```
[Identification]
    OptionType = NetAdapter
[PlatformsSupported]
    MCA
[Options]
    IBMTOKMC
    IBMTOKA
[FileConstants]
UtilityInf      = "UTILITY.INF"
subroutineinf   = "SUBROUTN.INF"
SoftwareType    = "driver"
Exit_Code       = 0
NetEventDLL     = "%SystemRoot%\System32\netevent.dll"
IoLogMsgDLL     = "%SystemRoot%\System32\IoLogMsg.dll"
IBMTOKMC_1BYTE  = 1
IBMTOKMC_2BYTE  = 224
IBMTOKMC_ID     = 57345
IBMTOKA_1BYTE  = 0
IBMTOKA_2BYTE  = 224
IBMTOKA_ID     = 57344
Manufacturer    = "Microsoft"
ProductMajorVersion     = "4"
ProductMinorVersion     = "0"
ProductVersion  = $(ProductMajorVersion)"."$(ProductMinorVersion)
ProductSoftwareName     = "IbmTok"
ProductSoftwareImagePath = "\SystemRoot\system32\drivers\ibmtok.sys"
NetRuleSoftwareType     = "ibmtokSys ndisDriver ibmtokDriver"
NetRuleSoftwareUse      = $(SoftwareType)
NetRuleSoftwareBindForm = """IBMTokSys"" yes no container"
NetRuleSoftwareClass    = {"ibmtokriver basic"}
NetRuleSoftwareBindable = {"ibmtokDriver ibmtokAdapter non exclusive 100",+
                           "ibmtokDriver ibmtokmcAdapter non exclusive 100"}
ProductHardwareName     = "IbmTokMC"
NetRuleHardwareType     = "ibmtokmc ibmtokmcAdapter"
NetRuleHardwareBindForm = " yes yes container"
NetRuleHardwareClass    = {"ibmtokmcAdapter basic"}
ProductOpSupport     = 134
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)
        ifstr(i) $($1) == ""
           goto returnoptions
        endif
        set PlatformList = ^(PlatformsSupported, 1)
        Ifcontains(i) $($1) in $(PlatformList)
           goto returnoptions
        else
           set Status = STATUS_NOTSUPPORTED
           goto finish_ReturnOptions
        endif
    else
        set Status = STATUS_NOLANGUAGE
        goto finish_ReturnOptions
    endif
returnoptions = +
    set OptionList     = ^(Options, 1)
    set OptionTextList = ^(OptionsText$($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 $(LanguageList)
        Return STATUS_NOLANGUAGE
    endif
    Debug-Output "OEMNADTM.INF: STF_CWDIR is: "$(!STF_CWDIR)
    Debug-Output "OEMNADTM.INF: STF_LANGUAGE is: "$(!STF_LANGUAGE)
    set-subst LF = "\n"
    read-syms GeneralConstants
    read-syms FileConstants
    read-syms DialogConstants$(!STF_LANGUAGE)
    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
        set CommonStatus = STATUS_REBOOT
        Ifstr(i) $(ProductKeyName) == $(!NTN_RegBase)
            Debug-Output "Cannot configure the ibm token ring driver software."
            Shell $(UtilityInf),RegistryErrorString,CANNOT_CONFIGURE_SOFTWARE
            ifint $($ShellCode) != $(!SHELL_CODE_OK)
                Debug-Output "ShellCode error: cannot get an error string."
                goto ShellCodeError
            endif
            set Error = $($R0)
            set from = end
            set to = end
            goto nonfatalinfo
        endif
        set StartLabel = configureadapter
    else
        set StartLabel = installadapter
        set OEM_ABANDON_OPTIONS = {}
        set OEM_ABANDON_SOFTWARE = FALSE
        set OEM_ABANDON_ON = TRUE
    endif
    set from = $(fatal)
    set to = $(fatal)
    goto $(StartLabel)
installadapter = +
    OpenRegKey $(!REG_H_LOCAL) "" $(ProductKeyName) $(MAXIMUM_ALLOWED) KeyProduct
    Ifstr $(KeyProduct) != $(KeyNull)
        CloseRegKey $(KeyProduct)
        ifstr(i) !(NTN_RegBase) == $(ProductKeyName)
           Shell $(UtilityInf), VerExistedDlg, $(ProductSoftwareTitle),+
               $(ProductVersion)
           ifint $($ShellCode) != $(!SHELL_CODE_OK)
               Debug-Output "ShellCode error: cannot get an error string."
               goto ShellCodeError
           endif
           goto end
        else
           Shell $(UtilityInf), CardExistedDlg
           ifint $($ShellCode) != $(!SHELL_CODE_OK)
               Debug-Output "ShellCode error: cannot get an error string."
               goto ShellCodeError
           endif
           ifstr(i) $($R1) != "OK"
               Set CommonStatus = STATUS_USERCANCEL
               goto end
           endif
           set OldVersionExisted = $(TRUE)
        endif
    endif
    goto nextstep
configureadapter = +
    OpenRegKey $(!REG_H_LOCAL) "" $(!NTN_RegBase) $(MAXIMUM_ALLOWED) NetCardKey
    ifstr(i) $(NetCardKey) == ""
        set CommonStatus = STATUS_USERCANCEL
        goto successful
    endif
```

```
    GetRegValue $(NetCardKey) "ServiceName" ServiceInfo
    set ServiceName = *($(ServiceInfo),4)
    CloseRegKey $(NetCardKey)
    OpenRegKey $(!REG_H_LOCAL) "" $(!NTN_ServiceBase)"\"$(ServiceName)"\Parameters"
$(MAXIMUM_ALLOWED) ParamKey
    ifstr(i) $(ParamKey) != ""
        GetRegValue $(ParamKey),"NetworkAddress", NetworkAddressInfo
        set NetworkAddress = *($(NetworkAddressInfo), 4)
        read-syms FileDependentDlg$(!STF_LANGUAGE)
        ui start "NetworkAddress"
        ifstr(i) $(DLGEVENT) == "CONTINUE"
            set NetworkAddress = *($(EditTextOut),1)
            SetRegValue $(ParamKey) {NetworkAddress,$(NoTitle),$(!REG_VT_SZ),$
(NetworkAddress)}
            ui pop 1
        else
            set CommonStatus = STATUS_USERCANCEL
            ui pop 1
        endif
        CloseRegKey $(ParamKey)
    endif
    goto successful
nextstep = +
    StartWait
installproduct = +
    ifstr(i) $(Option) == "IBMTOKMC"
        set NETCARD_ID = $(IBMTOKMC_ID)
        Shell $(UtilityInf), MCAFindBus, $(IBMTOKMC_1BYTE), $(IBMTOKMC_2BYTE)
    else
        set NETCARD_ID = $(IBMTOKA_ID)
        Shell $(UtilityInf), MCAFindBus, $(IBMTOKA_1BYTE), $(IBMTOKA_2BYTE)
    endif
    ifint $($ShellCode) != $(!SHELL_CODE_OK)
        goto ShellCodeError
    endif
    ifstr $($R0) != "NO_ERROR"
        set Error = $($R0)
        goto fatal
    endif
    ifstr(i) $($R1) == {}
        set Error = $(CANNOT_FIND_ANY_CARD)
        set CommonStatus = STATUS_USERCANCEL
        set from = "end"
        goto nonfatal
    endif
    set AdapterList = $($R1)
    ifint $(OldVersionExisted) == $(FALSE)
        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
      Shell $(UtilityInf), AddSoftwareComponent, $(Manufacturer), +
          $(ProductSoftwareName), +
          $(ProductSoftwareName), +
          $(ProductSoftwareTitle), $(STF_CONTEXTINFNAME), +
          $(ProductSoftwareImagePath), "kernel", "NDIS", {}, "",+
          $(NetEventDLL)
      Set OEM_ABANDON_SOFTWARE = TRUE
      ifint $($ShellCode) != $(!SHELL_CODE_OK)
          goto ShellCodeError
      endif
      set RegistryErrorIndex = $($R0)
      Ifstr(i) $(RegistryErrorIndex) != "NO_ERROR"
          EndWait
          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)},+
                          {ServiceName,$(NoTitle),$(!REG_VT_SZ),$
(ProductSoftwareName)},+
                          {InstallDate,$(NoTitle),$(!REG_VT_DWORD),*($(Now),1)}}
      Shell  $(UtilityInf), AddValueList, $(SoftProductKey), $(NewValueList)
      ifint $($ShellCode) != $(!SHELL_CODE_OK)
          goto ShellCodeError
      endif
      set RegistryErrorIndex = $($R0)
      Ifstr(i) $(RegistryErrorIndex) != "NO_ERROR"
```

```
            EndWait
            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)
            goto ShellCodeError
        endif
        set RegistryErrorIndex = $($R0)
        CloseRegKey $(SoftProductKey)
        CloseRegKey $(SoftNetRuleKey)
        CloseRegKey $(SoftServiceKey)
        CloseRegKey $(SoftParameterKey)
        CloseRegKey $(SoftLinkageKey)
        Ifstr(i) $(RegistryErrorIndex) != "NO_ERROR"
            EndWait
            goto fatalregistry
        endif
    endif
    ForListDo $(AdapterList)
        set BusNum = *($($),1)
        set SlotNum = *($($),2)
        Debug-Output $(BusNum)
        Debug-Output $(SlotNum)
        Shell $(UtilityInf), IsNetCardAlreadyInstalled, $(BusNum), +
            $(SlotNum), $(ProductHardware$(Option)Description), $
(ProductHardwareName)
        ifint $($ShellCode) != $(!SHELL_CODE_OK)
            goto ShellCodeError
        endif
        ifstr $($R0) != "NO_ERROR"
            set Error = $($R0)
            goto fatal
        endif
        ifstr(i) $($R1) != "YES"
                Shell $(UtilityInf), AddHardwareComponent, $(ProductHardwareName),$
(STF_CONTEXTINFNAME),$(ProductKeyName)
                ifint $($R4) != -1
                    Set OEM_ABANDON_OPTIONS = >($(OEM_ABANDON_OPTIONS), $(!
NTN_SoftwareBase)"\Microsoft\Windows NT\CurrentVersion\NetworkCards\"$($R4))
                endif
                ifint $($ShellCode) != $(!SHELL_CODE_OK)
                    goto ShellCodeError
                endif
                set RegistryErrorIndex = $($R0)
```

```
                Ifstr(i) $(RegistryErrorIndex) != "NO_ERROR"
                    EndWait
                    CloseRegKey $($R1)
                    CloseRegKey $($R2)
                    CloseRegKey $($R3)
                    goto fatalregistry
                endif
                Set HardNetCardKey      = $($R1)
                Set HardNetRuleKey      = $($R2)
                Set HardParameterKey    = $($R3)
                set AdapterNumber       = $($R4)
                set NewValueList = {{Manufacturer,$(NoTitle),$(!REG_VT_SZ),$
(Manufacturer)},+
                                    {Title,$(NoTitle),$(!REG_VT_SZ),"["$($R4)"] "$
(ProductHardware$(Option)Title)},+
                                    {Description,$(NoTitle),$(!REG_VT_SZ),$
(ProductHardware$(Option)Description)},+
                                    {ProductName,$(NoTitle),$(!REG_VT_SZ),$
(ProductHardwareName)},+
                                    {ServiceName,$(NoTitle),$(!REG_VT_SZ),$($R5)},+
                                    {OperationsSupport,$(NoTitle),$(!REG_VT_DWORD),$
(ProductOpSupport)},+
                                    {InstallDate,$(NoTitle),$(!REG_VT_DWORD),*($
(Now),1)}}
                Shell  $(UtilityInf), AddValueList, $(HardNetCardKey), $
(NewValueList)
                ifint $($ShellCode) != $(!SHELL_CODE_OK)
                    goto ShellCodeError
                endif
                ifstr(i) $(!STF_GUI_UNATTENDED) == "YES"
                    set NetworkAddress = ""
                    goto WriteNetworkAddr
                endif
                read-syms FileDependentDlg$(!STF_LANGUAGE)
                ui start "NetworkAddress"
                set NetworkAddress = *($(EditTextOut),1)
WriteNetworkAddr = +
                set NewValueList = {{BusNumber,$(NoTitle),$(!REG_VT_DWORD),$
(BusNum)},+
                                    {MediaType,$(NoTitle),$(!REG_VT_DWORD),2},+
                                    {NetworkAddress,$(NoTitle),$(!REG_VT_SZ),$
(NetworkAddress)},+
                                    {McaPosId,$(NoTitle),$(!REG_VT_DWORD),$
(NETCARD_ID)},+
                                    {BusType,$(NoTitle),$(!REG_VT_DWORD),3},+
                                    {SlotNumber,$(NoTitle),$(!REG_VT_DWORD),$
(SlotNum)}}
                Shell  $(UtilityInf), AddValueList, $(HardParameterKey), $
(NewValueList)
                ifstr(i) $(!STF_GUI_UNATTENDED) == "YES"
                    Shell $(UtilityInf),AddDefaultNetCardParameters,$
(HardParameterKey)
                endif
                ifint $($ShellCode) != $(!SHELL_CODE_OK)
                    goto ShellCodeError
                endif
                set TempProdName = """"$(ProductHardwareName)$(AdapterNumber)""""
                set TempBindForm = $(TempProdName)$(NetRuleHardwareBindForm)
                set NewValueList = {{type,$(NoTitle),$(!REG_VT_SZ),$
```

```
(NetRuleHardwareType)},+
                                    {bindform,$(NoTitle),$(!REG_VT_SZ),$
(TempBindForm)}, +
                                    {class,$(NoTitle),$(!REG_VT_MULTI_SZ),$
(NetRuleHardwareClass)}, +
                                    {InfOption,$(NoTitle),$(!REG_VT_SZ),$(Option)}}
                Shell  $(UtilityInf), AddValueList, $(HardNetRuleKey), $
(NewValueList)
                CloseRegKey $(HardNetCardKey)
                CloseRegKey $(HardNetRuleKey)
                CloseRegKey $(HardParameterKey)
                ifint $($ShellCode) != $(!SHELL_CODE_OK)
                    goto ShellCodeError
                endif
        endif
    EndForListDo
    goto writeparameters
writeparameters = +
    EndWait
    goto successful
bindingadapter =+
    set Error = "Binding: Sorry, not yet implemented."
    goto fatal
removeadapter = +
    Ifstr(i) $(ProductKeyName) == $(!NTN_RegBase)
        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
    else
        Shell $(UtilityInf), RemoveHardwareComponent, $(Manufacturer), +
            $(ProductSoftwareName), $(!NTN_RegBase)
        ifint $($ShellCode) != $(!SHELL_CODE_OK)
            Debug-Output "ShellCode error"
            goto ShellCodeError
        endif
        set RegistryErrorIndex = $($R0)
        Ifstr(i) $(RegistryErrorIndex) != NO_ERROR
            goto fatalregistry
        endif
    endif
    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)}
```

```
        CloseRegKey $(KeyProduct)
    else
        goto fatalregistry
    endif
    set iSearch = 1
nextnetcard = +
    Shell $(UtilityInf), FindNextNetworkCard, $(ProductHardwareName), $(iSearch)
    set KeyNetcard = $($R0)
    set iSearch = $($R1)
    Debug-Output "OemNadEp.Inf: FindNextNetworkCard "$(KeyNetcard)","$(iSearch)
    Ifstr $(KeyNetcard) != $(KeyNull)
        Debug-Output "OemNadEp.Inf: Setting OperationsSupport value"
        SetRegValue $(KeyNetcard) {OperationsSupport,$(NoTitle),$(!REG_VT_DWORD),$
(ProductOpSupport)}
        CloseRegKey $(KeyNetcard)
        goto nextnetcard
    Endif
    goto end
successful = +
    goto end
abandon = +
    ForListDo $(OEM_ABANDON_OPTIONS)
        Shell $(UtilityInf), RemoveHardwareComponent, $(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
    EndForListDo
    Ifstr(i) $(OEM_ABANDON_SOFTWARE) == TRUE
        Shell $(UtilityInf), RemoveSoftwareComponent, $(Manufacturer), +
            $(ProductSoftwareName), FALSE
        ifint $($ShellCode) != $(!SHELL_CODE_OK)
            Debug-Output "ShellCode error"
            goto ShellCodeError
        endif
        set RegistryErrorIndex = $($R0)
        Ifstr(i) $(RegistryErrorIndex) != NO_ERROR
            goto fatalregistry
        endif
    endif
    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 abandon
    endif
```

```
    goto end
end = +
    goto term
term = +
    Return $(CommonStatus)
[Install-Option]
    set STF_VITAL = ""
    ifstr(i) $(AddCopy) == "YES"
        AddSectionFilesToCopyList Files-IBMTOKMC $(SrcDir) $(!STF_WINDOWSSYSPATH)\
drivers
    endif
    ifstr(i) $(DoCopy) == "YES"
       set !STF_NCPA_FLUSH_COPYLIST = TRUE
       CopyFilesInCopyList
    endif
    ifstr(i) $(DoConfig) == "YES"
    endif
    Exit
[Install-Update]
   set STF_VITAL        = ""
   set STF_OVERWRITE    = "VERIFYSOURCEOLDER"
   AddSectionFilesToCopyList Files-IBMTOKMC $(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-IBMTOKMC]
1,IBMTOK.SYS , SIZE=999
[LanguagesSupported]
    ENG
[OptionsTextENG]
    IBMTOKMC     = "IBM Token Ring 4/16 Adapter /A"
    IBMTOKA      = "IBM Token Ring Adapter /A"
[FileConstantsENG]
ProCaption   = "Windows NT ��ı���"
ProCancel    = "��ﻮ"
ProCancelMsg = "Windows NT ò[illegible]ŧİ§���Ă��· ���B"+
               "[illegible]~���· ���?"
ProCancelCap = "ò[illegible] ��ı��  γ[illegible]"
ProText1     = "[illegible]:"
ProText2     = "[illegible]:"
FunctionTitle   = "IBM Token Ring [illegible] ı���"
ProductSoftwareDescription      = "IBM Token Ring [illegible] ��"[illegible]"
ProductHardwareIBMTOKADescription      = "IBM Token Ring [illegible] /A"
ProductHardwareIBMTOKMCDescription     = "IBM Token Ring 4/16 [illegible] /A"
CANNOT_FIND_ANY_CARD            = "[illegible]ò[illegible] ···· [illegible]~[illegible]B"
ProductSoftwareTitle    = "IBM Token Ring [illegible] ��"[illegible]"
ProductHardwareIBMTOKATitle     = "IBM Token Ring [illegible] /A"
ProductHardwareIBMTOKMCTitle    = "IBM Token Ring 4/16 [illegible] /A"
```

```
ShellCodeErrorTitle     = "▒▒: "$(FunctionTitle)
ShellCodeErrorText      = "▒▒▒ ▒▒▒▒ ▒▒▒▒▒▒B"
[DialogConstantsENG]
Help        = "▒▒▒▒(&H)"
Exit        = "▒▒▒▒"
OK          = "OK"
HelpContext = ""
Continue    = "▒▒▒s(&O)"
Cancel      = "▒▒▒▒(&A)"
[FileDependentDlgENG]
Edit1Label = "▒▒▒ ▒▒▒▒(&N):"
DlgType = "Edit"
DlgTemplate = "NETWORK_NUMBER"
Caption = $(FunctionTitle)
HelpContext = $(!IDH_DB_OEMNADTM_INS)
EditTextLim = 17
EditTextIn  = $(NetworkAddress)
EditFocus   = "ALL"
RCCtlFocusOn = 403
```