status
decode "WQ", echo, 64, answer
```

HNF-1569
Revision 1

```
dWQ = answer

If DisplayStuff = True Then
    MsgBox "Displacer weight (WQ) = " & dWQ
End If

'place result into boxes where they can use them
frmMain.txtCustomCommand.Text = dWQ
frmService.txtItem.Text = dWQ

Dim R%
R = MsgBox("Set S1, S3 and RM to 15 gram delta?", vbOKCancel + vbQuestion, "Adjust Setpoints?")
If vbOK Then
    Dim ddSET As Double
    Dim Error As Boolean
    ddSET = CDbl(dWQ)
    ddSET = ddSET - 15
    EnterPassword (R)
    If R = 0 Then
        Exit Function
    End If
    ciucom "ZS1=" & FormatTensionString(ddSET, Error), echo, status
    If Error Then
        ciucom "ZEX", echo, status
        MsgBox "Error entering values. Aborting!"
        For L = 1 To 5
            GetALevel
            DoEvents
        Next L
        GoTo Aborting
    End If
    ciucom "ZS3=" & FormatTensionString(ddSET, Error), echo, status
    If Error Then
        ciucom "ZEX", echo, status
        MsgBox "Error entering values. Aborting!"
        For L = 1 To 5
            GetALevel
            DoEvents
        Next L
        GoTo Aborting
    End If
    ciucom "ZRM=" & FormatTensionString(ddSET, Error), echo, status
    If Error Then
        ciucom "ZEX", echo, status
        MsgBox "Error entering values. Aborting!"
        For L = 1 To 5
            GetALevel
            DoEvents
        Next L
        GoTo Aborting
    End If
    ciucom "ZEX", echo, status
    For L = 1 To 5
        GetALevel
        DoEvents
    Next L
End If

Aborting:

If nResume = 1 Then
    frmMain.mnuConnectCIU_Click
End If

frmMain.sBar.SimpleText = ""

Dim PNS
ciucom "ZPN", echo, status
decode "PN", echo, 32, answer
PN = answer
' Record in Activity.log
```

HNF-1569
Revision 1

```
Dim F

F = FreeFile
Open App.Path & "\DENSITY\ACTIVITY.LOG" For Append As F
Print #F, "=====
Print #F, "          WEIGH DISPLACER RESULTS"
Print #F, "          Gauge S/N: " & PN
If TestMode = 1 Then
Print #F, "          DEMO MODE RUN, SIMULATED DATA"
End If
Print #F, "Date", "Time", "Weight"
Print #F, "=====
Print #F, Date$, Time$, Format(Str(dWQ), "###.##")
Print #F, "^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^"
Close #F
WeighDisplacer = dWQ
End Function
```

End Globals.bas

```
=====
Begin FRMLEVEL.FRM
```

VERSION 5.00

Begin VB.Form frmLevelCal

```
Caption      = "854 Level Calibration"
ClientHeight = 3195
ClientLeft   = 2310
ClientTop    = 4230
ClientWidth  = 5760
LinkTopic    = "Form1"
PaletteMode  = 1 'UseZOrder
ScaleHeight  = 3195
ScaleWidth   = 5760
StartPosition = 2 'CenterScreen
```

Begin VB.CommandButton cmdUN

```
Caption      = "UN"
Height       = 372
Left         = 2760
TabIndex     = 6
Top          = 1680
Width        = 372
```

End

Begin VB.CommandButton cmdFR

```
Caption      = "FR"
Height       = 372
Left         = 2760
TabIndex     = 5
Top          = 1320
Width        = 372
```

End

Begin VB.CommandButton cmdCA

```
Caption      = "CA"
Height       = 372
Left         = 2760
TabIndex     = 4
Top          = 960
Width        = 372
```

End

Begin VB.CommandButton cmdAbort

```
Caption      = "&Abort"
Height       = 252
Left         = 4560
TabIndex     = 3
Top          = 2520
Width        = 1092
```

End

Begin VB.CommandButton cmdClose

```
Caption      = "&Close"
Height       = 252
Left         = 2400
TabIndex     = 2
Top          = 2520
Width        = 1092
```

```
End
Begin VB.CommandButton cmdCalibrate
    Caption       = "C&alibrate"
    Height        = 252
    Left          = 120
    TabIndex      = 12
    Top           = 2520
    Width         = 1092
End
Begin VB.Frame Frame2
    Caption       = "Calibration Log"
    Height        = 2292
    Left          = 3240
    TabIndex      = 1
    Top           = 120
    Width         = 2412
    Begin VB.TextBox txtLog
        Height      = 1812
        Left        = 120
        MultiLine   = -1 'True
        ScrollBars  = 3 'Both
        TabIndex    = 15
        Top         = 360
        Width       = 2172
    End
End
Begin VB.Frame Frame1
    Caption       = "Input Data"
    Height        = 2292
    Left          = 120
    TabIndex      = 0
    Top           = 120
    Width         = 2532
    Begin VB.TextBox txtTargetCA
        Height      = 288
        Left        = 1320
        TabIndex    = 10
        Top         = 1800
        Width       = 972
    End
End
Begin VB.TextBox txtSetPoint
    Height        = 288
    Left          = 1320
    TabIndex      = 9
    Top           = 1320
    Width         = 972
End
Begin VB.TextBox txtGaugeID
    Height        = 288
    Left          = 1320
    TabIndex      = 8
    Top           = 840
    Width         = 972
End
Begin VB.TextBox txtCRL
    Height        = 288
    Left          = 1320
    TabIndex      = 7
    Top           = 360
    Width         = 972
End
Begin VB.Label Label4
    Alignment     = 1 'Right Justify
    Caption       = "Targe CA Level"
    Height        = 372
    Left          = 120
    TabIndex      = 16
    Top           = 1800
    Width         = 972
End
Begin VB.Label Label3
```


HNF-1569
Revision 1

```

        Alignment      = 1 'Right Justify
        Caption        = "Setpoint Delta"
        Height         = 372
        Left           = 120
        TabIndex       = 14
        Top            = 1320
        Width          = 972
    End
    Begin VB.Label Label2
        Alignment      = 1 'Right Justify
        Caption        = "Gauge ID"
        Height         = 252
        Left           = 240
        TabIndex       = 13
        Top            = 840
        Width          = 852
    End
    Begin VB.Label Label1
        Alignment      = 1 'Right Justify
        Caption        = "Correct RL"
        Height         = 252
        Left           = 240
        TabIndex       = 11
        Top            = 360
        Width          = 852
    End
End
Begin VB.Timer Timer1
    Enabled          = 0 'False
    Left             = 2040
    Top              = 120
End
Begin VB.Label lblStatus
    BackColor        = &H0000FFFF&
    ForeColor         = &H00000000&
    Height           = 252
    Left             = 120
    TabIndex         = 17
    Top              = 2880
    Width            = 5532
End
Begin VB.Menu mnuFile
    Caption          = "&File"
    Begin VB.Menu mnuFileSaveLog
        Caption      = "&Save Log"
    End
    Begin VB.Menu mnuFilePrintLog
        Caption      = "&Print Log"
    End
    Begin VB.Menu mnuFileBar1
        Caption      = "-"
    End
    Begin VB.Menu mnuFileClose
        Caption      = "&Close"
    End
    Begin VB.Menu mnuFileAbort
        Caption      = "&Abort"
    End
End
End
Attribute VB_Name = "frmLevelCal"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
Option Explicit
'=====
' This module/form is not yet tested and implemented as of
' 12/21/98. The form is, therefore not accessible. It is
' retained for future development.
'=====
```

HNF-1569
Revision 1

```
Dim GaugeIDS
Dim Msg$
Dim CR
Dim Reply%
Dim answer$
Dim TankLevel As Double
' CL As Double ' current level reading
Dim L2$ ' line to of gauge display
Dim echo$
Dim timeFlag%
Dim AsFoundRL$
Dim CorrectRL$
Dim AsFoundTT$
Dim AsFoundLevel$
Dim AsLeftRL$
Dim AsLeftTT$
Dim AsLeftLevel$
Dim ASFoundWQ$
Dim AsLeftWQ$
Dim AsFoundDW$
Dim AsLeftDW$
Dim ProgrammedTT$
Dim VisualInspectFlag%
Dim AbortCal%
Dim PartNo$
Dim DW$ 'programmed displacer weight

Private Sub cmdAbort_Click()
AbortCal = 1
End Sub

Private Sub cmdCA_Click()
Dim echo$
If ja = 0 Then
    checkGageAddress ta, ja
End If
    ciucom "ZCA", echo, status
End Sub

Private Sub cmdCalibrate_Click()

CR = Chr(13) + Chr(10)
Dim i%, j%, k%
AbortCal = 0

If VisualInspectFlag Then GoTo visual_inspect
'Set button stuff
cmdCalibrate.Enabled = False
cmdClose.Enabled = False
lblStatus.Caption = "...Initializing/Gathering Data..."
'Make sure we have communication
If gaugeChecked = 0 Then
    checkGageAddress ta, ja
End If

If ja = 0 Then
    lblStatus.Caption = "Gauge connection failed..."
    GoSub GetOut
End If

GetTankDimensions
ciucom "ZPN", echo, status
decode "PN", echo, 32, PartNo

txtLog.Text = "Enraf Level Calibration" & vbNewLine
txtLog.Text = txtLog.Text & "===== " & vbNewLine
txtLog.Text = txtLog.Text & Time$ & " " & Date$ & vbNewLine
txtLog.Text = txtLog.Text & "Gauge S/N: " & PartNo & vbNewLine
If txtGaugeID.Text = "" Then
    Msg = "You have not entered a Gauge ID." + CR
```

HNF-1569
Revision 1

```

Msg = Msg + "Would you like to enter one at this time?" + CR
Msg = Msg + "Click YES to enter a new value, NO to continue without, or Cancel to abort..."
Reply = MsgBox(Msg, 3 + 32, "Error?")
If Reply = 2 Then
    txtLog.Text = txtLog.Text + "Aborting...." + CR
    lblStatus.Caption = "Aborting..."
    AbortCal = 1
    GoSub check_abort
ElseIf Reply = 6 Then
    txtGaugeID.Text = InputBox("Please enter a new Gauge ID", "Setpoint Delta Entry")
End If
End If

txtLog.Text = txtLog.Text & "Gauge ID: " & txtGaugeID.Text & vbNewLine

ciucum "Z1B", echo, status
decode "1B", echo, 32, answer
If Z1BLevel(echo) > (RL - 50) Then
    Msg = "It appears the displacer may not be at waste level." + CR
    Msg = Msg + "Click YES to send the displacer down" + CR
    Msg = Msg + "Click NO to continue from here."
    Reply = MsgBox(Msg, 4 + 32, "Send Displacer Down?")
    If Reply = 6 Then
        lblStatus.Caption = "Sending displacer to waste level..."
        ciucum "ZUN", echo, status
    End If
End If

GetTankDimensions

txtLog.Text = txtLog.Text + "Programmed RL= " + Str(RL) + CR
'txtLog.text = txtLog.text + "Correct RL= " + txtCRL.text + CR

'Validate input data
change_programmed_RL:
If txtCRL.Text = "" Then txtCRL.Text = "0"
If RL <> Cdbl(txtCRL.Text) Then
    frmRL.Show 1
    If frmRL.optNew.Value = True Then
        txtCRL.Text = InputBox("Please enter a Correct RL", 0 + 16, "Correct RL Entry")
    ElseIf frmRL.optYours.Value = True Then

        txtLog.Text = txtLog.Text + "Changing Prog. RL to Correct RL" + CR

        ciucum "ZW2=ENRAF2", echo, status
        If InStr(echo, "&") Then
            ciucum "ZRL=+" & Format$(txtCRL.Text, "00000.00"), echo, status
            Debug.Print "Entering RL= "; "ZRL=+" & Format$(txtCRL.Text, "00000.00")
            GoSub check_errors
        Else
            MsgBox "Sorry. We have a communication error."

            txtLog.Text = txtLog.Text + "Communication error. Aborting..." + CR
            txtLog.Text = txtLog.Text + Time$ + " " + Date$ + CR

            GoSub GetOut
        End If
        ciucum "ZRL", echo, status
        decode "RL", echo, 32, answer
        Debug.Print "New Gage RL= "; answer
        RL = Cdbl(answer)
        If RL = Cdbl(Trim(txtCRL.Text)) Then
            MsgBox "RL change successful!"
            txtLog.Text = txtLog.Text + "Correct RL Entered...." + CR
        Else
            txtLog.Text = txtLog.Text + "RL Change Unsuccessful. Aborting..." + CR
            txtLog.Text = txtLog.Text + Time$ + " " + Date$ + CR
            Msg = "Sorry, RL change was not successful." + CR
            Msg = Msg + "Please try manual change."
            MsgBox Msg, 16, "Error!"
        End If
    End If
End If

```

```

        GoSub GetOut
    End If
    ElseIf frmRL.optProg.Value = True Then
        ciucom "ZRL", echo, status
        decode "RL", echo, 32, answer
        txtCRL.Text = answer
    ElseIf frmRL.lblRL.Caption = "ABORT" Then
        txtLog.Text = txtLog.Text + "Aborting..."
        lblStatus.Caption = "Aborting..."
        AbortCal = 1
        GoSub check_abort
    End If
End If

CorrectRL = txtCRL.Text
txtLog.Text = txtLog.Text + "Using Correct RL = " + CorrectRL + CR

If txtSetPoint.Text = "" Then
    txtSetPoint.Text = "0"
End If

If CDbl(txtSetPoint.Text) > 20 Then
    Msg = "You have entered a Setpoint Delta equal to " + txtSetPoint.Text + CR
    Msg = Msg + "It seems a little high. Would you like to accept this value?" + CR
    Msg = Msg + "Click YES to accept, NO to change, or Cancel to abort..."
    Reply = MsgBox(Msg, 3 + 32, "Error?")
    If Reply = 2 Then
        txtLog.Text = txtLog.Text + "Aborting...." + CR
        lblStatus.Caption = "Aborting..."
        AbortCal = 1
        GoSub check_abort
    ElseIf Reply = 6 Then
        txtSetPoint.Text = "15"
    Else
        txtSetPoint.Text = InputBox("Please enter a new Setpoint Delta", 0 + 16, "Setpoint Delta Entry")
    End If
End If

ElseIf CDbl(txtSetPoint.Text) < 10 Then
    Msg = "You have entered a Setpoint Delta equal to " + txtSetPoint.Text + CR
    Msg = Msg + "It seems a little low. Would you like to accept the default (15)?" + CR
    Msg = Msg + "Click YES to accept, NO to change, or Cancel to abort..."
    Reply = MsgBox(Msg, 3 + 32, "Error?")
    If Reply = 2 Then
        txtLog.Text = txtLog.Text + "Aborting...." + CR
        lblStatus.Caption = "Aborting..."
        AbortCal = 1
        GoSub check_abort
    ElseIf Reply = 6 Then
        txtSetPoint.Text = "15"
    Else
        txtSetPoint.Text = InputBox("Please enter a new Setpoint Delta", "Setpoint Delta Entry")
    End If
End If

txtLog.Text = txtLog.Text + "Setpoint Delta= " + txtSetPoint.Text + CR

If txtTargetCA.Text = "" Then
    Msg = "You have not entered a Target CA level." + CR
    Msg = Msg + "Click YES to accept the RL as the target, " + CR
    Msg = Msg + "Or click NO to enter a target level"
    Reply = MsgBox(Msg, 4 + 16, "Use RL as Target CA Level?")
    If Reply = 6 Then
        txtTargetCA.Text = Str(RL)
    Else
        txtTargetCA.Text = InputBox("Please Enter a Target CA Level", "Target CA Level")
    End If
End If

```

```
End If

'>>>INPUT DATA IS VALID<<<<<

'Get a current level reading
'Making sure displacer is at waste level
Do
  GetALevel
GoSub check_abort
If InStr(frmMain.txtLevelDisplay.Text, "INN") Then
  TankLevel = CL
  AsFoundLevel = Str(TankLevel)
  Exit Do
Else
  Msg = "Level Readings are not stable. It is recommended that" + CR
  Msg = Msg + "you wait for 'INN' to appear in the display, then" + CR
  Msg = Msg + "click OK to continue." + CR
  Msg = Msg + "click CANCEL to stop waiting."
  frmMsg.lblMsg.Caption = Msg
  frmMsg.Caption = "Level Readings Unstable. Please wait..."
  frmMsg.cmdCancel.Visible = True
  frmMsg.Show
End If
If frmMsg.lblStatus.Caption = "CANCEL" Then
  GetALevel
  TankLevel = CL
  AsFoundLevel = Str(TankLevel)
  txtLog.Text = txtLog.Text + "Stable level reading not acheived" + CR
  txtLog.Text = txtLog.Text + "As Found Level= " + AsFoundLevel + CR
  Exit Do
End If
Loop
If AsFoundLevel <> "" Then
  GetALevel
  TankLevel = CL
  AsFoundLevel = Str(TankLevel)
  txtLog.Text = txtLog.Text + "As Found Level= " + AsFoundLevel + CR
End If

GetALevel
TankLevel = CL

'Raise the displacer and check the weight.
lblStatus.Caption = "Preparing to raise displacer..."
'but first warn if necessary
Msg = "We are about to raise the displacer. Please make sure" + CR
Msg = Msg + "Radiological Coverage is available before continuing..."
Reply = MsgBox(Msg, 1 + 48, "Radiological Coverage Required!")
If Reply <> 1 Then
  txtLog.Text = txtLog.Text + "Aborting...." + CR
  lblStatus.Caption = "Aborting..."
  MsgBox "Aborting..."
  Exit Sub
End If
Debug.Print "TankLevel= "; TankLevel
Debug.Print "TT= "; TT

If TankLevel >= (TT - 18) Then
  Debug.Print "Displacer close to TT"
  ciucom "ZCA", echo, status
  GetALevel
  lblStatus.Caption = "Raising Displacer..."
  Delay (2)
  ciucom "ZFR", echo, status
  GetALevel
  GoTo Above_Ball
Else
```

HNF-1569
Revision 1

```
Debug.Print "Displacer NOT close to TT"
ciucum "ZCA", echo, status
GetALevel
lblStatus.Caption = "Raising Displacer..."
End If
'Timer1.Interval = scanRate

'Timer1.Enabled

' Waiting for displacer to reach target level
Do '>>>>>DO LOOP START<<<<<<<
GoSub check_abort
GoSub check_errors
Delay (scanRate / 1000)

GetALevel
TankLevel = CL
If TankLevel < RL Then
    If TankLevel > (RL - 50) Then
        ciucum "ZFR", echo, status
        GetALevel
        TankLevel = CL
        GoSub check_echo
        GoSub warn
        ciucum "ZCA", echo, status
        GoSub check_echo
    End If
End If

GetALevel
TankLevel = CL

If TankLevel <> 0 And TankLevel >= CDBl(txtTargetCA.Text) Then
    lblStatus.Caption = "Displacer has reached target CA level..."
    ciucum "ZFR", echo, status
    Exit Do
ElseIf TankLevel > (TT - 4) Then
    lblStatus.Caption = "Displacer is at maximum permissible height..."
    ciucum "ZFR", echo, status
    Exit Do
End If
Loop

Above_Ball:
txtLog.Text = txtLog.Text + "Displacer Above Ball Valve..." + CR
MsgBox "Please close ball valve...", 0 + 16, "Close the Ball Valve!"
lblStatus.Caption = "Visually inspect displacer..."
Msg = "Please click OK and inspect the displacer." + CR
Msg = Msg + "Use the CA and FR buttons to adjust the" + CR
Msg = Msg + "Displacer position. Click the CALIBRATE button" + CR
Msg = Msg + "when visual inspection is complete. The program" + CR
Msg = Msg + "will return to this point in the calibration. If" + CR
Msg = Msg + "a flush is required, you must close the 854 Level" + CR
Msg = Msg + "Calibration Window to restart."
MsgBox Msg, 0 + 16, "Visually Inspect the Displacer!"
GoSub check_abort
VisualInspectFlag = 1
lblStatus.Caption = "Click Calibrate to continue...."
GoSub GetOut
visual_inspect:
'VisualInspectFlag = 0
'reset buttons
cmdCalibrate.Enabled = False
cmdClose.Enabled = False

'Weigh the displacer.
lblStatus.Caption = "Weighing the displacer..."
ciucum "ZMF", echo, status

'Wait for MF to complete
```

HNF-1569
Revision 1

```
Do
GoSub check_abort
GetALevel
Delay (1)
ciucom "Z2B", echo, status
decode "2B", echo, 32, answer
If InStr(answer, "FR") Then
    Exit Do
End If
Loop

ciucom "ZWQ", echo, status
decode "WQ", echo, 64, answer

ASFoundWQ = answer
txtLog.Text = txtLog.Text + "As found WQ= " + ASFoundWQ + CR

lblStatus.Caption = "Displacer weight(WQ)= " + ASFoundWQ

'Check results against requirements.

'Requirement: Max 280 grams
If CDBl(ASFoundWQ) > 280 Then
    Msg = "Displacer weight is greater than 280 grams." + CR
    Msg = Msg + "Please initiate flush sequence and repeat" + CR
    Msg = Msg + "this calibration."
    MsgBox Msg, 0 + 16, "Displacer too heavy!"
    txtLog.Text = txtLog.Text + "Displacer too heavy..."
    GoSub GetOut
End If
GoSub check_abort

'If necessary, change DW, S1, S2, S3, RM
'get DW
ciucom "ZDW", echo, status
decode "DW", echo, 32, answer
ASFoundDW = answer
txtLog.Text = txtLog.Text + "As found DW= " + ASFoundDW + CR
If CDBl(ASFoundDW) <> CDBl(ASFoundWQ) Then
    Msg = "As Found DW= " + ASFoundDW + CR
    Msg = Msg + "As Found WQ= " + ASFoundWQ + CR
    Msg = Msg + "Okay to change?"
    Reply = MsgBox(Msg, 4 + 32, "Make DW and WQ Match?")
    If Reply = 6 Then
        lblStatus.Caption = "Changing DW to match WQ..."
        ciucom "ZW2=ENRAF2", echo, status
        GoSub check_echo
        ciucom "ZDW=" + ASFoundWQ, echo, status
        GoSub check_errors
        ciucom "ZEX", echo, status
        lblStatus.Caption = "Waiting for gauge to reinitialize..."
        'Loop until TG is in 2B
        Do
            GoSub check_abort
            Delay (1)
            GetALevel
            If InStr(L2, "TG") <> 0 Then
                Exit Do
            End If
        Loop
        'If performing TG, loop until TG not in 2B
        If InStr(L2, "TG") <> 0 Then
            Do
                Delay (1)
                GetALevel
                If InStr(L2, "TG") = 0 Then
                    Exit Do
                End If
            Loop
        End If
        txtLog.Text = txtLog.Text + "DW changed to match WQ...." + CR
    End If
End If
```

HNF-1569
Revision 1

```

Else
    txtLog.Text = txtLog.Text + "DW Not changed to match WQ..." + CR
End If
End If

'Lower to top of ball valve.
lblStatus.Caption = "Now performing level calibration..."
ciucom "ZUN", echo, status
Do
GoSub check_abort
Delay (1)
GetALevel
TankLevel = CL

If InStr(frmMain.txtLevelDisplay.Text, "INN") <> 0 Then
    Exit Do
End If
Loop

'if level reading is off, Change and accept, exit, Record
GetALevel
AsFoundRL = Str(CL)
txtLog.Text = txtLog.Text + "As Found RL= " + AsFoundRL + CR
If (Cdbl(txtCRL.Text) + 0.1) < Cdbl(answer) Or (Cdbl(txtCRL.Text) - 0.1) > answer Then
    Msg = "The Correct RL and As Found RL values" + CR
    Msg = Msg + "do not match to within 0.1 inch. Do" + CR
    Msg = Msg + "you want to accept the current displacer" + CR
    Msg = Msg + "position as the Reference Level?" + CR
    Msg = Msg + "Correct RL= " + txtCRL.Text + CR
    Msg = Msg + "As Found RL= " + Str(answer)
    Reply = MsgBox(Msg, 4 + 32, "RL Mismatch!")
    'if yes, do it here, else get on with life
    If Reply <> 6 Then
        Msg = "Selecting NO at this point may be a procedure violation!" + CR
        Msg = Msg + "Are you sure you intended to select 'YES'?"
        Reply = MsgBox(Msg, 4 + 48, "Violation!")
        If Reply = 6 Then
            lblStatus.Caption = "Aborting Level Calibration..."
            txtLog.Text = txtLog.Text + "Aborting..." + CR
            Exit Sub
        End If
    End If
End If
ciucom "ZW2=ENRAF2", echo, status
If InStr(echo, "&") <> 0 Then
    ciucom "ZRL=+" + Format$(Trim(txtCRL.Text), "00000.00"), echo, status
    Debug.Print "Entering RL= "; "ZRL=+" + Format$(Trim(txtCRL.Text), "00000.00")
    GoSub check_errors
    ciucom "ZRL", echo, status
    decode "RL", echo, 32, answer
    Debug.Print "New Gauge RL= "; answer
    If Trim(Cdbl(answer)) <> Trim(Cdbl(txtCRL.Text)) Then
        Do
            Reply = MsgBox("RL entry check failed. Retry entry?", 3 + 48, "Problem...")
            If Reply = 2 Then GoSub GetOut
            ciucom "ZRL=+" + Format$(Trim(txtCRL.Text), "00000.00"), echo, status
            Debug.Print "Entering RL= "; "ZRL=+" + Format$(Trim(txtCRL.Text), "00000.00")
            GoSub check_errors
            ciucom "ZRL", echo, status
            decode "RL", echo, 32, answer
            If Trim(Cdbl(answer)) = Trim(Cdbl(txtCRL.Text)) Then
                Exit Do
            End If
        Loop Until Reply <> 6
    End If
    ciucom "ZAR", echo, status
    'wait for EX to finish
    lblStatus.Caption = "Waiting for gauge to reinitialize..."
    Do
    GoSub check_abort
    Delay (1)
    GetALevel

```


HNF-1569
Revision 1

```

    TankLevel = CL
    If InStr(L2, "TG") <> 0 Then
        Exit Do
    End If
    Loop
    Do
        GoSub check_abort
        Delay (1)
        GetALevel
        TankLevel = CL
        If InStr(L2, "TG") = 0 Then
            Exit Do
        End If
    Loop
    lblStatus.Caption = ""
    txtLog.Text = txtLog.Text + "New RL Accepted by gauge..."
End If

End If
    ciucom "ZRL", echo, status
    decode "RL", echo, 32, answer
    AsLeftRL = answer
    txtLog.Text = txtLog.Text + "As left RL= " + AsLeftRL + CR

Reply = MsgBox("Okay to check TT?", 4 + 32, "Check TT?")
If Reply = 6 Then
    lblStatus.Caption = "Now checking TT..."

    ciucom "ZCA", echo, status
    Dim oldlevel As Double
    TankLevel = 0
    Do
        GoSub check_abort
        GoSub check_errors
        GetALevel
        oldlevel = CL
        Delay (1)
        GetALevel
        TankLevel = CL
        ' when both readings are equal, displacer is at TT (stopped
        ' moving)
        If oldlevel = TankLevel Then Exit Do
    Loop

    'get programmed TT
    ciucom "ZTT", echo, status
    decode "TT", echo, 32, answer
    ProgrammedTT = answer
    txtLog.Text = txtLog.Text + "Programmed TT= " + answer + CR

    GoSub check_abort

    'get as found TT
    GetALevel
    AsFoundTT = Str(CL)
    txtLog.Text = txtLog.Text + "As found TT= " + answer + CR

    GoSub check_abort

    'check them
    Debug.Print "AsfoundTT= "; AsFoundTT
    Debug.Print "ProgrammedTT= "; ProgrammedTT
    If CDBl(AsFoundTT) <> CDBl(ProgrammedTT) Then
        'change programmed TT
        lblStatus.Caption = "Changing TT's..."
        ciucom "ZW2=ENRAF2", echo, status
        decode "W2=ENRAF2", echo, 4, answer
        If InStr(answer, "&") <> 0 Then
            ciucom "ZTT=" + Format$(AsFoundTT, "00000.00"), echo, status
            GoSub check_errors
        End If
    End If
End If

```

HNF-1569
Revision 1

```
AsLeftTT = AsFoundTT
txtLog.Text = txtLog.Text + "As left TT= " + AsLeftTT + CR
End If 'okay to check TT
'Prompt to open ball valve.
MsgBox "Please open ball valve, then click OK..."

'Place into service.
lblStatus.Caption = "Returning gauge to service..."
ciucom "ZUN", echo, status

'wait for displacer to get to surface...
Do
GetALevel
GoSub check_abort
GoSub check_errors
Delay (scanRate / 1000)
GoSub check_echo
If InStr(frmMain.txtLevelDisplay.Text, "INN") <> 0 Then
Exit Do
End If
Loop

AsLeftLevel = frmMain.txtLevelDisplay.Text
txtLog.Text = txtLog.Text + "As left Level= " + AsLeftLevel + CR

txtLog.Text = txtLog.Text + "Level Calibration complete..." + CR
lblStatus.Caption = "Complete. Have a nice day!"
txtLog.Text = txtLog.Text + Time$ + " " + Date$ + CR
VisualInspectFlag = 0
GoSub GetOut

warn:
Msg = "The displacer is now less than 50 inches below" + CR
Msg = Msg + "the ball valve. Ensure radiological protection" + CR
Msg = Msg + "is present before continuing."
MsgBox Msg, 0 + 16, "Radiological Hold Point!"
Return

check_echo:
If Len(echo) <= 3 And i = numretries Then
AbortCal = 1
lblStatus.Caption = "Could not get a Reponse from CIU/PET...Aborting!"
GoSub check_abort
Else
Return
End If

check_abort:
If AbortCal = 1 Then
txtLog.Text = txtLog.Text + "Calibration" + CR + "ABORTED!" + CR
GoSub GetOut
Else
Return
End If

check_errors:
ciucom "ZES", echo, status
decode "ES", echo, 64, answer
answer = Trim(answer)
If answer <> "0000" Then
MsgBox "Gauge SPU Error " + answer + " encountered..." + CR + "Exiting...", vbOKOnly +
vbExclamation
GoSub GetOut
End If
ciucom "ZEP", echo, status
decode "EP", echo, 64, answer
answer = Trim(answer)
If InStr(echo, "!0") <> 0 Then
Dim m$
```

HNF-1569
Revision 1

```
        m = GetErrorMessage(Mid(echo, InStr(echo, "!0"), 4))
        m = m + CR
        m = m + "Aborting..."
        MsgBox m, 1 + 16, "Gauge Error Code!"
        GoSub GetOut
    End If
    Return

    cmdCalibrate.Enabled = True
    cmdClose.Enabled = True

Errhandler:
    'Debug.Print "Error in GetDensity, getting out"
    MsgBox "Error encountered during level calibration..."
    GoSub GetOut

GetOut:
    cmdCalibrate.Enabled = True
    cmdClose.Enabled = True
    If txtLog.Text <> "" Then
        mnuFileSaveLog_Click
    End If
End Sub

Private Sub cmdClose_Click()
    frmLevelCal.Hide
End Sub

Private Sub cmdFR_Click()
    Dim echo$
    If ja = 0 Then
        checkGageAddress ta, ja
    End If
    ciucom "ZFR", echo, status
End Sub

Private Sub cmdUN_Click()
    Dim echo$
    If ja = 0 Then
        checkGageAddress ta, ja
    End If
    ciucom "ZUN", echo, status
End Sub

Private Sub mnuFileAbort_Click()
    AbortCal = 1
End Sub
Private Sub mnuFileClose_Click()
    frmLevelCal.Hide
End Sub

Private Sub mnuFilePrintLog_Click()
    Printer.Print txtLog.Text
    Printer.EndDoc
End Sub

Private Sub mnuFileSaveLog_Click()
    Dim F
    F = FreeFile
    Open App.Path & "\LOG\C" & Right(Date$, 7) & ".LOG" For Output As #F
    Print #F, txtLog.Text
    Close #F
End Sub

Private Sub Timer1_Timer()
    timeFlag = 1
End Sub

Private Sub txtLog_Change()
    txtLog.SelStart = Len(txtLog.Text)
End Sub
```

```

=====
End FRMLEVEL.FRM
=====
Begin FRMRL.FRM

VERSION 5.00
Begin VB.Form frmRL
    BorderStyle      = 3   'Fixed Dialog
    Caption          = "RL Resolution"
    ClientHeight     = 3435
    ClientLeft       = 4860
    ClientTop        = 6030
    ClientWidth      = 3270
    LinkTopic        = "Form1"
    MaxButton        = 0   'False
    MinButton        = 0   'False
    PaletteMode      = 1   'UseZOrder
    ScaleHeight      = 3435
    ScaleWidth       = 3270
    ShowInTaskbar    = 0   'False
    StartUpPosition  = 2   'CenterScreen
    Begin VB.CommandButton cmdAbort
        Caption       = "&Abort"
        Height        = 372
        Left          = 2280
        TabIndex      = 6
        Top           = 3000
        Width         = 852
    End
    Begin VB.CommandButton cmdOkay
        Caption       = "&OK"
        Height        = 372
        Left          = 120
        TabIndex      = 5
        Top           = 3000
        Width         = 852
    End
    Begin VB.Frame Frame1
        Height        = 2052
        Left          = 120
        TabIndex      = 0
        Top           = 840
        Width         = 3012
        Begin VB.OptionButton optNew
            Caption     = "Enter a different RL Value"
            Height      = 372
            Left        = 360
            TabIndex    = 3
            Top         = 1560
            Width       = 2292
        End
        Begin VB.OptionButton optProg
            Caption     = "Change Your RL Value to the Programmed RL Value currently in the"
            Height      = 612
            Left        = 1560
            TabIndex    = 2
            Top         = 840
            Width       = 2532
        End
        Begin VB.OptionButton optYours
            Caption     = "Accept Your RL Value and update gauge Programmed RL"
            Height      = 612
            Left        = 360
            TabIndex    = 1
            Top         = 240
            Width       = 2532
        End
    End
    Begin VB.Label lblRL
        Height        = 252
        Left          = 1200
        TabIndex      = 7
    End
End

```

HNF-1569
Revision 1

```

Top          = 3000
Visible      = 0 'False
Width        = 852
End
Begin VB.Label Label1
Caption      = "You have entered an RL that does not match the Programed RL. Please
resolve by selecting one of the following:"
Height      = 612
Left         = 120
TabIndex    = 4
Top          = 120
Width        = 3012
End
End
Attribute VB_Name = "frmRL"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
' =====
' This form is used with the frmLevelCal form. It is simply
' used to specify options. All options could have been placed
' on the level cal form, but we're trying to keep the forms as
' small as possible so they all fit on a 640x480 screen.
' =====
Option Explicit

Private Sub cmdAbort_Click()
lblRL.Caption = "ABORT"
frmRL.Hide
End Sub

Private Sub cmdOkay_Click()
frmRL.Hide
End Sub

Private Sub Form_Load()

End Sub

End FRMRL.FRM
=====
Begin frmWeigh.frm
VERSION 5.00
Object = "{6B7E6392-850A-101B-AFC0-4210102A8DA7}#1.2#0"; "COMCTL32.OCX"
Begin VB.Form frmWeigh
Caption      = "Displacer Weight "
ClientHeight = 2544
ClientLeft  = 48
ClientTop   = 312
ClientWidth = 4524
LinkTopic   = "Form1"
ScaleHeight = 2544
ScaleWidth  = 4524
StartupPosition = 3 'Windows Default
Begin VB.CommandButton cmdClose
Caption      = "&Close"
Height       = 252
Left         = 3480
TabIndex    = 12
Top          = 1200
Width        = 972
End
Begin VB.CommandButton cmdAbort
Caption      = "&Abort"
Enabled      = 0 'False
Height       = 252
Left         = 1800
TabIndex    = 11
Top          = 1200
Width        = 1452
End
End

```

HNF-1569
Revision 1

```
Begin VB.Frame Frame1
Caption      = "Options"
Height      = 975
Left        = 120
TabIndex    = 6
Top         = 120
Width       = 4332
Begin VB.CheckBox chkSETDW
Caption      = "Set DW = WQ"
Height      = 192
Left        = 240
TabIndex    = 10
Top         = 600
Width       = 1572
End
Begin VB.TextBox txtDelta
Enabled      = 0 'False
Height      = 288
Left        = 3240
TabIndex    = 8
Text        = "15"
Top         = 360
Width       = 372
End
Begin VB.CheckBox chkSET
Caption      = "Set S1, S3 and RM"
Height      = 435
Left        = 240
TabIndex    = 7
Top         = 240
Width       = 1815
End
Begin VB.Label Label4
Caption      = "grams"
Height      = 252
Left        = 3720
TabIndex    = 13
Top         = 480
Width       = 492
End
Begin VB.Label Label3
Alignment    = 1 'Right Justify
Caption      = "Setpoint Delta:"
Height      = 252
Left        = 2040
TabIndex    = 9
Top         = 360
Width       = 1092
End
End
Begin VB.TextBox txtWQ
Height      = 288
Left        = 1920
TabIndex    = 4
Top         = 1920
Width       = 2412
End
Begin VB.TextBox txtDW
Height      = 288
Left        = 1920
TabIndex    = 2
Top         = 1560
Width       = 2412
End
Begin ComctlLib.StatusBar sBar
Align       = 2 'Align Bottom
Height      = 240
Left        = 0
TabIndex    = 1
Top         = 2304
Width       = 4524
End
```

HNF-1569
Revision 1

```
_ExtentX      = 7980
_ExtentY      = 423
Style         = 1
SimpleText    = ""
_Version      = 327682
BeginProperty Panels {0713E89E-850A-101B-AFC0-4210102A8DA7}
  NumPanels    = 1
  BeginProperty Panel1 {0713E89F-850A-101B-AFC0-4210102A8DA7}
    Object.Tag  = ""
  EndProperty
EndProperty
End
Begin VB.CommandButton cmdWeigh
  Caption      = "Weigh Displacer"
  Height       = 252
  Left         = 120
  TabIndex     = 0
  Top          = 1200
  Width        = 1452
End
Begin VB.Label Label2
  Alignment     = 1 'Right Justify
  Caption       = "Measured Weight WQ:"
  Height        = 252
  Left          = 120
  TabIndex     = 5
  Top           = 1920
  Width         = 1692
End
Begin VB.Label Label1
  Alignment     = 1 'Right Justify
  Caption       = "Programmed DW:"
  Height        = 252
  Left          = 480
  TabIndex     = 3
  Top           = 1560
  Width         = 1332
End
End
Attribute VB_Name = "frmWeigh"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
Dim Weighing As Boolean

Option Explicit

Private Sub chkSET_Click()
If chkSET.Value = Unchecked Then
  txtDelta.Enabled = True
Else
  txtDelta.Enabled = False
End If
End Sub

Private Sub cmdAbort_Click()
Abort = 1
End Sub

Private Sub cmdClose_Click()
If Not Weighing Then
  Unload Me
End If
End Sub

Private Sub cmdWeigh_Click()
' *****
' This function causes the gauge to weigh the displacer

```

HNF-1569
Revision 1

```
' The ACTIVITY.LOG file is appended with the results.
' *****
Dim echo$, answer$, dWQ$
Dim nResume%, R%, m$
Abort = 0

If chkSETDW.Value = checked Or chkSET.Value = checked Then
    m = "Proceeding from this point will cause values in the gauge" & vbCrLf
    m = m & "to be modified. Do you want to continue? To continue" & vbCrLf
    m = m & "without modifying gauge parameters, click CANCEL"
    R = MsgBox(m, vbExclamation + vbYesNoCancel, "Warning!")
    If R = vbNo Then
        Abort = 1
        GoSub CheckAbort
    End If
    If R = vbCancel Then
        chkSETDW.Value = Unchecked
        chkSET.Value = Unchecked
    End If
End If

cmdAbort.Enabled = True
cmdClose.Enabled = False
sBar.SimpleText = "...Stand by..."

' IF scanning, stop
If gettingScans = 1 Then
    frmMain.mnuDisconnect_Click
' ELSE Check for connection and connect
ElseIf gaugeChecked = 0 Then
    frmMain.txtDisplay2.Text = "Initializing..."
    checkGageAddress ta, ja
    frmMain.txtDisplay2.Text = ""
End If
If ja = 0 Then Exit Sub
Weighing = True
sBar.SimpleText = "Getting gauge parameters..."
GetTankDimensions
ciucom "ZDW", echo, status
decode "DW", echo, 32, answer
txtDW.Text = answer

' Remember current level/status
GetALevel s1B, s2B
GoSub CheckAbort

Dim L$
' Make sure displacer is low enough
If CL >= (TT - 7) Then
    sBar.SimpleText = "Displacer is too high...Lowering!"
    Delay (1)
    GetALevel
    ciucom "ZUN", echo, status
    For L = 1 To 4
        GetALevel
        DoEvents
        GoSub CheckAbort
    Next L
    ciucom "ZFR", echo, status
    sBar.SimpleText = ""
    GoSub CheckAbort
End If

' Raise it a little
sBar.SimpleText = "Raising displacer..."
ciucom "ZUN", echo, status
ciucom "ZLT", echo, status
'Wait 3 seconds
For L = 1 To 3
    GetALevel
    DoEvents
```



```

    GoSub CheckAbort
Next L

'Stop the displacer
ciucom "ZFR", echo, status
GetALevel
GoSub CheckAbort

sBar.SimpleText = "Weighing displacer..."
'Weigh the displacer.
ciucom "ZMF", echo, status

'Wait for MF to complete
Do
    GetALevel s1B, s2B
    If InStr(s2B, "FR") <> 0 Then
        Exit Do
    End If
    DoEvents
    GoSub CheckAbort
Loop

sBar.SimpleText = "Weight obtained..."
ciucom "ZWQ", echo, status
decode "WQ", echo, 64, answer
txtWQ.Text = answer
dWQ = answer

'place result into boxes where they can use them
frmMain.txtCustomCommand.Text = dWQ
frmService.txtItem.Text = dWQ
GoSub CheckAbort

Dim ddSET As Double
Dim Error As Boolean
If chkSETDW.Value = checked Or chkSET.Value = checked Then
sBar.SimpleText = "Entering protection level 2..."
L = 0
EnterPassword (L)
Debug.Print "EnterPassword= "; L
If L = 1 Then
    Abort = 1
    GoSub CheckAbort
End If
GoSub CheckAbort
End If

If chkSETDW.Value = checked Then
sBar.SimpleText = "Setting DW..."
ddSET = CDb1(answer)
'ciucom "ZDW=" & FormatTensionString(ddSET, Error), echo, status
Debug.Print "FTS= "; FormatTensionString(ddSET, Error)
Debug.Print "ZDW String= "; "ZDW=" & txtWQ.Text
ciucom "ZDW=" & txtWQ.Text, echo, status
If Error Then
    ciucom "ZEX", echo, status
    MsgBox "Error entering values. Aborting!"
    For L = 1 To 5
        GetALevel
        DoEvents
    Next L
    Abort = 1
    GoSub CheckAbort
End If
End If

If chkSET.Value = checked Then
If txtDelta.Text = "" Then
    txtDelta.Text = "15"
End If
sBar.SimpleText = "Setting S1, S3 and RM..."

```

```

ddSET = CDb1(dwQ)
ddSET = ddSET - CDb1(txtDelta.Text)
Debug.Print "ddSET - Delta= "; ddSET
' Debug step REMOVE!
''ddSET = ddSET + 0.000444444
ciucum "ZS1=" & FormatTensionString2(ddSET), echo, status
'ciucum "ZS1=" & FormatTensionString(ddSET, Error), echo, status
Debug.Print "FTS2 ciucum ZS1: "; "ZS1=" & FormatTensionString2(ddSET)
If Error Then
    ciucum "ZEX", echo, status
    MsgBox "Error entering values. Aborting!"
    For L = 1 To 5
        sBar.SimpleText = "Gauge initializing..."
        GetALevel
        DoEvents
        GoSub CheckAbort
    Next L
    sBar.SimpleText = ""
    Abort = 1
    GoSub CheckAbort
End If
GoSub CheckAbort
'ciucum "ZS3=" & FormatTensionString(ddSET, Error), echo, status
ciucum "ZS3=" & FormatTensionString2(ddSET), echo, status
If Error Then
    ciucum "ZEX", echo, status
    MsgBox "Error entering values. Aborting!"
    For L = 1 To 5
        sBar.SimpleText = "Gauge initializing..."
        GetALevel
        DoEvents
        GoSub CheckAbort
    Next L
    sBar.SimpleText = ""
    Abort = 1
    GoSub CheckAbort
End If
GoSub CheckAbort
'ciucum "ZRM=" & FormatTensionString(ddSET, Error), echo, status
ciucum "ZRM=" & FormatTensionString2(ddSET), echo, status
If Error Then
    ciucum "ZEX", echo, status
    MsgBox "Error entering values. Aborting!"
    For L = 1 To 5
        sBar.SimpleText = "Gauge initializing..."
        GetALevel
        DoEvents
        GoSub CheckAbort
    Next L
    sBar.SimpleText = ""
    Abort = 1
    GoSub CheckAbort
End If
GoSub CheckAbort
ciucum "ZEX", echo, status
'wait until gauge reinitializes
sBar.SimpleText = "Waiting for gauge to reinitialize..."
'Delay (10) ' allow TG to start
Do
    GoSub CheckAbort
    GetALevel s1B, s2B
    Delay (scanRate / 1000)
    If InStr(s2B, "TG") <> 0 Then
        Exit Do
    End If
    DoEvents
Loop

'wait for test gauge to complete
sBar.SimpleText = "Waiting for Test Gauge to Complete..."

```

```

Do
    GoSub CheckAbort
    GetALevel s1B, s2B
    Delay (scanRate / 1000)
    If InStr(s2B, "TG") = 0 Then
        Exit Do
    End If
    DoEvents
Loop
ciucom "ZFR", echo, status
ciucom "ZDW", echo, status
decode "DW", echo, 32, answer
txtDW.Text = answer

sBar.SimpleText = "Finished..."
End If

Aborting:
If Abort = 1 Then
    sBar.SimpleText = "Aborted!"
End If

Dim PN$
ciucom "ZPN", echo, status
decode "PN", echo, 32, answer
PN = answer

If nResume = 1 Then
    frmMain.mnuConnectCIU_Click
End If
' Record in Activity log
Dim F
F = FreeFile
Open App.Path & "\DENSITY\ACTIVITY.LOG" For Append As F
    Print #F, "=====
    Print #F, "          WEIGH DISPLACER RESULTS"
    Print #F, "          Gauge S/N: " & PN
    If TestMode = 1 Then
        Print #F, "          DEMO MODE RUN, SIMULATED DATA"
    End If
    Print #F, "Date", "Time", "Weight"
    Print #F, "=====
    Print #F, Date$, Time$, Format(dWQ, "###.##")
    If Abort = 1 Then
        Print #F, "          ABORTED!"
        Abort = 0
    End If
    Print #F, "=====
Close #F
Weighing = False
cmdAbort.Enabled = False
cmdClose.Enabled = True
Exit Sub

CheckAbort:
    If Abort = 1 Then
        GoTo Aborting
    End If
    Return

End Sub

Private Sub Form_Load()
Me.Left = frmMain.Left
Me.Top = frmMain.Top + frmMain.Height
End Sub

                End frmWeigh.frm
=====
                Begin frmDemo.frm

VERSION 5.00
Begin VB.Form frmDemo

```

```
Caption      = "Dummy Gauge"
ClientHeight = 3945
ClientLeft   = 60
ClientTop    = 345
ClientWidth  = 3090
LinkTopic    = "Form1"
ScaleHeight  = 3945
ScaleWidth   = 3090
StartPosition = 3 'Windows Default
Begin VB.CommandButton cmdLevel
    Caption      = "Change"
    Height       = 375
    Left         = 2040
    TabIndex     = 7
    Top          = 1200
    Width        = 855
End
Begin VB.TextBox txtLevel
    Height       = 285
    Left         = 2040
    TabIndex     = 6
    Top          = 840
    Width        = 855
End
Begin VB.CommandButton cmd1BtoExcl
    Caption      = "1B to !!!"
    Height       = 375
    Left         = 360
    TabIndex     = 5
    Top          = 3360
    Width        = 1335
End
Begin VB.CommandButton cmd1BtoINN
    Caption      = "1B to INN"
    Height       = 375
    Left         = 360
    TabIndex     = 4
    Top          = 2760
    Width        = 1335
End
Begin VB.CommandButton cmd2BtoI2
    Caption      = "2B to I2"
    Height       = 375
    Left         = 360
    TabIndex     = 3
    Top          = 2160
    Width        = 1335
End
Begin VB.CommandButton cmd2BtoI1
    Caption      = "2B to I1"
    Height       = 375
    Left         = 360
    TabIndex     = 2
    Top          = 1560
    Width        = 1335
End
Begin VB.CommandButton cmd2BtoMF
    Caption      = "2B to MF"
    Height       = 375
    Left         = 360
    TabIndex     = 1
    Top          = 960
    Width        = 1335
End
Begin VB.CommandButton cmd2BtoFR
    Caption      = "2B to FR"
    Height       = 375
    Left         = 360
    TabIndex     = 0
    Top          = 360
    Width        = 1335
```

```
End
Begin VB.Label Label1
    Caption       = "Change 1B Level"
    Height        = 375
    Left          = 1920
    TabIndex      = 8
    Top           = 360
    Width         = 975
End
End
Attribute VB_Name = "frmDemo"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
Option Explicit

Sub cmd1BtoExcl_Click()
On Error Resume Next
Dim i%
For i = 1 To TestItemCount
    If InStr(TestItems(i), "1B=") Then
        Debug.Print "Found 1B= "; TestItems(i)
        Debug.Print "Length 1B= "; Len(TestItems(i))
        Debug.Print "Last Char in 1B= "; Mid(TestItems(i), Len(TestItems(i)))
        Mid(TestItems(i), Len(TestItems(i)) - 5) = "!! !!!"
        Debug.Print "Changed 1B= "; TestItems(i)
    End If
Next i
End Sub

Sub cmd1BtoINN_Click()
On Error Resume Next
Dim i%
For i = 1 To TestItemCount
    If InStr(TestItems(i), "1B=") Then
        Debug.Print "Found 1B= "; TestItems(i)
        Debug.Print "Length 1B= "; Len(TestItems(i))
        Debug.Print "Last Char in 1B= "; Mid(TestItems(i), Len(TestItems(i)))
        Mid(TestItems(i), Len(TestItems(i)) - 5) = "in INN"
        Debug.Print "Changed 1B= "; TestItems(i)
    End If
Next i
End Sub

Sub cmd2BtoFR_Click()
On Error Resume Next
Dim i%
For i = 1 To TestItemCount
    If InStr(TestItems(i), "2B=") Then
        Debug.Print "Found 2B= "; TestItems(i)
        Debug.Print "Length 2B= "; Len(TestItems(i))
        Debug.Print "Last Char in 2B= "; Mid(TestItems(i), Len(TestItems(i)))
        Mid(TestItems(i), Len(TestItems(i)) - 1) = "FR"
        Debug.Print "Changed 2B= "; TestItems(i)
    End If
Next i
End Sub

Sub cmd2BtoI1_Click()
On Error Resume Next
Dim i%
For i = 1 To TestItemCount
    If InStr(TestItems(i), "2B=") Then
        Debug.Print "Found 2B= "; TestItems(i)
        Debug.Print "Length 2B= "; Len(TestItems(i))
        Debug.Print "Last Char in 2B= "; Mid(TestItems(i), Len(TestItems(i)))
        Mid(TestItems(i), Len(TestItems(i)) - 1) = "I1"
        Debug.Print "Changed 2B= "; TestItems(i)
    End If
Next i
```

End Sub

```
Sub cmd2BtoI2_Click()  
On Error Resume Next  
Dim i%  
For i = 1 To TestItemCount  
    If InStr(TestItems(i), "2B=") Then  
        Debug.Print "Found 2B= "; TestItems(i)  
        Debug.Print "Length 2B= "; Len(TestItems(i))  
        Debug.Print "Last Char in 2B= "; Mid(TestItems(i), Len(TestItems(i)))  
        Mid(TestItems(i), Len(TestItems(i)) - 1) = "I2"  
        Debug.Print "Changed 2B= "; TestItems(i)  
    End If  
Next i  
End Sub
```

```
Sub cmd2BtoMF_Click()  
On Error Resume Next  
Dim i%  
For i = 1 To TestItemCount  
    If InStr(TestItems(i), "2B=") Then  
        Debug.Print "Found 2B= "; TestItems(i)  
        Debug.Print "Length 2B= "; Len(TestItems(i))  
        Debug.Print "Last Char in 2B= "; Mid(TestItems(i), Len(TestItems(i)))  
        Mid(TestItems(i), Len(TestItems(i)) - 1) = "MF"  
        Debug.Print "Changed 2B= "; TestItems(i)  
    End If  
Next i  
End Sub
```

```
Sub cmdLevel_Click()  
On Error Resume Next  
Dim i%  
For i = 1 To TestItemCount  
    If InStr(TestItems(i), "1B=") Then  
        Debug.Print "Found 1B= "; TestItems(i)  
        Debug.Print "Length 1B= "; Len(TestItems(i))  
        Debug.Print "Last Char in 1B= "; Mid(TestItems(i), Len(TestItems(i)))  
        Mid(TestItems(i), 5, 8) = Format$(txtLevel.Text, "00000.00")  
        Debug.Print "Changed 1B= "; TestItems(i)  
    End If  
Next i  
End Sub
```

```
Private Sub Form_Load()  
Me.Left = frmMain.Left + frmMain.Width  
Me.Top = frmMain.Top  
  
End Sub
```

End frmDemo.frm

=====