```
[Identification]
    OptionType = NetTransport
[Options]
    DLC
[FileConstants]
UtilityInf      = "UTILITY.INF"
subroutineinf   = "SUBROUTN.INF"
SoftwareType    = "transport"
Exit_Code       = 0
NetEventDLL     = "%SystemRoot%\System32\netevent.dll"
IoLogMsgDLL     = "%SystemRoot%\System32\IoLogMsg.dll"
Manufacturer    = "Microsoft"
ProductMajorVersion     = "4"
ProductMinorVersion     = "0"
ProductVersion  = $(ProductMajorVersion)"."$(ProductMinorVersion)
ProductSoftwareName     = "DLC"
ProductSoftwareImagePath = "\SystemRoot\System32\drivers\dlc.sys"
NetRuleSoftwareType     = "dlc dlcDriver dlcDriver"
NetRuleSoftwareUse      = $(SoftwareType)" yes yes"
NetRuleSoftwareBindForm = """Dlc"" yes yes container"
NetRuleSoftwareClass    = {"dlcDriver basic"}
NetRuleSoftwareBindable = {"dlcDriver ndisDriver non non 100"}
ProductOpSupport     = 132
ProductKeyName  = $(!NTN_SoftwareBase)"\"$(Manufacturer)"\"$
(ProductSoftwareName)"\CurrentVersion"
ParamKeyName    = $(!NTN_ServiceBase)"\"$(ProductHardwareName)"\Parameters"
[GeneralConstants]
from      = ""
to        = ""
ExitCodeOk     = 0
ExitCodeCancel = 1
ExitCodeFatal  = 2
KeyNull         = ""
MAXIMUM_ALLOWED   = 33554432
RegistryErrorIndex = NO_ERROR
KeyProduct      = ""
KeyParameters   = ""
TRUE            = 1
FALSE           = 0
NoTitle            = 0
ExitState   = "Active"
OldVersionExisted = $(FALSE)
DriverPath      = $(!STF_NTPATH)\drivers
[date]
    Now = {} ? $(!LIBHANDLE) GetSystemDate
[Identify]
    read-syms Identification
    set Status     = STATUS_SUCCESSFUL
    set Identifier = $(OptionType)
    set Media      = #("Source Media Descriptions", 1, 1)
    Return $(Status) $(Identifier) $(Media)
[ReturnOptions]
    set Status        = STATUS_FAILED
    set OptionList     = {}
    set OptionTextList = {}
    set LanguageList = ^(LanguagesSupported, 1)
    Ifcontains(i) $($0) in $(LanguageList)
        goto returnoptions
    else
        set Status = STATUS_NOLANGUAGE
        goto finish_ReturnOptions
    endif
returnoptions = +
    set OptionList     = ^(Options, 1)
```

```
    set OptionTextList = ^(OptionsText$($0), 1)
    set Status         = STATUS_SUCCESSFUL
finish_ReturnOptions = +
    Return $(Status) $(OptionList) $(OptionTextList)
[InstallOption]
    set Option   = $($1)
    set SrcDir   = $($2)
    set AddCopy  = $($3)
    set DoCopy   = $($4)
    set DoConfig = $($5)
    set LanguageList = ^(LanguagesSupported, 1)
    Ifcontains(i) $($0) NOT-IN $(LanguageList)
        Return STATUS_NOLANGUAGE
    endif
    Debug-Output "OEMNXPDL.INF: STF_CWDIR is: "$(!STF_CWDIR)
    Debug-Output "OEMNXPDL.INF: STF_LANGUAGE is: "$(!STF_LANGUAGE)
    set-subst LF = "\n"
    read-syms GeneralConstants
    read-syms FileConstants
    read-syms DialogConstants$(!STF_LANGUAGE)
    ifstr(i) $(!NTN_Origination) == "NCPA"
        set Continue = $(OK)
    endif
    read-syms FileConstants$(!STF_LANGUAGE)
    detect date
    set-title  $(FunctionTitle)
    set to   = Begin
    set from = Begin
    set CommonStatus = STATUS_SUCCESSFUL
    EndWait
Begin = +
    Ifstr(i) $(!NTN_InstallMode) == deinstall
        set StartLabel = removeadapter
    else-Ifstr(i) $(!NTN_InstallMode) == Update
        set StartLabel = UpgradeSoftware
    else-Ifstr(i) $(!NTN_InstallMode) == bind
        set StartLabel = bindingadapter
    else-Ifstr(i) $(!NTN_InstallMode) == configure
        Shell $(UtilityInf),RegistryErrorString,CANNOT_CONFIGURE_SOFTWARE
        ifint $($ShellCode) != $(!SHELL_CODE_OK)
            Debug-Output "OEMNXPDL.INF: ShellCode error: cannot get an error
string."
            goto ShellCodeError
        endif
        set Error = $($R0)
        set from = end
        set to = end
        goto nonfatalinfo
    else
        set StartLabel = installadapter
    endif
    set from = $(fatal)
    set to = $(fatal)
    goto $(StartLabel)
installadapter = +
    OpenRegKey $(!REG_H_LOCAL) "" $(ProductKeyName) $(MAXIMUM_ALLOWED)
KeyProduct
    Ifstr $(KeyProduct) != $(KeyNull)
        CloseRegKey $(KeyProduct)
        Shell $(UtilityInf), VerExistedDlg, $(ProductSoftwareTitle),+
            $(ProductVersion)
        ifint $($ShellCode) != $(!SHELL_CODE_OK)
            Debug-Output "ShellCode error: cannot get an error string."
            goto ShellCodeError
```

```
        endif
        goto end
    endif
    CloseRegKey $(KeyProduct)
    goto nextstep
nextstep = +
    StartWait
    ifstr(i) $(!NTN_InstallMode) == "install"
        Ifstr(i) $(DoCopy) == "YES"
           Shell $(UtilityInf), DoAskSource, $(!STF_CWDDIR), $(SrcDir) YES
           Ifint $($ShellCode) != $(!SHELL_CODE_OK)
               Goto ShellCodeError
           Else-Ifstr(i) $($R0) == STATUS_FAILED
               Shell $(UtilityInf) RegistryErrorString "ASK_SOURCE_FAIL"
               ifint $($ShellCode) != $(!SHELL_CODE_OK)
                   goto ShellCodeError
               endif
               set Error = $($R0)
               Goto fatal
           Else-Ifstr(i) $($R0) == STATUS_USERCANCEL
               Goto successful
           Endif
           Set SrcDir = $($R1)
        Endif
        install "Install-Option"
        ifstr(i) $(STF_INSTALL_OUTCOME) != STF_SUCCESS
           Shell $(UtilityInf) RegistryErrorString "UNABLE_COPY_FILE"
           ifint $($ShellCode) != $(!SHELL_CODE_OK)
               goto ShellCodeError
           endif
           set Error = $($R0)
           goto fatal
        endif
    endif
    ifint $(OldVersionExisted) == $(FALSE)
        set OEM_ABANDON_ON = TRUE
        Shell $(UtilityInf), AddSoftwareComponent, $(Manufacturer), +
            $(ProductSoftwareName), +
            $(ProductSoftwareName), +
            $(ProductSoftwareDisplayName), $(STF_CONTEXTINFNAME), +
            $(ProductSoftwareImagePath), "kernelauto", "", {}, "",+
            $(IoLogMsgDLL)
        ifint $($ShellCode) != $(!SHELL_CODE_OK)
            Debug-Output "ShellCode error"
            goto ShellCodeError
        endif
        set RegistryErrorIndex = $($R0)
        Ifstr(i) $(RegistryErrorIndex) != NO_ERROR
            EndWait
            Debug-Output "Registry error: add software components"
            CloseRegKey $($R1)
            CloseRegKey $($R2)
            CloseRegKey $($R3)
            CloseRegKey $($R4)
            CloseRegKey $($R5)
            goto fatalregistry
        endif
        Set SoftProductKey      = $($R1)
        Set SoftNetRuleKey      = $($R2)
        Set SoftServiceKey      = $($R3)
        Set SoftParameterKey    = $($R4)
        Set SoftLinkageKey      = $($R5)
        set NewValueList = {{SoftwareType,$(NoTitle),$(!REG_VT_SZ),$
(SoftwareType)},+
```

```
                            {MajorVersion,$(NoTitle),$(!REG_VT_DWORD),$
(ProductMajorVersion)},+
                            {MinorVersion,$(NoTitle),$(!REG_VT_DWORD),$
(ProductMinorVersion)},+
                            {Title,$(NoTitle),$(!REG_VT_SZ),$
(ProductSoftwareTitle)},+
                            {Description,$(NoTitle),$(!REG_VT_SZ),$
(ProductSoftwareDescription)},+
                            {OperationsSupport,$(NoTitle),$(!REG_VT_DWORD),$
(ProductOpSupport)},+
                            {ServiceName,$(NoTitle),$(!REG_VT_SZ),$
(ProductSoftwareName)},+
                            {InstallDate,$(NoTitle),$(!REG_VT_DWORD),*($
(Now),1)}}
        Shell  $(UtilityInf), AddValueList, $(SoftProductKey), $(NewValueList)
        ifint $($ShellCode) != $(!SHELL_CODE_OK)
            Debug-Output "ShellCode error."
            goto ShellCodeError
        endif
        set RegistryErrorIndex = $($R0)
        Ifstr(i) $(RegistryErrorIndex) != NO_ERROR
            EndWait
            Debug-Output "Registry error: add software components"
            CloseRegKey $(SoftProductKey)
            CloseRegKey $(SoftNetRuleKey)
            CloseRegKey $(SoftServiceKey)
            CloseRegKey $(SoftParameterKey)
            CloseRegKey $(SoftLinkageKey)
            goto fatalregistry
        endif
        set NewValueList = {{type,$(NoTitle),$(!REG_VT_SZ),$
(NetRuleSoftwareType)},+
                            {use,$(NoTitle),$(!REG_VT_SZ),$
(NetRuleSoftwareUse)}, +
                            {bindform,$(NoTitle),$(!REG_VT_SZ),$
(NetRuleSoftwareBindForm)}, +
                            {class,$(NoTitle),$(!REG_VT_MULTI_SZ),$
(NetRuleSoftwareClass)}, +
                            {bindable,$(NoTitle),$(!REG_VT_MULTI_SZ),$
(NetRuleSoftwareBindable)}, +
                            {InfOption,$(NoTitle),$(!REG_VT_SZ),$(Option)}}
        Shell  $(UtilityInf), AddValueList, $(SoftNetRuleKey), $(NewValueList)
        ifint $($ShellCode) != $(!SHELL_CODE_OK)
            Debug-Output "ShellCode error."
            goto ShellCodeError
        endif
        set RegistryErrorIndex = $($R0)
        CloseRegKey $(SoftProductKey)
        CloseRegKey $(SoftNetRuleKey)
        CloseRegKey $(SoftServiceKey)
        CloseRegKey $(SoftParameterKey)
        CloseRegKey $(SoftLinkageKey)
        Ifstr(i) $(RegistryErrorIndex) != NO_ERROR
            EndWait
            Debug-Output "Resgitry error: add value list."
            goto fatalregistry
        endif
    endif
    LibraryProcedure Result, $(!NCPA_HANDLE), CPlAddMonitor
    goto writeparameters
writeparameters = +
    CloseRegKey $(KeyParameters)
    EndWait
    goto successful
```

```
bindingadapter =+
    set Error = "Binding: Sorry, not yet implemented."
    goto fatal
removeadapter = +
    Shell $(UtilityInf), RemoveSoftwareComponent, $(Manufacturer), +
        $(ProductSoftwareName)
    ifint $($ShellCode) != $(!SHELL_CODE_OK)
        Debug-Output "ShellCode error"
        goto ShellCodeError
    endif
    set RegistryErrorIndex = $($R0)
    Ifstr(i) $(RegistryErrorIndex) != NO_ERROR
        goto fatalregistry
    endif
    LibraryProcedure Result, $(!NCPA_HANDLE), CPlDeleteMonitor
    goto end
UpgradeSoftware = +
    OpenRegKey $(!REG_H_LOCAL) "" $(ProductKeyName) $(MAXIMUM_ALLOWED)
KeyProduct
    Ifstr $(KeyProduct) != $(KeyNull)
        install "Install-Update"
        ifstr(i) $(STF_INSTALL_OUTCOME) != STF_SUCCESS
            goto fatal
        endif
        SetRegValue $(KeyProduct) {MajorVersion,$(NoTitle),$(!REG_VT_SZ),$
(ProductMajorVersion)}
        SetRegValue $(KeyProduct) {MinorVersion,$(NoTitle),$(!REG_VT_SZ),$
(ProductMinorVersion)}
        SetRegValue $(KeyProduct) {Description,$(NoTitle),$(!REG_VT_SZ),$
(ProductSoftwareDescription)}
        SetRegValue $(KeyProduct) {OperationsSupport,$(NoTitle),$(!
REG_VT_DWORD),$(ProductOpSupport)}
        CloseRegKey $(KeyProduct)
    else
        goto fatalregistry
    endif
    goto end
successful = +
    goto end
warning = +
    Shell $(subroutineinf) SetupMessage, $(!STF_LANGUAGE), "WARNING", $(Error)
    ifint $($ShellCode) != $(!SHELL_CODE_OK)
        goto ShellCodeError
    endif
    ifstr(i) $($R1) == "OK"
        goto $(to)
    else-ifstr(i) $($R1) == "CANCEL"
        goto $(from)
    else
        goto "end"
    endif
nonfatalinfo = +
    Set CommonStatus = STATUS_USERCANCEL
    Set Severity = STATUS
    goto nonfatalmsg
nonfatal = +
    Set Severity = NONFATAL
    goto nonfatalmsg
nonfatalmsg = +
    ifstr(i) $(Error) == ""
        Set Severity = NONFATAL
        Shell $(UtilityInf) RegistryErrorString "SETUP_FAIL"
        ifint $($ShellCode) != $(!SHELL_CODE_OK)
            goto ShellCodeError
```

```
        endif
        set Error = $($R0)
    endif
    Shell $(subroutineinf) SetupMessage, $(!STF_LANGUAGE), $(Severity), $(Error)
    ifint $($ShellCode) != $(!SHELL_CODE_OK)
        goto ShellCodeError
    endif
    ifstr(i) $($R1) == "OK"
        goto $(from)
    else
        goto "end"
    endif
fatalregistry = +
    Shell $(UtilityInf) RegistryErrorString $(RegistryErrorIndex)
    ifint $($ShellCode) != $(!SHELL_CODE_OK)
        goto ShellCodeError
    endif
    set Error = $($R0)
    goto fatal
fatal = +
    ifstr(i) $(Error) == ""
        Shell $(UtilityInf) RegistryErrorString "SETUP_FAIL"
        ifint $($ShellCode) != $(!SHELL_CODE_OK)
            goto ShellCodeError
        endif
        set Error = $($R0)
    endif
    Shell $(subroutineinf) SetupMessage, $(!STF_LANGUAGE), "FATAL", $(Error)
    ifint $($ShellCode) != $(!SHELL_CODE_OK)
        goto ShellCodeError
    endif
    goto setfailed
ShellCodeError = +
    set DlgType      = "MessageBox"
    set STF_MB_TITLE = $(ShellCodeErrorTitle)
    set STF_MB_TEXT  = $(ShellCodeErrorText)
    set STF_MB_TYPE  = 1
    set STF_MB_ICON  = 3
    set STF_MB_DEF   = 1
    ui start "Error Message"
    goto setfailed
setfailed = +
    set CommonStatus = STATUS_FAILED
    ifstr(i) $(OEM_ABANDON_ON) == TRUE
        set OEM_ABANDON_ON = FALSE
        goto removeadapter
    endif
    goto end
end = +
    goto term
term = +
    Return $(CommonStatus)
[Install-Option]
    set STF_VITAL = ""
    ifstr(i) $(AddCopy) == "YES"
        AddSectionFilesToCopyList Files-$(Option)API $(SrcDir) $(!
STF_WINDOWSSYSPATH)
        AddSectionFilesToCopyList Files-$(Option) $(SrcDir) $(!
STF_WINDOWSSYSPATH)\drivers
    endif
    ifstr(i) $(DoCopy) == "YES"
       set !STF_NCPA_FLUSH_COPYLIST = TRUE
       CopyFilesInCopyList
    else
```

```
       Debug-Output "OEMNSVCU.INF: Copy Single File "$(SrcDir)\hpmon.dll" to "$
(!STF_WINDOWSSYSPATH)\hpmon.dll
       LibraryProcedure STATUS,$(!NCPA_HANDLE), CopySingleFile $(!STF_HWND) $
(SrcDir)\hpmon.dll $(!STF_WINDOWSSYSPATH)\hpmon.dll
    endif
    ifstr(i) $(DoConfig) == "YES"
    endif
    Exit
[Install-Update]
   set STF_VITAL        = ""
   set STF_OVERWRITE    = "VERIFYSOURCEOLDER"
   AddSectionFilesToCopyList Files-$(Option)API $(SrcDir) $(!STF_WINDOWSSYSPATH)
   AddSectionFilesToCopyList Files-$(Option) $(SrcDir) $(!STF_WINDOWSSYSPATH)\
drivers
   exit
[Source Media Descriptions]
    1 = "Windows NT Workstation CD-ROM" , TAGFILE = cdrom_w.40
[Signature]
    FileType = MICROSOFT_FILE
[GetSignature]
    read-syms Signature
    return $(FileType)
[ProductType]
STF_PRODUCT  = Winnt
STF_PLATFORM = I386
[Files-Inf]
2,    oemsetup.inf,      SIZE=1000, RENAME=$(!UG_Filename)
[Files-DLC]
1,DLC.SYS , SIZE=55120, OVERWRITE=OLDER
[Files-DLCAPI]
1,DLCAPI.DLL , SIZE=13072
1,HPMON.DLL , SIZE=60176
1,HPMON.HLP , SIZE=999
[LanguagesSupported]
    ENG
[OptionsTextENG]
    DLC     = "DLC Protocol"
[FileConstantsENG]
ProCaption   = "Windows NT Setup"
ProCancel    = "Cancel"
ProCancelMsg = "Windows NT Networking is not correctly installed.  "+
               "Are you sure you want to cancel copying files?"
ProCancelCap = "Network Setup Message"
ProText1     = "Copying:"
ProText2     = "To:"
FunctionTitle   = "DLC Setup"
ProductSoftwareDescription      = "Data Link Control enables this computer to
connect to an IBM mainframe and set up printers attached directly to network
control."
ProductSoftwareDisplayName      = "DLC Protocol"
ProductSoftwareTitle    = "DLC Protocol"
ShellCodeErrorTitle     = "Error: "$(FunctionTitle)
ShellCodeErrorText      = "Shell Code Error."
[DialogConstantsENG]
Help        = "&Help"
Exit        = "Cancel"
OK          = "OK"
HelpContext = ""
Continue    = "Continue"
Cancel      = "Cancel"
[FileDependentDlgENG]
```