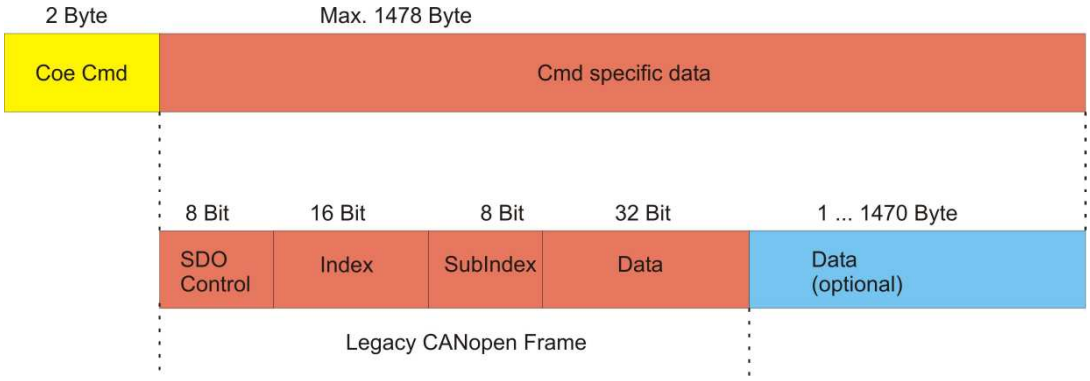
Sybera TechNews

11.4.2018

SDO Handling mit dem EtherCAT Master

Die meisten EtherCAT Geräte verwenden im Application Layer für die Kommunikation und Parametrierung CANopen. Mit dem Protokoll CANopen over EtherCAT (CoE) wird die Nutzung des CANopen ermöglicht.
Unter CANopen werden Kommunikationsobjekte, s.g. "Service Data Objects" (SDOs) verwendet. Mit diesen Objekten können Verzeichniseinträge (über Index und SubIndex) gelesen und geschrieben werden, um somit z.B. das PDO-Mapping zu realisieren.

Aufbau Service Data Object (Quelle: EtherCAT Technologie Group)



SDO Header Word and Command Byte (expedited transfer)

CANopen Header	Number	Unsigned9	0x00
	Reserved	Unsigned3	0x00
	Service	Unsigned4	0x02: SDO Request
SDO	Size Indicator	Unsigned1	0x01: size of Data in Data Set Size specified
	Transfer Type	Unsigned1	0x01: Expedited transfer
	Data Set Size	Unsigned2	0x00: 4 Octet Data 0x01: 3 Octet Data 0x02: 2 Octet Data 0x03: 1 Octet Data
	Complete Access	Unsigned1	0x00: entry addressed with index and subindex will be downloaded 0x01: complete object will be uploaded, subindex shall be zero (subindex 0 included) or one (subindex 0 excluded)
	Command Specifier	Unsigned3	0x01: Download Request
	Index	WORD	Index of the Object
	Subindex	BYTE	Subindex of the Object, shall be zero or one if Complete Access = 0x01
	Data	BYTE[4]	Data of the Object,

Beispiel

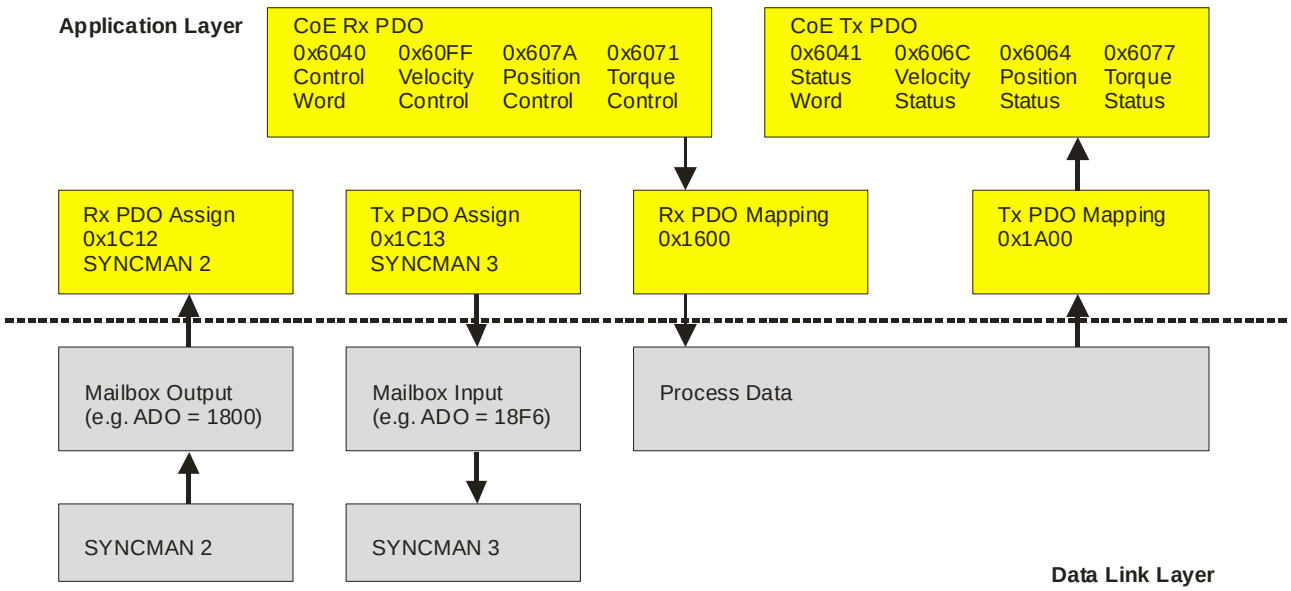
```
|NumServ|Cmd|Index  |SubIndex
                    |Data
|       |   |       |   |               |
 00  20  2F  12  1C  00  00  00  00  00 <- COE Cmd0
```

Der EtherCAT Master von Sybera bietet 2 Möglichkeiten, um SDOs an die EtherCAT Gerät zu übertragen:

SDO Parametrierung über ECATDEVICE.PAR

Mit der Software ECATVERIFY können SDO Legacy Objekte (max. 64) in der Konfigurationsdatei ECATDEVICE.PAR eingetragen werden. Mit der Interface-Funktion Ecat64PdoAssignment() werden diese SDOs dann an die Geräte übertragen (SDO Download).
ECATVERIFY verfügt über einen implementierten XML-Parser, der es ermöglicht, ESI Dateien (Electronic Sheet Information) mit SDOs in ein natives Format zu konvertieren und in der Parameterdatei ECATDEVICE.PAR zu speichern. Im ESI-Konverter von ECATVERIFY kann die ESI Datei nach Name, Produktcode, Hersteller-ID oder Revisionsnummer durchsucht werden. Es ist auch möglich, die gesamte ESI-Datei in das native Format zu konvertieren.
Der integrierte PDO-Konfigurator ermöglicht zudem einfache Festlegung des PDO-Mapping und erzeugt entsprechende SDOs in der Datei ECATDEVICE.PAR.

Beispiel: PDO Mapping nach DS402



Sybera TechNews

11.4.2018

Beispiel ECATDEVICE.PAR:

```
[SDO]
00 20 2f 13 1c 00 00 00 00 00
00 20 2b 13 1c 01 00 1a 00 00
00 20 2f 13 1c 00 01 00 00 00
00 20 2f 00 1a 00 00 00 00 00
00 20 23 00 1a 01 10 00 41 60
00 20 23 00 1a 02 08 00 61 60
00 20 23 00 1a 03 20 00 64 60
00 20 23 00 1a 04 20 0c 1e 30
00 20 23 00 1a 05 10 00 3f 60
00 20 2f 00 1a 00 05 00 00 00
00 20 2f 12 1c 00 00 00 00 00
00 20 2b 12 1c 01 00 16 00 00
00 20 2f 12 1c 00 01 00 00 00
00 20 2f 00 16 00 00 00 00 00
00 20 23 00 16 01 10 00 40 60
00 20 23 00 16 02 08 00 60 60
00 20 23 00 16 03 20 00 7a 60
00 20 23 00 16 04 20 00 81 60
00 20 23 00 16 05 20 00 83 60
00 20 23 00 16 06 20 00 84 60
00 20 2f 00 16 00 06 00 00 00
00 20 2b 06 30 3d 01 00 00 00
00 20 2b 06 30 38 00 00 00 00
00 20 2b 06 30 18 ff ff 00 00
00 20 2b 06 30 06 01 00 00 00
00 20 23 06 30 07 00 00 02 00
00 20 23 06 30 08 01 00 00 00
00 20 2b 12 30 06 e8 03 00 00
00 20 2b 13 30 06 e8 03 00 00
00 20 2f 60 60 00 08 00 00 00
00 20 23 33 1c 03 90 d0 03 00
```

Sybera TechNews

11.4.2018

SDO Parametrierung über die Mailbox-Kommunikation

Sequentieller SDO Download

Eine flexible SDO Parametrierung bieten die Interface-Funktionen des EtherCAT Masters für die Mailbox-Kommunikation. Dabei können auch variable SDOs (variable Datenlängen) übertragen werden. Folgende Kommandos stehen zur Verfügung:

Write command to mailbox

```
ULONG Result = Ecat(32/64)MailboxWrite(
                                    PSTATION_INFO pStation,
                                    PUCHAR pData,
                                    USHORT DataSize,
                                    UCHAR MailboxType)
```

Read command from mailbox

```
ULONG Result = Ecat(32/64)MailboxRead(
                                    PSTATION_INFO pStation,
                                    PUCHAR pData)
```

Check mailbox for pending response

```
ULONG Result = Ecat(32/64)MailboxCheck(PSTATION_INFO pStation)
```

Beispiel: SDO Download (variable SDO Länge)

```
UCHAR __Sdo1[] = { 0x00, 0x20, 0x31, 0x00, 0x80, 0x01, 0x12, 0x00,
                   0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00, 0x00,
                   0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
                   0x00, 0x00, 0x00, 0x40 };

UCHAR __Sdo2[] = { 0x00, 0x20, 0x31, 0x10, 0x80, 0x01, 0x12, 0x00,
                   0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00, 0x00,
                   0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
                   0x00, 0x00, 0x00, 0x40 };

UCHAR __Sdo3[] = { 0x00, 0x20, 0x31, 0x20, 0x80, 0x01, 0x12, 0x00,
                   0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00, 0x00,
                   0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
                   0x00, 0x00, 0x00, 0x40 };
```


```
__inline BOOLEAN __SdoVariableDownload(
                                    PSTATION_INFO pStation,
                                    PUCHAR pSdo,
                                    USHORT SdoLen)
{
      ULONG Result = -1;

      //Allocate mailbox data
      USHORT MailboxSize = pStation->SyncManList[0].s.Length;
      PUCHAR pMailbox = (PUCHAR)malloc(MailboxSize);
      if (pMailbox)
      {
            //Reset mailbox data
            memset(pMailbox, 0, MailboxSize);
            memcpy(pMailbox, pSdo, SdoLen);

            //Check mailbox for pending response
            //Write COE command from mailbox
            //Read COE command from mailbox
            if (ERROR_SUCCESS == (Result = Ecat64MailboxCheck(pStation)))
            if (ERROR_SUCCESS == (Result = Ecat64MailboxWrite(
                                                pStation,
                                                pMailbox,
                                                SdoLen,
                                                MBX_TYPE_COE)))

            if (ERROR_SUCCESS == (Result = Ecat64MailboxRead(
                                                      pStation,
                                                      pMailbox)))
            {
                  //Check SDO command
                  PSDO_INIT_HDR pSdoInitHdr = (PSDO_INIT_HDR)&pSdo[sizeof(MBX_HDR) +
                  sizeof(COE_HDR)];
                  if (pSdoInitHdr->s.bits.Command != SDO_INIT_DOWNLOAD_RESP)
                        Result = -1;
            }
            //Release mailbox
            free(pMailbox);
      }

      //Something failed
      return (ERROR_SUCCESS == Result) ? TRUE : FALSE;
}


//Write variable SDOs
__SdoVariableDownload(pStation, __Sdo1, sizeof(__Sdo1));
__SdoVariableDownload(pStation, __Sdo2, sizeof(__Sdo2));
__SdoVariableDownload(pStation, __Sdo3, sizeof(__Sdo3));
```

Sybera TechNews

11.4.2018

Paralleler SDO Download

Um die Hochlaufzeit bei einer großen Anzahl von Stationen zu verkürzen, kann für die Parametrierung die parallele Mailbox-Kommunikation genutzt werden. Dabei stehen folgende Kommandos zur Verfügung:

(siehe ECAT64MAILBOXDEF.H)

```
typedef struct _MAILBOX_INFO
{
      UCHAR  Buffer[MAX_ECAT_DATA];
      USHORT Len;
      ULONG  StationIndex;

} MAILBOX_INFO, *PMAILBOX_INFO;
```

Write command to mailbox (parallel)

```
ULONG Result = Ecat(32/64)MailboxWriteAll(
                              PMAILBOX_INFO pInfoList,
                              ULONG InfoNum,
                              UCHAR MailboxType)
```

Read command from mailbox (parallel)

```
ULONG Result = Ecat(32/64)MailboxReadAll(
                              PMAILBOX_INFO pInfoList,
                              ULONG InfoNum)
```

Check mailbox for pending response

This function checks a mailbox for pending response

```
ULONG Result = Ecat(32/64)MailboxCheckAll(
                              PMAILBOX_INFO pInfoList,
                              ULONG InfoNum)
```

Beispiel:Paralleler SDO Download von ECATDEVICE.PAR

```
__inline BOOLEAN __SdoDownloadParallel(
                              PSTATION_INFO pStationList,
                              ULONG SdoIndex)
{
      ULONG Result = -1;
      int InfoNum = 0;

      //Allocate memory for mailbox information list
      PMAILBOX_INFO pMailboxInfo = (PMAILBOX_INFO) malloc(
                                        __StationNum * sizeof(MAILBOX_INFO));
      if(pMailboxInfo)
      {
            //Reset memory
            memset(pMailboxInfo, 0, __StationNum*sizeof(MAILBOX_INFO));

            //Loop throug all stations
            for (int i=0; i<__StationNum; i++)
            {
                  //Check station for SDO list
                  PSTATION_INFO pStation = (PSTATION_INFO)&pStationList[i];
                  pStation->bUpdate = FALSE;

                  if (pStation->SdoNum)
                  if (pStation->SdoNum > SdoIndex)
                  {
                        //Init mailbox information
                        pMailboxInfo[InfoNum].StationIndex = pStation->Index;
                        pMailboxInfo[InfoNum].Len = sizeof(SDO_LEGACY);
                        memcpy((PUCHAR)&pMailboxInfo[InfoNum].Buffer,
                               (PUCHAR)&pStation->SdoList[SdoIndex].bytes,
                               sizeof(SDO_LEGACY));

                        //Increase mail
                        InfoNum++;
                  }
            }

            //Check mailbox information list
            if (InfoNum)
            {
                  //Check mailbox for pending response
                  //Write COE command from mailbox
                  //Read COE command from mailbox
                  if (ERROR_SUCCESS == (Result =
                     Ecat64MailboxCheckAll(pMailboxInfo, InfoNum)))
                  if (ERROR_SUCCESS == (Result =
                     Ecat64MailboxWriteAll(pMailboxInfo, InfoNum, MBX_TYPE_COE)))
                  if (ERROR_SUCCESS == (Result =
                     Ecat64MailboxReadAll( pMailboxInfo, InfoNum)))
                  {
                        //Check SDO response
                        for(int i=0;i<InfoNum;i++)
                        {
```


```
                                //Check SDO command
                                PSDO_INIT_HDR pSdoInitHdr =
   (PSDO_INIT_HDR)&pMailboxInfo[i].Buffer[sizeof(MBX_HDR) + sizeof(COE_HDR)];
                                if (pSdoInitHdr->s.bits.Command !=
                                    SDO_INIT_DOWNLOAD_RESP)
                                {
                                      Result = -1;
                                      break;
                                }
                          }
                    }
              }

              //Release TX data
              free(pMailboxInfo);
   }

   //Something failed
   return (ERROR_SUCCESS == Result) ? TRUE : FALSE;
}



void SdoDownload(void)
{
      int i=0;
      while (__SdoDownloadParallel(__pUserList, i++));
}
```

Der Sybera EtherCAT Master erlaubt auch eine Kombination der Parametrierung mit ECATDEVICE.PAR (Ecat64PdoAssignment) und den Mailbox-Funktionen. Neben dem CoE Protokoll können mit der Mailbox-Kommunikation auch andere Protokoll-Daten (z.B. EoE, FoE) an die Geräte übertragen werden.