Gebruik van de XpressNet DLL

-

Inleiding

V1.3
Oktober 2024
Aiko Pras

1. Algemeen

Het doel van deze XpressNet bibliotheek is het schrijven van (MS-Windows) applicatie programma's te vereenvoudigen, door alle complexiteit van het XpressNet interface in één bibliotheek te concentreren.

De XpressNet bibliotheek implementeert een groot aantal XpressNet commando's, alsmede een beperkt aantal Z21 commando's. Deze handleiding bespreekt slechts een klein deel van deze commando's. Op GitHub zijn lijsten beschikbaar met daarin alle ondersteunde commando's:
https://github.com/aikopras/XpressNet/tree/main/Documentation/Commands

De XpressNet bibliotheek wordt aangeboden in de vorm van Visual Basic 6 (VB6) broncode, maar ook als DLL. Voor het gemak wordt in de rest van dit document meestal de term DLL gebruikt. Voor installatie wordt verwezen naar:
https://github.com/aikopras/XpressNet/blob/main/Code/Install/Installeren-Library.md

In dit document wordt op een aantal plaatsen verwezen naar de XpressNet specificatie. Het betreft hier de volgende documenten:
- V3.6 = Befehlsbeschreibung XpressNet V3.6 mit LAN/USB Interface 23151 Kommunikation auf den Schnittstellen, Doku Version 1.2 05/2019 https://www.lenz-elektronik.de/src/pdf/23151_Info.zip
- V4 = Dokumentation XpressNet Version 4.0, 02/2022: https://www.lenz-elektronik.de/src/pdf/Lenz_XpressNet_Doku.pdf

De DLL implementeert een groot aantal methoden (subroutines) en attributen (variabelen). Slechts een deel hiervan mag door de applicatie gebruikt worden; het overige deel is voor intern (DLL) gebruik. Om beide uit elkaar te kunnen houden, beginnen de methoden / attributen die door de gebruiker aangeroepen mogen worden, met een hoofdletter. Methoden / attributen die met een kleine letter beginnen, mogen dus *niet* door de applicatie worden gebruikt. De reden dat locale methoden en attributen ook zichtbaar zijn voor de gebruiker, ligt in beperkingen van VB6.

2. Gebruik van de XpressNet DLL

Om de XpressNet DLL te gebruiken, moet het applicatie programma een object van het type `XpressNetClass` declareren en aanmaken.

Een voorbeeld is hieronder gegeven:

```
Public WithEvents myXpressNet As XpressNetClass

Public Sub Form_Load()
  Set myXpressNet = New XpressNetClass
End Sub
```

In bovenstaand voorbeeld wordt één XpressNet object aangemaakt. Een XpressNet object kan XpressNet berichten verzenden en ontvangen over één interface. Dat interface kan een TCP/IP (LAN) verbinding zijn, een USB interface of een serieel interface. Per XpressNet object kan dus één DCC centrale bedient worden.

Indien meerdere XpressNet interfaces / centrales aangesloten worden, moet er per interface / centrale een eigen Xpressnet object aangemaakt worden. Het voorbeeld hieronder kan twee XpressNet interfaces / centrales bedienen.

```
Public WithEvents Schakelen As XpressNetClass
Public WithEvents Melden As XpressNetClass

Public Sub Form_Load()
  Set Schakelen = New XpressNetClass
  Set Melden = New XpressNetClass
End Sub
```

Om het applicatie programma te informeren, gebruikt de DLL *events*. Om events te kunnen gebruiken, moet het hoofdprogramma, als onderdeel van de object declaratie, het sleutelwoord `WithEvents` toevoegen.

In de voorbeelden hierboven wordt steeds een `myXpressNet` object gedeclareerd van de klasse `XpressNetClass`. De term `WithEvents` geeft aan dat de applicatie *events* wil ontvangen die door het object `myXpressNet` worden gegenereerd.

Voor ieder event dat het applicatie programma wenst te ontvangen, moet een eigen subroutine worden geschreven (event handler). Dergelijke subroutines krijgen als naam `ObjectName_EventName`. Het underscore karakter _ vertelt VB6 dat event en object bij elkaar horen. Dit underscore karakter is dus essentieel, en mag niet worden weggelaten.

3. TCP/IP verbinding opzetten

Om een TCP verbinding op te zetten, moet de methode `connectTcp` aangeroepen worden. Als parameter moet in ieder geval het IP/DNS adres meegegeven worden. Het TCP poort nummer is optioneel en heeft, indien weggelaten, als default waarde 5550.

Onderstaand drie voorbeelden:

```
Private Sub cmdTcpConnectAll_Click()
  Call myXpressNet1.connectTcp("192.168.2.101")
  Call myXpressNet2.connectTcp(“192.168.2.102”, “5550”)
  Call myXpressNet3.connectTcp(“schakelen.local”)
End Sub
```

De TCP verbinding kent de volgende Events:

- `Event InterfaceConnected()`
- `Event TcpClosedByRemoteHost()`
- `Event TcpError(ByRef Description As String)`

Deze Events kunnen worden gebruikt om bijvoorbeeld een bepaald element van kleur te laten veranderen of een status bericht weer te geven:

```
Private Sub XpressNet_InterfaceConnected()
  ledInterfaceStatus.FillColor = vbGreen
  LabelStatus = "TCP Connection established"
End Sub

Private Sub XpressNet_TcpClosedByRemoteHost()
  ledInterfaceStatus.FillColor = vbRed
  LabelStatus = "TCP connection closed by remote host"
End Sub

Private Sub XpressNet_TcpError(ByRef ErrorDescription As String)
  ledInterfaceStatus.FillColor = vbRed
  LabelStatus = ErrorDescription
End Sub
```

4. USB verbinding opzetten

Om een USB verbinding op te zetten, moet de methode `connectUsb` aangeroepen worden. Als parameter moet in de USB poort meegegeven worden.

Onderstaand een voorbeeld waarbij USB poort 3 wordt gebruikt:

```
Private Sub cmdUsbConnect_Click()
  Call myXpressNet.connectUsb(3)
End Sub
```

De USB verbinding kent het volgende Event:

- `Event InterfaceConnected()`

Dit Events kan worden gebruikt om bijvoorbeeld een bepaald element van kleur te laten veranderen of een status bericht weer te geven:

```
Private Sub XpressNet_InterfaceConnected()
  ledInterfaceStatus.FillColor = vbGreen
  LabelStatus = "TCP Connection established"
End Sub
```

5. Wissel commando’s

Het onderstaande commando kan gebruikt worden om wissels om te zetten, signalen te veranderen, of andere acties uit te voeren door middel van DCC commando’s:

```
AccessoryCommandAuto(ByVal Address As Integer,
  ByVal Position As Byte)
```

De `Address` parameter is het adres van de wissel en heeft als laagste waarde 1 (dus niet 0). Het hoogste adres heeft de waarde 1024 (≤ V3.6), of 2048 (≥ 3.8). De DLL controleert zelf of de waarde past bij de gebruikte centrale. Intern leest de DLL hiervoor de variabele `ZentraleVersion`.

Indien de adres waarde niet past bij de gebruikte centrale / XpressNet versie, zal het commando genegeerd worden waarna een foutmelding in de logfile wordt gezet.

De `Position` parameter heeft als waarde `0` (rechtdoor) of `1` (af). Waarden >1 geven geen foutmelding, maar worden gezien als 1.

5.1. Activeren / Deactiveren

Een wisselaandrijving die uit spoelen bestaat, kan doorbranden als de spanning op de spoel aanwezig blijft. Ook motorische wisselaandrijvingen kunnen doorbranden, als de spanning op de motor aanwezig blijft. Goede wisselaandrijvingen hebben daarom vaak schakelaars ingebouwd, die de spanning weghalen nadat de eindstand is bereikt. Ook kunnen veel wisseldecoders de spanning weghalen, nadat een bepaalde tijd is verstreken.

Niet alle wisselaandrijvingen / decoders halen echter de spanning weg nadat de eindstand is bereikt, of een vooringestelde tijd is verstreken. Het DCC kent daarom een tweede commando, waarmee de spanning weer weggehaald kan worden. Dit tweede commando wordt door deze DLL automatisch verstuurd na 250 milliseconde (deze tijd is instelbaar).

5.2. Verdere informatie

Verdere informatie is te vinden in de XpressNet specificatie onder de term *Schaltbefehl* (V3.6 §3.2.24, V4: §3.39 en §3.40).

Een discussie over het gebruik van wissel activate / deactivate is the vinden onder: https://www.stummiforum.de/t218248f5-Problem-mit-Lenz-Schaltempfaenger-LS-an-verschiedenen-Zentralen-Lenz-YaMorC.html#msg2596407

6. Terugmeldingen

De DLL kan het applicatie programma informeren of
1) Sporen bezet zijn (bezetmeldingen), en
2) Wat de stand van wissels is (wisselmeldingen).
Dit informeren kan op verzoek gebeuren, maar ook ongevraagd, nadat er een verandering heeft plaatsgevonden. Met name deze ongevraagde status veranderingen zijn belangrijk voor het applicatie programma.

6.1. Bezetmeldingen

Om bezetmeldingen te ontvangen, wordt het event met de naam `Feedback` gebruikt.
Een voorbeeld van een event handler subroutine is hieronder gegeven:

```
Private Sub myXpressNet_Feedback(ByVal address As Integer,
  ByVal nibble As Byte, ByVal dataBits As Byte)
  ' Do something
End Sub
```

Bovengenoemd event kent drie variabelen:
- **address**: Dit is het RS-Bus adres van de terugmeld decoder. Het laagste RS-Bus adres is 1; het hoogste is 128[1].
- **nibble**: Bij ieder RS-Bus adres behoren 8 bits (het maximale aantal terug meldingen is dus 8 x 128 = 1024). Per event worden echter maar vier bits gemeld: bits 0 t/m 3 of bits 4 t/m 8. Indien het nibble bit = 0, dan vertegenwoordigen de dataBits de eerste vier bits (0 t/m 3). Indien het nibble bit = 1, dan vertegenwoordigen de dataBits de hoogste vier bits (4 t/m 7).
- **dataBits**: vier bits voor vier terugmeldingen.

6.2. Wisselmeldingen

Om wisselmeldingen te ontvangen, wordt het event met de naam `SwitchFeedback` gebruikt. Per event wordt de stand van twee wissels (4 bits) teruggemeld.
Een voorbeeld van een event handler subroutine is hieronder gegeven:

```
Private Sub myXpressNet_Switch Feedback(ByVal address As Integer,
  ByVal ttBits As Byte, ByVal iBit As Byte, ByVal dataBits As Byte)
  ' Do something
End Sub
```

Bovengenoemd event kent vier variabelen:
- **address**: Dit is het adres van de eerste wissel, dus die met het laagste adres[2]. Het adres van de tweede wissel is 1 hoger.
- **ttBits**: Er zijn twee soorten decoders waarmee wissels geschakeld kunnen worden:
  1) wisseldecoders *zonder* terugmelding, (zoals de Lenz LS150), en
  2) wisseldecoders *met* terugmelding (zoals de Lenz LS101).
  Voor decoders *zonder* terugmelding meldt de centrale zelf, wanneer de stand van de wissel veranderd. Het analyseert hiertoe het laatste wisselcommando.
  Voor decoders *met* terugmelding geeft de centrale de terugmelding door, die het van de decoder heeft ontvangen.
  De TTBits geven aan met welk van deze twee soorten decoder we te maken hebben.
  1) Indien de waarde van de ttBits = 0, dan is de terugmelding door de centrale gegenereerd.
  2) Indien de waarde van de ttBits = 1, dan is de terugmelding door een wisseldecoder met terugmelding gegenereerd.
- **iBit:** Als het iBit = 1, dan is het wisselcommando nog in uitvoering en heeft de wissel zijn eindstand nog niet bereikt.

---

[1] Afhankelijk of men de Lenz XpressNet of de LH100 specificatie leest, hebben de RS-Bus adressen een bereik van 0..127 dan wel van 1..128. In deze library is het bereik 1..128.

[2] In de XpressNet berichten worden decoder adressen gebruikt. De DLL vertaald deze adressen naar wissel adressen

- **dataBits**: vier bits (0..3) voor twee wissels.
  Bit 0 en 1 worden gebruikt voor de wissel met het laagste adres; bit 2 en 3 worden gebruikt voor de wissel met het hoogste adres.
  Tijdens normaal gebruik hebben beide bits tegenovergestelde waarden. Als de wissel in de éne stand staat, is het éne bit 1 en het andere 0. Als de wissel in de andere stand staat, zijn de waarden omgedraaid.
  Beide bits kunnen in de opstart fase ook de waarden 0 hebben, als er nog geen wisselcommando heeft plaatsgevonden en de centrale niet weet in welke stand de wissel staat.

6.3. Wanneer kunnen wisselmeldingen gebruikt worden?

Het is mogelijk de applicatie te schrijven zonder de events voor wisselmelding te gebruiken. Indien de applicatie bijvoorbeeld het gebruikersinterface aanpast, onmiddellijk nadat een wisselcommando is verstuurd, is het niet nodig de wisselmelding events te gebruiken.

Als er echter meerdere apparaten (PCs, handhelds) zijn die wisselcommando's kunnen versturen, zou in het bovenstaande voorbeeld alleen op het apparaat dat het wisselcommando heeft verstuurd, het gebruikersinterface worden aangepast. De gebruikersinterfaces op de overige apparaten laten de nieuwe wisselstand dan niet zien. In dergelijke gevallen kan het daarom soms beter zijn op op alle apparaten pas het gebruikersinterface aan te passen, nadat een wisselmelding is ontvangen.

Een tweede voorbeeld waarbij het gebruik van wisselmeldingen handig kan zijn, is wanneer de mogelijkheid bestaat dat gebruikers de stand van wissels handmatig veranderen. Dit is vaak het geval bij traditionele wisselaandrijvingen, die uit twee spoelen bestaan. Afhankelijk van het type decoder en aandrijving, kan de werkelijke stand van de wissel worden teruggemeld.
Bij andere soorten wisselaandrijvingen, zoals telefoonrelais, motoren of servo's, kan de stand van de wissel meestal niet handmatig worden veranderd, en geven wisselmeldingen dus geen extra informatie.

Een probleem bij alle terugmeld systemen (dus niet alleen XpressNet), is dat de informatie van meerdere wissels in één terugmelding worden samengevoegd. Het probleem kan zich voordoen als een applicatie programma twee wisselcommando's onmiddellijk na elkaar verstuurd, en vervolgens een terugmelding ontvangt waarin staat dat de eerste wissel is omgezet, maar de tweede (nog?) niet. Het applicatieprogramma weet nu niet of de tweede terugmelding later komt, of dat er iets mis is gegaan met het tweede wisselcommando.

7. Lok commando's

7.1. Lok aanmelden

Voordat de snelheid of de functies F0 t/m F12 van een lok worden aangepast, moet eerst de centrale gevraagd worden welke informatie het al over de lok heeft opgeslagen.
Het onderstaande commando kan gebruikt worden om lok en informatie op te halen

```
CmdLokInfoAnfordern(ByVal Address As Integer)
```

De centrale beantwoord deze aanvraag met het volgende event:

```
Event NormaleLokinfo(ByVal Besetzt As Boolean, ByVal Steps As
 Integer, ByVal Speed As Integer, ByVal Forward As Boolean,
 ByVal F0_F4 As Byte, ByVal F5_F8 As Byte, ByVal F9_F12 As Byte)
```

De parameters hebben de volgende betekenis:

- `Besetzt`: geeft aan of de lok reeds door een ander XpressNet apparaat in gebruik is.
- `Steps`: geeft aan of de lok met 14, 27, 28 of 128 stappen aangestuurd wordt
- `Speed`: geeft de snelheid aan waarmee de lok momenteel onderweg is
- `Forward`: geeft de richting aan
- `F0_F4`: Geeft aan of de Functies F0 t/m F4 actief zijn of niet.
  Merk op dat F1=Bit0, F2=Bit1, F3=Bit2, F4=Bit3 maar dat F0=Bit4.
- `F5_F8`: Geeft aan of de Functies F5 t/m F8 actief zijn of niet. Hierbij is F5=Bit0 en F8=Bit3.
- `F9_F12`: Geeft aan of de Functies F9 t/m F12 actief zijn of niet.

7.2. Snelheid en richting

Het onderstaande commando kan gebruikt worden om de snelheid en richting van een lok aan te passen:

```
CmdFahrbefehl(ByVal Address As Integer, ByVal Speed As Integer,
 ByVal Steps As Integer, ByVal Forward As Boolean)
```

Bovengenoemd commando kent vier variabelen:

- De `Address` parameter is het adres van de lok en heeft als laagste waarde 0 en als hoogste 9999.
- De `Speed` parameter heeft als laagste waarde 0. De hoogste waarde hangt af van de decoder, en wordt door `Steps` bepaald.
- De `Steps` parameter geeft aan of de decoder is ingesteld voor 14, 27, 28 of 128 stappen.
- De `Forward` parameter geeft aan of de lok vooruit (True) of achteruit rijdt

7.3. Functies schakelen

Bij XpressNet V3.6 kunnen 28, en bij V4 68 functies aan en uit gezet worden. Deze functies zijn verdeeld over 10 groepen (de groepen 6 t/m 10 zijn alleen geïmplementeerd bij V4):

Groep	Functies	Groep	Functies
1	0-4	6	29-36
2	5-8	7	37-44
3	9-12	8	45-52
4	13-20	9	53-60
5	21-28	10	61-68

Met onderstaande commando worden alle functies binnen een groep aangepast:

```
CmdLokFunktionBefehl(ByVal address As Integer, ByVal group As Byte,
 ByVal Data As Byte)
```

Bovengenoemd commando kent die variabelen:
- `Address`: het adres van de lok; heeft als laagste waarde 0 en als hoogste 9999.
- `Group`: de functie groep; getal tussen1 en 10.
- `Data`: de waarde van *alle* functie bits die bij deze groep horen.

Met bovengenoemd commando worden dus altijd meerdere functies gelijktijdig gezet. Het is daarom belangrijk om, voordat één functie bit gezet wordt, alle functiewaarden die bij deze groep behoren worden opgevraagd bij de centrale.

7.4. Functie waarden opvragen

De waarden van de lokfuncties, zoals de centrale die kent, kan worden opgevraagd met speciale commando's. Het commando voor F0-F12 is boven beschreven (`CmdLokInfoAnfordern`).
Voor F13-F28 en F29-F68 zijn er de commando's:
Voor F13-F28:

```
CmdFunktionsZustandF13F28Anfordern(ByVal address As Integer)
```

Voor F29-F69:

```
CmdFunktionsZustandF13F28Anfordern(ByVal address As Integer)
```

De centrale beantwoord deze aanvraag met één van volgende events:

```
Event FunctionsZustandF13F20(ByVal F13_F20 As Byte,
  ByVal F21_F28 As Byte)

Event FunctionsZustandF29F68(ByVal F29_F36 As Byte,
  ByVal F37_F44 As Byte, ByVal F45_F52 As Byte,
  ByVal F53_F60 As Byte, ByVal F61_F68 As Byte)
```

7.5. Functie status

De functie status geeft aan of een functie aan/uit (toggle), of "continue" is (zolang de knop is ingedrukt).

Het commando om de status op een bepaalde waarde te zetten, is:

```
CmdLokFunktionStatus(ByVal address As Integer, ByVal group As Byte,
  ByVal data As Byte)
```

Afhankelijk van de gevraagde functies, is het commando om de status op te vragen:

```
CmdFunktionsStatusAnfordern(ByVal address As Integer)
CmdFunktionsStatusF13F28Anfordern(ByVal address As Integer)
CmdFunktionsStatusF29F68Anfordern(ByVal address As Integer)
```

De volgende events horen hierbij:

```
Event FunctionStatusdF0F12(ByVal F0_F4 As Byte,
  ByVal F5_F8 As Byte, ByVal F9_F12 As Byte)

Event FunctionStatusdF13F20(ByVal F13_F20 As Byte,
  ByVal F21_F28 As Byte, ByVal RefreshMode As Byte)

Event FunctionStatusdF29F68(ByVal F29_F36 As Byte,
  ByVal F37_F44 As Byte, ByVal F45_F52 As Byte,
  ByVal F53_F60 As Byte, ByVal F61_F68 As Byte)
```

8. Gegevens ophalen

8.1. Gegevens betreffende de Centrale

Nadat een (TCP/IP, USB of Serial) verbinding door de DLL is opgebouwd, stuurt de DLL een aantal XpressNet commando's naar de centrale, om de versie en status van de centrale te bepalen. Nadat hierop antwoorden zijn ontvangen, vult de DLL onderstaande variabelen (attributen) automatisch in. Achter de naam van de variabelen staat het type, en een verwijzing naar een beschrijving zoals die is te vinden in Lenz handleiding voor XpressNet V3.6 dan wel XpressNet V4.0.

- `ZentraleVersion` String (V3.6: §3.1.3)
- `ZentraleKennung` (=ID) Byte (V3.6: §3.1.3)
- `ZentraleStartmodeAuto` Boolean (V3.6: §3.1.4)
- `ZentraleKaltstart` Boolean (V3.6: §3.1.4)
- `ZentraleRamCheckFehler` Boolean (V3.6: §3.1.4)
- `ZentraleNotAus` Boolean (V3.6: §3.1.1.2, §3.1.4)
- `ZentraleNotHalt` Boolean (V3.6: §3.1.1.3, §3.1.4)
- `ZentraleProgrammierMode` Boolean (V3.6: §3.1.1.4, §3.1.4)

Deze variabelen worden door de DLL ook intern gebruikt, bijvoorbeeld om te bepalen of wissel adressen > 1024 zijn toegestaan (≥V3.8) of niet (≤3.6).

De versie van de centrale wordt doorgegeven aan het hoofdprogramma dmv. het volgende event:

- `ZentraleVersion(version As String)`

De meeste van bovengenoemde variabelen zijn statisch, en veranderen niet van waarde nadat de verbinding is opgebouwd. Een uitzondering hierop zijn de variabelen `ZentraleNotAus`, `ZentraleNotHalt` en `ZentraleProgrammierMode`, die ook later, na ontvangst van een broadcast (V3.6: §3.1.1), van waarde kunnen veranderen. Als dit gebeurt wordt de DLL gebruiker geïnformeerd door middel van één van de onderstaande events:

- `AllesAn()`
- `NotAus()`
- `NotHalt()`
- `ProgrammierMode()`

8.2. Gegevens betreffende XpressNet

Het XpressNet adres en versie worden opgevraagd door de volgende commando's (Methods):

- `CmdXpressNetAddress (Optional Address As Integer)` (V3.6: §1.7)
- `CmdXpressNetVersion()` (V3.6: §2.2)

`Address` is een optionele parameter. Indien deze wordt ingevuld met een waarde tussen 1 en 31, dan wordt deze waarde als nieuw adres ingesteld.

De antwoorden worden aan de DLL applicatie doorgegeven via onderstaande events:

- `XpressNetAddress (Address As Byte)`
- `XpressNetVersion (Version As String)`

`Version` wordt als string doorgegeven, omdat het applicatie programma normaal gesproken deze variabele niet intern in berekeningen gebruikt, maar meteen toont aan de gebruiker.

8.3. Gegevens betreffende het Interface

De versie van het LAN/USB interface en het aantal vrije TCP verbindingen (indien de LAN poort wordt gebruikt), worden opgevraagd door de volgende commando's (Methods):

- `CmdInterfaceVersion()` (V3.6: §1.6)
- `CmdInterfaceFreieVerbindungen()` (V3.6: §2.3)

De antwoorden worden aan de DLL applicatie doorgegeven via onderstaande events:

- `InterfaceVersion(version As String, code As String)`

• `InterfaceFreieVerbindungen(number As Byte)`

`Version` en `code` worden als string doorgegeven, omdat het applicatie programma normaal gesproken deze variabelen niet intern in berekeningen gebruikt, maar meteen toont aan de gebruiker.

`InterfaceFreieVerbindungen` geeft aan hoeveel vrije TCP verbindingen het interface nog beschikbaar heeft. Het 231515 LAN Interface kan maximaal 8 TCP verbindingen gelijktijdig ondersteunen.

8.4. Extra informatie betreffende de centrale

Vanaf XpressNet V4 is het mogelijk om extra informatie op te vragen omtrent de versie en “build” van de centrale, terugmeldprocessor en bootloader. Hiertoe kan het volgende commando worden gebruikt:

• `CmdErweiterteZentralenVersionsinformation()`

ALs resultaat genereert de DLL het volgende event:

• `Event ErweiterteZentraleVersion(BuildZentrale, VersionRuckMelder, BuildRuckMelder, VersionBootloader)`

Alle vier parameters zijn van het type String.

8.5. Aantal XpressNet berichten

Het hoofdprogramma kan twee variabelen (attributen) gebruiken die bijhouden hoeveel XpressNet berichten zijn ontvangen / verstuurd:

• `NumberOfExpressNetMessagesReceived`
• `NumberOfExpressNetMessagesTransmitted`

9. Modeltijd

Vanaf XpressNet V4 houdt de centrale een modeltijd bij waarmee apparaten die op XpressNet zijn aangesloten geïnformeerd worden over de huidje modeltijd.

Hiervoor stuurt de DLL het volgende Event:
• `Event Modellzeit(day, hour, minute, factor)`
Alle vier variabelen zijn van het type Byte.
Dit event treed op als er een modelminuut is verstreken, of als de gebruiker de tijd expliciet opvraagt door middel van het `CmdModellzeitAnfordern()` commando.
• `day`: 0 = maandag, 6 = zondag
• `factor`: Als de gebruiker zelf de modeltijd opvraagt, geeft factor de snelheid aan waarmee de klok loopt; 0 is uit, 1 is langzaam en 31 is snel.

Het hoofdprogramma kan bijvoorbeeld onderstaande code gebruiken voor het tonen van de huidige modeltijd (uren en minuten)

```
Private Sub XpressNet_Modellzeit(ByVal day As Byte,
  ByVal hour As Byte, ByVal minute As Byte, ByVal Faktor As Byte)
   TxtHours = hour
     If minute < 10 Then
     TxtMinutes = "0" & minute
   Else
     TxtMinutes = minute
   End If
End Sub
```

De volgende commando's (methods) kunnen gebruikt worden om de modeltijd te starten, stoppen dan wel op te vragen
• `CmdModellzeitStarten()`
• `CmdModellzeitAnhalten()`
• `CmdModellzeitAnfordern()`

Het gebruikersprogramma kan ik de modeltijd opnieuw instellen met behulp van onderstaand commando:
• `CmdModellzeitStellen(day, hour, minute, factor)`

10.Logfile

De naam van de logfile is "TMC-LogFile.txt". Indien gewenst, kan deze naam in de programma code van de logfile class worden aangepast. De logfile wordt opgeslagen in de directory "Users\Public".

Door de bibliotheek wordt één logfile bijgehouden, ongeacht of er op dat moment meerdere verbindingen naar verschillende centrales zijn opgebouwd. Alle verbindingen delen dus dezelfde logfile. Om desondanks te zien bij welke verbinding een verstuurd / ontvangen bericht behoort, wordt in de logfile aan het einde van een regel de verbindingsgegevens (IP adres, Com poort) tussen haakje toegevoegd.

Als het programma wordt gestart, wordt de bestaande logfile verder aangevuld. Na een programmastart wordt in principe dus *geen* nieuwe, lege logfile aangemaakt. Een uitzondering hierop is als de grootte van de reeds bestaande logfile meer is dan bepaald maximum. In dat geval zal de bestaande logfile een volgnummer krijgen en hernoemd worden. Maximaal drie oude versies blijven bestaan; eventueel oudere versies worden verwijderd.
De maximale log file afmeting is 100 KByte. Indien gewenst, kan deze waarde in de programma code van de logfile class worden aangepast.

Indien gewenst, kan de logfile ook door het hoofdprogramma worden gebruikt. Hiervoor dient de Append methode aangeroepen te worden (zie voorbeeld hieronder).

Het hoofdprogramma kan bepalen welke gegevens wel en welke niet in de logfile worden opgeslagen. Standaard worden alle gegevens opgeslagen. Indien er veel XpressNet berichten worden uitgewisseld, kan het om performance redenen verstandig zijn het aantal berichten te beperken. Hiervoor kan het hoofdprogramma de onderstaande (boolean) variabelen gebruiken; de standaard waarde van de meeste variabelen is True, maar kan door het hoofdprogramma op False worden gezet.

- logErrors (Default: True)
- logWarnings (Default: True)
- logHexMessages (Default: True)
- logDetails (Default: False)

Een voorbeeld van het gebruik van de logfile is hieronder gegeven.

```
Public WithEvents XpressNet As XpressNetClass

Public Sub Form_Load()
  Set XpressNet = New XpressNetClass
  XpressNet.logHexMessages = False
  XpressNet.logfile.Append ("My message")
End Sub
```

11.Installatie van de Xpressnet library

Om de XpressNet DLL te kunnen gebruiken, moet deze eerst gedownload worden van GitHub en in een permanente folder worden gezet. Een goede locatie is `C:\XpressNetDll`. Vervolgens wordt de Windows Command Line aangeroepen en naar bovengenoemde folder gesprongen. Daarna wordt het commando `regsvr32 XpressNet.dll` aangeroepen, die gegevens omtrent de DLL in de windows registry zet. Nadat in de VB6 IDE een reference is toegevoegd naar XpressNet.dll, kan de applicatie gebruik maken van de XpressNet library. Verdere details zijn te vinden in de GitHub repository.