

SEMI E54.16-1106

LONWORKS によるセンサ／アクチュエータネットワーク通信の仕様

本スタンダードは、global Information & Control Committee で技術的に承認されている。現版は 2006 年 8 月 24 日、global Audits and Reviews Subcommittee にて発行が承認された。2006 年 10 月に www.semi.org で、そして 2006 年 11 月に CD-ROM で入手可能となる。初版は 2005 年 7 月発行に発行された。

注意: 本文書は 2005 年に SEMI E54.6 と差替えられた。

注意: 本文書は、SEMI E54 一連のスタンダードとして出版されず、単独で発行および販売される。

1 目的

1.1 本スタンダードの仕様では、半導体製造装置の SEMI 指定のデバイスモデル (共通仕様およびデバイス特定) に従って動作するセンサ／アクチュエータネットワーク (SAN) 上のインテリジェントなデバイス間の通信を可能とする ANSI/EIA/CEA-709.1 コントロールネットワークキングプロトコル (LONWORKS) に基づいた通信仕様を定義する。

1.2 本文書では、LONWORKS のデバイスに対して確立された *LONMARK Interoperability Guidelines* (*LONMARK 相互運用ガイドライン*) を使用した SEMI 共通デバイスモデル (CDM) の LONWORKS テクノロジーへのマッピングを指定する。LONMARK International Association は、このネットワーク通信に関する文書に記載された強化された SEMI LONMARK 機能プロファイルを検討、承認し、必要な SEMI プロファイルを *LONMARK インテオペラビリティガイドライン* に組み込む予定である。

1.3 背景および動機付け — LONWORKS 通信システムにより、高速な応答時間の低コストの工業用ネットワークで、センサ、アクチュエータ、コントローラなどのスマートな制御デバイスを相互接続できる。LONWORKS により、複数のデバイスが 1 つのネットワークを共有し、コントローラ、センサ、アクチュエータ間のポイントツーポイントの配線を大幅に削減できる。LONWORKS ネットワーク通信スタンダード (NCS) は、7 層の ANSI/EIA/CEA-709.1 (LONWORKS) プロトコルに基づき、ニューロンチップ、物理層のトランシーバおよびオプションのホストプロセッサによって実装される。ANSI/EIA/CEA-709.1 (LONWORKS) コントロールネットワークキングプロトコルは、Echelon によって開発され、任意のハードウェアプラットフォームに実装するために無償でライセンスできる。LONWORKS 用の SEMI の NCS は、LONMARK 相互運用ガイドラインに基づき、ANSI/EIA/CEA-709.1 (LONWORKS) プロトコルを 1~6 層およびアプリケーション層で相互に使用するためのフレームワークを提供する。LONMARK 相互運用ガイドラインが、SEMI 共通デバイスモデル (CDM) および SEMI 特定デバイスモデル (SDM) に必要な機能を提供していない場合は、そのガイドラインは SEMI 固有の強化されたプロファイル要求条件によって拡張される。

2 範囲

2.1 本文書では、ANSI/EIA/CEA-709.1 コントロールネットワークキングプロトコル (LONWORKS) 仕様に基づいた SAN 通信仕様スタンダードを規定し、これによって SEMI SAN 共通デバイスモデルおよび適切な SEMI SAN 特定デバイスモデル (SDM) の仕様に従って構成された SAN デバイスによる通信が可能となる。

2.2 本文書では、この半導体製造装置センサ／アクチュエータネットワークによって情報を交換するために、準拠しているインテリジェントなデバイスがサポートしなければならないサービスのための LONWORKS テクノロジーの使用方法を指定する。

2.3 本文書では、CDM および適切な SDM で指定されているデバイスの構造および挙動を LONWORKS ネットワーク上で表現するための LONWORKS テクノロジーの利用方法を定義する。

2.4 本文書は、SEMI SAN 共通デバイスモデル仕様および 1 つ以上の SEMI SAN 特定デバイスモデル仕様 (例えば、マスフローコントローラ) と共に使用される。同時に、このモデルの文書では、SEMI 準拠の SAN システム内で LONWORKS テクノロジーを使用してデバイスの構造および挙動を記述する。

2.5 この文書では、センサ／アクチュエータネットワークの共通デバイスモデルと1つ以上のセンサ／アクチュエータネットワークの特定デバイスモデルの文書、ANSI/EIA/CEA-709.1 (LONWORKS)プロトコル仕様、およびLONMARKインタオペラビリティガイドラインと併せ、LONWORKSテクノロジーに基づくSEMI SAN実装のための要求条件を指定する。

注意: このスタンダードはその使用に関連したすべての安全問題を取り扱うことを意図していない。このスタンダードの使用者は、その責任において、適切な安全および健康上実施すべき事柄を確立し、また使用前に法規制やその他の制限への適用性を判断するものである。

3 制限

3.1 本文書は、LONWORKSテクノロジーのみに基づいた半導体製造装置SANを規定している、そしてANSI/EIA/CEA-709.1 (LONWORKS)コントロールネットワークプロトコル仕様の付属書である。したがって、本スタンダードの完全な仕様には必然的にLONWORKS仕様が含まれる。半導体製造装置の他のSAN通信オプションがある。これらのオプションの仕様は、ここには含まれていない。

3.2 本文書は、LONMARKインタオペラビリティガイドラインに現在必要な機能を超える追加機能を提供する強化点を指定する。文書の一貫性に関する問題を避けるために、本文書に関連するLONWORKSテクノロジーの情報については、本文書では重ねて記載しない。本文書では、LONWORKSテクノロジーの強化点や制限事項および本仕様によって課せられるLONMARKインタオペラビリティガイドラインを記述することだけに制限している。

3.3 コンFORMANCEテストプロシージャの完全な仕様には、適切なLONMARKインタオペラビリティテストの仕様が含まれること。CONFORMANCEテストには、本スタンダード仕様に必要なLONMARKインタオペラビリティガイドラインに対する強化点が含まれること。

4 参照スタンダードおよび文書

4.1 SEMI スタンダード

SEMI E30 — Generic Model for Communications and Control of SEMI Equipment (GEM)

SEMI E39 — Object Services Standard: Concepts, Behavior, and Services

SEMI E54 — Sensor/Actuator Network Standard

SEMI E54.1 — Standard for Sensor/Actuator Network Common Device Model

SEMI E54.3 — Specification for Sensor/Actuator Network Specific Device Model for Mass Flow Device

SEMI E54.10 — Specification for Sensor/Actuator Network Specific Device Model for An In-Situ Particle Monitor Device

SEMI E54.11 — Specification for Sensor/Actuator Specific Device Model for an Endpoint Device

SEMI E54.18 — Specification for Sensor/Actuator Network Specific Device Model for Vacuum Pump Device

4.2 ISO スタンダード¹

ISO 7498 — Basic Reference Model for Open Systems Interconnection

ISO/CEN EN14908 Specification for Control Networking Protocol

ANSI/EIA/CEA-709.1 (LONWORKS) Control Networking Protocol Specification, Rev B-2000

4.3 その他の文書

LONMARK Layers 1–6 Interoperability Guidelines, LONMARK International.

LONMARK Application Layer Interoperability Guidelines, LONMARK International.

Neuron Chip Data Book, Toshiba and Cypress Corporation.

Smart Transceiver Data Book, Echelon Corporation.

¹ International Organization for Standardization, 1 rue de Varembe, Case postale 56, CH-1211, Geneva 20, Switzerland



Neuron C Programmer's Guide, Revision 2, Echelon Corporation

Neuron C Reference Guide, Revision 2, Echelon Corporation

Standard Device Resource Report, <http://types.lonmark.org>, LONMARK International.

Standard Program ID Definitions, <http://types.lonmark.org/spid>, LONMARK International.

SCPT Master List and Programmer's Guide, <http://types.lonmark.org/scpt>, LONMARK International.

SNVT Master List and Programmer's Guide, <http://types.lonmark.org/snvt>, LONMARK International.

注意: 別途指示のない場合, 引用されたすべての文書は最新の出版バージョンである。

5 用語

5.1 このSANシリーズのすべての文書に共通な用語は, センサ/アクチュエータネットワークスタンダード (SEMI E54 § 4.1 参照) でも定義されていることがある。用語によっては, 他のSEMI文書で定義されている用語と重複していることがある。

5.2 略語および頭字語

5.2.1 *APDU (Application Protocol Device Unit)* – アプリケーションプロトコルデバイスユニット

5.2.2 *CDM (Common Device Model)* – 共通デバイスモデル

5.2.3 *CP (Configuration Parameter)* – 構成パラメータ

5.2.4 *DM (Device Manager)* – デバイスマネージャ

5.2.5 *DS (Device Status)* – デバイスステータス

5.2.6 *NCS (Network Communications Standard)* – ネットワーク通信スタンダード

5.2.7 *Neuron* – ニューロンチップは, ANSI/EIA/CEA-709.1 (LONWORKS) コントロールネットワークプロトコルの 8 ビットのハードウェア実装である。

5.2.8 *NV (Network Variable)* – ネットワーク変数

5.2.9 *NVI (Network Variable Input)* – ネットワーク変数入力

5.2.10 *NVO (Network Variable Output)* – ネットワーク変数出力

5.2.11 *OSI (Open Systems Interconnect)* – オープンシステム相互接続

5.2.12 *OSS (Object Services Standard)* – オブジェクトサービススタンダード

5.2.13 *RO (Read Only)* – 読み出し専用

5.2.14 *RW (Read/Write)* – 読み/書き

5.2.15 *SAC (Sensor, Actuator, Controller (object))* – センサ, アクチュエータ, コントローラ (オブジェクト)

5.2.16 *SAN (Sensor/Actuator Network)* – センサ/アクチュエータネットワーク

5.2.17 *SCPT (Standard Configuration Parameter Type)* – 標準構成パラメータタイプ

5.2.18 *SDM (Specific Device Model)* – 特定デバイスモデル

5.2.19 *SNVT (Standard Network Variable Type)* – 標準ネットワーク変数タイプ

5.2.20 *SPID (Standard Program ID)* – 標準プログラムID

5.3 デバイスコンポーネントの定義 – 本スタンダードでは, ネットワークでのCDMデータ構造と挙動の表現やマッピングを定義するので, SEMI E54.1-CDM文書の多くの用語を利用する。表 1 には, CDM文書の基礎用語の本文書への, およびLONWORKS定義へのマッピングを示す。記号“=”は, その定義がCDM仕様での指定とまったく同様に使用されることを示す。

5.4 以降の項では、ANSI/EIA/CEA-709.1 (LONWORKS) プロトコルの文脈において、これらのいくつかの用語をさらに明確にする。

表 1 CDM の NCS への用語のマッピング

CDM の用語	NCS 等価	LONWORKS 等価
デバイス	=	デバイスまたはノード
デバイスモデル	=	デバイスインタフェース
オブジェクト	=, Class	機能プロファイル, クラス
インスタンス	=	機能ブロック
属性	=	ネットワーク変数または構成プロパティ
挙動	=	=
サービス	=	ネットワーク変数関数またはアプリケーションメッセージ
状態図	=	=
バイト	=	=
ニブル	=	=
文字列	=	ASCII 文字列または国際文字列

5.4.1 属性 (attributes) — 属性は、入力ネットワーク変数 (NVI)、出力ネットワーク変数 (NVO)、コンフィグレーションプロパティ (CP) のどれかである。入力ネットワーク変数、出力ネットワーク変数、コンフィグレーションプロパティは、デバイス自体によって読み書きされ、すべての属性はネットワーク経由でポーリングできる。さらに、入力ネットワーク変数とコンフィグレーションプロパティはネットワーク経由で更新でき、そのような更新を受信するとデバイスのアプリケーション層にイベントを送信できる。これは、ネットワーク変数を所有するオブジェクトの RW (読み書き) 属性に対応する。出力ネットワーク変数は、ネットワーク経由では更新できない。これは、ネットワーク変数を所有するオブジェクトの RO (読み出し専用) 属性に対応する。デバイス自体がその出力ネットワーク変数のどれかを更新すると、その変数の値をネットワーク経由でインストール時に決定された送信先アドレスに送信できる。最後に、コンフィグレーションプロパティは、通常は不揮発性メモリに格納される属性であり、デバイスをリセットしたり、電源を入り切りしても保持される。

5.4.2 挙動 (behavior) — オブジェクトの一般的な挙動は、LONMARK アプリケーション層インタオペラビリティガイドラインによって指定される。オブジェクト固有の追加の挙動は、機能プロファイルの手段によって指定される。

5.4.3 デバイス (device) — デバイス (またはノード) は、通常、ANSI/EIA/CEA-709.1 (LONWORKS) プロトコルの物理層を実装する 1 つのネットワークランシーバ、LONTALK プロトコルのその他の層を実装する関連ファームウェアを搭載した 1 つのニューロンチップ、およびデバイスから外部のセンサやアクチュエータハードウェアの物理インタフェースを実装する入力 / 出力ハードウェアから構成される。LONWORKS デバイスは、オプションで ANSI/EIA/CEA-709.1 (LONWORKS) プロトコルのアプリケーション層を実装するホストプロセッサと関連ソフトウェアまたはファームウェアを含むことができる。

5.4.4 デバイスモデル (device model) — デバイスモデルは、インタオペラビリティ可能なネットワークについてデバイスの外部インタフェースを完全に記述する複数の要素から構成される。インタフェースは、次の要素から作られる。それらは、デバイスマネージャ (DM) オブジェクト、センサ / アクチュエータ / コントローラ (SAC) オブジェクトおよびセンサ、アクチュエータ、コントローラなどの機能ブロック、個々のネットワーク変数、および構成プロパティである。

5.4.5 インスタンス (instance) — 真のデバイスは、定義されたそれぞれの機能ブロックのゼロ個以上のインスタンスを持つことができる。オブジェクトのインスタンスは、デバイス内のインスタンス番号によって識別される。

5.4.6 オブジェクト (object) — 標準ネットワーク変数タイプ (Standard Network Variable Types) として実装される 1 つまたは複数のネットワーク変数入力やネットワーク変数出力の集合および標準構成プロパティ型 (Standard Configuration Property Types) として実装される構成プロパティの集合として定義される機能ブロック。機能ブロックは、アプリケーション層におけるインタオペラビリティの基礎を形成する。機能ブロックは、情報がデバイスに入力さ

れる方法、情報がデバイスから出力される方法、ネットワーク上の他のデバイスと情報を共有する方法について標準のフォーマットを記述する。

5.4.7 サービス(service) — 要求サービスは、デバイスアプリケーションに設定され、送出されるANSI/EIA/CEA-709.1 入力ネットワーク変数(NVI)によって表現される。通知サービスは、デバイスアプリケーションに設定され、ネットワーク全体に送出されるANSI/EIA/CEA-709.1 出力ネットワーク変数によって表現される。

5.4.8 状態遷移図(state diagram) — LONWORKSデバイス内では、状態はアプリケーションプログラムのローカル変数およびネットワーク変数の値の集積によって表現される。状態間の遷移は、外部イベント(ネットワーク変数更新の受信やその他のI/Oイベントなど)または内部イベント(タイマ切れなど)の結果である。

5.5 LONWORKS固有の定義(LONWORKS-Specific Definitions) — ビット, 1/2 バイト, バイト, 文字に対する標準のデータ型の定義に加え, ANSI/EIA/CEA-709.1 (LONWORKS) プロトコルでは, 属性値として使用するための標準データ表現の集合を定義している。

5.5.1 バインディング(binding) — 同じデバイスまたは異なるデバイスのネットワーク変数をバインディングとして知られるネットワーク管理サービスの手段によって互いに関連付けることができる。バインディングは、その集合内のすべてのネットワーク変数が同じデータ型である場合のみ可能である。バインドされているネットワーク変数の値はANSI/EIA/CEA-709.1 (LONWORKS) プロトコルによってネットワークに送信される。表 2 に、更新やポーリングが可能なネットワーク変数の組み合わせを示す。

表 2 ネットワーク変数の更新およびポーリング

ネットワーク変数クラス	ネットワークからの更新	デバイスからの更新	ネットワークからのポーリング	デバイスからのポーリング
Input	はい	はい	はい	はい ^{注2}
Output	いいえ	はい ^{注1}	はい	いいえ

注 1 デバイスがそれ自体の出力ネットワーク変数の 1 つを更新する場合、この出力ネットワーク変数にバインドされている入力ネットワーク変数がネットワークから更新を受信する。

注 2 デバイスがそれ自体の入力ネットワーク変数または構成ネットワーク変数をポーリングする場合、この入力ネットワーク変数または構成ネットワーク変数にバインドされている出力ネットワーク変数がネットワークからポーリングを受信する。

5.5.2 コンフィギュレーションプロパティ(configuration properties) — センサのゲインおよびオフセット, リニアライズテーブル, サンプリング速度など, 機能ブロックのアプリケーション固有の挙動を構成するために使用する機能ブロックの属性がある。これらの属性は、通常、デバイスが取り付けられた、環境設定された、校正された、あるいは不揮発性メモリに保存されたときに更新される。

5.5.3 デバイスインタフェース(device interface) — デバイスインタフェースは、機能ブロックの挙動をネットワーク変数の値に関連付けるよう定義づける、1 つまたは複数の機能ブロックの仕様である。1 つのデバイス内の機能ブロックのコレクションは、そのデバイスのSEMI SANデバイス固有モデルに対応する。それぞれのタイプのデバイスインタフェースは、標準プログラムID (SPID) によって識別される。

5.5.4 ネットワーク変数(network variable) — これは、明確に定義されたデータタイプを有するデバイスのネットワークとして認識されるデータ属性である。ネットワーク変数は、入力変数, 出力変数, コンフィギュレーションネットワーク変数のどれかである。ネットワーク変数の値は、デバイス自体あるいは何らかの他のデバイスからネットワーク経由で更新できる。これは、SEMI固有のサービスであるSetAttribute操作に対応する。ネットワーク変数の値は、何らかの他のデバイスによってネットワーク経由でポーリングするか、デバイス自体によって取り出すことができる。これは、SEMI固有のサービスであるGetAttribute操作に対応する。

5.5.5 機能ブロック(functional block) — 機能ブロックは機器のアプリケーションの一部であり、構成設定や動作データ入力を受け付け、そのデータを処理し、動作データ出力を送ることで作業を遂行するものである。機能ブロック1個が機能的な様態の実装1つに対応するものである。デバイスの外部インタフェースの文書では、デバイス内に格納されている機能ブロックの固有識別子を指定する。この文書は、デバイスからアップロードでき、デバイスによって実装された機能プロファイルおよび各機能ブロック内に格納されているネットワーク変数およびコンフィギュレーションプロパティを完全に指定できる。LonMark International 発行の 'LonMark Application Layer Interoperability Guidelines' の § 4.3 を参考として参照されたい。

5.5.6 機能プロファイル (functional profile) — 機能プロファイルは、抽象エンティティの属性および挙動の定義である。それぞれの型の機能ブロックは、機能プロファイル番号によって識別される。特定のデバイス型は、これらの 1 つまたは複数の機能プロファイルのインスタンスから成る。機能ブロックは、ネットワーク変数および構成プロパティのコレクションとして実装される。

5.5.7 標準構成プロパティタイプ (standard configuration property types) — CSPTとしても知られるこれらのデータタイプは、機能ブロックのコンフィギュレーションプロパティのデータタイプの定義および挙動の意味を定義する。使用可能なすべてのSCPTおよびその定義の詳細のリストは、SCPT マスタリストおよびプログラマーズガイド (SCPT Master List and Programmer's Guide) に記載されている。

5.5.8 標準ネットワーク変数タイプ (standard network variable types) — SNVTとしても知られるこれらのデータタイプは、異なるメーカーによって製造されたデバイス間の通信のために明確に定義されたインタフェースを提供することによってインタオペラビリティを容易にする。デバイスをネットワークにインストールでき、データタイプが一致する限り、ネットワーク変数により他のデバイスに論理的に接続できる。使用可能なすべてのSNVTおよびその定義の詳細のリストは、SNVT マスタリストおよびプログラマーズガイド (SNVT Master List and Programmer's Guide) に記載されている。

6 通信プロトコルの上位構造

6.1 ANSI/EIA/CEA-709.1 (LONWORKS) プロトコルは、7 層のアーキテクチャに基づいている。各層において、その層内で提供されるサービスの記述がある。上位プロトコルアーキテクチャは図 1 に示す通りである。

6.1.1 図 1 は、デバイスのアーキテクチャを概念的に示した図である。実装においては、通常、ニューロンチップおよびその関連ファームウェアを使用し、準拠した実装を 2 層～6 層まで提供する。LonMark インタオペラビリティガイドラインでは、物理層 (ネットワークトランシーバ) およびアプリケーション層 (オブジェクトモデル) で使用されるプロトコルオプションを最も明確に指定する。

6.2 物理層 — LONMARK 1～6 層インタオペラビリティガイドラインで指定されているように、デバイスは LONMARK 承認の物理チャンネルのどれかを採用しなければならない。LONWORKS ベースの SEMI SAN 準拠のデバイスは、ネットワーク接続には 2 ピンねじ込み端子オープン接続可能なコネクタ (Weidmüller-Klippon SL2, Phoenix Combicon, または相当品) をデフォルトで使用しなければならない。特定のデバイスタイプについて特殊な要求条件が適用される場合には、デフォルトのコネクタ仕様に優先するコネクタを使用できるが、そのようなコネクタは §9, 「特定のデバイスタイプの情報」に記載されていないなければならない。この接続には、極性を区別する。半導体製造装置の要求条件は、次に記載するツイストチャンネル仕様のどれかと合致していることがある。

6.2.1 TP/XF-1250 ツイストペア — このツイストペアチャンネルは、1,250kbps のビットレートで動作し、トランス結合トランシーバを使用したバスプロジをサポートする。

6.2.2 TP/FT-10 ツイストペア — このツイストペアチャンネルは、78kbps のビットレートで動作し、フリースプロジとバスプロジの配線の両方およびオプションのリンクパワーをサポートする。

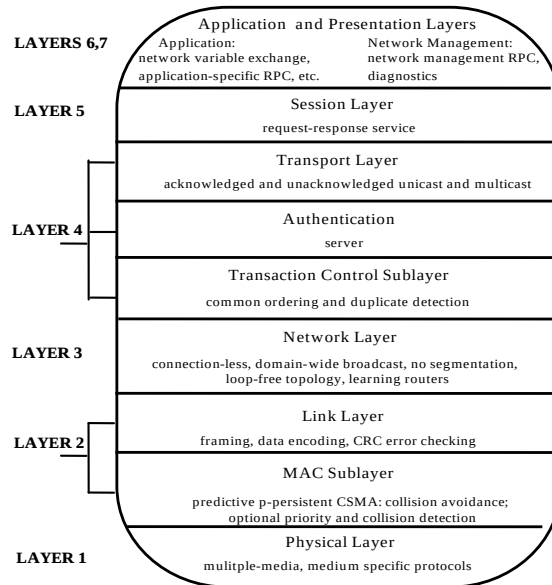


図 1

ANSI/EIA/CEA-709.1 (LonWorks) プロトコルの層構造

6.2.3 LONMARK 1～6 層インタオペラビリティガイドラインには、これらのトランシーバの特性に固有な詳細が記載されている。この文書には、デバイスのインタオペラビリティを保証するために使用する配線のタイプおよび相互接続トポロジの仕様も記載されている。ANSI/EIA/CEA-709.1 (LonWorks) プロトコルは、異種のネットワークをサポートすることに注意すること。異なるトランシーバを有するデバイスは、ルータやリピータ経由で相互接続し、通信できる。同様に、ルータやリピータを使用して、デバイス数、配線長、あるいは選択されたトランシーバに伴うその他の物理的制限を超えて物理チャンネルを拡張できる。

6.2.4 ANSI/EIA/CEA-709.1 (LonWorks) プロトコルでは、複数の物理層プロトコルおよびデータコード化方法が使用される。ツイストペア物理層では差分マンチェスタコード化が使用される。

6.3 リンク層 — デバイスは、ANSI/EIA/CEA-709.1 (LonWorks) プロトコルのリンク層の仕様に準拠しなければならない。この層には、メディアアクセスコントロールサブ層が含まれる。単純性やマルチキャストプロトコルとの互換性も含め、多くの理由により、ANSI/EIA/CEA-709.1 (LonWorks) プロトコルは、シンプルなコネクションレスのサービスをサポートする。その機能は、フレーミング、フレームコード化、エラー検出に制限されている。

6.3.1 メディアアクセスコントロールサブ層 — 潜在的に衝突検出がないさまざまなメディアを取り扱うために、MAC (メディアアクセスコントロール: Media Access Control) サブ層は、Predictive *p*-persistent CSMA (キャリアセンス、マルチプルアクセス) と呼ぶ衝突回避アルゴリズムを採用する。

6.4 ネットワーク層 — デバイスはANSI/EIA/CEA-709.1 (LonWorks) プロトコルのネットワーク層の仕様に準拠しなければならない。この層は、単一のドメイン内でのパケットの送出を取り扱い、ドメイン間通信については規定をしない。ネットワークサービスは、コネクションレスであり、確認応答をおこなわず、メッセージのセグメント化や再構築はサポートしていない。トポロジを学習するためにネットワーク層によって採用されているルーティングアルゴリズムでは、ツリーのようなネットワークトポロジを仮定しており、構成テーブルを有するルータは通信パスが論理的にツリー状になっている限りトポロジ上で動作できる。この構成では、パケットを送信する側のルータにはパケットが複数回現れてはならない。ユニキャストルーティングアルゴリズムでは、オーバーヘッドを小さくし、ルーティングによるトラフィックをさらに発生させないために学習を使用する。構成されたルーティングテーブルの使用は、ユニキャストおよびマルチキャストのアドレスの両方についてサポートされている。

6.5 トランスポート層 — デバイスはANSI/EIA/CEA-709.1 (LonWorks) プロトコルのトランスポート層の仕様に準拠しなければならない。プロトコルの階層構造の心臓部は、トランスポート層とセッション層である。共通のトランザクションコントロールサブ層では、トランザクションのオーダーと両方の層の重複検出を取り扱う。トランスポート層はコネクションレスであり、単一の送信先と複数の送信先の両方への信頼性の高いメッセージ送信を提供する。メ

メッセージの送信者のIDの認証は、オプションの機能として提供される。認証サーバでは、トランザクションコントロールサブ層だけがこの機能を達成する必要がある。トランスポート層とセッション層のメッセージは、ブロードキャスト以外のANSI/EIA/CEA-709.1のすべてのアドレッシングモードを使用して認証できる。トランスポート層では、エンドツーエンドの応答確認されるサービスおよび認証なしの繰り返しサービスをサポートする。

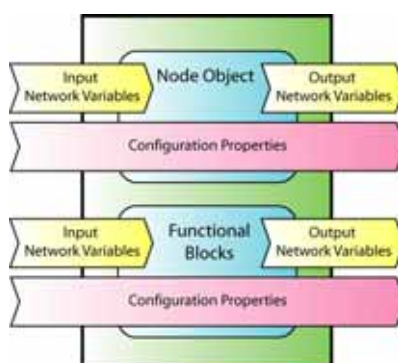
6.6 セッション層 — デバイスはANSI/EIA/CEA-709.1 (LonWORKS) プロトコルのセッション層の仕様に準拠しなければならない。この層では、リモートサーバにアクセスするためのシンプルな要求／応答メカニズムを実装する。このメカニズムは、アプリケーション固有のリモートプロシージャコールを組み込むことができるプラットフォームを提供する。ANSI/EIA/CEA-709.1 (LonWORKS) ネットワーク管理プロトコルは、例えば、アプリケーション層のインタフェースによってプロトコルにアクセスしてもセッション層の要求／応答メカニズムに依存している。

6.7 プレゼンテーション層 — デバイスはANSI/EIA/CEA-709.1 (LonWORKS) プロトコルのプレゼンテーション層の仕様に準拠しなければならない。プレゼンテーション層とアプリケーション層をいっしょにとるとLonWORKSデバイスの相互運用の基礎を形成する。アプリケーション層は、メッセージの送受信の通常のすべてのサービスを提供するが、ネットワーク変数の概念も含んでいる。プレゼンテーション層は、アプリケーションプロトコルデータユニット(APDU)のヘッダに、ネットワーク変数の更新についてAPDUの解釈の仕方に関する情報を提供する。このアプリケーションに依存しないデータの解釈により、事前にアレンジしなくてもデバイス間でデータを共有できる。異なるメーカのセンサ、アクチュエータなどのインテリジェントなコンポーネントについて使用するネットワーク変数について合意することによって、お互いの特性を事前に知らなくても協調して動作することができる。

6.8 アプリケーション層 — アプリケーション層では、機能ブロックおよび標準ネットワーク変数タイプ (SNVT) を使用することによって、LonWORKSベースのデバイス間の相互運用が容易になる。機能ブロックはネットワーク変数上に構築し、特定のデバイス機能のための意味を組み込む簡明なアプリケーション層インタフェースを提供する。機能ブロックは、データを搬送するために使用するSNVTを定義するだけでなく、通信する情報の意味も提供する。明解に定義された挙動による特定のデバイスモデルの仕様を支援するために、機能ブロックインタフェースと意味は機能プロファイルによって定義される。アプリケーション層には、LonWORKSファイル転送プロトコルも含まれ、データの任意の長さのファイルのセグメント化および再構築を提供する。このサービスを使用して、31 バイトのネットワーク変数のサイズの制限を超えるオブジェクトの属性を取得したり、設定したりできる。

6.8.1 オブジェクトモデル — ANSI/EIA/CEA-709.1 (LonWORKS) プロトコルは、オブジェクト属性とイベントの表現であるネットワーク変数およびコンフィギュレーションプロパティを定義し、アドレッシングするオブジェクト指向の仕様を提供する。デバイスは、本文書の § 7 で定義されるオブジェクトモデル仕様に準拠しなければならない。

6.8.2 機能ブロック構造体 — LonMARK アプリケーション層インタオペラビリティガイドラインでは、機能プロファイルの構造体を定義する。各機能プロファイルは、必須ネットワーク変数の集合、オプションのネットワーク変数の集合、コンフィギュレーションプロパティの集合 (必須とオプションの両方)、プロファイルに非運用性の拡張をおこなうことができるメーカ定義のセクションを持つことができる。これを図 2 に示す。この表記法は、LonMARK アプリケーション層インタオペラビリティガイドラインで定義されている。



注: 図の表記, 図 2 で定義されている矢印のような記号は LonMark アプリケーション層インタオペラビリティガイドラインで定義されている。

図 2
LonMark オブジェクトの構造

6.8.3 LONMARK アプリケーション層インタオペラビリティガイドラインは、標準ネットワーク変数タイプ、標準コンフィギュレーションプロパティタイプ、機能プロファイルを定義するために提供されている。§ 7 の SEMI CDM の LONMARK オブジェクト構造体へのマッピングでは、現在の SNVT リストおよび LONMARK インタオペラビリティガイドラインに対する拡張はアスタリスク(*)でマーキングされている。機能ブロック番号はガイドラインによって指定され、1 つのデバイスはノード Object タイプの 1 つのインスタンスおよび他の機能プロファイルの 1 つのまたは複数のインスタンスで構成できる。

6.9 ネットワーク管理 — ANSI/EIA/CEA-709.1 (LONWORKS) プロトコルでは、LONWORKS デバイスの完全なネットワーク管理および診断プロトコルが定義される。このプロトコルは、セッション層 (要求/応答サービス) より上の層であり、アプリケーションのダウンロード、デバイスアドレスの割り当て、暗黙のメッセージ送信のための送信先アドレスの配布、ルータの構成、デバイスレベルの診断のためのメカニズムを提供する。LONMARK アプリケーション層インタオペラビリティガイドラインでは、機能ブロックのためのデバイス管理層を定義する。

7 必須およびオプションのオブジェクトタイプ

7.1 LONMARK ガイドラインでは、複数の機能ブロックを含むデバイスにはノードオブジェクトの機能ブロックが必要であることを除き、LONMARK デバイスに準拠するためにデバイス内に特定のオブジェクトが存在している必要はない (SEMI 準拠のデバイスには、通常、ノードオブジェクトの機能ブロックがある)。本スタンダードでは、新しい LONMARK 標準プロファイルが定義され、LONMARK SEMI 準拠の SAN ネットワークにおいて相互運用可能かつインターチェンジ可能とするデバイス内に存在しなければならない機能ブロックを特定し、説明する。

7.1.1 LONMARK INTERNATIONAL ASSOCIATION は、さまざまなセンサ、アクチュエータ、コントロールオブジェクトの機能プロファイルを出版する。特定のデバイスは、これらのプロファイルに基づいた機能ブロックを使用して実装できる。共通デバイス仕様では、すべての SEMI 準拠の SAN デバイスに存在しなければならない 2 つの機能ブロック (具体的には、デバイスマネージャ (DM) およびセンサアクチュエータコントローラ (SAC) オブジェクト) を特定する。ここで記載されるネットワーク通信仕様を利用する SEMI 準拠 SAN デバイスに必要な機能プロファイルは、最低限、LONMARK 機能プロファイルの要求条件および CDM 仕様の要求条件の和集合を必然的に備えている。

7.1.2 必須およびオプションのオブジェクトタイプのリストを表 3 に示す。特定の SDM 内で指定される追加のオブジェクトは、その SDM 仕様内で与えられる識別子である。これらの識別子に対する LONMARK 固有の表現情報は、本文書の § 9 に記載されている。

表 3 共通デバイスモデルの必須およびオプションのオブジェクトタイプ

SEMI オブジェクト名	LONMARK SEMI クラス ID/ インスタンス ID ^{注1}	CDM タグ ^{注2}	LONMARK 機能 プロファイル名	LONMARK ^{注1} に必須	CDM ^{注2} に必須	NCS に必須
(DM) Device Manager ^{注4}	0 / 1	DmIO	ノードオブジェクト	はい	はい	はい
(SAC) Sensor/Actuator/ Controller ^{注4}	0 / 1	SacIO	ノードオブジェクト	はい	はい	はい
Assembly	180.81 / 1~i	Asm	メーカ固有の番号	いいえ	いいえ	いいえ
Local Link	ターンオーバーコネクション ^{注3}	Lnk	ターンオーバーコネクション ^{注3}	いいえ	いいえ	いいえ
Sensor-AI	180.11 / 1~k	Sai	SFPTsemiSensorAI	いいえ	いいえ	いいえ
Sensor-EI	180.22 / 1~l	Sei	SFPTsemiSensorEI	いいえ	いいえ	いいえ
Sensor-BI	180.18 / 1~m	Sbi	SFPTsemiSensorBI	いいえ	いいえ	いいえ
Actuator-AO	180.31 / 1~n	Aao	SFPTsemiActuatorAO	いいえ	いいえ	いいえ
Actuator-EO	180.34 / 1~o	Aeo	SFPTsemiActuatorEO	いいえ	いいえ	いいえ

SEMI オブジェクト名	LONMARK SEMI クラスID/ インスタンスID ^{注1}	CDM タグ ^{注2}	LONMARK 機能 プロファイル名	LONMARK ^{注1} に必須	CDM ^{注2} に必須	NCS に必須
Actuator-BO	180.33 / 1~p	Abo	SFPTsemiActuatorBO	いいえ	いいえ	いいえ
Controller	180.51 / 1~q	C	SFPTsemiController	いいえ	いいえ	いいえ
Sensor-BI-TH	180.19 / 1~s	Sbith	SFPTsemiSensorBITH	いいえ	いいえ	いいえ

注 1 LonMarkは、SEMIクラスIDを参照 180 としてグループ化し、特定のオブジェクト名のカテゴリをXXとして指定する。LONMARKアプリケーション層インタオペラビリティガイドラインおよび<http://types.lonmark.org>を参照すること。

注 2 詳細については、E54.1 – CDM 仕様を参照すること。

注 3 ターンオーバーパコネクションは、プロファイルではなく、リンクを作成するためのプロトコル固有のメソッドである。

注 4 LonMark アプリケーション層は、SEMI デバイスマネージャ(DM)およびセンサ/アクチュエータ/コントローラ(SAC)オブジェクトをLonMark ノードオブジェクトにマッピングする。

7.1.3 サービス要求コード — ほとんどのサービス要求は、指定された属性に対応するネットワーク変数にアドレスリングされるネットワーク変数更新として実装される。このマッピングに対する例外は、31 バイトを超える応答が必要な要求である。その場合、その要求はデバイスのノードオブジェクトの機能ブロックへのnviSemiReq入力に対するネットワーク変数更新として実装される。この入力は、次のようにSNVT_semi_reqタイプによって定義される。

```
typedef enum {
    struct {
        CMD_RESET                = 1,
        CMD_ABORT                = 2,
        CMD_RECOVER              = 3,
        CMD_GET_ATTRIBUTE        = 4,
        CMD_SET_ATTRIBUTE        = 5,
        CMD_OERATE               = 6,
        CMD_RESTOTE_DEFAULT     = 7,
        CMD_PUBLISH_ATTRIBUTE    = 8,
        CMD_LOCK                 = 9,
        CMD_UNLOCK              = 10,
        CMD_GET_EXCEPTION_QUEUE  = 11,
        CMD_CLEAR_EXCEPTION_QUEUE = 12,
        CMD_EXECUTE              = 13,
        CMD_PERFORM_DIAGNOSTICS  = 14,
    } semi_request_t;
    typedef struct {
        unsigned int flblock_index;
        semi_request_t service_code;
        unsigned int parameter;          // Optional parameter
    } SNVT_semi_req;
```

7.1.4 デバイスは、ノードオブジェクト機能ブロックのnvoSemiStat出力およびnvoFileStat出力を更新することによって、SNVT_semi_req要求を受け入れる。nvoSemiStat出力は、次のようにSNVT_semi_statusタイプによって定義される。

```
typedef enum {
    struct {
```

```
STAT_SUCCESS                = 0,
STAT_UNSUPPORTED_SERVICE    = 1,
STAT_UNSUPPORTED_ATTRIBUTE   = 2,
STAT_SET_ATTRIBUTE_ERROR     = 3,
STAT_GET_ATTRIBUTE_ERROR     = 4,
STAT_CALIBRATION_ERROR      = 5,
STAT_SERVICE_REQUEST_ERROR   = 6,
STAT_DIAGNOSTIC_ERROR       = 7,
} semi_status_t;
typedef struct {
semi_status_t status;
    unsigned long selected_file;
    struct address {          // Address of the requesting device
        unsigned short domain_id[6];
        unsigned short domain_length;
        unsigned short subnet;
        unsigned short node;
    }
} SNVT_semi_status;
```

7.1.5 **nvoSemiStat**出力および**nvoFileStat**出力が、その要求が受け入れられたことを示した場合、要求元のデバイスは**nvoSemiStat**応答によって報告されるファイルインデックスを有するファイルを送信することによって要求されたデータをフェッチする。

7.1.6 **オブジェクト属性** — SEMI GetAttributeサービス要求およびSEMI SetAttributeサービス要求は、指定された属性に対応するネットワーク変数にアドレッシングされるネットワーク変数のフェッチ、ポーリング、および更新の要求として実装される。これは、31 バイトを超える要求が不要である場合には適正である。GetAttributeサービス要求は、LonTalkプロトコルのネットワーク管理のNVフェッチメカニズムを使用して、LonTalk要求メッセージとして任意のネットワーク変数に直接アドレッシングできる。GetAttributeサービス要求も、NV選択メカニズムを使用し、LonTalk NVポーリングメッセージとして出力ネットワーク変数にアドレッシングできる。SetAttributeサービス要求は、NV選択メカニズムを使用し、LonTalk NV更新メッセージとして入力ネットワーク変数にアドレッシングできる。SetAttribute(ネットワーク変数更新)の確認は、LonTalkプロトコルのトランスポート層の確認応答されるサービスによって提供される。

7.1.7 **シーケンス番号** — 機能ブロック内の各ネットワーク変数は、デバイスのメモリに格納される自己文書化文字列の手段によって識別される。この文字列には、この変数が属する機能ブロックの機能ブロックインデックスおよび機能ブロック内のネットワーク変数のシーケンス番号が格納される。LONWORKSのNCSについては、このシーケンス番号はCDMタグによって指定される数値シーケンス番号と同じである。例として、デバイス内の 2 番目のオブジェクトのインスタンスとして宣言されるDevice Managerオブジェクトのインスタンスを使用すると、1 のインスタンスIDを有する。Device Managerの属性、Standard Revision Levelは、DmA2 というタグを有する。したがって、このネットワーク変数の自己文書化文字列は、“@12”と指定される。

7.1.8 **発行サービス** — 発行通知サービスは、出力ネットワーク変数が更新される場合には暗黙のものとなる。デバイスは、出力ネットワーク変数(読み出し専用属性と等価)の値を、バインディングできるすべての入力ネットワーク変数へ送信する。ANSI/EIA/CEA-709.1 (LONWORKS)プロトコルは、出力ネットワーク変数の伝搬のみをサポートする。CDMの用語では、このことは読み出し専用の属性だけが発行できるということを意味する。特定のデバイスモデルに、読み書きの属性の発行が必要である場合は、その値が入力(読み書き)ネットワーク変数をミラーリングする出力ネットワーク変数をオブジェクトの定義に導入することができる。

7.2 センサ／アクチュエータ／コントローラオブジェクト(*) — SEMI CDM SACオブジェクトは、デバイス内のセンサオブジェクト、アクチュエータオブジェクト、コントローラオブジェクトの機能を調整する。SACオブジェクトは、LONMARKのノードオブジェクトの機能ブロックにマッピングされる。それぞれのデバイスは、1つ(かつ1つのみ)のノードオブジェクトの機能ブロックをサポートしなければならない。SAC機能ブロックおよびその共通の必須およびオプションの属性、サービス、挙動は、SEMI E54.1 CDMスタンダード仕様の文書内で記述されている。現在のSNVTリストおよびLONMARKインタオペラビリティガイドラインに対する拡張はアスタリスク(*)でマーキングされている。

7.2.1 ノードオブジェクト機能プロファイルへの拡張は、LONMARKインタオペラビリティガイドラインで定義され、LONWORKSテクノロジーに基づくSEMI SAN準拠のデバイスのLONMARK機能プロファイルの一部を形成する。表 4 に、SACオブジェクトの属性を実装するネットワーク変数をまとめて示す。表 5 に、SACオブジェクトによって実装されるサービスをまとめて示す。

表 4 Sensor/Actuator/Controller (SAC) オブジェクトのネットワーク変数

Sensor/Actuator/Controller オブジェクト(SAC) プロファイル ID = 00, インスタンス ID = 01				
シーケンス番号	名前	記憶クラス	CDM タグ	標準 NV または CP データタイプ
1	Last Calibration Date	CP	SacA1	SCPTlastCalibrationDate (*) ^{注2}
2	Next Calibration Date	CP	SacA2	SCPTnextCalibrationDate (*) ^{注2}
3	Expiration Timer	NVO	SacA3	SNVT_time_hour
4	Expiration Warning Enable	CP	SacA4	SCPTexpirationWarningEnable (*) ^{注2}
5	Run Hours	NVO	SacA5	SNVT_time_hour
128	Attribute ID ^{注1} (*) ^{注2}	NVI	SacA65	SNVT_member (*) ^{注2}

注 1 この入力を使用して Restore Default サービスおよび Publish Attribute サービスの属性を指定する。

注 2 (*) LonMark インタオペラビリティの拡張。詳細については、§ 6.8.3を参照すること。

表 5 Sensor/Actuator/Controller (SAC) オブジェクトのサービス

Sensor/Actuator/Controller オブジェクト(SAC) プロファイル ID = 00, インスタンス ID = 01				
サービス 要求コード	サービス名	CDM タグ	要求パラメータ	結果パラメータ
1	Reset	SacS1	ノードオブジェクトの RQ_RESET	
2	Abort	SacS2	ノードオブジェクトの RQ_ABORT (*) ^{注2}	
3	Recover	SacS3	ノードオブジェクトの RQ_RECOVER (*) ^{注2}	
4	GetAttribute	SacS4	NV または CP の読み出し	NV または CP の値
5	SetAttribute	SacS5	NV または CP の書き込み	
6	Operate	SacS6	ノードオブジェクトの RQ_NORMAL	
7	Restore Default ^{注1}	SacS7	ノードオブジェクトの RQ_RESTORE (*) ^{注2}	NVO の値
8	Publish Attribute ^{注1}	SacS8	ノードオブジェクトの RQ_PROPAGATE (*) ^{注2}	NVO の値

注 1 これらのサービスは、新しい Attribute ID を使用してノードオブジェクトの機能ブロックに入力できる。

注 2 (*) LonMark インタオペラビリティの拡張。詳細な定義については、§6.8.3を参照すること。

7.3 Device Managerオブジェクト(*) — Device Managerオブジェクトは、デバイスの動作を管理し、連結する責任があるデバイスのコンポーネントである。Device Managerオブジェクトは、LONMARKのノードオブジェクトの機能ブロックにマッピングされる。それぞれのデバイスは、1つ(かつ1つのみ)のノードオブジェクトの機能ブロックをサポートしなければならない。DM機能ブロックおよびその共通の必須およびオプションの属性、サービス、挙動は、

SEMI E54.1 CDMスタンダード仕様の文書内で記述されている。現在のSNVTリストおよびLONMARKインタオペラビリティガイドラインに対する拡張はアスタリスク(*)でマーキングされている。

7.3.1 LONMARKのノードオブジェクトにマッピングされるSEMI CDMのDevice Manager機能ブロックは、報告メカニズムを除き、デバイスの自己文書化の属性を組み合わせる。したがって、新しい機能プロファイルは、次の必須のコンフィギュレーションパラメータ、ネットワーク変数、挙動により定義される。この機能プロファイルは、LONWORKSに基づくSEMI SAN準拠のデバイスのLONMARK機能プロファイルの一部を形成する。表 6 に、Device Manager機能ブロックの属性を実装するネットワーク変数をまとめて示す。

表 6 Device Manager (DM)オブジェクトのネットワーク変数

Device Manager オブジェクト (DM) プロファイル ID = 00, インスタンス ID = 01				
シーケンス 番号	名前	記憶クラス	CDM タグ	標準 NV または CP データタイプ
11	Device Type	CP	DmA1	SCPTdeviceType (*) ^{注 5}
12	Standard Revision Level	CP	DmA2	SCPTdeviceRevLevel (*) ^{注 5}
13	Device Manufacture Identifier	CP	DmA3	SCPTdeviceManufacturer (*) ^{注 5}
14	Manufacturer Model Number	CP	DmA4	SCPTdeviceModelNumber (*) ^{注 5}
15	S/W or F/W Revision Level	CP	DmA5	SCPTappRevLevel (*) ^{注 5}
16	Hardware Revision Level	CP	DmA6	SCPThardwareRevLevel (*) ^{注 5}
17	Serial Number	CP	DmA7	SCPTserialNumber
18	Device Configuration	CP	DmA8	SCPTdeviceConfiguration (*) ^{注 5}
19	Device Status	NVO	DmA9	SNVT_dev_status (*) ^{注 5}
20	Reporting Mode	CP	DmA10	SCPT_rept_mode (*) ^{注 5}
21	Exception Status Report Interval	CP	DmA11	SCPT_exc_sts (*) ^{注 5}
22	Exception Status	NVO	DmA12	SNVT_alarm_2
23	Exception Detail Alarm	NVO	DmA13	SNVT_exc_detail (*) ^{注 5}
24	Exception Detail Warning	NVO	DmA14	SNVT_exc_detail (*) ^{注 5}
25	Visual Indicator	CP	DmA15	SCPTvisualIndicator (*) ^{注 5}
26	Alarm Enable	NVI	DmA16	SNVT_obj_request
27	Warning Enable	NVI	DmA17	SNVT_obj_request (*-add RQ_WARNING_NOTIFY_ENABLED および RQ_WARNING_NOTIFY_DISABLED) ^{注 5}
28	Exception Detail Type	CP	DmA18	SCPTexceptionDetailType (*) ^{注 5}
29	Exception Detail Alarm Queue	NVO	DmA19	SNVT_exc_detail (*) ^{注 5}
30	Exception Detail Warning Queue	NVO	DmA20	SNVT_exc_detail (*) ^{注 5}
31	Date and Time	NVO	DmA21	SNVT_time_p (*) ^{注 5}
32	Date and Time Type	CP	DmA22	SCPTdateTimeType (*) ^{注 5}
66	Test ID ^{注 2} (*) ^{注 5}	NVI	DmA34	SNVT_test_id (*) ^{注 5}
67	Password ^{注 3} (*) ^{注 5}	NVI	DmA35	SNVT_password (*) ^{注 5}
68	Exception Queue ^{注 4} (*) ^{注 5}	NVI	DmA36	SNVT_exc_que (*) ^{注 5}
128	Attribute ID ^{注 1} (*) ^{注 5}	NVI	DmA33	SNVT_member (*) ^{注 5}

注 1 この入力を使用して Restore Default サービスおよび Publish Attribute サービスの属性を指定する。

注 2 この入力を使用して Perform Diagnostics サービスのテスト ID を指定する。

注 3 この入力を使用して Unlock サービスのパスワードを指定する。

注 4 この入力を使用して Get and Clear Exception Queue サービスのパスワードを指定する。

注 5 (*) LonMark インタオペラビリティの拡張。詳細については、§ 6.8.3 を参照すること。

7.3.2 Device Managerの機能ブロックのサービス — 表 7 に、Device Managerの機能ブロックによって実装されるサービスをまとめて示す。

表 7 Device Manager (DM)オブジェクトのサービス

Device Manager オブジェクト (DM) プロファイル ID = 00, インスタンス ID = 01				
サービス 要求コード	サービス名	CDM タグ	要求パラメータ	結果パラメータ
1	Reset	DmS1	ノードオブジェクトの RQ_RESET	
2	Abort	DmS2	ノードオブジェクトの RQ_ABORT (*) ^{注 1}	
3	Recover	DmS3	ノードオブジェクトの RQ_RECOVER (*) ^{注 1}	
4	GetAttribute	DmS4	NV または CP の読み出し	NV または CP の値
5	SetAttribute	DmS5	NV または CP の書き込み	
13	Execute	DmS6	ノードオブジェクトの RQ_ENABLE	
14	Perform Diagnostics	DmS7	ノードオブジェクトの RQ_SELF_TEST	
8	Publish Attribute	DmS8	ノードオブジェクトの RQ_PROPAGATE (*) ^{注 1}	NVO の値
9	Lock	DmS9	ノードオブジェクトの RQ_LOCK (*) ^{注 1}	
10	Unlock	DmS10	ノードオブジェクトの RQ_UNLOCK (*) ^{注 1}	
11	Get Exception Queue	DmS11	ノードオブジェクトの RQ_GET_EXC_QUEUE (*) ^{注 1}	
12	Clear Exception Queue	DmS12	ノードオブジェクトの RQ_CLEAR_EXC_QUEUE (*) ^{注 1}	

注 1 (*)LonMark インタオペラビリティの拡張。詳細については、§ 6.8.3 を参照すること。

7.3.3 Device Managerの例外ステータスのためのPublish (DmS8) 通知サービスは、ノードオブジェクトの機能ブロックへのnviRequestの入力についてRQ_PROPAGATE要求を受信すると、実装される。特定のネットワーク変数は、SNVT_memberの入力について指定できる。これによって、このネットワーク変数の値は、バインディングできる他の入力ネットワーク変数へネットワーク経由で送信される。Device Managerの機能ブロックの実装は、機能ブロックのReporting ModeコンフィギュレーションプロパティおよびException Status Reporting Intervalコンフィギュレーションプロパティによって指定される条件に従って、指定された出力ネットワーク変数を更新する。

7.3.4 Device Managerオブジェクトのコンフィギュレーションプロパティ — DMオブジェクトには、表 8 に示すように例外報告を管理するコンフィギュレーションプロパティがある。これらのパラメータは、標準コンフィギュレーションプロパティタイプ (SCPT) である。

7.3.4.1 SCPT_rept_modeタイプ (*) には、アラームおよび警告の状態を報告する方法を指定する 2 つの 4 ビットのフィールドが格納されている。例えば、ニューロンチップ上で使用されるアプリケーションプログラミング言語、Neuron Cでは、SCPT_rept_modeの宣言は次の通りである。

```
typedef enum {
    REP_REQUEST                = 0,
    REP_REQ_LATCHED            = 1,
    REP_EVT_TRIGD_ON           = 2,
    REP_EVT_TRIGD_ONOFF        = 3,
    REP_TIME_TRIGD             = 4,
    REP_EVT_ON_TIME_TRIGD      = 5,
    REP_EVT_ONOFF_TIME_TRIGD   = 6,
} rept_mode_t;

typedef struct {
    rept_mode_t alarm_rept_mode : 4;
```

```
rept_mode_t warn_rept_mode :4;
} SCPT_rept_mode;
```

7.3.4.2 SCPT_exc_stsタイプ(*)は、分解能が 0.01 秒で 0.00～655.35 秒の時間を表す 16 ビットの値である。このパラメータはオプションである。デフォルトのレポートモードはREP_REQUESTである。

7.3.4.3 Device Managerの機能ブロックの出力ネットワーク変数 — CDMのDevice Managerの機能ブロックは、属性変数、Device Statusを定義する。データタイプ SNVT_dev_statusは、列挙である。共用体の 1 つのフィールドは列挙であり、表、CDMのデバイスステータス属性値によって定義されるデバイスステータス属性に対応する。このタイプの値は表 8 で定義される。

表 8 デバイスステータス属性変数値の一覧表

値	列挙タグ
0	DS_UNKNOWN
1	DS_INIT_SELFTEST
2	DS_IDLE
3	DS_SELFTEST_EXCPT
4	DS_EXECUTING
5	DS_ABORT_1
6	DS_ABORT_2

7.3.4.4 SNVT_dev_statusを宣言するために使用されるアプリケーションプログラミング言語は次の通りである。

```
typedef enum {
    DS_UNKNOWN          = 0,
    DS_INIT_SELFTEST    = 1,
    DS_IDLE              = 2,
    DS_SELFTEST_EXCPT   = 3,
    DS_EXECUTING         = 4,
    DS_ABORT_1           = 5,
    DS_ABORT_2           = 6,
} dev_status_t;
```

7.3.4.4.1 DS_ABORT_1 の値は、DMオブジェクトのAbort from Idle (アイドルから中止)状態またはExecuting (実行中)状態に対応し、DS_ABORT_2 の値は、DMオブジェクトのAbort from Initialized/Self Testing (初期化/自己テストから中止)状態またはSelf Test Exception (自己テスト例外)状態に対応する。

7.3.4.5 SNVT_exc_queタイプ(*)は、例外キューおよび取り出す例外要素を指定する 4 つのフィールドを格納する。例えば、ニューロンチップで使用するアプリケーションプログラミング言語であるNeuron Cでは、SNVT_exc_que_modeの宣言は次の通りである。

```
typedef enum {
    REP_ALARM          = 0,
    REP_WARNING         = 1,
} que_t;
typedef struct {
    que_t exception_type :1;
    int element_number   :1;
    SNVT_state exception_list[13];
    int exception_clear  :1;
```

```
} SNVT_exc_que;
```

7.3.4.6 SNVT_exc_detailタイプ(*)は、配列を格納している 2 つの構造体のシーケンスである。ANSI/EIA/CEA-709.1 プロトコルでは、このタイプ が 31 バイトのネットワーク変数のサイズの制限に適合するように、これらの配列のそれぞれのサイズを 9 バイトに制限している。例えば、Neuron CではSNVT_exc_detailの宣言は次の通りである。

```
typedef struct {
    u_char comn_exc_size;
    int     calibration      : 1;
    int     real_time        : 1;
    int     communic        : 1;
    int     RAM              : 1;
    int     EEPROM          : 1;
    int     EPROM           : 1;
    int     microproc       : 1;
    int     diagnostic       : 1;
    /*-----*/
    int     resvd1          : 1;
    int     reset           : 1;
    int     notify_mfr      : 1;
    int     maintenance     : 1;
    int     power_inputV    : 1;
    int     power_outptV    : 1;
    int     power_resvd     : 1;
    int     power_overC     : 1;
    u_char comn_exc_dtl[7];
    /*-----*/
    u_char dev_exc_size;
    u_char dev_exc_dtl[9];
    /*-----*/
    u_char mfr_exc_size;
    u_char mfr_exc_dtl[9];
} SNVT_exc_detail;
```

7.4 センサ, アクチュエータ, コントローラの機能ブロック — これらの機能ブロックは必然的に特定デバイスモデルに固有のものである。LONMARK アプリケーション層インタオペラビリティガイドラインは、機能プロファイルを定義するための枠組みを提供する。特定デバイスモデルは、これらの機能プロファイルを採用したり、独自の機能プロファイルおよびデバイス固有の要求条件に合わせて標準ネットワーク変数タイプ を定義したりできる。機能プロファイル定義のガイドラインに従う限り、これらのプロファイルは新しい標準プロファイルとしてLONMARK Interoperability Associationに提案できる。

7.5 Assemblyオブジェクト(Asm) – Assembly(Asm)オブジェクトのインスタンスを使用し、デバイス内の 1 つまたは複数の機能ブロックからの複数の属性を単一のデータ構造体にグループ化して、LonWorks経由で通信できる。Assemblyオブジェクトのインスタンスの属性の表現を表 9 に示す。Assemblyオブジェクトのサービスの表現を表 10 に示す。

表 9 Assembly オブジェクトのネットワーク変数

Assembly オブジェクト (Asm) プロファイル ID = 180.81, インスタンス ID = 01 ~ i				
シーケンス 番号	名前	記憶クラス	CDM タグ	標準 NV または CP データタイプ
1	Data	File	AsmA1	LONMARK ファイル転送 NV およびメッセージングインタフェース

表 10 Assembly オブジェクトのネットワークサービス

Assembly オブジェクト (Asm) プロファイル ID = 180.81, インスタンス ID = 01 ~ i				
サービス 要求コード	サービス名	CDM タグ	要求パラメータ	結果パラメータ
4	GetAttribute	AsmS4	ファイル転送読み出し	ファイル転送 – コンテキストに固有
5	SetAttribute	AsmS5	ファイル転送書き込み – コンテキストに固有	

7.6 Local Link オブジェクト (Lnk) — Local Link (Lnk) オブジェクトのインスタンスを使用して、あるオブジェクトのインスタンスを別のオブジェクトのインスタンスに「リンク」できる。このオブジェクトの詳細な説明と使用方法については、SEMI E54.1 – CDMスタンダードを参照すること。Local Link オブジェクトのインスタンスの属性の表現を表 11 に示す。Local Link オブジェクトのサービスの表現を表 12 に示す。

表 11 Local Link オブジェクトのネットワーク変数

Assembly オブジェクト (Asm) プロファイル ID = 180.81, インスタンス ID = 01 ~ i				
シーケンス 番号	名前	記憶クラス	CDM タグ	標準 NV または CP データタイプ
1	Source Object Class	NV Config Table/Address Table ^{注 1}	LnkA1	SNVTlnk_context. この SNVT の構造はコンテキスト固有のものである。
2	Source Object Instance	NV Config Table/Address Table ^{注 1}	LnkA2	SNVTlnk_context. この SNVT の構造はコンテキスト固有のものである。
3	Source Object Attribute	NV Config Table/Address Table ^{注 1}	LnkA3	SNVTlnk_context. この SNVT の構造はコンテキスト固有のものである。
4	Destination Object Class	NV Config Table/Address Table ^{注 1}	LnkA4	SNVTlnk_context. この SNVT の構造はコンテキスト固有のものである。
5	Destination Object Instance	NV Config Table/Address Table ^{注 1}	LnkA5	SNVTlnk_context. この SNVT の構造はコンテキスト固有のものである。
6	Destination Object Attribute	NV Config Table/Address Table ^{注 1}	LnkA6	SNVTlnk_context. この SNVT の構造はコンテキスト固有のものである。
7	Commit	NV Config Table/Address Table ^{注 1}	LnkA7	SNVT_switch (TRUE, FALSE).

注 1 これは、NV や CP ではない。これは、デバイス上の複数のネットワーク変数間のネットワーク変数コネクションを指定する、デバイスの NV コンフィギュレーションテーブルおよびアドレステーブル内に格納されているコンフィギュレーション情報である。同じタイプのネットワーク変数の接続だけに使用できる。リンクは、テーブルが更新されると直ちに作成されるので、Commit 操作は別のステップではない。