文書番号：

TOPPERS/OSEK
System Generation
通信ミドルウェア対応 OIL 拡張仕様書

Ver.1.01

2007/04/16

株式会社ヴィッツ　開発第３部

TOPPERS/OSEK System Generation
Toyohashi Open Platform for Embedded Real-Time Systems/
OSEK System Generation

＜目次＞

1. 概要

本仕様書は、CAN 通信ミドルウェア（OS 対応／非 OS 対応）、LIN 通信ミドルウェア（OS 対応／非 OS 対応）における「OSEK/VDX System Generation OIL:OSEK Implementation Language Ver2.5」仕様に準拠した OSEK Implementation Language（以降、OIL とする）仕様を規定したものである。

1.1. 本書の位置付け

本仕様書では、CAN 通信ミドルウェア（OS 対応／非 OS 対応）、LIN 通信ミドルウェア（OS 対応／非 OS 対応）の OIL 仕様を規定するものであり、TOPPRES/OSEK における OIL 仕様に関しては別紙「SG 取扱説明書」を参照のこと。また、OIL 仕様自体については、「OSEK/VDX System Generation OIL:OSEK Implementation Language Ver2.5」を参照のこと。

1.2. 関連文書

本仕様書を策定するにあたり以下の資料を参照した。

- OSEK/VDX Communication Version 3.0.3
- OSEK/VDX NetworkManagement Concept and Application Programming Interface Version 2.5.2
- TOPPERS_OSEK_COM 外部仕様書 v.1.0.3
- TOPPERS_OSEK_NM 外部仕様書 v.1. 0.0
- TLIN レファレンス・コード ユーザ・ガイド v.1.1
- TLIN アプリケーションガイド v.1.0
- TLIN レファレンス・コード プログラム・ビルド・ガイド v.1.0

2. モジュール

通信ミドルウェアは、次のモジュールから構成されている。

- CAN 通信ミドルウェア（OS 対応／非 OS 対応）
  - COM モジュール
  - NM モジュール
  - DRV モジュール
  - 共通モジュール
- LIN 通信ミドルウェア（OS 対応／非 OS 対応）
  - LINMW モジュール

CAN 通信ミドルウェア（OS 対応／非 OS 対応）／LIN 通信ミドルウェア（OS 対応／非 OS 対応）の各モジュールに規定されたデータ出力を満たすため、次の OIL オブジェクトを定義する。また、既存の OIL オブジェクトに対しても前述出力内容を満たすため OIL 仕様の追加定義を行う。

2.1. CAN 通信ミドルウェア（OS 対応／非 OS 対応）

CAN 通信ミドルウェア（OS 対応／非 OS 対応）の OIL オブジェクト概要について以下に記載する。

2.1.1. COM（OS 対応）

○新規 OIL オブジェクト定義

- CANCOMMON … 複数ネットワーク対応用に CAN ドライバ共通情報を設定
- CANDRV … CAN ドライバ情報の設定

○既存 OIL オブジェクトへの追加定義

- MESSAGE … 属性の型を変更
- NETWORKMESSAGE … 属性の型を変更
- IPDU … 属性の型を変更

2.1.2. NM（OS 対応）

○新規 OIL オブジェクト定義

- NMNODE … ネットワーク内のノードを定義

○既存 OIL オブジェクトへの追加定義

- NM … CAN 通信ミドルウェアに対応した情報を設定

2.1.3. COM（非 OS 対応）

○新規 OIL オブジェクト定義

- CANCOM … CAN ネットワーク内のノードを定義

○既存 OIL オブジェクトへの追加定義

- MESSAGE … CAN 通信ミドルウェアに対応した情報を設定
- NETWORKMESSAGE … CAN 通信ミドルウェアに対応した情報を設定
- IPDU … CAN 通信ミドルウェアに対応した情報を設定

2.1.4. NM（非 OS 対応：Indirect）

○新規 OIL オブジェクト定義

- CANNM … CAN 通信ミドルウェア（Indirect）に対応した情報を設定

2.1.5. NM（非 OS 対応：Direct）

○新規 OIL オブジェクト定義

- CANNM … CAN 通信ミドルウェア（Direct）に対応した情報を設定

2.1.6. CANDRV（非 OS 対応）

○新規 OIL オブジェクト定義

- CANDRV … CAN ドライバ情報を設定

2.1.7. 共通モジュール（非 OS 対応）

○新規 OIL オブジェクト定義

- CANCOMMON … COM オブジェクト、CANDRV オブジェクトの共通情報を設定
- CANNODE … COM オブジェクト、NM オブジェクトの共通情報を設定

2.2. LIN 通信ミドルウェア（OS 対応／非 OS 対応）

LIN 通信ミドルウェアでは、OS 対応／非 OS 対応どちらにも対応可能な OIL オブジェクト構成となっている。OIL オブジェクト概要について以下に記載する。

2.2.1. LINMW（OS 対応／非 OS 対応）

○新規 OIL オブジェクト定義

- LINSIGNAL … シグナル(送受信データ単位)の情報を設定
- LINFRAME … シグナルを格納するフレームの情報を設定
- LINSCHEDULE … スケジュールテーブルの情報を設定
- LINNNODE … ノードに関する情報を設定

3. オブジェクト

CAN 通信ミドルウェア（OS 対応／非 OS 対応）／LIN 通信ミドルウェア（OS 対応／非 OS 対応）の各 OIL オブジェクトに規定した属性について記載する。

3.1. COM モジュール（OS 対応）

3.1.1. COM

本オブジェクトでは、CAN 通信ミドルウェア（OS 対応）における COM 通信機能を実現するために、既存の COM オブジェクトを用いる。COM オブジェクトのインプリメンテーション部を下記に記載する。尚、各属性の設定値については、「OSEK/VDX Implementation Language (OIL) Ver2.5」参照のこと。

```
COM {
        FLOAT           COMTIMEBASE                 = 0.001;        // 未サポート
        BOOLEAN         COMERRORHOOK                = FALSE;
        BOOLEAN         COMUSEGETSERVICEID          = FALSE;
        BOOLEAN         COMUSEPARAMETERACCESS       = FALSE;
        BOOLEAN         COMSTARTCOMEXTENSION        = FALSE;
        SYMBOLNAME      COMAPPMODE[];
        ENUM [
                COMSTANDARD,
                COMEXTENDED
        ] COMSTATUS     = COMSTANDARD;
        STRING          USE [];                                     // 未サポート
};
```

3.1.1.1. 記述例

```
COM com {
        COMERRORHOOK                = FALSE;
        COMUSEGETSERVICEID          = FALSE;
        COMUSEPARAMETERACCESS       = FALSE;
        COMSTARTCOMEXTENSION        = FALSE;
        COMAPPMODE                  = comappmode_a;
        COMAPPMODE                  = comappmode_b;
        COMSTATUS                   = COMSTANDARD;
};
```

3.1.2. MESSAGE

本オブジェクトでは、CAN 通信ミドルウェア（OS 対応）における COM 通信機能を実現するために、既存の MESSAGE オブジェクトに対して属性の型変更を行った。

MESSAGE オブジェクトのインプリメンテーション部を下記に記載する。

※変更属性については水色の枠組みで記載する。

```
MESSAGE {
    ENUM {
    SEND_STATIC_INTERNAL {
        DATATYPE    CDATATYPE    = NO_DEFAULT;
    },
    SEND_STATIC_EXTERNAL {
        DATATYPE    CDATATYPE    = NO_DEFAULT;
        ENUM  WITH_AUTO [
                TRIGGERD,
                PENDING
        ] TRANSFERPROPERTY        = AUTO;
        ENUM [
                ALWAYS,
                NEVER,
                MASKEDNEWEQUALSX {
                        UINT64          MASK    = NO_DEFAULT;
                        UINT64          X       = NO_DEFAULT;
                },
                MASKEDNEWDIFFERSX {
                        UINT64          MASK    = NO_DEFAULT;
                        UINT64          X       = NO_DEFAULT;
                },
                NEWISEQUAL,
                NEWISDIFFERENT,
                MASKEDNEWEQUALSMASKEDOLD {
                        UINT64          MASK    = NO_DEFAULT;
                },
                MASKEDNEWDIFFERSMASKEDOLD {
                        UINT64          MASK    = NO_DEFAULT;
                }.
                NEWISWITHIN {
```

```
                UINT64          MIN     = NO_DEFAULT;
                UINT64          MAX     = NO_DEFAULT;
            },
            NEWISOUTSIDE {
                UINT64          MIN     = NO_DEFAULT;
                UINT64          MAX     = NO_DEFAULT;
            },
            NEWISGREATER,
            NEWISLESSOREQUAL,
            NEWISLESS,
            NEWISGREATEROREQUAL,
            ONEEVERYN {
                UINT64          PERIOD          = NO_DEFAULT;
                UINT64          OFFSET          = NO_DEFAULT;
        ] FILTER        = ALWAYS;
        SYMBOLNAME      NETWORKORDERCALLOUT                 = "";
        SYMBOLNAME      CPUORDERCALLOUT                     = "";
        UINT64      WITH_AUTO   INITIALVALUE   = AUTO;
        NETWORKMESSAGE_TYPE     NETWORKMESSAGE;
    },
    SEND_DYNAMIC_EXTERNAL {
        ENUM  WITH_AUTO [
            TRIGGERD,
            PENDING
        ] TRANSFERPROPERTY      = AUTO;
        SYMBOLNAME      NETWORKORDERCALLOUT                 = "";
        SYMBOLNAME      CPUORDERCALLOUT                     = "";
        UINT64      WITH_AUTO   INITIALVALUE   = AUTO;
        NETWORKMESSAGE_TYPE     NETWORKMESSAGE;
    },
    SEND_ZERO_INTERNAL {
    },
    SEND_ZERO_EXTERNAL {
        SYMBOLNAME      NETWORKORDERCALLOUT                 = "";
        SYMBOLNAME      CPUORDERCALLOUT                     = "";
```

```
        NETWORKMESSAGE_TYPE     NETWORKMESSAGE;
    },
    RECEIVE_ZERO_INTERNAL {
        MESSAGE_TYPE        SENDINGMESSAGE;
    },
    RECEIVE_ZERO_EXTERNAL {
        SYMBOLNAME          NETWORKORDERCALLOUT                 = "";
        SYMBOLNAME          CPUORDERCALLOUT                     = "";
        NETWORKMESSAGE_TYPE     NETWORKMESSAGE;
    },
    RECEIVE_UNQUEUED_INTERNAL {
        MESSAGE_TYPE        SENDINGMESSAGE;
        ENUM [
                ALWAYS,
                NEVER,
                MASKEDNEWEQUALSX {
                        UINT64          MASK    = NO_DEFAULT;
                        UINT64          X       = NO_DEFAULT;
                },
                MASKEDNEWDIFFERSX {
                        UINT64          MASK    = NO_DEFAULT;
                        UINT64          X       = NO_DEFAULT;
                },
                NEWISEQUAL,
                NEWISDIFFERENT,
                MASKEDNEWEQUALSMASKEDOLD {
                        UINT64          MASK    = NO_DEFAULT;
                },
                MASKEDNEWDIFFERSMASKEDOLD {
                        UINT64          MASK    = NO_DEFAULT;
                }.
                NEWISWITHIN {
                        UINT64          MIN     = NO_DEFAULT;
                        UINT64          MAX     = NO_DEFAULT;
                },
```

```
            NEWISOUTSIDE {
                UINT64          MIN     = NO_DEFAULT;
                UINT64          MAX     = NO_DEFAULT;
            },
            NEWISGREATER,
            NEWISLESSOREQUAL,
            NEWISLESS,
            NEWISGREATEROREQUAL,
            ONEEVERYN {
                UINT64          PERIOD          = NO_DEFAULT;
                UINT64          OFFSET          = NO_DEFAULT;
        ] FILTER        = ALWAYS;
        UINT64          INITIALVALUE    = 0;
    },
    RECEIVE_QUEUED_INTERNAL {
        MESSAGE_TYPE        SENDINGMESSAGE;
        ENUM [
            ALWAYS,
            NEVER,
            MASKEDNEWEQUALSX {
                UINT64          MASK    = NO_DEFAULT;
                UINT64          X       = NO_DEFAULT;
            },
            MASKEDNEWDIFFERSX {
                UINT64          MASK    = NO_DEFAULT;
                UINT64          X       = NO_DEFAULT;
            },
            NEWISEQUAL,
            NEWISDIFFERENT,
            MASKEDNEWEQUALSMASKEDOLD {
                UINT64          MASK    = NO_DEFAULT;
            },
            MASKEDNEWDIFFERSMASKEDOLD {
                UINT64          MASK    = NO_DEFAULT;
            }.
```

```
                NEWISWITHIN {
                        UINT64          MIN     = NO_DEFAULT;
                        UINT64          MAX     = NO_DEFAULT;
                },
                NEWISOUTSIDE {
                        UINT64          MIN     = NO_DEFAULT;
                        UINT64          MAX     = NO_DEFAULT;
                },
                NEWISGREATER,
                NEWISLESSOREQUAL,
                NEWISLESS,
                NEWISGREATEROREQUAL,
                ONEEVERYN {
                        UINT64          PERIOD          = NO_DEFAULT;
                        UINT64          OFFSET          = NO_DEFAULT;
        ] FILTER        = ALWAYS;
        UINT64          INITIALVALUE    = 0;
        UINT32          QUEUESIZE       = NO_DEFAULT;
    },
    RECEIVE_UNQUEUED_EXTERNAL {
        DATATYPE    CDATATYPE    = NO_DEFAULT;
        ENUM [
                ALWAYS,
                NEVER,
                MASKEDNEWEQUALSX {
                        UINT64          MASK    = NO_DEFAULT;
                        UINT64          X       = NO_DEFAULT;
                },
                MASKEDNEWDIFFERSX {
                        UINT64          MASK    = NO_DEFAULT;
                        UINT64          X       = NO_DEFAULT;
                },
                NEWISEQUAL,
                NEWISDIFFERENT,
                MASKEDNEWEQUALSMASKEDOLD {
```

```
                UINT64          MASK    = NO_DEFAULT;
            },
            MASKEDNEWDIFFERSMASKEDOLD {
                UINT64          MASK    = NO_DEFAULT;
            },
            NEWISWITHIN {
                UINT64          MIN     = NO_DEFAULT;
                UINT64          MAX     = NO_DEFAULT;
            },
            NEWISOUTSIDE {
                UINT64          MIN     = NO_DEFAULT;
                UINT64          MAX     = NO_DEFAULT;
            },
            NEWISGREATER,
            NEWISLESSOREQUAL,
            NEWISLESS,
            NEWISGREATEROREQUAL,
            ONEEVERYN {
                UINT64          PERIOD          = NO_DEFAULT;
                UINT64          OFFSET          = NO_DEFAULT;
        ] FILTER        = ALWAYS;
        BOOLEAN [
            TRUE {
                MESSAGE_TYPE            RECEIVEMESSAGE;
            },
            FALSE {
                SYMBOLNAME              NETWORKORDERCALLOUT     = "";
                SYMBOLNAME              CPUORDERCALLOUT         = "";
                NETWORKMESSAGE_TYPE     NETWORKMESSAGE;
            }
        ] LINK  = NO_DEFAULT;
        UINT64          WITH_AUTO       INITIALVALUE    = AUTO;
    },
    RECEIVE_QUEUED_EXTERNAL {
        DATATYPE    CDATATYPE    = NO_DEFAULT;
```

```
UINT32          QUEUESIZE       = NO_DEFAULT;
ENUM [
        ALWAYS,
        NEVER,
        MASKEDNEWEQUALSX {
                UINT64          MASK    = NO_DEFAULT;
                UINT64          X       = NO_DEFAULT;
        },
        MASKEDNEWDIFFERSX {
                UINT64          MASK    = NO_DEFAULT;
                UINT64          X       = NO_DEFAULT;
        },
        NEWISEQUAL,
        NEWISDIFFERENT,
        MASKEDNEWEQUALSMASKEDOLD {
                UINT64          MASK    = NO_DEFAULT;
        },
        MASKEDNEWDIFFERSMASKEDOLD {
                UINT64          MASK    = NO_DEFAULT;
        }.
        NEWISWITHIN {
                UINT64          MIN     = NO_DEFAULT;
                UINT64          MAX     = NO_DEFAULT;
        },
        NEWISOUTSIDE {
                UINT64          MIN     = NO_DEFAULT;
                UINT64          MAX     = NO_DEFAULT;
        },
        NEWISGREATER,
        NEWISLESSOREQUAL,
        NEWISLESS,
        NEWISGREATEROREQUAL,
        ONEEVERYN {
                UINT64          PERIOD          = NO_DEFAULT;
                UINT64          OFFSET          = NO_DEFAULT;
```

```
    ] FILTER        = ALWAYS;
    BOOLEAN [
        TRUE {
            MESSAGE_TYPE        RECEIVEMESSAGE;
        },
        FALSE {
            SYMBOLNAME          NETWORKORDERCALLOUT   = "";
            SYMBOLNAME          CPUORDERCALLOUT       = "";
            NETWORKMESSAGE_TYPE     NETWORKMESSAGE;
        }
    ] LINK  = NO_DEFAULT;
    UINT64      WITH_AUTO     INITIALVALUE   = AUTO;
},
RECEIVE_DYNAMIC_EXTERNAL {
    BOOLEAN [
        TRUE {
            MESSAGE_TYPE        RECEIVEMESSAGE;
        },
        FALSE {
            SYMBOLNAME          NETWORKORDERCALLOUT   = "";
            SYMBOLNAME          CPUORDERCALLOUT       = "";
            NETWORKMESSAGE_TYPE     NETWORKMESSAGE;
        }
    ] LINK  = NO_DEFAULT;
    UINT64      WITH_AUTO     INITIALVALUE   = AUTO;
},
RECEIVE_ZERO_SENDERS {                                          // 未サポート
    DATATYPE    CDATATYPE     = NO_DEFAULT;                     // 未サポート
    UINT64      INITIALVALUE  = 0;                              // 未サポート
}                                                               // 未サポート
] MESSAGEPROPERTY         = NO_DEFAULT;
ENUM [
NONE,
ACTIVATETASK {
    TASK_TYPE   TASK;
```

```
    },
    SETEVENT {
        TASK_TYPE    TASK;
        EVENT_TYPE   EVENT;
    },
    COMCALLBACK {
        SYMBOLNAME          CALLBACKROUTINENAME          = NO_DEFAULT;
        MESSAGE_TYPE        MESSAGE;
    },
    FLAG {
        SYMBOLNAME          FLAGNAME          = NO_DEFAULT;
    },
    INMCALLBACK {                                                    // 未サポート
        SYMBOLNAME          CALLBACKROUTINENAME          = NO_DEFAULT;    // 未サポート
        UINT32              MONITOREDIPDU                = NO_DEFAULT;    // 未サポート
    }                                                                // 未サポート
    ] NOTIFICATION                  = NONE;
    ENUM [
    NONE,
    ACTIVATETASK {
        TASK_TYPE    TASK;
    },
    SETEVENT {
        TASK_TYPE    TASK;
        EVENT_TYPE   EVENT;
    },
    COMCALLBACK {
        SYMBOLNAME          CALLBACKROUTINENAME          = NO_DEFAULT;
        MESSAGE_TYPE        MESSAGE;
    },
    FLAG {
        SYMBOLNAME          FLAGNAME          = NO_DEFAULT;
    },
    INMCALLBACK {                                                    // 未サポート
        SYMBOLNAME          CALLBACKROUTINENAME          = NO_DEFAULT;    // 未サポート
```

```
        UINT32              MONITOREDIPDU                   = NO_DEFAULT;       // 未サポート
    }                                                                           // 未サポート
    ] NOTIFICATIONERROR         = NONE;
};
```

3.1.2.1. CDATATYPE

送受信のデータ型を指定する。型名に適していない場合はエラーを出力する。

3.1.2.2. NETWORKORDERCALLOUT

NETWORKORDERCALLOUT の関数名を指定する。関数名に適していない場合はエラーを出力する。

3.1.2.3. CPUORDERCALOUT

CPUORDERCALLOUT の関数名を指定する。関数名に適していない場合はエラーを出力する。

3.1.2.4. CALLBACKROUTINENAME

CALLBACKROUTINENAME の関数名を指定する。関数名に適していない場合はエラーを出力する。

3.1.2.5. FLAGNAME

FLAGNAME を指定する。関数名に適していない場合は、エラーを出力する。

3.1.2.6. 記述例

```
MESSAGE msg1 {
    MESSAGEPROPERTY         RECEIVE_UNQUEUED_EXTERNAL {
        CDATATYPE           = “UINT32”;
        FILTER      = NEVER;
        LINK        = FALSE {
                    NETWORKORDERCALLOUT         = “networkcallout1”;
                    CPUORDERCALLOUT             = “cpuordercallout1”;
                    NETWORKMESSAGE              = netmsg1;
        };
        INITIALVALUE            = 0;
    };
    NOTIFICATION    = ACTIVATETASK {
                    TASK        = task1;
    };
    NOTIFICATIONERROR       = COMCALLBACK {
```

```
                CALLBACKROUTINENAME             = “callback1”;
        };
};
```

3.1.3. NETWORKMESSAGE

本オブジェクトでは、CAN 通信ミドルウェア（OS 対応）における COM 通信機能を実現するために、既存の NETWORKMESSAGE を用いる。尚、オブジェクトのインプリメンテーション部は「OSEK/VDX Implementation Language (OIL) Ver2.5」参照のこと。

```
NETWORKMESSAGE {
        IPDU_TYPE           IPDU;
        ENUM [
                STATIC {
                        UINT32          SIZEINBITS                      = NO_DEFAULT;
                        ENUM [
                                LITTLEENDIAN,
                                BIGENDIAN
                        ] BITORDERING                                   = NO_DEFAULT;
                        UINT32          BITPOSITION                     = NO_DEFAULT;
                        ENUM [
                                UNSIGNEDINTEGER,
                                BYTEARRAY
                        ] DATAINTERPRETATION                            = NO_DEFAULT;
                        UINT64 WITH_AUTO  INITIALVALUE                  = AUTO;
                        ENUM [
                                SENT {
                                        ENUM WITH_AUTO [
                                                TRIGGERED,
                                                PENDING
                                        ] TRANSFERPROPERTY              = AUTO;
                                },
                                RECEIVED {
                                }
                        ] DIRECTION             = NO_DEFAULT;
                },
                DYNAMIC {
```

```
                UINT32 MAXIMUMSIZEINBITS                = NO_DEFAULT;
                ENUM [
                        LITTLEENDIAN,
                        BIGENDIAN
                ] BITORDERING                           = NO_DEFAULT;
                UINT32;         BITPOSITION             = NO_DEFAULT
                UINT64 WITH_AUTO        INITIALVALUE    = AUTO;
                ENUM [
                        SENT {
                                ENUM WITH_AUTO [
                                        TRIGGERED,
                                        PENDING
                                ] TRANSFERPROPERTY = AUTO;
                        },
                        RECEIVED {
                        }
                ] DIRECTION             = NO_DEFAULT;
        },
        ZERO {
        }
    ] MESSAGEPROPERTY               = NO_DEFAULT;
};
```

3.1.3.1. 記述例

```
NETWORKMESSAGE netmsg1 {
        IPDU    = ipdu1;
        MESSAGEPROPERTY         = STATIC {
                SIZEINBITS      = 8;
                BITORDERING     = LITTLEENDIAN;
                BITPOSITION     = 0;
                DATAINTERPRETATION      = UNSIGNEDINTEGER;
                INITIALVALUE    = 0;
                DIRECTION       = SENT {
                        TRANSFERPROPERTY        = TRIGGERED;
                };
```

```
        };
};
```

3.1.4. IPDU

本オブジェクトでは、OS 対応 CAN 通信における COM 通信機能を実現するために、既存の IPDU オブジェクトに対して属性の型変更を行った。

IPDU オブジェクトのインプリメンテーション部を下記に記載する。

※変更属性については水色の枠組みで記載する。

```
IPDU {
    UINT32          SIZEINBITS      = NO_DEFAULT;
    ENUM [
        SENT {
            ENUM [
                DIRECT {
                    UINT64          MINIMUMDELAYTIME = 0;
                },
                PERIODIC {
                    UINT64          TIMEPERIOD          = NO_DEFAULT;
                    UINT64          WITH_AUTO   TIMEOFFSET  = NO_DEFAULT;
                },
                MIXED {
                    UINT64          TIMEPERIOD  = NO_DEFAULT;
                    UINT64          WITH_AUTO   TIMEOFFSET  = NO_DEFAULT;
                    UINT64          MINIMUMDELAYTIME = 0;
                }
            ] TRANSMISSIONMODE      = NO_DEFAULT;
            UINT64          TIMEOUT     = 0;
        }.
        RECEIVED {
            UINT64          TIMEOUT     = 0;
            UINT64          WITH_AUTO   FIRSTTIMEOUT        = AUTO;
        },
    ] IPDUPROPERTY      = NO_DEFAUTL;
    SYMBOLNAME          IPDUCALLOUT         = "";
    ENUM [
        CAN,
```

```
            LIN                                                    // 未サポート
        ] LAYERUSED         = NO_DEFAULT;
        UINT32          NETWORKID           = NO_DEFAULT;
};
```

3.1.4.1. LAYERUSED

使用するレイヤーを指定する。

3.1.4.2. NETWORKID

NETWORKID を指定する。LAYERUSED が CAN の場合は CANID、LIN の場合は LINID となる。

3.1.4.3. 記述例

```
IPDU ipdu1 {
        SIZEINBITS      = 64;
        IPDUPROPERTY    = SENT {
                TRANSMISSIONMODE        = DIRECT {
                        MINIMUMDELAYTIME        = 10;
                };
                TIMEOUT         = 0;
        };
        IPDUCALLOUT     = “ipducall”;
        LAYERUSED       = CAN;
        NETWORKID       = 0x0501;
};
```

3.2. NM モジュール（OS 対応）

3.2.1. NM

本オブジェクトでは、CAN 通信ミドルウェア（OS 対応）における NM 機能を実現するために、既存の NM オブジェクトに対して属性の追加を行った。

NM オブジェクトのインプリメンテーション部を下記に記載する。

```
NM {
        NMNODE_TYPE                 NMNODE[];
        NMNODE_TYPE                 MYNODE;
        UINT32                      NETWORKID = NO_DEFALUT;
        ENUM [
            DIRECT {
```

```
UINT32  [70..110]               TTYP;
UINT32  [220..284]              TMAX;
UINT32  [1..65535]              TERROR;
UINT32  [1..65535]              TWAITBUSSLEEP;
UINT32  [1..65535]              TTX;
UINT32  [1..255]                NMRXLIMIT;
UINT32  [1..255]                NMTXLIMIT;
UINT32  [0x01, 0x02, 0x04, 0x08, 0x10, 0x20, 0x40, 0x80]
                        OPCODE_ALIVE = NO_DEFALUT;
UINT32  [0x01, 0x02, 0x04, 0x08, 0x10, 0x20, 0x40, 0x80]
                        OPCODE_RING = NO_DEFALUT;
UINT32  [0x01, 0x02, 0x04, 0x08, 0x10, 0x20, 0x40, 0x80]
                        OPCODE_LIMPHONE = NO_DEFALUT;
UINT32  [0x01, 0x02, 0x04, 0x08, 0x10, 0x20, 0x40, 0x80]
                        OPCODE_SLEEPIND = NO_DEFALUT;
UINT32  [0x01, 0x02, 0x04, 0x08, 0x10, 0x20, 0x40, 0x80]
                        OPCODE_SLEEPACK = NO_DEFALUT;
UINT32  [0x00000700, 0x00000780, 0x000007C0, 0x000007E0, 0x000007F0, 0x000007F8]
                        NMMSG_WINDOW_MASK = NO_DEFALUT;
UINT32  [0x00000000..0x000007F8]
                        NMMSG_ID_BASE = NO_DEFALUT;
BOOLEAN                         NMSLEEPIND ;
UINT32  [2..64]         NMNUMRCVQUE = NO_DEFALUT;
BOOLEAN [
        TRUE {
                BOOLEAN         PRESENT_NETWORK;
                BOOLEAN         OPERATION_MODE_IF;
                BOOLEAN         NMACTIVE;
                BOOLEAN         NMON;
                BOOLEAN         NMLIMPHOME;
                BOOLEAN         NMBUSSLEEP;
                BOOLEAN         NMTWBS_NOR_LIMP;
                BOOLEAN         RING_DATA_ALLOWED;
                BOOLEAN         GMODE_BUSSLEEP;
        },
```

```
        FALSE
    ] NMSMASK;
    UINT32  [0..48]     TNUM_RINGDATA = NO_DEFALUT;
    ENUM [
        ACTIVATETASK {
            TASK_TYPE       TASK;
        },
        SETEVENT {
            EVENT_TYPE      EVENT;
            TASK_TYPE       TASK;
        },
        NMCALLBACK {
            STRING          NMCALLBACK;
        },
        NONE
    ] NMMSGNOTIFICATION;
    ENUM [MAX_NM, MIN_NM]       NMSCALEABILITY;
    },
    INDIRECT {
        BOOLEAN [
            TRUE {
                UINT32      THRESHOLD = NO_DEFALUT;
                UINT32      DELTAINC = NO_DEFALUT;
                UINT32      DELTADEC = NO_DEFALUT;
            },
            FALSE
        ] NMEXTEND;
        UINT32              TERROR = NO_DEFALUT;
        UINT32              TOB = NO_DEFALUT;
        BOOLEAN [
            TRUE {
                BOOLEAN     OPERATION_MODE_IF;
                BOOLEAN     NMON;
                BOOLEAN     NMLIMPHOME;
                BOOLEAN     NMBUSSLEEP;
```

```
                        BOOLEAN     NMWAIT_BUSSLEEP;
                    },
                    FALSE
                ] NMSMASK;
                BOOLEAN                 NMDYNASTATESMONI;
                BOOLEAN                 NMSTATICSTATESMONI;
                BOOLEAN [
                    TRUE {
                        UINT32      TWAITBUSSLEEP = NO_DEFALUT;
                    },
                    FALSE
                ] NMBUSSLEEP
        }
    ] NMTYPE;
    BOOLEAN         BUSINIT _ROUTINE
    BOOLEAN         BUSSHUTDOWN _ROUTINE
    BOOLEAN         BUSAWAKE _ROUTINE
    BOOLEAN         BUSSLEEP _ROUTINE
    BOOLEAN         BUSRESTART _ROUTINE
    ENUM [
        ACTIVATETASK {
                TASK_TYPE           TASK;
        },
        SETEVENT {
                EVENT_TYPE          EVENT;
                TASK_TYPE           TASK;
        },
        NMCALLBACK {
                STRING              NMCALLBACK;
        },
        NONE
    ] NMCMASKNOTIFICATION;
    ENUM [
        ACTIVATETASK {
                TASK_TYPE           TASK;
```

```
                },
                SETEVENT {
                        EVENT_TYPE          EVENT;
                        TASK_TYPE           TASK;
                },
                NMCALLBACK {
                        STRING              NMCALLBACK;
                },
                NONE
        ] NMSMASKNOTIFICATION;
};
```

3.2.1.1. NMNODE

ネットワークに参加しているノードのリストを指定する。

3.2.1.2. MYNODE

ネットワークに参加しているノードの中から自分のノードを指定する。
NMNODE でリストされていないノードを指定した場合及び、NMNODE を設定した数が(2^(NMMSG_WINDOW_MASK 下位ビットで 0 が連続する数))より多い場合はエラーとする。

3.2.1.3. NETWORKID

ネットワークの ID を指定する。

3.2.1.4. NMTYPE

使用するNMの監視方法を指定する。[1]

- DIRECT　：ダイレクト NM
- INDIRECT：インダイレクト NM

3.2.1.5. TTYP

トークンを保持できる時間を７０から１１０で指定する。

3.2.1.6. TMAX

バス上にメッセージが流れていないことを認知するまでの時間を２２０から２８４で指定する。

[1]制限事項：INDIRECT設定不可

3.2.1.7. TERROR

故障状態のノードが LimpHome メッセージを送信する間隔を１から６５５３５で指定する。

3.2.1.8. TWAITBUSSLEEP

BusSleep に入る前に待機する時間を１から６５５３５で指定する。

3.2.1.9. TTX

送信要求が拒否された場合に再送するまでの間隔を１から６５５３５で指定する。

3.2.1.10. NMRXLIMIT

受信不能カウンタ閾値を１から２５５で指定する。

3.2.1.11. NMTXLIMIT

送信不能カウンタ閾値を１から２５５で指定する。

3.2.1.12. OPCODE_ALIVE

OPCODE の Alive メッセージビットを 0x01, 0x02, 0x04, 0x08, 0x10, 0x20, 0x40, 0x80 のいずかで指定する。
OPCODE_ALIVE,OPCODE_RING,OPCODE、LIMPHONE, OPCODE_SLEEPIND,
OPCODE_SLEEPACK に全て異なる値を設定されていない場合はエラーとする。

3.2.1.13. OPCODE_RING

OPCODE の Ring メッセージビットを 0x01, 0x02, 0x04, 0x08, 0x10, 0x20, 0x40, 0x80 のいずかで指定する。
OPCODE_ALIVE,OPCODE_RING,OPCODE、LIMPHONE, OPCODE_SLEEPIND,
OPCODE_SLEEPACK に全て異なる値を設定されていない場合はエラーとする。

3.2.1.14. OPCODE_LIMPHONE

OPCODE の LimpHome メッセージビットを 0x01, 0x02, 0x04, 0x08, 0x10, 0x20, 0x40, 0x80 のいずかで指定する。
OPCODE_ALIVE,OPCODE_RING,OPCODE、LIMPHONE, OPCODE_SLEEPIND,
OPCODE_SLEEPACK に全て異なる値を設定されていない場合はエラーとする。

3.2.1.15. OPCODE_SLEEPIND

OPCODE の Sleep ind ビットを 0x01, 0x02, 0x04, 0x08, 0x10, 0x20, 0x40, 0x80 のいずかで指定する。
OPCODE_ALIVE,OPCODE_RING,OPCODE、LIMPHONE, OPCODE_SLEEPIND,
OPCODE_SLEEPACK に全て異なる値を設定されていない場合はエラーとする。

3.2.1.16. OPCODE_SLEEPACK

OPCODE の Sleep ack ビットを 0x01, 0x02, 0x04, 0x08, 0x10, 0x20, 0x40, 0x80 のいずかで指定する。
OPCODE_ALIVE,OPCODE_RING,OPCODE、LIMPHONE, OPCODE_SLEEPIND,
OPCODE_SLEEPACK に全て異なる値を設定されていない場合はエラーとする。

3.2.1.17. NMMSG_WINDOW_MASK

WindowMask を 0x00000700, 0x00000780, 0x000007C0, 0x000007E0, 0x000007F0, 0x000007F8 のいずかで指定する。

3.2.1.18. NMMSG_ID_BASE

IDBase を 0x00000000 から 0x000007F8 で指定する。
NMMSG_WINDOW_MASK で 1 でないビットが 1 の場合はエラーとする。

3.2.1.19. NMSLEEPIND

Sleep/Awake 要求メッセージ受信時アプリケーション通知有無を指定する。

- TRUE：通知する
- FALSE：通知しない

3.2.1.20. NMNUMRCVQUE

受信キューの数を２から６４で指定する。

3.2.1.21. NMSMASK

ステータスマスクを使用するか指定する。

- TRUE　：使用する
- FALSE：使用しない

3.2.1.22. PRESENT_NETWORK

ネットワーク安定状態が変化した場合のステータスマスクを使用するかを指定する。

- TRUE　：使用する
- FALSE：使用しない

3.2.1.23. OPERATION_MODE_IF

ネットワークインターフェース作動状態が変化した場合のステータスマスクを使用するかを指定する。

- TRUE　：使用する
- FALSE：使用しない

3.2.1.24. NMACTIVE

NM Passive,NM Active 状態が変化した場合のステータスマスクを使用するかを指定する。

- TRUE ：使用する
- FALSE：使用しない

3.2.1.25. NMON

NM On, NM OFF 状態が変化した場合のステータスマスクを使用するかを指定する。

- TRUE ：使用する
- FALSE：使用しない

3.2.1.26. NMLIMPHOME

非 NM LimpHome,NM LimpHome 状態が変化した場合のステータスマスクを使用するかを指定する。

- TRUE ：使用する
- FALSE：使用しない

3.2.1.27. NMBUSSLEEP

非 NM BusSleep,NM BusSleep 状態が変化した場合のステータスマスクを使用するかを指定する。

- TRUE ：使用する
- FALSE：使用しない

3.2.1.28. NMTWBS_NOR_LIMP

NMTwbsNormal,NMTwbsLimpHome 状態が変化した場合のステータスマスクを使用するかを指定する。

- TRUE ：使用する
- FALSE：使用しない

3.2.1.29. RING_DATA_ALLOWED

RingMsg の Data Field 書き換え状態が変化した場合のステータスマスクを使用するかを指定する。

- TRUE ：使用する
- FALSE：使用しない

3.2.1.30. GMODE_BUSSLEEP

GotoModeGotoMode(Awake)がコールされた,GotoMode(BusSleep)がコールされた場合のステータスマスクを使用するかを指定する。

- TRUE ：使用する
- FALSE：使用しない

3.2.1.31. NMWAIT_BUSSLEEP

NM WaitBusSleep が変化した場合のステータスマスクを使用するかを指定する。

- TRUE　：使用する
- FALSE：使用しない

3.2.1.32. TNUM_RINGDATA

リンクデータサイズを０から４８で指定する。

3.2.1.33. NMMSGNOTIFICATION

データが付加されているリングメッセージを受信した場合のアクションを指定する。

- ACTIVATETASK：起動するタスクを指定する
- SETEVENT：起動するイベントを指定する
- NMCALLBACK：起動するコールバックを指定する
- NONE：指定なし

3.2.1.34. NMSCALEABILITY

NMの規模を指定する。[2]

- MIN_NM：Min NM を指定する
- MAX_NM：Max NM を指定する。

3.2.1.35. NMEXTEND

NM 拡張監視を使用するか指定する。

- TRUE：拡張する
- FALSE：拡張しない

3.2.1.36. THRESHOLD

拡張監視用閾値を指定する。

3.2.1.37. DELTAINC

拡張監視用ノードステータスカウンタの増減値を指定する。

3.2.1.38. DELTADEC

拡張監視用ノードステータスカウンタの減少値を指定する。

[2]制限事項：MIN_NM設定不可

3.2.1.39. TERROR

故障状態のノードが LimpHome メッセージを送信する間隔を指定する。

3.2.1.40. TOB

ノードを監視するためのタイムアウト時間を指定する。

3.2.1.41. NMDYNASTATESMONI

ダイナミックステータスモニタリングを使用するか指定する。

- TRUE：使用する
- FALSE：使用しない

3.2.1.42. NMSTATICSTATESMONI

スタティックステータスモニタリングを使用するか指定する。

- TRUE：使用する
- FALSE：使用しない

3.2.1.43. NMBUSSLEEP

BusSleep を使用するか指定する。

- TRUE：使用する
- FALSE：使用しない

3.2.1.44. TWAITBUSSLEEP

BusSleep に入る前に待機する時間を指定する。

3.2.1.45. BUSINIT_ROUTINE

ネットワークの開始時に一度バスハードウェアを初期化するためのルーチンを参照するか指定する。

3.2.1.46. BUSSHUTDOWN_ROUTINE

バスハードウェアをシャットダウンするためのルーチンを参照するか指定する。

3.2.1.47. BUSAWAKE_ROUTINE

パワーダウンモードから復帰するときにバスハードウェアを再初期化するためのルーチンを参照するか指定する。

3.2.1.48. BUSSLEEP_ROUTINE

バスハードウェアのパワーダウンモードを初期化するためのルーチンを参照するか指定する。

3.2.1.49. BUSRESTART_ROUTINE

FATAL バスエラーの場合にバスハードウェアを再開するためのルーチンを参照するか指定する。

3.2.1.50. NMCMASKNOTIFICATION

コンフィグマスクの状態が変化した場合のアクションを指定する。

- ACTIVATETASK：起動するタスクを指定する
- SETEVENT：起動するイベントを指定する
- NMCALLBACK：起動するコールバックを指定する
- NONE：指定なし

3.2.1.51. SMASKSTATUSNOTIFICATION

NM 状態が変化した場合のアクションを指定する。

- ACTIVATETASK：起動するタスクを指定する
- SETEVENT：起動するイベントを指定する
- NMCALLBACK：起動するコールバックを指定する
- NONE：指定なし

3.2.1.52. 記述例

NM オブジェクトの記述例を記載する。

```
NM nm {
        NMNODE                  = nmnode1;
        NMNODE                  = nmnode2;
        NMNODE                  = nmnode3;
        MYNODE                  = nmnode1;
        NETWORKID               = 0x01;
        NMTYPE = DIRECT {
            TTYP                = 70;
            TMAX                = 200;
            TERROR              = 250;
            TWAITBUSSLEEP       = 200;
            TTX                 = 200;
            NMRXLIMIT           = 1;
            NMTXLIMIT           = 255;
            OPCODE_ALIVE        = 0x01;
            OPCODE_RING         = 0x02;
            OPCODE_LIMPHONE     = 0x04;
```

```
                OPCODE_SLEEPIND             = 0x10;
                OPCODE_SLEEPACK             = 0x20;
                NMMSG_ID_BASE               = 0x00000700;
                NMSLEEPIND                  = TRUE;
                NMMSG_WINDOW_MASK           = 0x00000700;
                NMNUMRCVQUE                 = 2;
                NMSMASK                     = FALSE;
                TNUM_RINGDATA               = 32;
                NMMSGNOTIFICATION = NMCALLBACK {
                        NMCALLBACK          = “nmmsgcb”;
                };
                NMSCALEABILITY              = MAX_NM;
        };
        BUSINIT _ROUTINE                    = TRUE;
        BUSSHUTDOWN _ROUTINE                = FALSE;
        BUSAWAKE _ROUTINE                   = FALSE;
        BUSSLEEP _ROUTINE                   = TRUE;
        BUSRESTART _ROUTINE                 = TRUE;
        NMCMASKNOTIFICATION = NONE;
        NMSMASKNOTIFICATION = ACTIVATETASK {
            TASK        = nmchkhdltask;
        };
};
```

3.2.2. NMNODE

本オブジェクトでは、CAN 通信ミドルウェア（OS 対応）における自ノード（自 ECU）が所属しているネットワークに関連しているノード情報定義を行う。ノード数分オブジェクトの定義を行う。
NMNODE オブジェクトのインプリメンテーション部を下記に記載する。

```
NMNODE {
    UINT32                  NODEID = NO_DEFALUT;
    BOOLEAN [
        TRUE {
            BOOLEAN     ACTUAL_CONFIG;
            BOOLEAN     LIMP_HOME_CONFIG;
        },
```

```
            FALSE
        ] NMCMASK;
};
```

3.2.2.1. NODEID

ノード ID を指定する。

3.2.2.2. NMCMASK

コンフィグマスクを指定する。

- TRUE　：使用する
- FALSE：使用しない

3.2.2.3. ACTUAL_CONFIG

Actual コンフィグが変化した場合のコンフィグマスクを使用するかを指定する。

- TRUE　：使用する
- FALSE：使用しない

3.2.2.4. LIMP_HOME_CONFIG

リンプホームコンフィグが変化した場合のコンフィグマスクを使用するかを指定する。

- TRUE　：使用する
- FALSE：使用しない

3.2.2.5. 記述例

NMNODE オブジェクトの記述例を記載する。

```
NMNODE nmnode1 {
        NODEID                  = 1;
        NMCMASK = TRUE {
                ACTUAL_CONFIG           = TRUE;
                LIMP_HOME_CONFIG        = TRUE;
        }
};
```

```
NMNODE nmnode2 {
        NODEID                  = 2;
        NMCMASK = TRUE {
                ACTUAL_CONFIG           = TRUE;
```

```
                LIMP_HOME_CONFIG        = FALSE;
        }
};
```

```
NMNODE nmnode3 {
        NODEID                  = 3;
        NMCMASK                 = FALSE;
};
```

3.3. DRV モジュール（OS 対応）

3.3.1. CANDRV

本オブジェクトでは、CAN 通信ミドルウェア（OS 対応）における CAN ドライバに関する定義を行う。
CANDRV オブジェクトのインプリメンテーション部を下記に記載する。

```
CANDRV {
        UINT32 [250, 500]       BAUDRATE            = NO_DEFAULT;
        UINT32 [0, 1]           CHANNEL             = NO_DEFAULT;
        UINT32 [20, 30, 32]     CLOCK               = NO_DEFAULT;
        UINT32 [16, 20]         NUMTQ               = NO_DEFAULT;
        UINT32 [77, 83, 95]     PORTIN              = NO_DEFAULT;
        UINT32 [76, 82, 96]     PORTOUT             = NO_DEFAULT;
        BOOLEAN                 USESLEEPMODE        = TRUE;
        BOOLEAN                 CANRCVINT           = TRUE;
        BOOLEAN                 USENMSND            = TRUE;
        BOOLEAN                 USECOMSND           = TRUE;
        BOOLEAN                 USENMRCV            = FALSE;
        BOOLEAN                 USEPOLRCV           = FALSE;
        BOOLEAN                 USECOMRCV           = FALSE;
        BOOLEAN                 USEERRCALL          = FALSE;
        UINT32 [1..7]           WKUPINTLEVEL        = NO_DEFAULT;
        UINT32 [1..7]           RCVINTLEVEL         = NO_DEFAULT;
        UINT32 [1..7]           SNDINTLEVEL         = NO_DEFAULT;
        UINT32 [1..7]           ERRINTLEVEL         = NO_DEFAULT;
        UINT32 [0..2047]        RCVCANID[]          = NO_DEFAULT;
};
```

3.3.1.1. BAUDRATE

ボーレートの設定をする。

3.3.1.2. CHANNEL

使用チャンネル CH1 / CH2 を設定する。

3.3.1.3. CLOCK

CPU クロック周波数 20MHz / 30MHz / 32MHz を設定する。

3.3.1.4. NUMTQ

ビットタイミング 16Tq / 20Tq を設定する。

3.3.1.5. PORTIN

使用入力ポート P77 / P83 / P95 を設定する。

3.3.1.6. PORTOUT

使用出力ポート P76 / P82 / P96 を設定する。

3.3.1.7. USESLEEPMODE

CAN の Sleep 許可・禁止を設定する。

- NM モジュール併用時
  - TRUE 固定とする。
- NM モジュール未使用時
  - FALSE 固定とする。

3.3.1.8. CANRCVINT

CAN 受信割込みの使用・未使用を設定する。TURE 固定とする。

3.3.1.9. USENMSND

NM 送信の使用・未使用を設定する。

- NM モジュール併用時
  - TRUE 固定とする。
- NM モジュール未使用時
  - FALSE 固定とする。

3.3.1.10. USECOMSND

COM 送信の使用・未使用を設定する。TRUE 固定とする。

3.3.1.11. USENMRCV

NM 受信の使用・未使用を設定する。

- NM モジュール併用時
  - TRUE 固定とする。
- NM モジュール未使用時
  - FALSE 固定とする。

3.3.1.12. USEPOLRCV

CAN ポーリング受信の使用・未使用を設定する。FALSE 固定とする。

3.3.1.13. USECOMRCV

COM 受信の使用・未使用を設定する。TURE 固定とする。

3.3.1.14. USEERRCALL

エラーコールの使用・未使用を設定する。

3.3.1.15. WKUPINTLEVEL

ウェイクアップ割込みのレベルを設定する。

3.3.1.16. RCVINTLEVEL

受信割込みのレベルを設定する。

3.3.1.17. SNDINTLEVEL

送信割込みのレベルを設定する。

3.3.1.18. ERRINTLEVEL

エラー割込みのレベルを設定する。

3.3.1.19. RCVCANID

自ノードが受信する全ての CANID を設定する。

3.3.1.20. 記述例

CANDRV オブジェクトの記述例を記載する。

```
CANDRV candrv {
        BAUDRATE                = 500;
        CHANNEL                 = 1;
```

```
        CLOCK                   = 30;
        NUMTQ                   = 16;
        PORTIN                  = 77;
        PORTOUT                 = 96;
        USESLEEPMODE            = TRUE;
        CANRCVINT               = TRUE;
        USENMSND                = TRUE;
        USECOMSND               = TRUE;
        USENMRCV                = FALSE;
        USEPOLRCV               = FALSE;
        USEERRCALL              = FALSE;
        WKUPINTLEVEL            = 4;
        RCVINTLEVEL             = 5;
        SNDINTLEVEL             = 5;
        ERRINTLEVEL             = 2;
        RCVCANID                = 0x0200;
};
```

3.4. 共通モジュール（OS 対応）

3.4.1. CANCOMMON

本オブジェクトでは、CAN 通信ミドルウェア（OS 対応）における COM モジュール、CANDRV モジュールの共通設定に関する定義を行う。

CANCOMMON オブジェクトのインプリメンテーション部を下記に記載する。

```
CANCOMMON {
        UINT32 [1..255]     MAINCYCLE           = NO_DEFAULT;
        UINT32 [0..255]     SOURCEID            = NO_DEFAULT;
        UINT32 [0..255]     IDBASE              = NO_DEFAULT;
        UINT32 [0..2040]    WINDOWMASK          = NO_DEFAULT;
        BOOLEAN             COMUSERCVINT        = FALSE;
};
```

3.4.1.1. MAINCYCLE

システムにおけるメイン周期を設定する。

3.4.1.2. SOURCEID

NMID は SOURCEID と IDBASE にて構成されている。（合計 11bit の ID）

そのため SOURCEID 値を 3 ～ 8 bit の範囲（11bit – IDBASE 属性値）で設定する。

3.4.1.3. IDBASE

IDBase 値を 3 ～ 8 bit の範囲で設定する。

3.4.1.4. WINDOWMASK

IDBase のマスク値を設定する。

3.4.1.5. COMUSERCVINT

COM 受信割り込みの使用可否を設定する。

3.4.1.6. 記述例

CANCOMMON オブジェクトの記述例を記載する。

```
CANCOMMON cancommon {
        MAINCYCLE           = 20;
        SOURCEID            = 30;
        IDBASE              = 4;
        WINDOWMASK          = 0x07e0;
        COMUSERCVINT        = FALSE;
};
```

3.5. COM モジュール（非 OS 対応）

3.5.1. CANCOM

本オブジェクトでは、CAN 通信ミドルウェア（非 OS 対応）における COM 通信機能を実現するために必要な CANCOM オブジェクトに関する定義を行う。非 OS 対応時のみ有効なオブジェクトである。
CANCOM オブジェクトのインプリメンテーション部を下記に記載する。

```
CANCOM {
    UINT32 [1..65535]   SNDINTERVAL         = NO_DEFAULT;
    UINT32 [1..65535]   SNDSTOP             = NO_DEFAULT;
    UINT32 [0..65535]   INITINTERVAL        = NO_DEFAULT;
    UINT32 [1..65535]   EVNINTERVAL         = NO_DEFAULT;
    UINT32 [1..64]      RCVQUEUENUM         = NO_DEFAULT;
    UINT32 [1..8]       SNDQUEUENUM         = NO_DEFAULT;
};
```

3.5.1.1. SNDINTERVAL

COM メッセージ送信間隔における確保時間を設定する。

3.5.1.2. SNDSTOP

通信途絶判定を実施する時間を設定する。

3.5.1.3. INITINTERVAL

電源挿入後から初期化完了までの時間を設定する。

3.5.1.4. EVNINTERVAL

イベント感覚保証時間を設定する。

3.5.1.5. RCVQUEUENUM

受信キューの数を設定する。尚、受信キューへの設定値”１”は CANCOMMON オブジェクトの COMUSERCVINT 属性未使用時のみ設定可能である。

3.5.1.6. SNDQUEUENUM

送信キューの数を設定する。

3.5.1.7. 記述例

CANCOM オブジェクトの記述例を記載する。

```
CANCOM com {
        SNDINTERVAL                     = 1000;
        SNDSTOP                         = 1000;
        INITINTERVAL                    = 10;
        EVNINTERVAL                     = 10;
        RCVQUEUENUM                     = 32;
        SNDQUEUENUM                     = 32;
};
```

3.5.2. MESSAGE

本オブジェクトでは、CAN 通信ミドルウェア（非 OS 対応）における COM 通信機能を実現するために必要な MESSAGE オブジェクトに関する定義を行う。非 OS 対応時のみ有効なオブジェクトである。
MESSAGE オブジェクトのインプリメンテーション部を下記に記載する。

```
MESSAGE {
    ENUM {
```

```
        SEND,
        RECEVIE
    ] MESSAGEPROPERTY       = NO_DEFAULT;
    UINT32 [0..255]         INITIALVALUE        = NO_DEFAULT;
    UINT32 [1..64]          MESSAGESIZE         = NO_DEFAULT;
    NETWORKMESSAGE_TYPE     NETWORKMESSAGE;
};
```

3.5.2.1. MESSAGEPROPERTY

メッセージの送受信プロパティを設定する。

3.5.2.2. INITIALVALUE

メッセージにおける初期値を設定する。

3.5.2.3. MESSAGESIZE

メッセージのバッファサイズを設定する。

3.5.2.4. NETWORKMESSAGE

MESSGE オブジェクトに関連付ける NETWORKMESSAGE オブジェクト名を設定する。

3.5.2.5. 記述例

MESSAGE オブジェクトの記述例を記載する。

```
MESSAGE msg {
    MESSAGEPROPERTY     = RECEVIE;
    INITIALVALUE        = 0x00;
    MESSAGESIZE         = 64;
    NETWORKMESSAGE      = “nwm_001”;
};
```

3.5.3. NETWORKMESSAGE

本オブジェクトでは、CAN 通信ミドルウェア（非 OS 対応）における COM 通信機能を実現するために必要な NETWORKMESSAGE オブジェクトに関する定義を行う。非 OS 対応時のみ有効なオブジェクトである。NETWORKMESSAGE オブジェクトのインプリメンテーション部を下記に記載する。

```
NETWORKMESSAGE {
  IPDU_TYPE             IPDU;
  UINT32 [0..63]        BITPOSITION         = NO_DEFAULT;
```

```
};
```

3.5.3.1. IPDU

NETWORKMESSAGE に関連付ける IPDU オブジェクト名を設定する。

3.5.3.2. BITPOSITION

IPDU に格納する場所を BIT 単位で設定する。

3.5.3.3. 記述例

NETWORKMESSAGE オブジェクトの記述例を記載する。

```
NETWORKMESSAGE netmsg {
    IPDU                        = “ipdu_001”;
    BITPOSITION                 = 0x04;
};
```

3.5.4. IPDU

本オブジェクトでは、CAN 通信ミドルウェア（非 OS 対応）における COM 通信機能を実現するために必要な IPDU オブジェクトに関する定義を行う。非 OS 対応時のみ有効なオブジェクトである。IPDU オブジェクトのインプリメンテーション部を下記に記載する。

```
IPDU {
        UINT32 [1..64]  SIZEINBITS    = NO_DEFAULT;
        ENUM [
                SENT {
                        ENUM [
                                DIRECT {
                                        BOOLEAN     USESNDINTERVAL       = NO_DEFAULT;
                                },
                                PERIODIC {
                                        UINT64      TIMEPERIOD   = NO_DEFAULT;
                                        UINT64      WITH_AUTO    TIMEOFFSET  = NO_DEFAULT;
                                },
                                MIXED {
                                        UINT64      TIMEPERIOD   = NO_DEFAULT;
                                        UINT64      WITH_AUTO    TIMEOFFSET  = NO_DEFAULT;
                                        BOOLEAN     USESNDINTERVAL       = NO_DEFAULT;
                                },
```

```
                        MIXEDEXT {
                            UINT64          TIMEPERIOD   = NO_DEFAULT;
                            UINT64          WITH_AUTO    TIMEOFFSET  = NO_DEFAULT;
                            BOOLEAN         USESNDINTERVAL        = NO_DEFAULT;
                        }
                    ] TRANSMISSIONMODE      = NO_DEFAULT;
                    UINT64          TIMEOUT       = 0;
                },
                RECEIVED {
                    UINT64          TIMEOUT       = 0;
                    UINT64          WITH_AUTO     FIRSTTIMEOUT          = AUTO;
                    ENUM [
                            GETDATA8BYTE,
                            GETDATACANCEL
                    ] DLCSIZEOVER               = NO_DEFAULT;
                    ENUM [
                            GETDATAONLY,
                            GETDATAAND0,
                            GETDATACANCEL
                    ] DLCSIZEUNDER              = NO_DEFALUTL;
                },
        ] IPDUPROPERTY      = NO_DEFAUTL;
        SYMBOLNAME          IPDUCALLOUT         = "";
        ENUM [
                CANID,
                DATA {
                        UINT32 [0.255]  DATAID                      = NO_DEFAULT;
                }
        ] MESSAGEFORMAT   = NO_DEFAULT;
        UINT32 [0..2047]        CANID           = NO_DEFAULT;
};
```

3.5.4.1. SIZEINBITS

IPDU のサイズを BIT 単位で設定する。

3.5.4.2. IPDUPROPERTY

IPDU の送受信プロパティを設定する。

3.5.4.3. TRANSMISSIONMODE

送信制御の方法を設定する。設定値を以下に示す。

- DIRECT　　：イベント送信
- PERIODIC　：周期送信
- MIXED　　：イベント＋周期送信（タイプ 1）
- MIXEDEXT　：イベント＋周期送信（タイプ 2）

3.5.4.4. USESNDINTERVAL

送信確保時間の使用有無を設定する。

3.5.4.5. TIMEPERIOD

周期送信のサイクルを設定する。

3.5.4.6. TIMEOFFSET

周期送信のオフセット値を設定する。

3.5.4.7. TIMEOUT

受信または送信のタイムアウト時間を設定する。

3.5.4.8. FIRSTTIMEOUT

受信時、最初のタイムアウト時間を設定する。

3.5.4.9. DLCSIZEOVER

受信したメッセージのメッセージ長が DLC サイズを超えていた場合の、DLC チェック処理を設定する。
IPDUPROPERTY 属性が RECEIVED の場合、有効となる。

- GETDATA8BYTE　　：データを 8 バイト全て取り込む
- GETDATACANCEL　：受信したデータを破棄する

3.5.4.10. DLCSIZEUNDER

受信したメッセージのメッセージ長が DLC サイズ未満だった場合の DLC チェック処理を設定する。
IPDUPROPERTY 属性が RECEIVED の場合有効とする。

- GETDATAONLY　　：受信データのみ取り込む
- GETDATAAND0　　：受信したデータ以降は 0 で埋める
- GETDATACANCEL　：受信したデータを破棄する

3.5.4.11. IPDUCALLOUT

IPDUCALLOUT 時に呼ばれるモジュールを設定する。

3.5.4.12. MESSAGEFORMAT

CAN データ送信書式を設定する。

- CANID：CAN データ送信書式１
- DATA：CAN データ送信書式２

3.5.4.13. DATAID

CAN データ送信書式２の場合、データ ID を設定する。

3.5.4.14. CANID

IPDU に関連付ける CANID の値を設定する。

3.5.4.15. 記述例

IPDU オブジェクトの記述例を記載する。

```
IPDU ipdu {
        SIZEINBITS              = 0x40;
        IPDUPROPERTY            = RECEIVED {
                TIMEOUT                 = 0x010;
                FIRSTTIMEOUT            = 0x40;
                DLCSIZEOVER             = GETDATA8BYTE;
                DLCSIZEUNDER            = GETDATAONLY;
        };
        IPDUCALLOUT             = “ipducallout_func”;
        CANID                   = 0x0008;
};
```

3.6. NM モジュール（非 OS 対応：Indirect）

3.6.1. CANNM

本オブジェクトでは、CAN 通信ミドルウェア（非 OS 対応）における IndirectNM 機能を実現するために必要な CANNM オブジェクトに関する定義を行う。非 OS 対応時のみ有効なオブジェクトである。
CANNM オブジェクトのインプリメンテーション部を下記に記載する。

```
CANNM {
   UINT32 [1..65535]            TERROR                  = NO_DEFAULT;
```

```
    UINT32 [1..65535]               TWAITBUSSLEEP           = NO_DEFAULT;
    ENUM {
            TOB {
                    UINT32 [1..65535]   TOBTIME             = NO_DEFAULT;
            },
            OMT
    ] USEMONITOR                    = NO_DEFAULT;
    BOOLEAN                         NMUSESLEEP              = FALSE;
    UINT32 [0..255]                 MNNODEID                = NO_DEFAULT;
    UINT32 [1..255]                 NETSHRESHOLD            = NO_DEFAULT;
    UINT32 [1..255]                 CFGSHRESHOLD            = NO_DEFAULT;
    UINT32 [1..255]                 NETDELTAINC             = NO_DEFAULT;
    UINT32 [1..255]                 NETDELTADEC             = NO_DEFAULT;
};
```

3.6.1.1. TERROR

LimpHome 送信周期を設定する。尚、CANCOMMON オブジェクトの MAINCYCLE 属性値の倍数になるように設定する必要がある。

3.6.1.2. TWAITBUSSLEEP

BusSleep 前の待機時間を設定する。尚、CANCOMMON オブジェクトの MAINCYCLE 属性値の倍数になるように設定する必要がある。

3.6.1.3. USEMONITOR

ノード監視方法を選択する。

3.6.1.4. TOBTIME

TOB 選択時のタイムアウト時間を設定する。

3.6.1.5. NUMUSESLEEP

NM Sleep の使用・未使用を設定する。

3.6.1.6. MNNODEID

ノード ID を設定する。

3.6.1.7. NETSHRESHOLD

拡張ネットワークステータス用カウンタ閾値を設定する。

3.6.1.8. CFGSHRESHOLD

拡張コンフィグ用カウンタ閾値を設定する。

3.6.1.9. NETDELTAINC

拡張ネットワークステータス用カウンタ⊿INC を設定する。

3.6.1.10. NETDELTADEC

拡張ネットワークステータス用カウンタ⊿DEC を設定する。

3.6.1.11. 記述例

CANNM オブジェクトの記述例を記載する。

```
CANNM cannm {
    TERROR                  = 1000;
    TWAITBUSSLEEP           = 1200;
    USEMONITOR              = OMT;
    NMUSESLEEP              = FALSE
    MNNODEID                = 1;
    NETSHRESHOLD            = 500;
    CFGSHRESHOLD            = 500;
    NETDELTAINC             = 125;
    NETDELTADEC             = 125;
};
```

3.7. NM モジュール（非 OS 対応：Direct）

3.7.1. CANNM

本オブジェクトでは、CAN 通信ミドルウェア（非 OS 対応）における DirectNM 機能を実現するために必要な CANNM オブジェクトに関する定義を行う。非 OS 対応時のみ有効なオブジェクトである。
CANNM オブジェクトのインプリメンテーション部を下記に記載する。

```
CANNM {
    UINT32 [1..255]         NODENUM;
    UINT32 [70..110]        TTYP;
    UINT32 [220..284]       TMAX;
    UINT32 [1..65535]       TERROR;
```

```
    UINT32 [1..65535]       TWAITBUSSLEEP;
    UINT32 [1..255]         NMRXLIMIT;
    UINT32 [1..255]         NMTXLIMIT;
    UINT32 [2..8]           NMNUMSNDQUE;
    UINT32 [2..64]          NMNUMRCVQUE;
    UINT32 [2..8]           DLC;
};
```

3.7.1.1. NODENUM

ネットワークに参加しているノードの数を指定する。
CANCOMMON オブジェクト MAINCYCLE 属性値の倍数になるように設定する必要である。

3.7.1.2. TTYP

ダイレクト NM にて、トークンを保持できる時間を指定する。
CANCOMMON オブジェクト MAINCYCLE 属性値の倍数になるように設定する必要である。

3.7.1.3. TMAX

バス上にメッセージが流れていないことを認知するまでの時間を指定する。
CANCOMMON オブジェクト MAINCYCLE 属性値の倍数になるように設定する必要である。

3.7.1.4. TERROR

故障状態のノードが LimpHome メッセージを送信する間隔を指定する。
CANCOMMON オブジェクト MAINCYCLE 属性値の倍数になるように設定する必要である。

3.7.1.5. TWAITBUSSLEEP

BusSleep に入る前に待機する時間を指定する。
CANCOMMON オブジェクト MAINCYCLE 属性値の倍数になるように設定する必要である。

3.7.1.6. NMRXLIMIT

受信不能カウンタ閾値を１から２５５で指定する。

3.7.1.7. NMTXLIMIT

送信不能カウンタ閾値を１から２５５で指定する。

3.7.1.8. NMNUMSNDQUE

送信キューの数を２から８で指定する。

3.7.1.9. NMNUMRCVQUE

受信キューの数を２から６４で指定する。

3.7.1.10. DLC

データ長を２から８で指定する。

3.7.1.11. 記述例

CANNM オブジェクト例で示しているのは、記述例であって個々の属性値が正しいとは限りません。

```
CANNM cannmnetwork1 {
    NODEID              = 6;
    TTYP                = 10;
    TMAX                = 10;
    TERROR              = 10;
    TWAITBUSSLEEP       = 10;
    NMRXLIMIT           = 30;
    NMTXLIMIT           = 50;
    NMNUMSNDQUE         = 3;
    NMNUMRCVQUE         = 40;
    DLC                 = 6;
};
```

3.8. DRV モジュール（非 OS 対応）

3.8.1. CANDRV

本オブジェクトでは、CAN 通信ミドルウェア（非 OS 対応）における IndirectNM/DirectNM 機能を実現するために必要な CANDRV オブジェクトに関する定義を行う。非 OS 対応時のみ有効なオブジェクトである。CANDRV オブジェクトのインプリメンテーション部を下記に記載する。

```
CANDRV {
    UINT32 [250, 500]       BAUDRATE        = NO_DEFAULT;
    UINT32 [0, 1]           CHANNEL         = NO_DEFAULT;
    UINT32 [20, 30, 32]     CLOCK           = NO_DEFAULT;
    UINT32 [16, 20]         NUMTQ           = NO_DEFAULT;
    UINT32 [77, 83, 95]     PORTIN          = NO_DEFAULT;
    UINT32 [76, 82, 96]     PORTOUT         = NO_DEFAULT;
    BOOLEAN                 USESLEEPMODE    = TRUE;
    BOOLEAN                 CANRCVINT       = TRUE;
    BOOLEAN                 USENMSND        = TRUE;
```

```
        BOOLEAN             USECOMSND       = TRUE;
        BOOLEAN             USENMRCV        = FALSE;
        BOOLEAN             USEPOLRCV       = FALSE;
        BOOLEAN             USECOMRCV       = FALSE;
        BOOLEAN             USEERRCALL      = FALSE;
        UINT32 [1..7]       WKUPINTLEVEL    = NO_DEFAULT;
        UINT32 [1..7]       RCVINTLEVEL     = NO_DEFAULT;
        UINT32 [1..7]       SNDINTLEVEL     = NO_DEFAULT;
        UINT32 [1..7]       ERRINTLEVEL     = NO_DEFAULT;
        UINT32 [0..2047]    RCVCANID[]      = NO_DEFAULT;
};
```

3.8.1.1. BAUDRATE

ボーレートの設定をする。

3.8.1.2. CHANNEL

使用チャンネル CH1 / CH2 を設定する。

3.8.1.3. CLOCK

CPU クロック周波数 20MHz / 30MHz / 32MHz を設定する。

3.8.1.4. NUMTQ

ビットタイミング 16Tq / 20Tq を設定する。

3.8.1.5. PORTIN

使用入力ポート P77 / P83 / P95 を設定する。

3.8.1.6. PORTOUT

使用出力ポート P76 / P82 / P96 を設定する。

3.8.1.7. USESLEEPMODE

CAN の Sleep 許可・禁止を設定する。

- DirectNM
  - COM モジュールまたは NM モジュールを併用する場合、TRUE 固定とする。
- InDirectNM
  - NM モジュールを併用する場合、CANNM オブジェクトの NMUSESLEEP 属性と設定値が

一致している必要がある。

3.8.1.8. CANRCVINT

CAN 受信割込みの使用・未使用を設定する。

- DirectNM
  - COM モジュールまたは NM モジュールを併用する場合、TRUE 固定とする。
- InDirectNM
  - CANCOMMON オブジェクトの COMUSERCVINT 属性と設定値が一致している必要がある。

3.8.1.9. USENMSND

NM 送信の使用・未使用を設定する。

- DirectNM
  - COM モジュールまたは NM モジュールを併用する場合、TRUE 固定とする。
- InDirectNM
  - FALSE 固定とする。

3.8.1.10. USECOMSND

COM 送信の使用・未使用を設定する。DirectNM、InDirectNM 共に TRUE 固定とする。

3.8.1.11. USENMRCV

NM 受信の使用・未使用を設定する。

- DirectNM
  - COM モジュールまたは NM モジュールを併用する場合、TRUE 固定とする。
- InDirectNM
  - FALSE 固定とする。

3.8.1.12. USEPOLRCV

CAN ポーリング受信の使用・未使用を設定する。

- DirectNM
  - CANCOMMON 属性の COMUSERCVINT と排他関係にある。
- InDirectNM
  - CANCOMMON 属性の COMUSERCVINT と排他関係にある。

3.8.1.13. USECOMRCV

COM 受信の使用・未使用を設定する。DirectNM、InDirectNM ともに、CANCOMMON 属性の COMSERCVINT と一致させる必要がある。

3.8.1.14. USEERRCALL

エラーコールの使用・未使用を設定する。

3.8.1.15. WKUPINTLEVEL

ウェイクアップ割込みのレベルを設定する。

3.8.1.16. RCVINTLEVEL

受信割込みのレベルを設定する。

- DirectNM
  - 特に制限なし。
- InDirectNM
  - COM 受信割込みが使用しない設定になっている場合、0 を出力する。
  - 使用する設定になっている場合は設定値を出力する。

3.8.1.17. SNDINTLEVEL

送信割込みのレベルを設定する。

3.8.1.18. ERRINTLEVEL

エラー割込みのレベルを設定する。

3.8.1.19. RCVCANID

自ノードが受信する全ての CANID を設定する。

3.8.1.20. 記述例

CANDRV オブジェクトの記述例を記載する。

```
CANDRV candrv {
        BAUDRATE            = 500;
        CHANNEL             = 1;
        CLOCK               = 30;
        NUMTQ               = 16;
        PORTIN              = 77;
        PORTOUT             = 96;
        USESLEEPMODE        = TRUE;
        CANRCVINT           = TRUE;
        USENMSND            = TRUE;
```

```
        USECOMSND               = TRUE;
        USENMRCV                = FALSE;
        USEPOLRCV               = FALSE;
        USEERRCALL              = FALSE;
        WKUPINTLEVEL            = 4;
        RCVINTLEVEL             = 5;
        SNDINTLEVEL             = 5;
        ERRINTLEVEL             = 2;
        RCVCANID                = 0x0200;
};
```

3.9. 共通モジュール（非 OS 対応）

3.9.1. CANCOMMON

本オブジェクトでは、CAN 通信ミドルウェア（非 OS 対応）における COM 通信機能を実現するために必要なＣＯＭモジュール、CANDRV モジュールの共通設定に関する定義を行う。
CANCOMMON オブジェクトのインプリメンテーション部を下記に記載する。

```
CANCOMMON {
        UINT32 [1..255]     MAINCYCLE;              // メイン周期
        UINT32 [0..255]     SOURCEID;               // SOURCE ID
        UINT32 [0..255]     IDBASE;                 // ID BASE
        UINT32 [0..2040]    WINDOWMASK;             // WINDOWMASK
        BOOLEAN             COMUSERCVINT;           // COM 受信割り込み使用/未使用
};
```

3.9.1.1. MAINCYCLE

システムにおけるメイン周期を設定する。

3.9.1.2. SOURCEID

NMID は SOURCEID と IDBASE にて構成されている。（合計 11bit の ID）
そのため SOURCEID 値を３～８bit の範囲（11bit – IDBASE 属性値）で設定する。

3.9.1.3. IDBASE

IDBase 値を３～８bit の範囲で設定する。

3.9.1.4. WINDOWMASK

IDBase のマスク値を設定する。

3.9.1.5. COMUSERCVINT

COM 受信割り込みの使用可否を設定する。

3.9.1.6. 記述例

CANCOMMON オブジェクトの記述例を記載する。

```
CANCOMMON cancommon {
        MAINCYCLE           = 20;
        SOURCEID            = 30;
        IDBASE              = 4;
        WINDOWMASK          = 0x07e0;
        COMUSERCVINT        = FALSE;
};
```

3.9.2. CANNODE

本オブジェクトでは、CAN 通信ミドルウェア（非 OS 対応）における COM 通信機能を実現するために必要な CANNODE オブジェクト関する定義を行う。CANNODE オブジェクトのインプリメンテーション部を下記に記載する。

```
CANNODE {
    UINT32 [0..255]             NODEID              = NO_DEFAULT;
    BOOLEAN {
        TRUE {
            UINT32 [0..2047]    OWNCANID            = NO_DEFAULT;
        },
        FALSE {
            UINT32 [0..2047]    MAINUSECANID        = NO_DEFAULT;
        }
    ] OWNNODE                   = NO_DEFAULT;
    UINT32 [1..255]             DELTAINC            = NO_DEFAULT;
    UINT32 [1..255]             DELTADEC            = NO_DEFAULT;
};
```

3.9.2.1. NODEID

ノードに対応付ける ID を設定する。他のノードと重複している場合はエラーとなる。

3.9.2.2. OWNCANID

設定ノードがオーナーノードの場合、オーナーとなる CANID を設定する。
設定した ID が SENT 設定の IPDU オブジェクトの CANID に割当てられていない場合はエラーとなる。

3.9.2.3. OWNNODE

設定ノードのオーナー設定をする。
TRUE を指定すると設定ノードがオーナーノードとなる。
オーナーノードは全ノード中 1 つしか設定できない。未定義の場合、1 つ以上の場合はエラーとなる。

3.9.2.4. MAINUSECANID

設定ノードで自身時に主に使用する CANID を設定する。
設定した ID が RECEIVED 設定の IPDU オブジェクトの CANID に割当てられていない場合はエラーとなる。

3.9.2.5. DELTAINC

ノード用カウンタ⊿INC を設定する。

3.9.2.6. DELTADEC

ノード用カウンタ⊿DEC を設定する。

3.9.2.7. 記述例

CANNODE オブジェクトの記述例を記載する。

```
CANNODE cannode {
    NODEID                          = 0x0100;
    OWNNODE                         = TRUE {
                        OWNCANID            = 0x0433;
    };
    DELTAINC                        = 0x80;
    DELTADEC                        = 0x07;
};
```

3.10. LINMW モジュール（OS 対応／非 OS 対応）

3.10.1. LINNODE

本オブジェクトでは、LIN 通信ミドルウェア（OS 対応／非 OS 対応）で使用する LIN のノードに関する定義を行う。LINNODE オブジェクト 1 つにつき、ノード 1 つ分の定義となる。
LINNODE オブジェクトのインプリメンテーション部を下記に記載する。

```
LINNODE {
    ENUM [
        MASTER {
            LINSCHEDULE_TYPE        SCHEDULE[];
        },
        SLAVE
    ] NODEATTR = NO_DEFAULT;
    ENUM [
        USE {
            UINT32 [1..255]             DIAGADDR = NO_DEFAULT;
            ENUM [SEND, RECEIVE]        DIRECT = NO_DEFAULT;
        },
        UNUSE
    ] DIAG = UNUSE;
    LINFRAME_TYPE                       SENDFRAME[];
    LINSIGNAL_TYPE                      RECEIVESIGNAL[];
};
```

3.10.1.1. NODEATTR

ノードに対してマスターノード or スレーブノードの設定する。

3.10.1.2. SCHEDULE

マスターノードが管理するスケジュールテーブルを設定する。本属性は、NODEATTR 属性を MASTER に選択した場合のみ設定可能である。
該当する LINSCHEDULE オブジェクト名を指定する。（複数指定可能）

3.10.1.3. DIAG

ノードに対してダイアグの使用有無を設定する。
USE を指定するとダイアグ使用可能、UNUSE を指定するとダイアグ使用不可となる。

3.10.1.4. DIAGADDR

ノードで使用するダイアグアドレスを設定する。本属性は、DIAG 属性を USE に選択した場合のみ設定可能である。

3.10.1.5. DIRECT

ダイアグの送受信方向を設定する。本属性は、DIAG 属性を USE に選択した場合のみ設定可能である。

3.10.1.6. SENDFRAME

ノードが送信するフレームを設定する。該当する LINFRAME オブジェクト名を指定する。（複数指定可能）

3.10.1.7. RECEIVESIGNAL

ノードが受信するシグナルを設定する。該当する LINSIGNAL オブジェクト名を指定する。（複数指定可能）

3.10.1.8. 記述例

LINNODE オブジェクトの記述例を記載する。

```
LINNODE linnode {
    NODEATTR = MASTER {
            SCHEDULE            = ScheduleA;
            SCHEDULE            = ScheduleB;
    };
    ] DIAG = USE {
            DIAGADDR            = 0x0010;
            DIRECT              = SEND;
    };
    SENDFRAME                   = FrmSndDataA;
    SENDFRAME                   = FrmSndDataB;
    RECEIVESIGNAL               = SigRcvData;
};
```

3.10.2. LINSIGNAL（LINMW モジュール：OS 対応／非 OS 対応）

本オブジェクトでは、LIN 通信ミドルウェア（OS 対応／非 OS 対応）で使用する LIN のシグナルに関する定義を行う。LINSIGNAL オブジェクト 1 つにつき、シグナル 1 つ分の定義となる。
LINSIGNAL オブジェクトのインプリメンテーション部を下記に記載する。

```
LINSIGNAL {
    UINT32 [1..16]      SIZE = NO_DEFAULT;
    UINT32 [0..65535]   INITVALUE = NO_DEFAULT;
};
```

3.10.2.1. SIZE

シグナルのデータサイズを設定する。（ビット単位）

3.10.2.2. INITVALUE

シグナルの初期値を設定する。SIZE 属性で指定したビット数の範囲を超える数値を指定してはいけない。

3.10.2.3. 記述例

LINSIGNAL オブジェクトの記述例を記載する。

```
LINSIGNAL linsignal {
        SIZE                    = 8;
        INITVALUE               = 0;
};
```

3.10.3. LINFRAME（LINMW モジュール：OS 対応／非 OS 対応）

本オブジェクトでは、LIN 通信ミドルウェア（OS 対応／非 OS 対応）で使用する LIN のフレームに関する定義を行う。LINFRAME オブジェクト 1 つにつき、フレーム 1 つ分の定義となる。
LINFRAME オブジェクトのインプリメンテーション部を下記に記載する。

```
LINFRAME {
        UINT32 [0..15]                  FRAMEID = NO_DEFAULT;
        UINT32 ENUM WITH_AUTO [2,4,8]   FRAMESIZE = AUTO;
        ENUM [
                FORMAT {
                        LINSIGNAL_TYPE          SIGNAL;
                        UINT32 [0..63]          OFFSET = NO_DEFAULT;
                }
        ] USESIGNAL[];
};
```

3.10.3.1. FRAMEID

フレームのフレームID[3]を設定する。但し、FRAMESIZE属性に 8 を指定している場合は、FRAMEID 属性には 0～11 の値いずれかしか指定できない。

3.10.3.2. FRAMESIZE

フレームのフレームサイズを設定する。（バイト単位）
AUTO を指定した場合、フレームに入れるシグナルのサイズやオフセット応じてフレームサイズを自動で決定する。FRAMEID 属性が 12～15 のいずれかの場合は、FRAMESIZE 属性に 8 を指定してはいけ

[3] LIN仕様にて規定されているIDの下位 4bitを指す。

ない。

3.10.3.3. USESIGNAL

フレームに格納する対象シグナルとシグナルのデータ配置を設定する。（複数指定可能）

3.10.3.4. SIGNAL

フレームに格納する対象シグナルを設定する。
格納対象となる LINSIGNAL オブジェクト名を指定する。

3.10.3.5. OFFSET

SIGNAL 属性で指定したシグナルのデータ配置位置を設定する。（ビット単位）
0 がフレームの先頭である。
フレームサイズ範囲(FRAMESIZE * 8 - 1)を超えるオフセットを指定してはいけない。

3.10.3.6. 記述例

LINFRAME オブジェクトの記述例を記載する。

```
LINFRAME linframe {
    FRAMEID                 = 4;
    FRAMESIZE               = 64;
    USESIGNAL = FORMAT {
            SIGNAL          = SigCmbDataA;
            OFFSET          = 8;
    };
    USESIGNAL = FORMAT {
            SIGNAL          = SigCmbDataB;
            OFFSET          = 16;
    };
    USESIGNAL = FORMAT {
            SIGNAL          = SigCmbDataC;
            OFFSET          = 32;
    };
};
```

3.10.4. LINSCHEDULE（LINMW モジュール：OS 対応／非 OS 対応）

本オブジェクトでは、LIN 通信ミドルウェア（OS 対応／非 OS 対応）で使用する LIN のスケジュールに関する定義を行う。LINSCHEDULE オブジェクト 1 つにつき、フレーム 1 つ分の定義となる。

LINSCHEDULE オブジェクトのインプリメンテーション部を下記に記載する。

```
LINSCHEDULE {
      ENUM [
            FORMAT {
                  LINFRAME_TYPE       FRAME;
                  UINT32              DELAY = NO_DEFAULT;
            }
      ] SENDFRAME[];
};
```

3.10.4.1. SENDFRAME

スケジュールで送信するフレーム、ディレイ時間を設定する。（複数指定可能）

3.10.4.2. FRAME

スケジュールで送信するフレームを設定する。送信対象の LINFRAME オブジェクト名を指定する。

3.10.4.3. DELAY

フレーム送信間隔のディレイ時間を設定する。（us 単位）

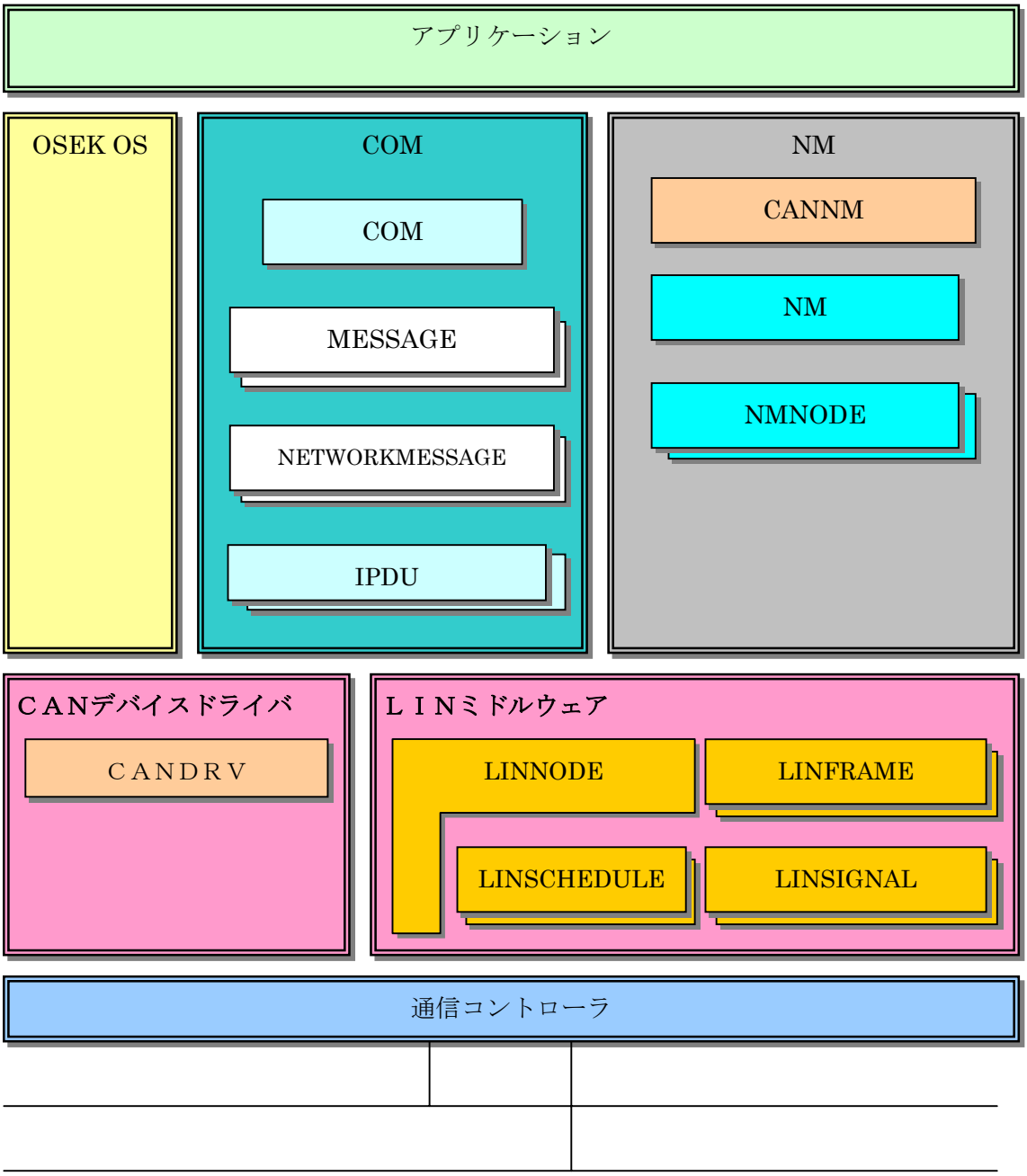
3.10.4.4. 記述例

LINSCHEDULE オブジェクトの記述例を記載する。

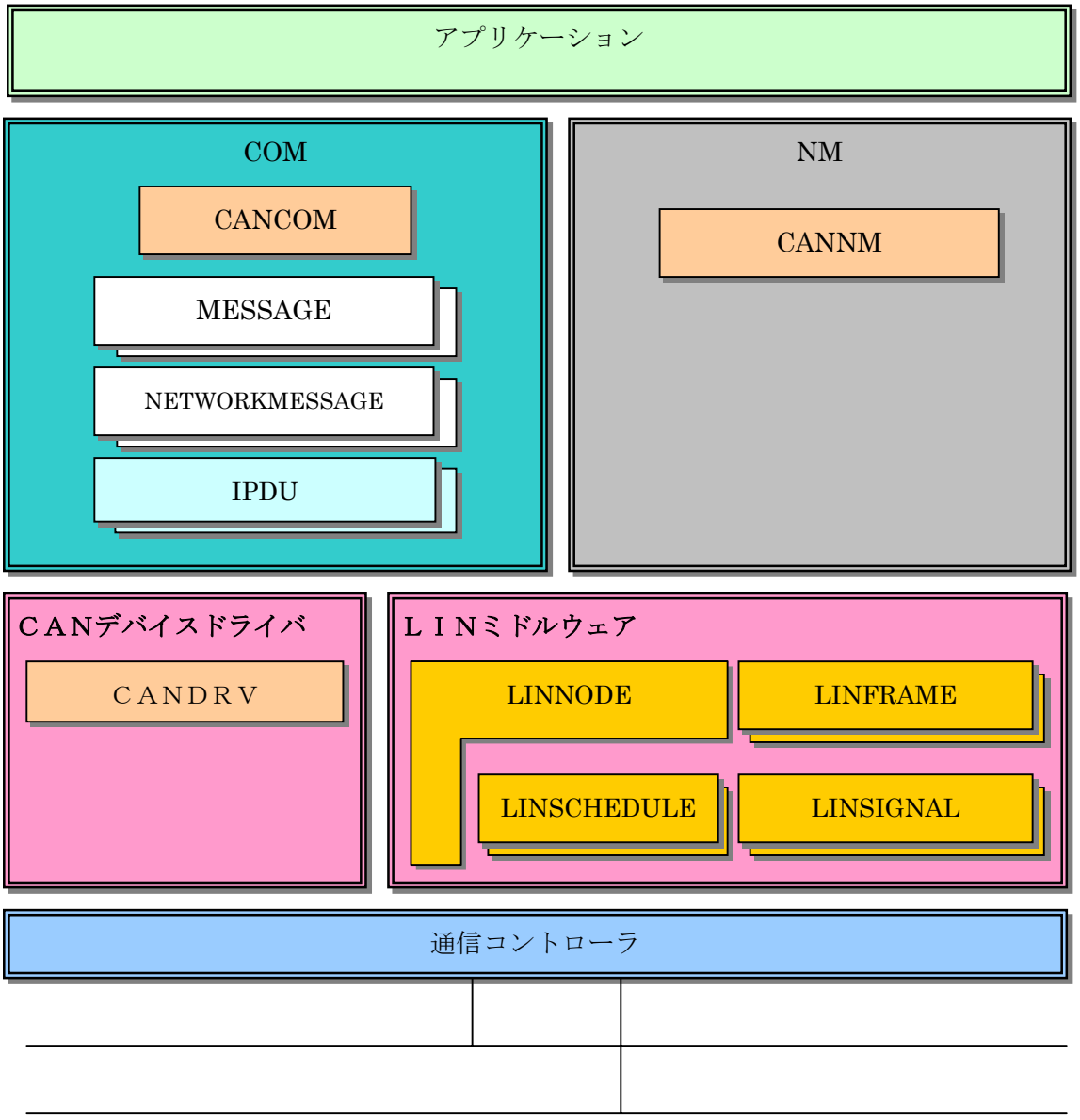
```
LINSCHEDULE linschedule {
      SENDFRAME = FORMAT {
                  FRAME               = FrmDataA;
                  DELAY               = 10000;
      };
      SENDFRAME = FORMAT {
                  FRAME               = FrmDataB;
                  DELAY               = 10000;
      };
      SENDFRAME = FORMAT {
                  FRAME               = FrmDataC;
                  DELAY               = 15000;
      };
};
```

3.11. オブジェクト関連図

通信ミドルウェア(OS 対応版)



3.11.1. 通信ミドルウェア(非 OS 対応版)



変更履歴

Version	Date	Detail	Editor
1.00	2007/02/27	・ 新規作成	ヴィッツ
1.01	2007/04/16	・ 表記統一	ヴィッツ
		・	