

COGNEX®

Checker® 4G 通信

およびプログラムガイド

9/3/2014

バージョン 4.1

この文書で説明しているソフトウェアは、ライセンスの下で提供されており、そのようなライセンスの条件に従って、さらにこのページに表示されている著作権の表示を含めている場合にだけ、使用または複製することができます。このライセンスを付与されている人以外の人には、このソフトウェアも、この文書も、あるいは一切の複製も提供したり、使用可能にすることはできません。このソフトウェアの一切の所有権は、Cognex Corporation またはそのライセンスを受けた人に留まります。Cognex Corporation は、Cognex Corporation が提供していない装置での本ソフトウェアの使用、あるいはその場合の信頼性に関して、一切の責任を認めません。Cognex Corporation は、ここで説明するソフトウェアに関して、明示的または暗示的を問わず、その商品性、非侵害、特定の目的への適合性を一切保証いたしません。

本文書に記載されている情報は、通知なしで変更されることがあり、Cognex Corporation の確約ともみなされません。Cognex Corporation は、本文書、または関連するソフトウェアに存在する可能性がある誤りについて、一切の責任を負いません。

本文書のいかなる部分であれ、Cognex Corporation の書面による許可がない場合、形式や手段を問わず、電子的であれ機械的であれ、目的によらず、他の媒体や言語へ複製または転送することはできません。

Copyright © 2014 Cognex Corporation. All Rights Reserved.

Cognex が提供するハードウェアおよびソフトウェアの一部は、以下に一覧する一件または複数の米国および外国の特許、ならびに米国および外国で申請中の特許によって保護されていることがあります。本文書の発行の日以降に申請された、そのような米国および外国で申請中の特許は、Cognex の Web サイト (<http://www.cognex.com/patents>) に一覧されています。

VisionPro

5481712, 5495537, 5548326, 5583954, 5602937, 5640200, 5751853, 5768443, 5825913, 5850466, 5872870, 5901241, 5943441, 5978080, 5978521, 5987172, 6005978, 6039254, 6064388, 6075881, 6137893, 6141033, 6167150, 6215915, 6240208, 6324299, 6381366, 6381375, 6411734, 6421458, 6459820, 6490375, 6516092, 6563324, 6658145, 6687402, 6690842, 6697535, 6718074, 6748110, 6771808, 6804416, 6836567, 6850646, 6856698, 6920241, 6959112, 6963338, 6973207, 6975764, 6985625, 6993177, 6993192, 7006712, 7016539, 7043081, 7058225, 7065262, 7088862, 7164796, 7190834, 7242801, 7251366, 7313761, EP0713593, JP3522280, JP3927239

Checker® 4G 通信およびプログラムガイド

9/3/2014 | バージョン 4.1

DataMan

5742037, 5943441, 6215915, 6236769, 6282328, 6381375, 6408109, 6457032, 6690842, 6941026, 7175090, 7181066, 7412106, 7427028, 7549582, 7604174, 7614563, 7617984, US-2005-0087601-A1, US-2006-0131418-A1, US-2006-0131419-A1, US-2006-0133757-A1, US-2007-0090193-A1, US-2007-0091332-A1, US-2007-0152064-A1, US-2007-0170259-A1, US-2008-0004822-A1, US-2008-0011855-A1, US-2008-0142604-A1, US-2008-0143838-A1, US-2008-0158365-A1, US-2009-0090781-A1, US-2009-0108073, US-2009-0121027-A1, US-2009-0166424-A1, US-2009-0294541-A1, WO06065619A1, EP1687752

CVL

5495537, 5548326, 5583954, 5602937, 5640200, 5717785, 5751853, 5768443, 5825483, 5825913, 5850466, 5859923, 5872870, 5901241, 5943441, 5949905, 5978080, 5987172, 5995648, 6002793, 6005978, 6064388, 6067379, 6075881, 6137893, 6141033, 6157732, 6167150, 6215915, 6240208, 6240218, 6324299, 6381366, 6381375, 6408109, 6411734, 6421458, 6457032, 6459820, 6490375, 6516092, 6563324, 6658145, 6687402, 6690842, 6718074, 6748110, 6751361, 6771808, 6798925, 6804416, 6836567, 6850646, 6856698, 6920241, 6959112, 6975764, 6985625, 6993177, 6993192, 7006712, 7016539, 7043081, 7058225, 7065262, 7088862, 7164796, 7190834, 7242801, 7251366, EP0713593, JP3522280, JP3927239

VGR

5495537, 5602937, 5640200, 5768443, 5825483, 5850466, 5859923, 5949905, 5978080, 5995648, 6002793, 6005978, 6075881, 6137893, 6141033, 6157732, 6167150, 6215915, 6324299, 6381375, 6408109, 6411734, 6421458, 6457032, 6459820, 6490375, 6516092, 6563324, 6658145, 6690842, 6748110, 6751361, 6771808, 6804416, 6836567, 6850646, 6856698, 6959112, 6975764, 6985625, 6993192, 7006712, 7016539, 7043081, 7058225, 7065262, 7088862, 7164796, 7190834, 7242801, 7251366

OMNIVIEW

6215915, 6381375, 6408109, 6421458, 6457032, 6459820, 6594623, 6804416, 6959112, 7383536

CVL Vision Library

5495537, 5548326, 5583954, 5602937, 5640200, 5717785, 5751853, 5768443, 5825483, 5825913, 5850466, 5859923, 5872870, 5901241, 5943441, 5949905, 5978080, 5987172, 5995648, 6002793, 6005978, 6064388, 6067379, 6075881, 6137893, 6141033, 6157732, 6167150, 6215915, 6240208, 6240218, 6324299, 6381366, 6381375, 6408109, 6411734, 6421458, 6457032, 6459820, 6490375, 6516092, 6563324, 6658145, 6687402, 6690842, 6718074, 6748110, 6751361, 6771808, 6798925, 6804416, 6836567, 6850646, 6856698, 6920241, 6959112, 6975764, 6985625, 6993177, 6993192, 7006712, 7016539, 7043081, 7058225, 7065262, 7088862, 7164796, 7190834, 7242801, 7251366, EP0713593, JP3522280, JP3927239

SMD 4

5995648, 5850466, 6751361, 6690842, 6563324, 6490375, 5949905, 5978080, 6137893, 6167150, 6075881, 6748110, 5859923, 6411734, 6324299, 6516092, 7190834, 6658145, 6836567, 6850646, 6975764, 6985625, 6993192, 7006712, 7043081, 7058225, 7065262, 7088862, 7164796, 7251366, 6856698, 6002793, 6005978, 6771808, 6804416, 7016539, 6959112, 5602937, 7242801, 5640200, 5495537, 5768443, 5825483, 6421458, 6459820, 6215915, 6381375, 6457032, 6157732, 6408109, 6141033, 6026176, 6442291, 6151406, 6396942, 6614926, 5371690, 5845007, 5943441, 6963338, 5805722, 5909504, 5933523, 5964844, 5974169, 5987172, 6078700, 6252986, 6278796, 6307210, 6408429, 6424734, 6526165, 6571006, 6639624, 6681039, 6748104, 6813377, 6853751, 6898333, 6950548, 6993177, 7139421, 5757956

BGA II および BGA III

5495537, 5602937, 5640200, 5768443, 5801966, 5825483, 5850466, 5859923, 5949905, 5978080, 5995648, 6002793, 6005978, 6026176, 6055328, 6075881, 6115042, 6118893, 6130959, 6137893, 6141009, 6141033, 6151406, 6157732, 6167150, 6215915, 6289117, 6324299, 6353676, 6381375, 6396942, 6408109, 6411734, 6421458, 6442291, 6457032, 6459820, 6490375, 6516092, 6563324, 6577775, 6614926, 6658145, 6690842, 6748110, 6751361, 6771808, 6804416, 6836567, 6850646, 6856698, 6959112, 6975764, 6985625, 6993192, 7006712, 7016539, 7043081, 7058225, 7065262, 7088862, 7164796, 7190834, 7242801, 7251366

Wire Bonder

5495537, 5532739, 5581632, 5602937, 5640199, 5640200, 5642158, 5676302, 5754679, 5757956, 5768443, 5825483, 5835622, 5850466, 5859923, 5861909, 5949905, 5978080, 5991436, 5995648, 6002793, 6005978, 6035066, 6061467, 6075881, 6137893, 6141033, 6157732, 6167150, 6215915, 6289492, 6324299, 6381375, 6408109, 6411734, 6421458, 6457032, 6459820, 6490375, 6516092, 6563324, 6658145, 6690842, 6748110, 6751361, 6771808, 6804416, 6836567, 6850646, 6856698, 6959112, 6975764, 6985625, 6993192, 7006712, 7016539, 7043081, 7058225, 7065262, 7088862, 7164796, 7171036, 7190834, 7242801, 7251366

以下は、Cognex Corporation の登録商標です：

acuReader® BGAI® Check it with Checker® Checker® Cognex Vision for Industry CVC-1000® CVL® DataMan® DisplayInspect® DVT® EasyBuilder® IDMax® In-Sight® In-Sight 2000® In-Sight® (insignia with cross-hairs) MVS-8000® OmniView® PatFind® PatFlex® PatInspect® PatMax® PatQuick® SensorView® SmartLearn® SmartView® SMD4® UltraLight® Vision Solutions® VisionPro® VisionView®

Checker® 4G 通信およびプログラムガイド

9/3/2014|バージョン 4.1

ページ | 3

以下は、Cognex Corporation の商標です：

3D-Locate™ 3DMax™ CheckPoint™ Cognex VSoC™ FFD™ iLearn™ InspectEdge™ Legend™ LineMax™ NotchMax™ ProofRead™ SmartAdvisor™ SmartSync™
SmartSystem™

ここに記載されている、その他の製品名や会社名は、それぞれの所有者の商標、登録商標です。

本マニュアルについて	10
ネットワーキング	11
IP アドレスの自動割り当て	11
スタティック IP アドレスの割り当て	12
サブネットをまたぐ Checker 4G の接続	13
産業用ネットワーク・プロトコル	14
Ethernet/IP	14
Common Industrial Protocol (CIP).....	15
CIP での EtherNet/IP の実装.....	17
CIP メッセージングのタイプ	18
暗示的メッセージング	18
デバイスの分類.....	19
明示的メッセージ・サーバー	20
明示的メッセージ・クライアント	20
I/O アダプター	21
I/O スキャナー	21
Checker と EtherNet/IP	21
入力.....	25
サービス.....	25
取得サービス.....	26

Checker® 4G 通信およびプログラムガイド

9/3/2014|バージョン 4.1

取得シーケンス.....	26
チェック/結果シーケンス	30
Inspection ビットの挙動.....	31
結果のバッファリング.....	33
アセンブリ・オブジェクト.....	34
入力アセンブリ.....	35
Rockwell AOP を使用してセンサーの結果データを有効にする.....	37
出力アセンブリ.....	38
スタティック・データおよび変動データの概要.....	39
Checker の実装.....	43
Ethernet 現場プロトコル・サブシステム.....	43
汎用的な暗示的メッセージング接続の確立.....	44
汎用暗示的メッセージング接続データへのアクセス.....	50
例.....	50
デバイス・アクセス・コード.....	51
PROFINET.....	52
Checker GUI で PROFINET を有効にする.....	52
PROFINET の設定.....	54
モジュール.....	60

デバイス制御モジュール	62
デバイス・ステータス・モジュール	63
取得制御モジュール	65
取得ステータス・モジュール	66
結果制御モジュール	67
結果ステータス・モジュール	68
入力モジュール	69
出力モジュール	70
センサー・メーター・モジュール	71
操作	71
取得シーケンス	71
チェック/結果シーケンス	75
チェックステータスレジスターの挙動	76
結果のバッファリング	77
Siemens の例 (S7-300)	78
シンボル・テーブル	78
変数テーブル	80
バッファリングなしのトリガ	82
バッファリングありのトリガ	85

ジョブの変更の例.....	87
UDT の使用	90
トリガとジョブの変更を交互に連続させる	91
例に関する注.....	94
Checker と Siemens S7-1200 PLC 間での PROFINET 通信の作成.....	95
Generic FFP	97
Checker GUI で Generic FFP を有効にして、設定する.....	97
24 仮想入出力.....	99
レポート・メーター値およびスレッシュホールド設定ポイント	99
パターンの再登録.....	100
内部照明の制御.....	101
チェックからの FTP 結果データ	101
ストロボの制御.....	102
操作.....	103
入力アセンブリ	104
出力アセンブリ	106
Checker の一般的な入力コマンドの処理.....	107
定義.....	107
入力コマンド.....	107

入力コマンドのソース.....	108
同時複数のコマンド.....	108
コマンドの重複.....	108
ポートの利用.....	109
付録 A – プロトコル設定、オブザーバ・ビットの変更	110

本マニュアルについて

Checker 4G 通信およびプログラムガイドは、以下を含めて、特定の環境に Checker 4G センサーを統合する方法についての情報を説明しています：

- ネットワーク構成
- 産業用ネットワーク・プロトコル
- PLC との統合
- ポートの利用

ネットワーキング

Checker 4G センサーは、シンプルな Ethernet 接続で接続することができます。Checker 4G センサーは、PC と同じサブネットに配置することもできますし、別のサブネットに配置することもできます。

Checker 4G センサーをマシンと同じサブネットに配置した場合は、以下の Ethernet ネットワーク割り当てをサポートできます：

- リンクローカルのアドレス指定
- DHCP クライアント
- スタティック IP
- IP アドレスなし

Checker 4G センサーは、デフォルトでは DHCP を使用するように設定されます。ただし、スタティック IP アドレスを割り当てて、使用することも可能です。

IP アドレスの自動割り当て

Checker 4G センサーを DHCP サーバーをサポートするように設定した場合 (デフォルト)、センサーはまず直接 PC に接続されているかチェックします。直接、接続されているなら、リンクローカルのアドレス指定によって接続が確立されます。こうすることで、DHCP サーバーが見つからなかった場合の接続時間を短縮できます。

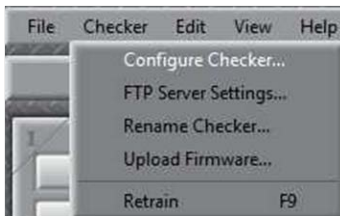
リンクローカルのアドレス指定が行われなかった場合、Checker 4G センサーは DHCP サーバーがアドレスを割り当ててのを待ちます。

DHCP サーバーが有効な IP アドレスを見つけることができなければ、Checker 4G センサーは通常、実行モードで機能します。センサーは、アドレスの割り当てをリスンし続けます。ただし、この間は、設定はできません。

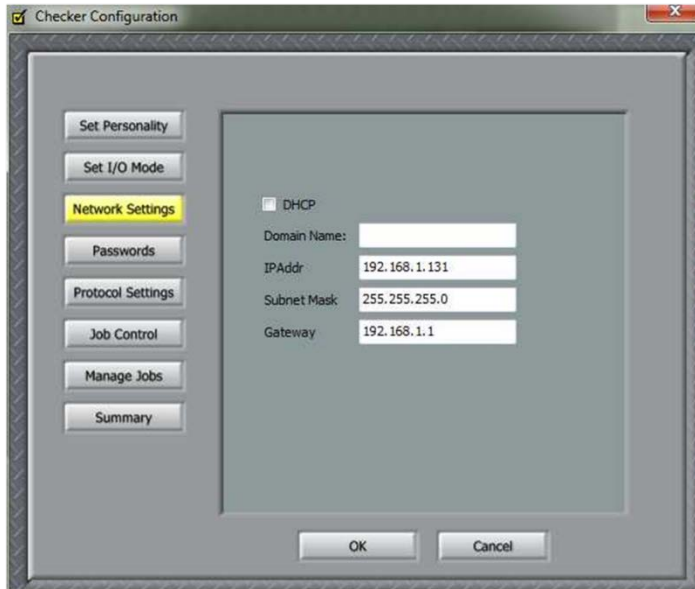
スタティック IP アドレスの割り当て

Checker 4G センサーが、スタティック IP アドレスを使用するように設定することができます。この場合、センサーは割り当てられたアドレスにだけ応答し、リンクローカルのアドレス指定や DHCP サーバーによるアドレス割り当ては使用しません。この場合、接続時間は短縮されます。

1. [メイン] ツールバーで、[Checker の設定] を選択します。



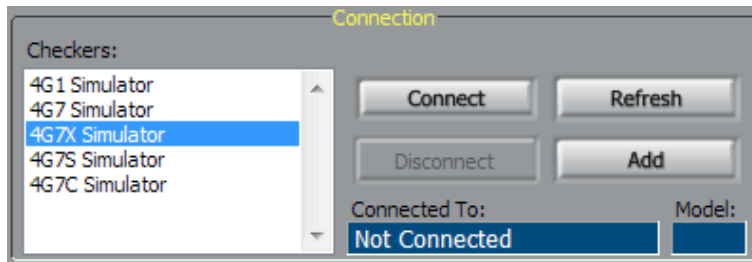
2. ポップアップ表示されるウィンドウで、必要に応じて IP 設定を行うことができます。



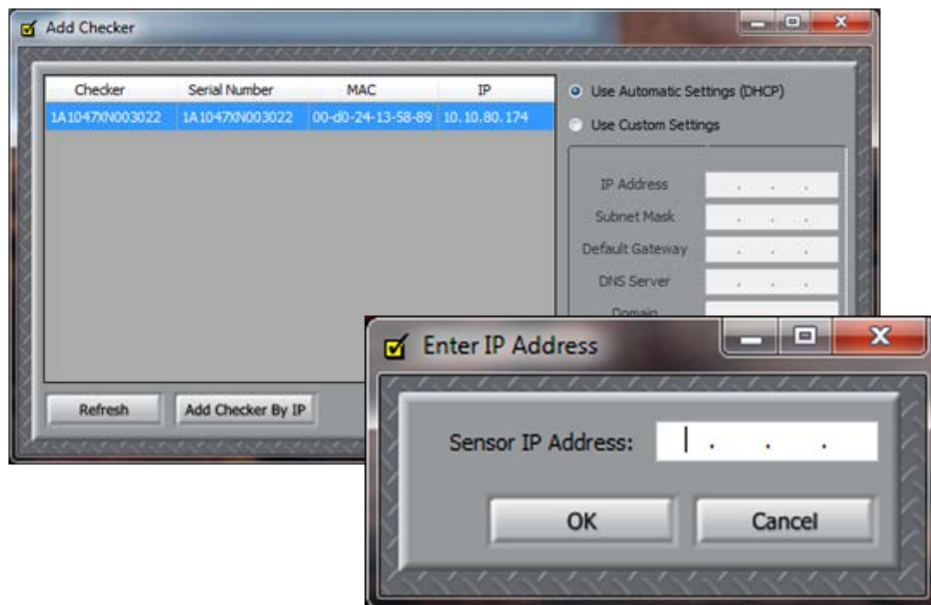
サブネットをまたぐ Checker 4G の接続

以下のオプションを使用すると、Checker 4G センサーの IP アドレスが分かっている場合に、サブネットにまたがって Checker 4G と PC を接続することができます。

1. [接続]のステップで、[接続] フレームの [追加] をクリックします。



2. [Checker を追加] ダイアログボックスで、[IP で Checker を追加] をクリックしてから、Checker 4G センサーの IP アドレスを入力します。



3. [OK] をクリックしてから、[Checker を追加] ダイアログボックスを閉じます。
これで、Checker はもう [Checkers] リストには表示されません。

IP アドレスが DHCP サーバーによって割り当てられるように Checker を設定しているのに、DHCP サーバーが存在しないネットワークに接続される場合、Checker は **[接続]** リストには表示されません。**[追加]** をクリックします。表示されるウィンドウに、Checker センサーが表われます。必要であれば、ここでさらにデバイスを設定することができます。

IP アドレスがスタティックに設定されるように Checker を設定しているのに、DHCP サーバーが存在するネットワークに接続されてしまい、IP の範囲が異なる場合、Checker は **[接続]** リストには表示されません。**[追加]** をクリックします。表示されるウィンドウに、Checker センサーが表われます。必要であれば、ここでさらにデバイスを設定することができます。

産業用ネットワーク・プロトコル

Checker 4G は、標準 Ethernet プロトコルに基づく産業ネットワーク・プロトコルを使用します。たとえば、EtherNet/IP™ や PROFINET などの、こうしたプロトコルは標準 Ethernet より高い信頼性を提供できるように拡張されています。また EtherNet/IP や PROFINET がサポートされていない場合の通信では、Generic 現場プロトコル (Generic FFP) が利用可能です。

デフォルトでは、Checker では産業用ネットワーク・プロトコルは無効になっています。プロトコルを有効にするには、Checker へ接続し、**[Checker]** -> **[Checker の設定]** と選択してから、**[プロトコル設定]** をクリックして、必要なプロトコルを有効にします。

Ethernet/IP

EtherNet/IP は、産業用の環境で使用するのに適している通信システムです。EtherNet/IP を使うと産業用デバイスは、速度が重視されるアプリケーション情報を交換することができます。こうしたデバイスとしては、センサー/アクチュエータなどの簡単な I/O デバイス、ならびにロボット、プログラマブル・ロジック・コントローラ、溶接機、プロセスコントローラなどの複雑な制御デバイスがあります。

Checker® 4G 通信およびプログラムガイド

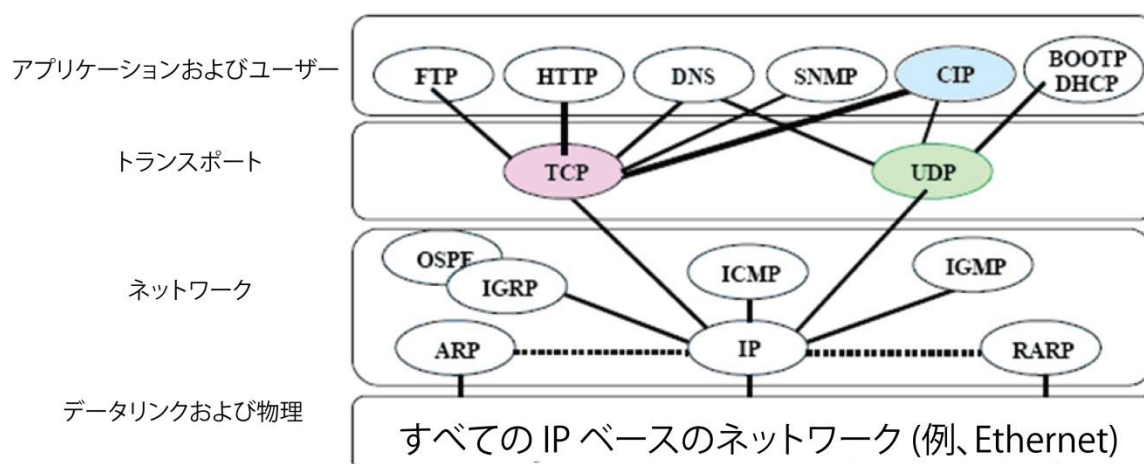
9/3/2014|バージョン 4.1

Ethernet/IP を理解するには、Common Industrial Protocol (CIP)を理解することが必要です。Ethernet/IP は、Common Industrial Protocol (CIP) の拡張であり、ラッパーであるというのが最も適切な説明です。TCP/IP プロトコル・スイートを使用して、標準 IEEE 802.3 Ethernet で動作します。Checker が使用している Ethernet/IP の実装は、Ethernet スタックのすぐ上の層であり、通常のソケット機能を使用しています。決定論を改善しようとする、非標準の追加であるといえます。

Common Industrial Protocol (CIP)

Common Industrial Protocol (CIP) は、オートメーション・アプリケーション用に設計された、メディア独立、接続ベース、オブジェクト指向のプロトコルです。オートメーション・アプリケーション向けの通信サービスの広範なセットを含んでいます。たとえば、制御、安全、同期、モーション、構成、情報などです。現在、CIP は of EtherNet/IP、DeviceNet、CompoNet、ControlNet のコア的な存在です。

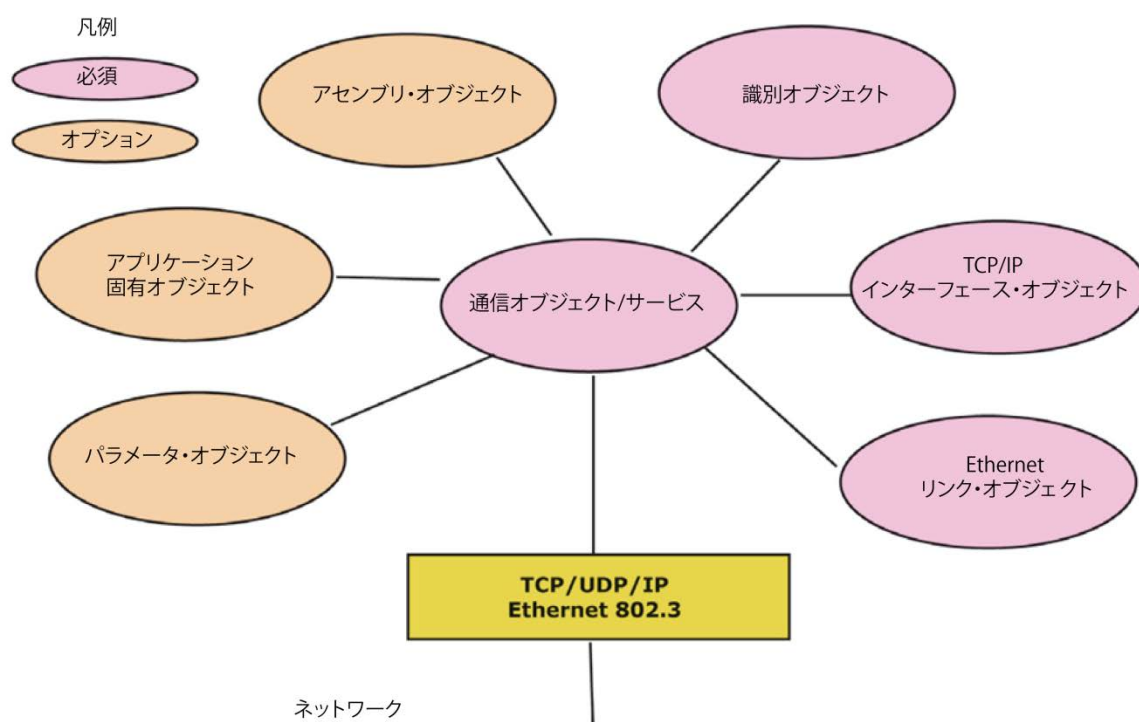
Ethernet 上で、CIP は UDP と TCP の両方にまたがります。CIP の「明示的」なメッセージングは、TCP を使って、速度が重視されないポイント・ツー・ポイントのメッセージやコマンドをデバイス間で送信します。CIP の「暗示的」なメッセージングは、速度が重視される制御や I/O のために UDP を使用します。



CIP は、オブジェクト・モデルに基づいています。このオブジェクトによって、デバイスのデータや機能（少なくとも、外部に公開したくない部分）をカプセル化します。モデルは、このデータや機能にアクセスする共通の方法となります。

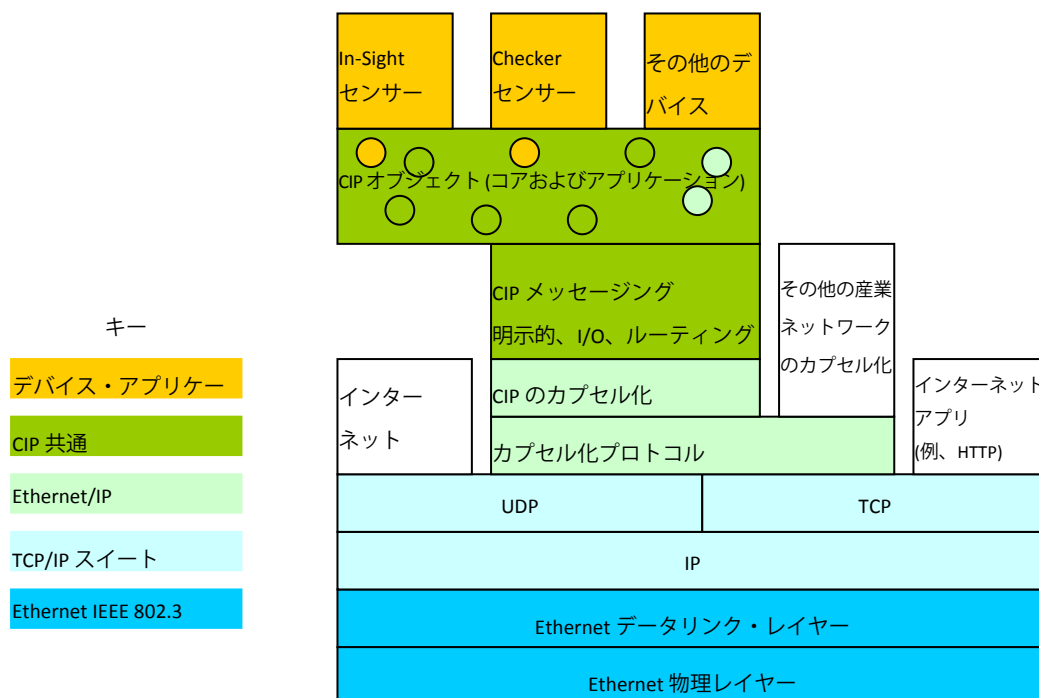
EtherNet/IP を完全に実装するには、多数の事前定義のオブジェクトが必要です。これらのオブジェクトが主に扱うのは、デバイスの識別と通信です。他にも、オプションとして事前定義のオブジェクトが含まれます。CIP の仕様には、事前定義のオブジェクトの完全なライブラリ（アセンブリ、離散的出力、アナログ入力、AC/DC ドライブなど）が含まれます。また、ベンダーもそれぞれのアプリケーションに固有のオブジェクトを作成することができます。

簡素化した EtherNet/IP のオブジェクト・モデルの概要は、以下の図を参照してください。



CIP での EtherNet/IP の実装

CIP、EtherNet/IP、TCP/IP の間の関係については、以下の図を参照してください。
EtherNet/IP の主要なコンポーネントは、2 つの新しい CIP オブジェクトとカプセル化のレイヤーです。



カプセル化レイヤーは、ネイティブな CIP が TCP/IP プロトコル・スイートを使用して、標準 IEEE 802.3 Ethernet で通信ができるようにするラッパーです。また、CIP のルーティング機能で Ethernet 独自の側面を取り扱えるようにする拡張も提供します。

EtherNet/IPをサポートするために、TCP/IP インターフェース・オブジェクトと Ethernet リンク・オブジェクトの 2 つの新しい CIP オブジェクトが追加されています。インターフェース・オブジェクトは、デバイスの TCP/IP ネットワーク・インターフェース (IP アドレス、ネットワーク・マスクなど) を設定するメカニズムを提供します。リンク・オブジェクトは、リンクに固有の構成、Ethernet 802.3 通信インターフェースのステータスや診断情報を保持します。

CIP メッセージングのタイプ

CIP は、リアルタイムの I/O または制御のために、“明示的な”メッセージングを提供します。また、CIP は、速度重視ではないデータ転送や情報メッセージのために“暗示的な”メッセージングも提供します。現在、Checker は明示的なメッセージだけをサポートしています。

伝送のタイプ	メッセージのタイプ	説明	例
情報	明示的	速度重視ではない情報	要求/応答コマンド (構成パラメータ値の設定など)
I/O データ	暗示的	リアルタイム I/O データ	リモート I/O デバイスからのリアルタイム・データ (センサー、スイッチ、モーター制御など)

暗示的メッセージング

多くの場合「I/O」と呼ばれる、このタイプの通信は、通常は、速度や低ディレイが重要であるリアルタイムでのデータ交換で使用されます。暗示的メッセージには、それらの意味についてあまり情報が含まれていないので、伝送はより効率的になりますが、柔軟性で明示的メッセージより劣ります。伝送されるデータの解釈は高速です。

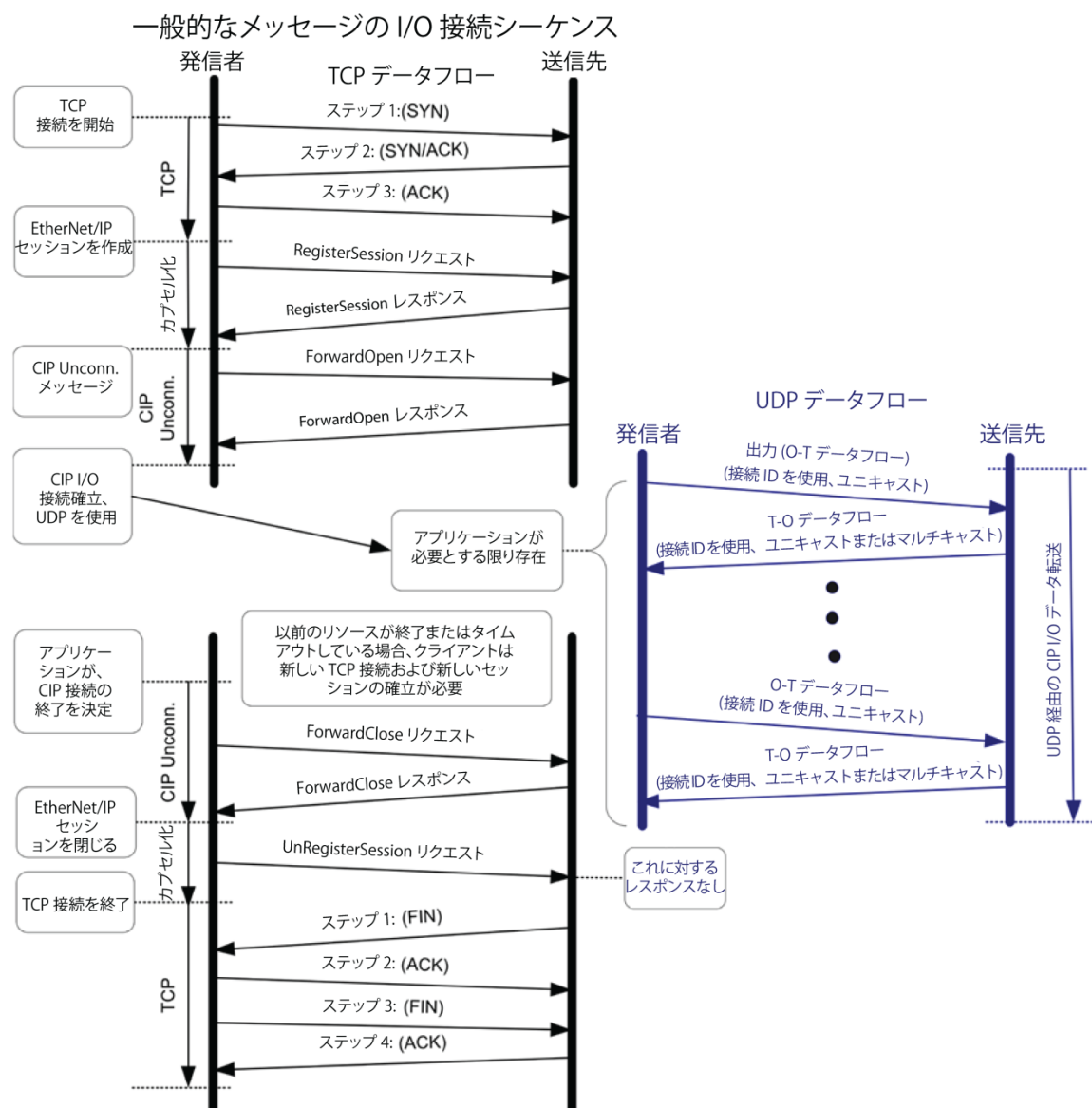
暗示的メッセージングで、2 つのデバイス間にアソシエーション (“CIP 接続”) を確立できます。接続を確立した後、事前に定義されているトリガのメカニズムに従って、デバイスは暗示的メッセージを作成します。トリガは、「周期的」(最も一般的)、「状態の変化」(CoS)、「アプリケーション固有」のいずれかです。両方のデバイスが、互いが使用するデータのフォーマットを知っていて、合意していることです(つまり、フォーマットは「暗示的」)。EtherNet/IP の場合、暗示的メッセージングでは UDP を使用し、マルチキャストまたはユニキャストが可能です。

Checker® 4G 通信およびプログラムガイド

9/3/2014 | バージョン 4.1

暗示的メッセージングは、アセンブリに基づいています。アセンブリ・オブジェクトの詳細については、「[アセンブリ・オブジェクト](#)」のセクションを参照してください。

一般的な I/O 接続のメッセージのシーケンスの概要は、以下の図を参照してください。



デバイスの分類

一般的な挙動や、サポートしている EtherNet/IP 通信のタイプによって、複数のデバイスの分類があります。大半のデバイスは、以下の分類のいずれかになります。ただし

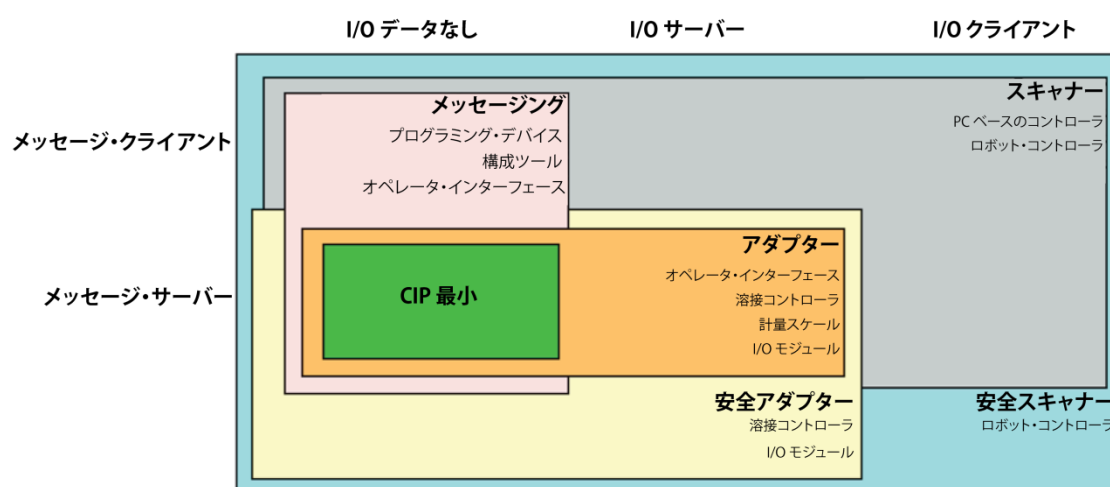
Checker® 4G 通信およびプログラムガイド

9/3/2014 | バージョン 4.1

、特定の場、デバイスが複数の分類に属することもあります。たとえば、Checker は主に I/O アダプターに分類されます。また、I/O モジュールへの接続で使用される、I/O スキャナーの機能もあります。

すべての EtherNet/IP デバイスは、最低でも明示的サーバー機能を有し、デバイスの識別や構成の要求に応答できる必要があります。

通信のタイプとデバイスの分類に関するマトリックスの概要は、以下の図を参照してください。



明示的メッセージ・サーバー

明示的メッセージ・サーバーは、明示的メッセージ・クライアントによって始動される通信を指向する要求/応答に対応します。明示的メッセージ・サーバーの例は、キオスクでのテキストのスクロール表示です。

明示的メッセージ・クライアント

明示的メッセージ・クライアントは、他のデバイスとの間での要求/応答指向の通信を始動します。メッセージのレートやレイテンシーの要件は、一般的にそれほど厳しいものではありません。明示的メッセージ・クライアントの例は、HMI デバイス、

プログラミング・ツール、あるいは制御デバイスからデータを収集する PC ベースのアプリケーションなどがあります。

I/O アダプター

I/O アダプターは、I/O スキャナーから暗示的通信接続要求を受信し、要求されたレートで I/O データを生成します。また、I/O アダプターは明示的メッセージ・サーバーでもあります。I/O アダプターとしては、簡単なデジタル入力デバイス、あるいはモジュール式の空気バルブシステムなどのより複雑なものなどがあります。

I/O スキャナー

I/O スキャナーは、I/O アダプターとの暗示的通信を始動します。どの接続を行うか、アダプター・デバイスをどのように設定するかなどの問題を取り扱うので、一般にスキャナーは最も複雑なタイプの EtherNet/IP デバイスです。また通常、スキャナーは明示的メッセージの始動をサポートします。プログラマブル・コントローラ (PLC) は I/O スキャナーの一例です。

Checker と EtherNet/IP

Checker 4G は、Common Industrial Protocol (CIP) に基づくアプリケーション・レベルのプロトコルである EtherNet/IP をサポートしています。EtherNet/IP は、Ethernet によるデータの転送や I/O のために幅広い種類のメッセージングのオプションやサービスを提供します。EtherNet/IP ネットワークにあるすべてのデバイスは、属性と呼ばれる一連のデータ値として、データをネットワークに送り出します。属性は、他の関連データと共に、アセンブリと呼ばれるセットにグループ化されます。

EtherNet/IP を有効にするには、以下のステップを実行します：

- マシンに Rockwell Software ツールがインストールされていることを確認します。
- Checker センサーが Rockwell Software ツールによって認識されるように設定します。

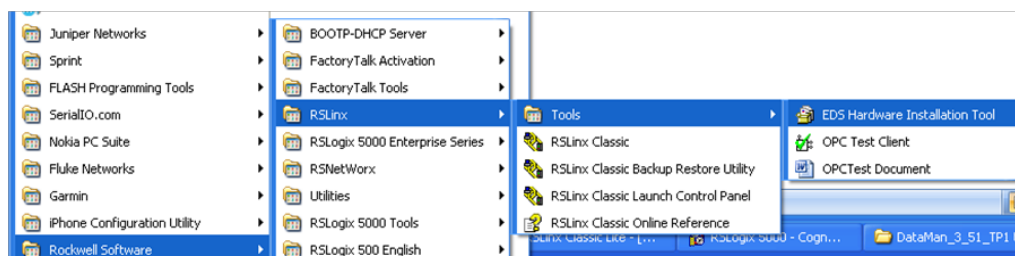
Checker® 4G 通信およびプログラムガイド

9/3/2014 | バージョン 4.1

- Checker センサー用に Checker Electronic Data Sheet (EDS) をインストールします。

以下のステップを実行して、EtherNet/IP をセットアップします：

1. Rockwell Software が PC にインストールされていることを確認します。
2. 次のディレクトリへ移動して、**Rockwell Add on Profiles** をインストールします：
C:\Program Files\Cognex\Checker vision Sensors x.x\EthernetIP\Rockwell AOP
3. 今までに Rockwell AOP をインストールしていない場合は、ここでこのディレクトリから MPSetup.exe を実行します。
4. [スタート] メニューで、[プログラム] → [Rockwell Software] → [RSLink] → [Tools] → [EDS ハードウェア・インストール・ツール] と移動します。



5. EDS インストール・ツールを実行します。

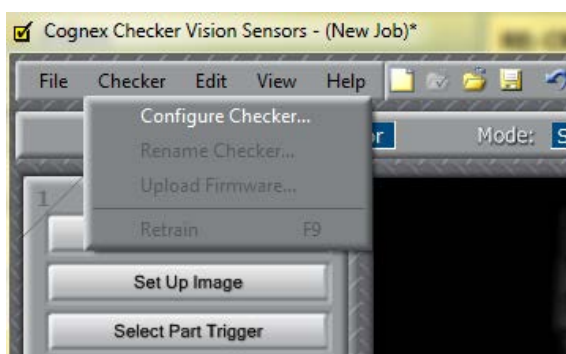
注

既存の EDS ファイルが存在する場合は、まずそれをアンインストールしてから、最新バージョンの EDS をインストールしてください。

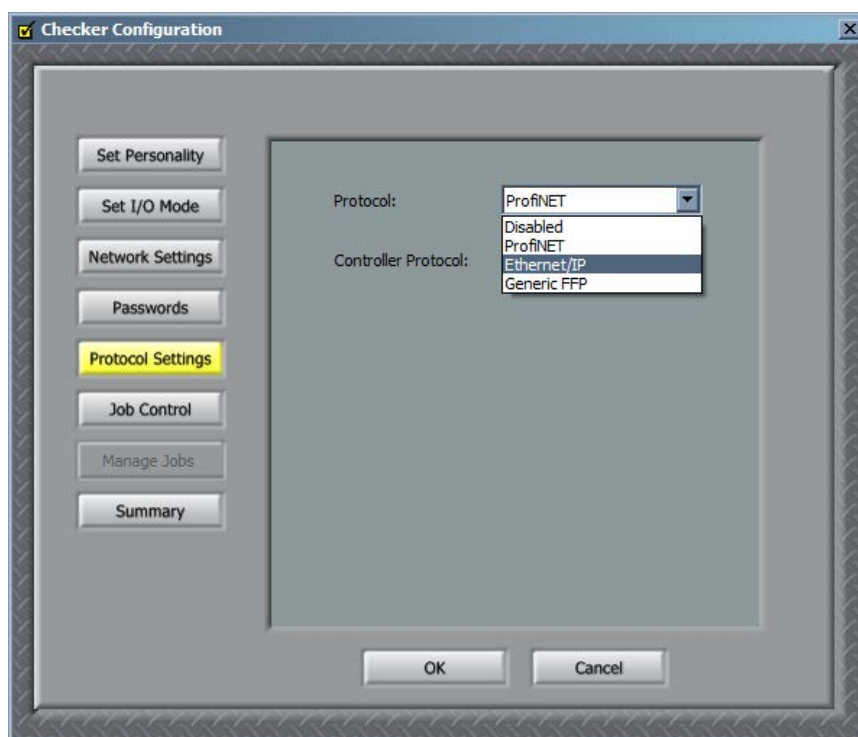
6. Checker PC アプリケーションを使って、Checker へ接続します。ファームウェアのアップグレードまたはダウングレードが必要であることを通知するポップアップ・ウィンドウが表示されたら、その指示に従います。それ以外の場合、次のステップへ進んでください。

7. デフォルトでは、Checker 4G では EtherNet/IP プロトコルは無効になっています。プロトコルを有効にするには、以下のステップを実行してください：

- a. 上のメニュー・バーで、[Checker] をクリックします。
- b. [Checker の設定] を選択します。



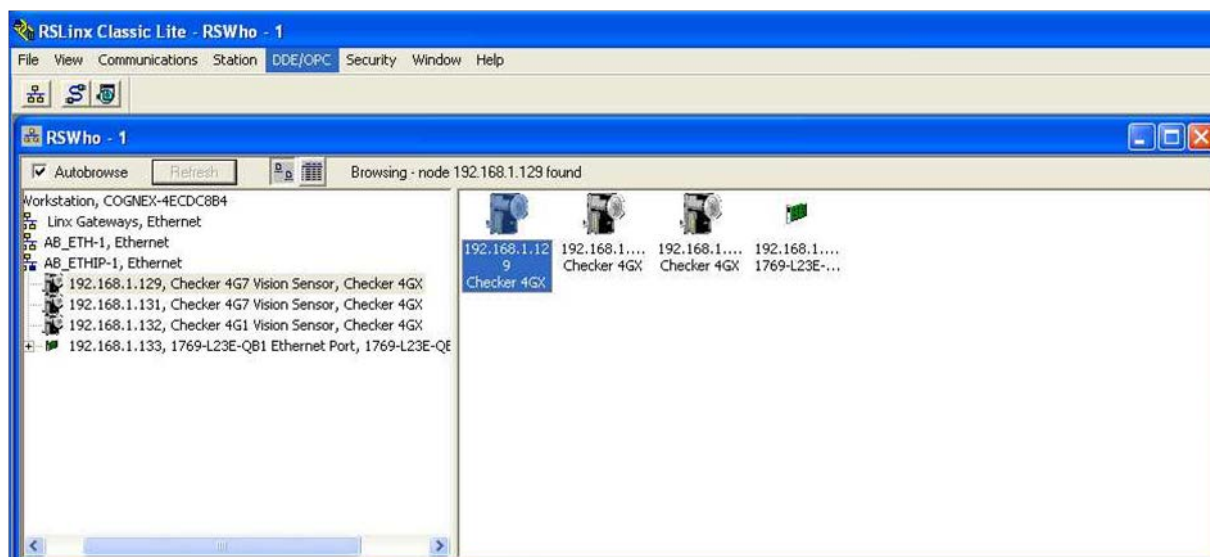
- c. ポップアップ表示される Checker 設定ウィンドウで、[プロトコル設定] をクリックし、[EtherNet/IP] を選択します。



- d. [OK] をクリックします。

これらの設定を有効にするために、センサーは自動的に再起動します。

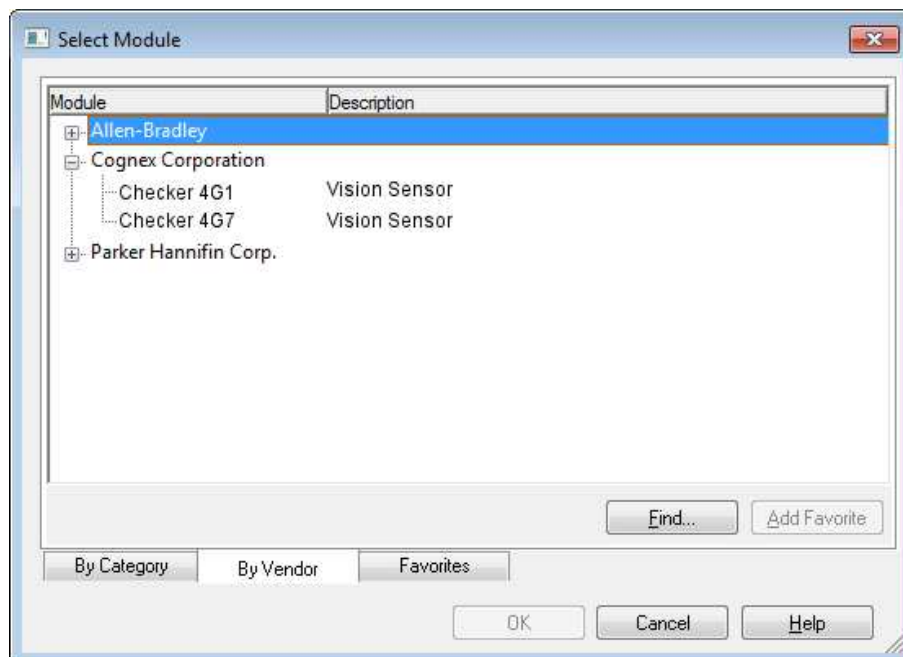
8. これで、Checker センサーが RSWHO に表示されます。



Checker 4G センサーは表示されるものの、アイコンが疑問符である場合は、EDS のインストールを繰り返してください。

9. ジョブを開き、プロファイルのアドオンを使って、Checker 4G センサーをプログラムに組み込みます。

あるいは、Checker 4G センサーをネットワーク上のモジュールとして追加することもできます。



入力

入力は、「入力」の入力として機能します。入力の値が 0 → 1 と変化すると、このイベントに関連付けられているアクションが実行されます。アクションが完了すると、対応する InputAck ビットが 1→0 と変化して、完了を通知します。対応する入力ビットを 0 に戻すと、確認ビットが 0 に戻ります。

サービス

Checker のオブジェクトは、以下の共通 CIP サービスをサポートしています：

サービスコード	サービス名	説明
0x05	リセット	Checker オブジェクトをリセットします。
0x0E	Get_Attribute_Single	指定された属性のコンテンツを返します。
0x10	Set_Attribute_Single	指定された属性を変更します。

取得サービス

取得サービスは、取得をトリガします (取得システムが画像取得の準備ができている場合)。取得をトリガできなかった場合、次に取得が成功するまで、Missed Acquisition ビットがセットされます。

取得シーケンス

Checker は、アセンブリ・オブジェクトを通じて、画像取得を明示的にトリガすることができます。

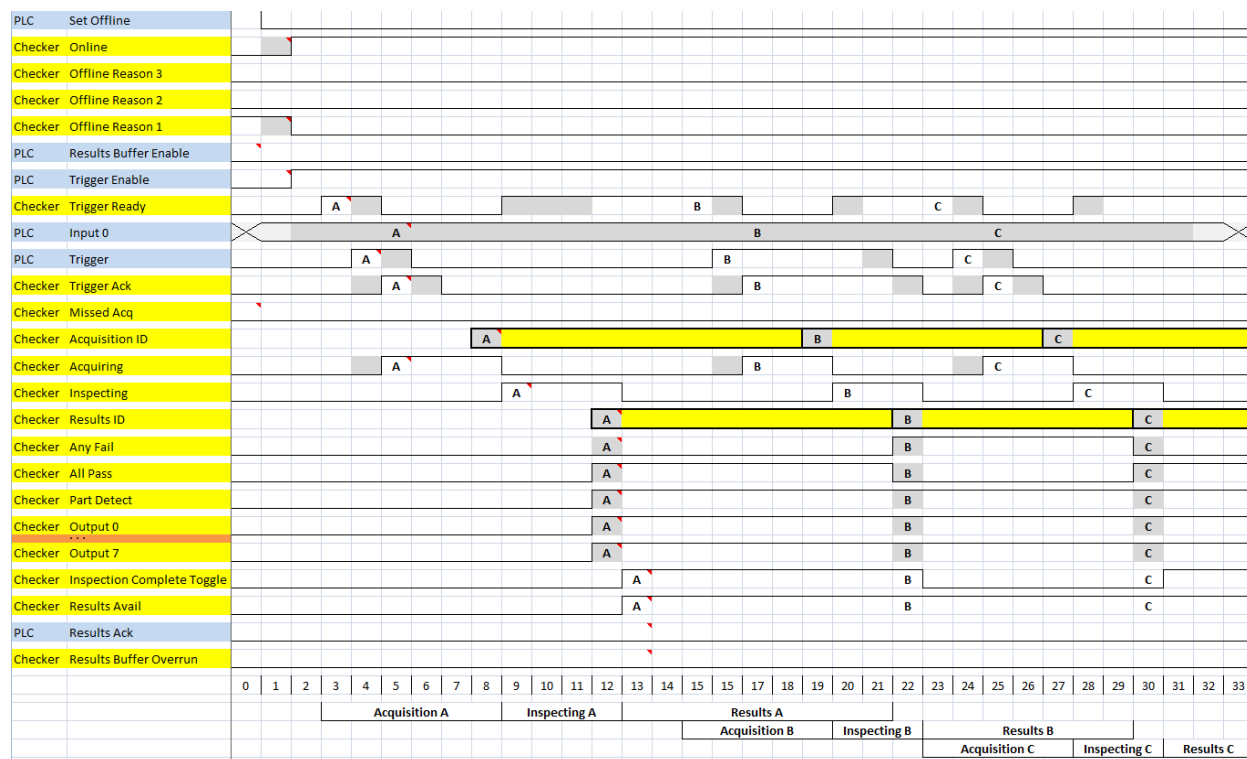
起動時には、TriggerEnable 属性は false です。トリガを有効にするには、true に設定する必要があります。デバイスがトリガを受信できるようになると、トリガ Ready ビットが true に設定されます。

TriggerEnable 属性が true で、トリガ Ready ビットが true であると、トリガ属性が 0 から 1 に変わったことを Checker オブジェクトが確認するたびに、画像取得が始動されます。アセンブリ・オブジェクトでこれを設定すると、トリガ Ack ビットで同じ値が確認されるまで、属性は新しい状態を保持します (これは、その変更が Checker オブジェクトで確認されることを保証するハンドシェークです)。

取得中に、トリガ Ready ビットがクリアされ、Acquiring ビットが true に設定されます。取得が完了すると、Acquiring ビットがクリアされ、取得システムの新しい画像取得の準備ができると、トリガ Ready ビットが true に設定されます。

トリガのメカニズムを強制的にリセットするには、トリガ Ready が 0 に設定されなくなるまで、TriggerEnable 属性を false に設定します。その後、TriggerEnable を true に設定して、取得を再度有効にすることができます。

トリガが使用できるのは、Checker が外部トリガモードである時だけです。結果のバッファリングがオフである場合の一般的な取得シーケンスの概要は、下の図を参照してください。



注

「いずれか不合格」、「すべて合格」、「製品検出」、「出力 0 - 7 ライン」の状態は、チェックの結果によります。示されている状態は、あくまでもサンプルです。

Results Buffer Enable=false であると、「Results Avail Ack」および「Results Buffer Overrun」が無効になります。最初に結果が設定された後、「Results Avail」は true のままになります。Inspection Complete Toggle を使用すると、IO バッファから PLC メモリへの結果をゲートすることができます。

Trigger Enable = true かつ Trigger Ready == false で、トリガの立ち上がりエッジが発生すると、Missed Ack は true になります。Missed Ack は、取得が成功するまで true のままです (Trigger Ready == true で、トリガの立ち上がりエッジが発生している場合)。

Set Offline が true であるか、Checker アプリケーションがセットアップ・モードになると、Online が false になり、Offline Reason 1 が true になります。

Checker® 4G 通信およびプログラムガイド

9/3/2014|バージョン 4.1

Trigger Enable が true になるのは、Online が最初に true になる時だけです。Trigger Ready に必要な信号のいずれかが true になると、Trigger Enable = true になります。

Trigger Ready が true になるのは、Online = true で、Trigger Enable = true で、Acquiring = false で、取得バッファが使用可能である場合です。

Trigger Ready が true になると、Trigger が true になります。Trigger Ack が true になると、Trigger は false になるはずですが、Trigger の立ち上がりエッジが true になると、Trigger Ready が false になり、その結果として Acquiring が true になります。

シーケンス中、入力 0 ラインは任意の状態になります。Acquiring 信号の立ち上がりエッジの値が、使用される値です。

Trigger が true になると、Trigger Ack が true になります。Trigger Ack が false になると、Trigger が false になります。

Trigger Ready == true であると、Trigger 信号の立ち上がりエッジ後は、Acquiring は true になります。取得とは、カメラが写真を撮り、画像が取得バッファに移動することを意味します。取得の立下りエッジは、Acquisition ID のゲート信号であり、これによってチェックが true になります。

[トリガ] 取得信号が false になると、Acquisition ID は有効のままです。この数は、[チェック] Results ID の同じ数を使って、取得した画像とチェック出力を一致させるのに使われます。

次のプロセスがパイプラインでつながっているため、取得が false になると、すぐにチェックが true になります。チェックの立下りエッジは、Results ID およびすべての結果データのゲート信号です。Inspection Complete Toggle および Results Avail は、チェックの立下りエッジで有効になります。

Results ID は、Results Available の立下りエッジ前は、有効のままです。チェックの立下りエッジでは、Inspection Complete Toggle が変化し、Results Available が true

になります。Results ID の値は、一致するカメラ画像の [トリガ] Acquisition ID の値と同じになります。

Any Fail は、Results Available の立下りエッジ前は、有効のままです。チェックの立下りエッジでは、Inspection Complete Toggle が変化し、Results Available が true になります。

「すべて合格」は、Results Available の立下りエッジ前は、有効のままです。チェックの立下りエッジでは、Inspection Complete Toggle が変化し、Results Available が true になります。

Part Detect は、Results Available の立下りエッジ前は、有効のままです。チェックの立下りエッジでは、Inspection Complete Toggle が変化し、Results Available が true になります。

出力 0 - 7 は、Results Available の立下りエッジ前は、有効のままです。チェックの立下りエッジでは、Inspection Complete Toggle が変化し、Results Available が true になります。

Inspection Complete Toggle は、チェックの立下りエッジと同時に変化します。Results ID は、Results Available が true になるたび (つまり、Inspection Complete Toggle の状態が変化) に増えます。

[チェック] Results Available はチェックの立下りエッジにすぐに従い、true になるのは、結果データが安定して有効である間だけです。

注

最初の結果の後、Results Buffer Enable = false となると、Results Available 信号は必ず true です。Inspection Complete Toggle を有効信号として使用し、IO 結果バッファからの新しい結果データを PLC メモリにコピーできます。

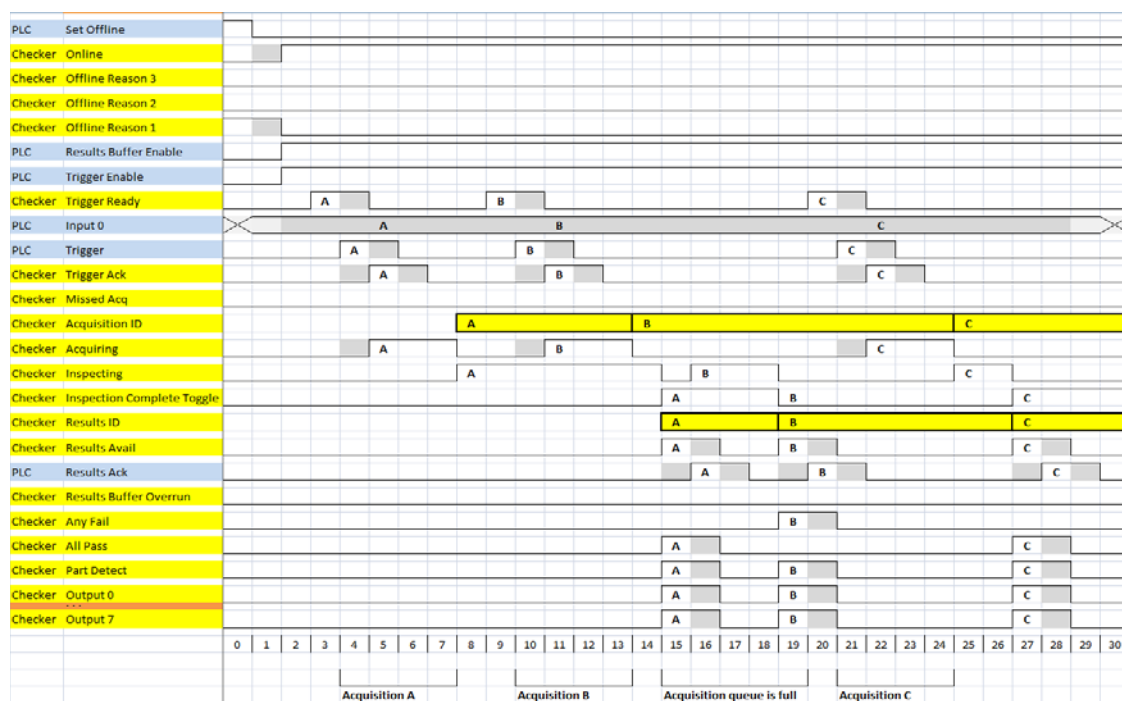
Checker® 4G 通信およびプログラムガイド

9/3/2014 | バージョン 4.1

Results Buffer Enable = false であれば、Results Ack は使用されません。

Results Buffer Enable = false であれば、Results Buffer Overrun は使用されません。

取得失敗とは、Checker が対応できない時点で、トリガが発行されたことを意味します。これによって、PLC が通常の処理ロジックをスキップして、新しいトリガを再発行することができます。チェックシーケンスより速い取得については、下の図を参照してください。



「いずれか不合格」、「すべて合格」、「製品検出」、「出力 0 - 7 ライン」の状態は、チェックの結果によります。示されている状態は、あくまでもサンプルです。

チェック/結果シーケンス

画像を取得すると、処理待ちのキューに入ります。画像の処理が実行中である時は、チェックステータスレジスタの Inspecting ビットがセットされます。チェックが完了すると、Inspecting ビットがクリアされ、Inspection Completed がトグルされます。

たとえば、速度やロバティクスを改善するために、取得とチェックで別個のインジケータを使用しているケース。取得中は、ロボットは静止したままのはずです。ただし、チェック中は、ロボットを動かすことはできません。

BufferResultsEnable 属性は、チェック結果が Checker オブジェクトによってどのように扱われるかを決定します。BufferResultsEnable 属性が false に設定されると、チェック結果はただちに InspectionResults 属性に配置され、Results Available が true に設定されます。

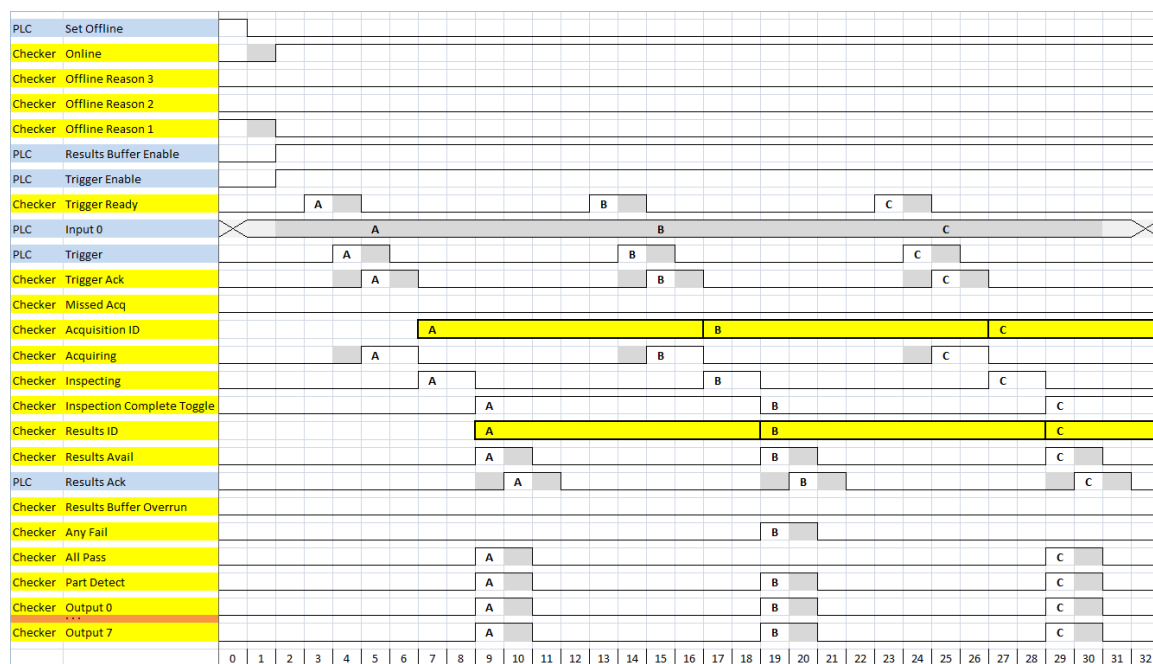
BufferResultsEnable 属性を true に設定すると、新しい結果はキューに入ります。内部トリガを使用している場合、次のトリガによって結果が上書きされます。外部トリガを使用している場合は、アプリケーションによって使用されていなくても、次の外部トリガによって結果が上書きされます。

Inspection ビットの挙動

ビット	ビット名	バッファリング 無効時の結果	バッファリング有効時の結果
1	Inspecting	画像のチェック 時にセット	画像のエンコード時にセット
2	Inspection Complete	画像のチェック 完了時にトグル	画像のチェック完了時にトグル
3	Results Buffer Overflow	ゼロにセットさ れたまま	クライアントが前の結果を確認できなかったためにチェック結果をキューに入れることができなかった場合にセットされます。チェック結果が正常にキューに入ればクリアされます。 オーバーフローは、何かが失われたことを

ビット	ビット名	バッファリング 無効時の結果	バッファリング有効時の結果
			知らせています。バッファリングを有効にしているかどうかによって、以前のチェックが失われたか (バッファリングなし)、最後の結果が失われた (バッファリングでキュー満杯) ことを意味します。
4	Results Available	最初のチェックの後、true になり、そのまま true です。	新しい結果が results ビットに配置されると (いずれか不合格、すべて合格、製品検出、出力)、セットされます。Results Available Ack を true に設定することで、結果を確認するまでセットされたままになります。 注:必ず、Results Available / Results Available Ack のハンドシェークのロジックが適用されます。

バッファリングが有効である場合のシーケンスについては、下の図を参照してください。



「いずれか不合格」、「すべて合格」、「製品検出」、「出力 0 - 7 ライン」の状態は、チェックの結果によります。示されている状態は、あくまでもサンプルです。

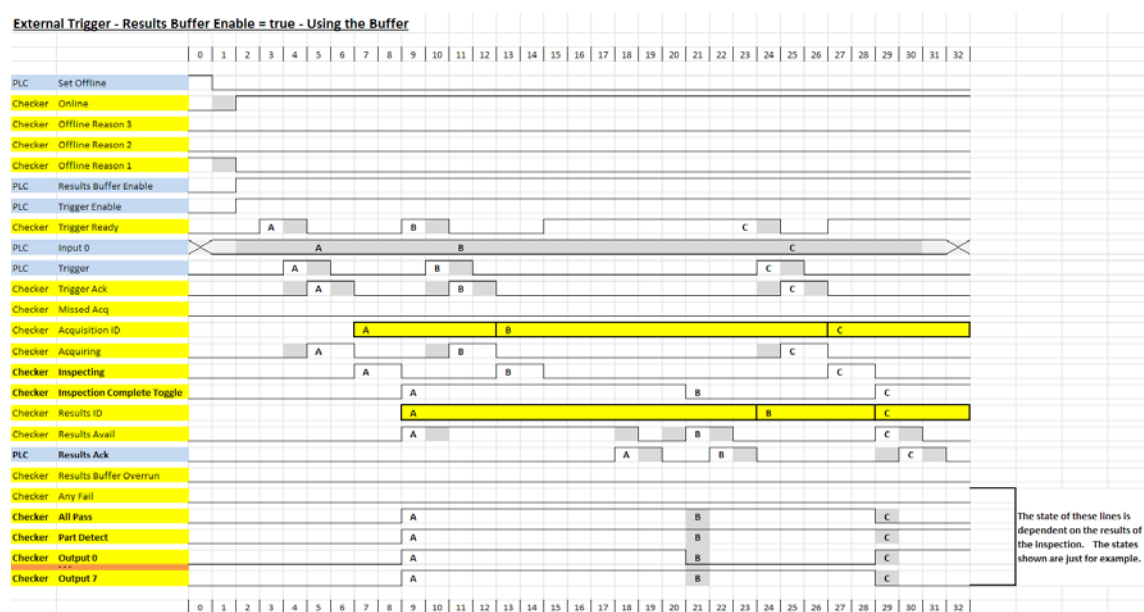
結果のバッファリング

チェック結果に対してキューを有効にするオプションがあります。有効にすると、クライアント (PLC) に読み取るための時間ができるまで、有限の数のチェック結果データがキューに入ります。これは、短い期間の間、システムの異なる部分 (外部 PLC を含む) がスローダウンする場合に、データの流れを円滑にするのに有益です。

一般に、結果が送信されるより速くチェックが行われる場合、バッファリングの有無による主な差異はどの結果が無視されるかです。バッファリングが有効でなければ、最も新しい結果が保持され、以前の結果 (PLC が十分速く読み取れなかった) は失われます。実質的には、単に最も新しい結果が以前の結果を上書きすることになります。バッファリングが有効であれば (および、キューが満杯)、結果キューに十分な空きができるまで、最も新しい結果が無視されます。

下の図に示されているように、A Acq に対応する最初の結果が、結果バッファおよび出力ラインに移動します。これに続いて、B の結果が結果バッファへ移動しますが、出力ラインには引き続き A の結果が残ります。PLC が最初の結果 (A) を確認すると、バッファおよび出力ラインから消えます。結果 B が出力に配置されます。PLC が B の読み取り結果を確認すると、バッファが空になり、Result Ack は低のままになります。

C Acq に対応する結果が、結果バッファおよび出力ラインへ移動します。



アセンブリ・オブジェクト

アセンブリ・オブジェクトは、暗示的メッセージングを使用します。抽象化によって、単なる暗示的メッセージング・パケットの未処理ペイロードとして送信されるデータのブロックとなります。これらの暗示的メッセージングのパケットは、事前に選択されたレート (10ms、200ms、等) で、繰返し生成 (送信) されます。

Checker® 4G 通信およびプログラムガイド

9/3/2014 | バージョン 4.1

一般に、アセンブリは、デバイス内の異なる CIP オブジェクトから選択された属性 (データ項目) の組み合わせになります。デバイスのベンダーは、必要性に応じてアセンブリを定義できます。アセンブリは、アプリケーションの要件に応じて、便利なグループ化でデータを組み合わせます。

入力および出力のアセンブリの指定は、若干混乱しがちです。Checker は I/O アダプターのクラスのデバイスです。アダプターは、入力アセンブリが別のデバイスへデータを生成 (送信) し (例、Checker → PLC)、出力アセンブリが別のデバイスからのデータを消費 (受信) します (例、PLC → Checker)。つまり、Checker は別のデバイスにとっての I/O モジュールとして機能することになります。

Checkers には、単一の入力アセンブリと単一の出力アセンブリがあります。これらのアセンブリは、Checker オブジェクトの選択した属性 (データ) を組み合わせてグループ化し、ネットワークの帯域を最小化しながら、効率的な制御や処理を維持することができます。また、こうしたアセンブリのデータは Checker オブジェクトから個別にアクセスすることもできます。ただし、アセンブリ・オブジェクトを使用するとさらに効率的です。これが、Checker と PLC の間でランタイム通信を行う主要な理由です。

入力アセンブリ

入力アセンブリのインスタンスは、ステータス情報、処理状態、チェック結果を提供します。下のテーブルは、入力アセンブリ (Checker から送信されるデータ) を示しています。ここで、予約済み¹とは将来の使用のために予約されていることを意味します。

Observer ビットが 1 に設定されると、EIP コントローラは Checker のステータスを変更することができず、その状態を観察することができるだけです。Observer ビットは PC アプリケーションによってのみ設定できます ([Checker] → [Checker の設定] → [プロトコル設定])。詳細情報は、付録 A – プロトコル設定、オブザーバ・ビットの変更を参照してください。

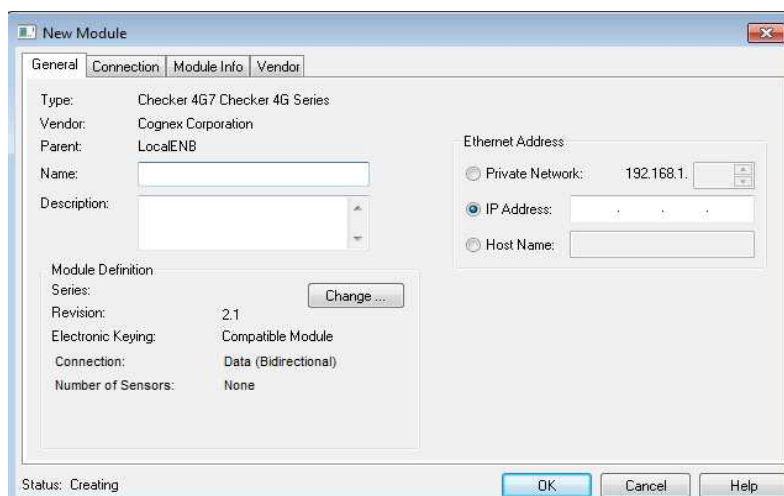
インスタンス	バイト	ビット 7	ビット 6	ビット 5	ビット 4	ビット 3	ビット 2	ビット 1	ビット 0
11	0	オンライン	オフラインの理由			Missed Acq (確認失敗)	Acquiring (取得)	Trigger Ack (トリガ確認)	Trigger Ready (トリガ・レディ)
			0 – Online (オンライン) 1 – Setup (設定) 2-7 – 予約済み						
	1	General Fault (一般的なフ ォールト)	Job Load Failed (ジョブのロ ード失敗)	Job Load Complete (ジョブのロ ード完了)	Observer (オブザー バー)	Results Available (結果使用可能)	Results Buffer Overrun (結果バッファ・ オーバーラン)	Inspection Complete Toggle (チェック完 了トグル)	Inspecting (チェック)
	2	予約済み ¹	予約済み ¹	予約済み ¹	Retrain Failed (再登録失 敗)	Retrain Completed (再登録完了)	Any Fail (いずれか不合格)	All Pass (すべて合格)	Part Detect (製品検出)
	3	出力7	出力6	出力5	出力4	出力3	出力2	出力1	出力0
	4	出力15	出力14	出力13	出力12	出力11	出力10	出力9	出力8
	5	出力23	出力22	出力21	出力20	出力19	出力18	出力17	出力16
	6	Acquisition ID (16-ビットの整数)							
	7								
	8	Inspection Result ID (16-ビットの整数)							
	9								
	10	予約済み							
	11	予約済み							
	12	センサー 1 結果データ 0							
	...	...							
	27	センサー 1 結果データ 15							
	28	センサー 2 結果データ 0							
	...	...							
	43	センサー 2 結果データ 15							
	44	センサー 3 結果データ 0							
	...	...							

インスタンス	バイト	ビット 7	ビット 6	ビット 5	ビット 4	ビット 3	ビット 2	ビット 1	ビット 0
	411	センサー 25 結果データ 15							
	412	センサー 26 結果データ 0							
	...	...							
	427	センサー 26 結果データ 15							

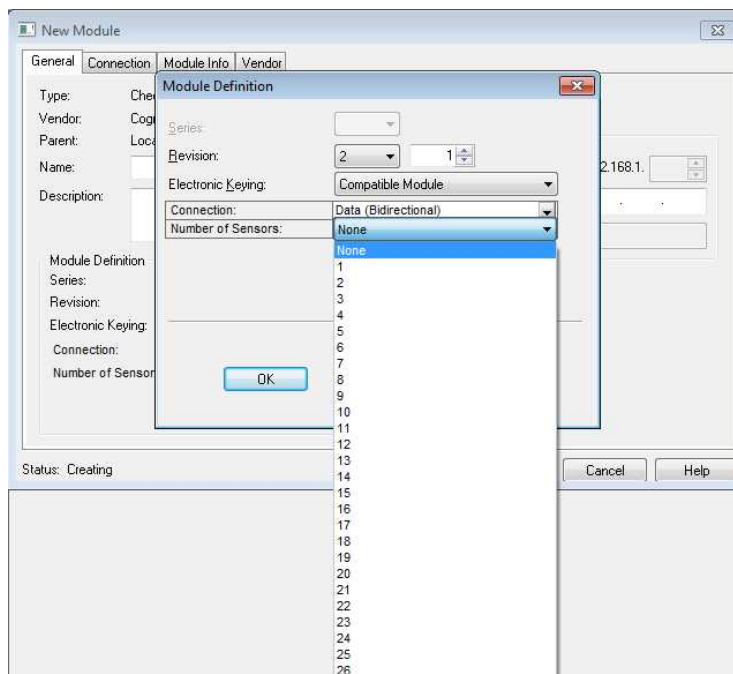
Rockwell AOP を使用してセンサーの結果データを有効にする

デフォルトですべてのセンサーが表示されているわけではないので、後で有効にします。これを行うには、以下のステップを実行します：

1. Checker モジュールを右クリックします。
2. [プロパティ] を選択します。
3. [General] タブの [Module Definition] セクションで **[Change]** をクリックします。



4. **[Number of Sensors]** ドロップダウンリストで、センサーを設定します。これはセンサーメーター値に関するものであり、なるべく機能できる範囲で最小の数を設定してください。



出力アセンブリ

出力アセンブリのインスタンスには、制御信号、ソフトウェア・イベント信号、トリガやチェックに必要なユーザーデータなどが含まれています。次のテーブルは、出力アセンブリのインスタンスを示しています (Checker へ送信されるデータ)。ここで、予約済み¹ とは将来の使用のために予約されていることを意味します。Lights Off ビットを 1 に設定すると、ジョブの設定に関わらず、外部照明がオフになります。SM Volatile ビットを 0 に設定すると、ジョブに対して変更されないスタティックな結果データを選択します。これを 1 に設定すると、製品によって異なる変動データが選択されます。このデータは、センサー・メーター・テーブルの値で返されます。

アセンブリの使用法に関する詳細は、デバイス・ステータス・モジュールのセクションを参照してください。

インスタンス	バイト	ビット 7	ビット 6	ビット 5	ビット 4	ビット 3	ビット 2	ビット 1	ビット 0
21	0	Set Offline (オフライン設定)	Lights Off (照明オフ)	Retrain (再登録)	Job Change (ジョブの変更)	Results Available Ack (結果使用可能確認)	Buffer Results Enable (バッファ結果有効)	Trigger (トリガ)	Trigger Enable (トリガ有効)
	1	入力7	入力6	入力5	入力4	入力3	入力2	入力1	入力0
	2	入力15	入力14	入力13	入力12	入力11	入力10	入力9	入力8
	3	入力23	入力22	入力21	入力20	入力19	入力18	入力17	入力16
	4	Job Number (8-ビットの整数)							
	5	予約済み ¹							SM Volatile (SM 変動)
	6	調整用バイト							
	7	調整用バイト							

スタティック・データおよび変動データの概要

センサー・メーター・テーブルで替えされるスタティックおよび変動のデータは、すべてのプロトコル (EtherNet/IP、PROFINET、Generic FFP) で同じです。

次のテーブルは、センサー・メーター・テーブルの値として返されるスタティック・データをまとめたものです。

表記：

- (背景が黄色) はスタティック・データであることを意味しています。
- 「該当なし」は、特定のセンサーのタイプでデータタイプが利用不可であることを意味しています。

スタティック・テーブル - 12 バイト

	設定ポイント 1	設定ポイント 2	センサー角度	センサー角度範囲 1	センサー角度範囲 2	センサー幅	センサー高さ
ビット	8	8	16	16	16	16	16
製品検出	X	該当なし	X	該当なし	該当なし	X	X
パターン・プレゼンス	X	該当なし	X	X	X	X	X
輝度プレゼンス	X	該当なし	X	該当なし	該当なし	X	X
コントラスト・プレゼンス	X	該当なし	X	該当なし	該当なし	X	X
エッジ・プレゼンス	X	X	X	X	X	X	X
輪郭プレゼンス	X	該当なし	X	X	X	X	X
カラー・プレゼンス	X	X	X	該当なし	該当なし	X	X
測定幅	X	X	X	該当なし	該当なし	X	X
測定高さ	X	X	X	該当なし	該当なし	X	X
測定直径	X	X	X	該当なし	該当なし	X	X

パターン位置	該当なし	該当なし	X	X	X	X	X
エリア (ブロブ) 位置	該当なし	該当なし	X	該当なし	該当なし	X	X
エッジ位置	該当なし	該当なし	X	X	X	X	X
輪郭位置	X	該当なし	X	X	X	X	X

次のテーブルは、センサー・メーター・テーブルの値として返される変動データをまとめたものです。

表記：

- X (背景が白色) は変動データであることを意味しています。
- X (背景が黄色) はスタティック・データであることを意味しています。
- 「該当なし」は、特定のセンサーのタイプでデータタイプが利用不可であることを意味しています。

スタティック・テーブル – 全体で 16 バイトおよび 26 センサーが可能									
	アナログ 出力	合格/検 出	センサー 中央 X	センサー 中央 Y	フィーチャ ー中央 X	フィーチャ ー中央 Y	フィーチャ ー幅	フィーチャ ー高さ	フィーチャ ー角度
ビット	8	8	16	16	16	16	16	16	16
製品検出	X	X	X	X	X	X	X	X	X

パターン・プレゼンス	X	X	X	X	該当なし	該当なし	該当なし	該当なし	該当なし
輝度プレゼンス	X	X	X	X	該当なし	該当なし	該当なし	該当なし	該当なし
コントラスト・プレゼンス	X	X	X	X	該当なし	該当なし	該当なし	該当なし	該当なし
エッジ・プレゼンス	X	X	X	X	該当なし	該当なし	該当なし	該当なし	該当なし
輪郭プレゼンス	X	X	X	X	X	X	X	X	X
カラー・プレゼンス	X	X	X	X	該当なし	該当なし	該当なし	該当なし	該当なし
測定幅	X	X	X	X	X	X	X	X	該当なし
測定高さ	X	X	X	X	X	X	X	X	該当なし
測定直径	X	X	X	X	X	X	X	X	該当なし
パターン位置	該当なし	X	X	X	X	X	X	X	X
エリア(プロブ)位置	該当なし	X	X	X	X	X	X	X	該当なし

エッジ 位置	該当なし	X	X	X	X	X	該当なし	X	X
輪郭位置	X	X	X	X	X	X	X	X	X

注

以下の値が、合格/検出フィールドで有効です：

バイナリ値 (10 進相当)	意味
00 (0)	未検出
10 (2)	検出したが不合格
11 (3)	検出、合格

Checker の実装

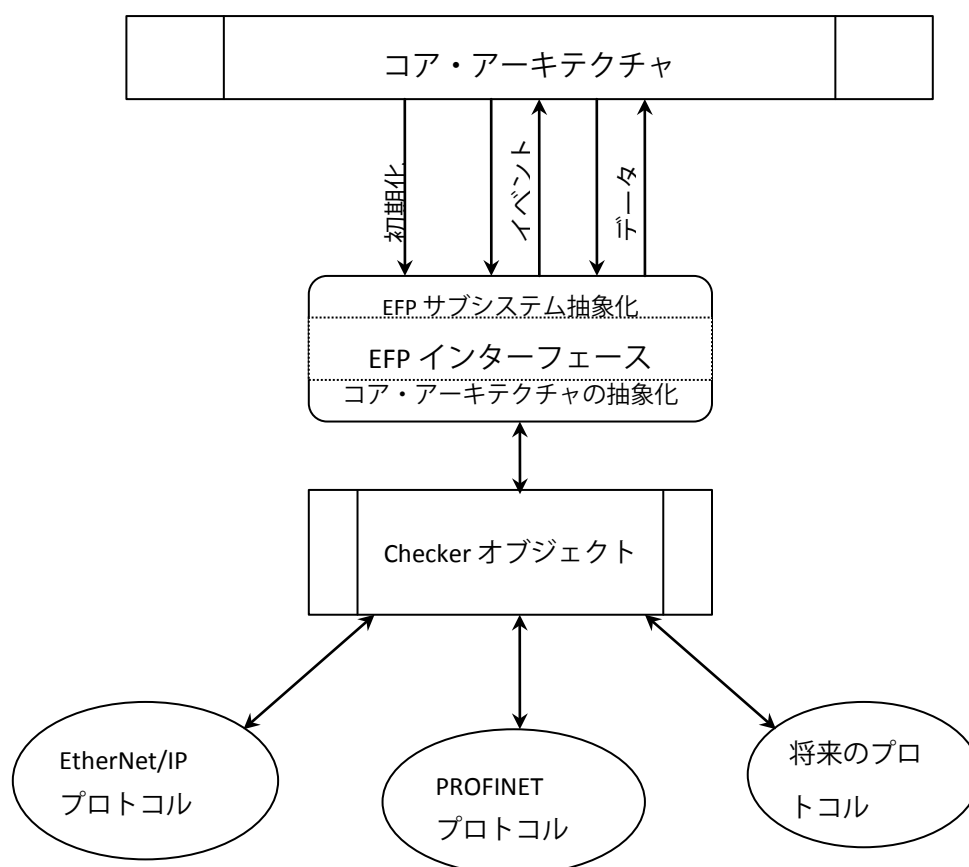
Ethernet 現場プロトコル・サブシステム

Ethernet Factory Protocol (EFP) サブシステムは、Checker 製品用に実装されている、すべての “工場現場” Ethernet 作業用プロトコルのグループ化された実装です。現在、これには EtherNet/IP と PROFINET が含まれています。

このサブシステムは、共通インターフェースに抽象化されます。このインターフェースは、初期化、イベントのトラフィック追跡、データフローの共通の方法となります。

このインターフェースは、両方向へ抽象化されます。まず、Checker のコア・アーキテクチャから EFP サブシステムの詳細を抽象化します。さらに、EFP サブシステムのプロトコルから、Checker のコア・アーキテクチャの詳細も抽象化します。

一般に、ある時点で動作しているプロトコルは 1 つだけです。EFP サブシステムの中心は Checker オブジェクトです。Checker オブジェクトは、Checker で使用可能なすべてのデータや機能（少なくとも、Ethernet の工場プロトコルを通じて、エンドユーザーが操作できるすべて）をモデル化します。Checker オブジェクトによって、Checker デバイスは、使用している工場プロトコルの種類に関わらず、共通のデータや動作を持つことができます。



汎用的な暗示的メッセージング接続の確立

まだ AOP または EDS のファイルをインストールしていなくても、このセクションの説明に従って Checker を使用することができます。Checker と ControlLogix コントローラの

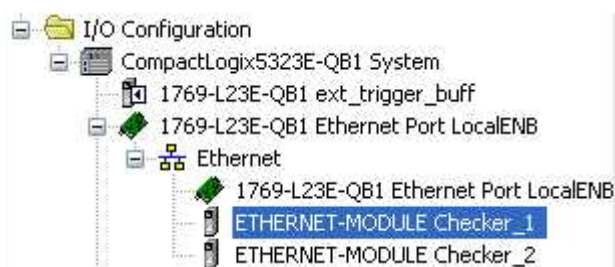
Checker® 4G 通信およびプログラムガイド

9/3/2014 | バージョン 4.1

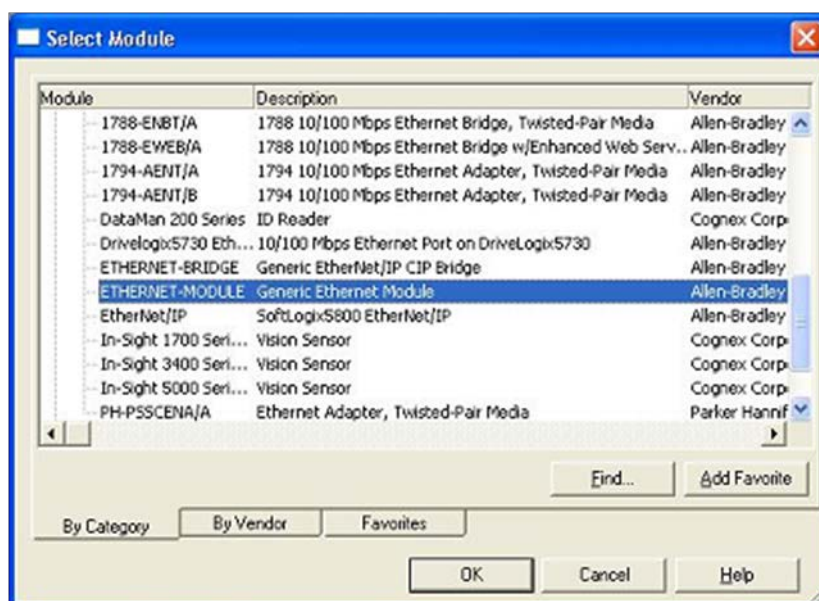
間で EtherNet/IP 暗示的メッセージング接続を設定するには、まず Checker センサーを ControlLogix I/O の構成ツリーに追加する必要があります。これは、Rockwell が提供する汎用プロファイルで実現できます。

ControlLogix PLC で、汎用暗示的メッセージング接続を確立するには：

1. RSLogix5000 を起動して、プロジェクトをロードします (あるいは、[ファイル] -> [新規作成...] と選択して新規に作成)。
2. I/O 構成ノードのプロジェクトの Ethernet モジュールで、[Ethernet] ノードを選択してから、アイコンを右クリックし、メニューで [新規モジュール] を選択します：



3. [Select Module] ダイアログボックスで、[Allen-Bradley Generic Ethernet Module] を選択します。

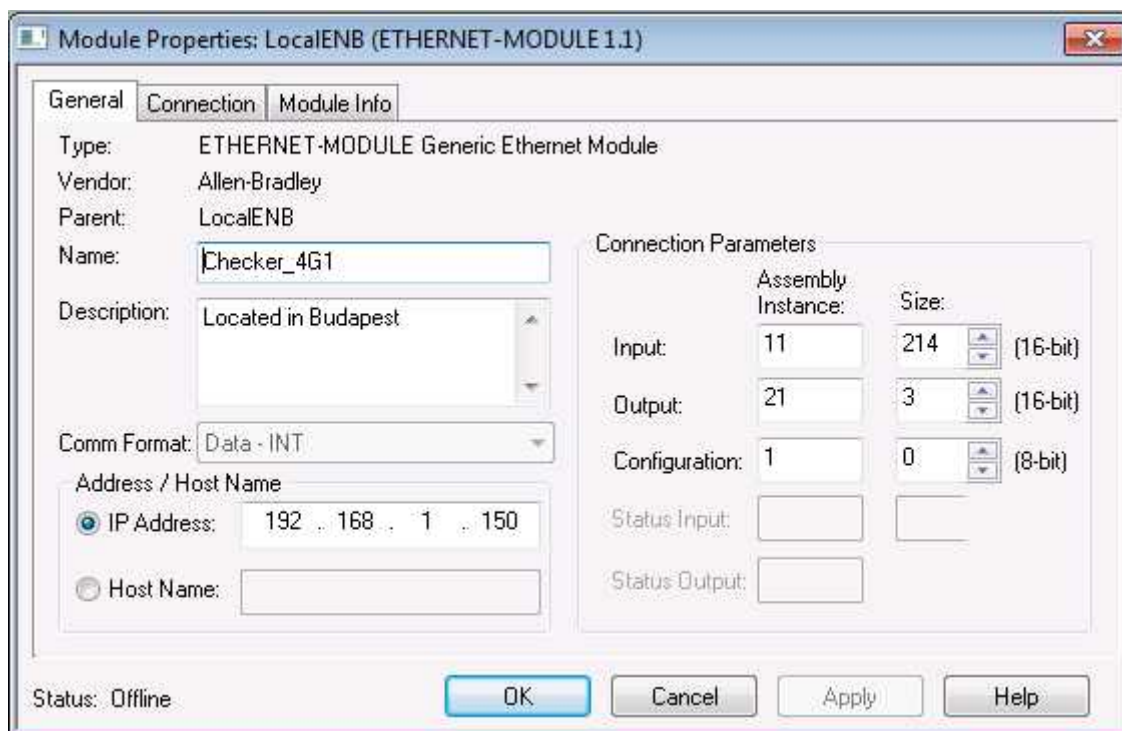


Checker® 4G 通信およびプログラムガイド

9/3/2014 | バージョン 4.1

4. 選択が終わると、Generic Ethernet モジュールの設定ダイアログボックスが表示されます。以下のように、設定します：

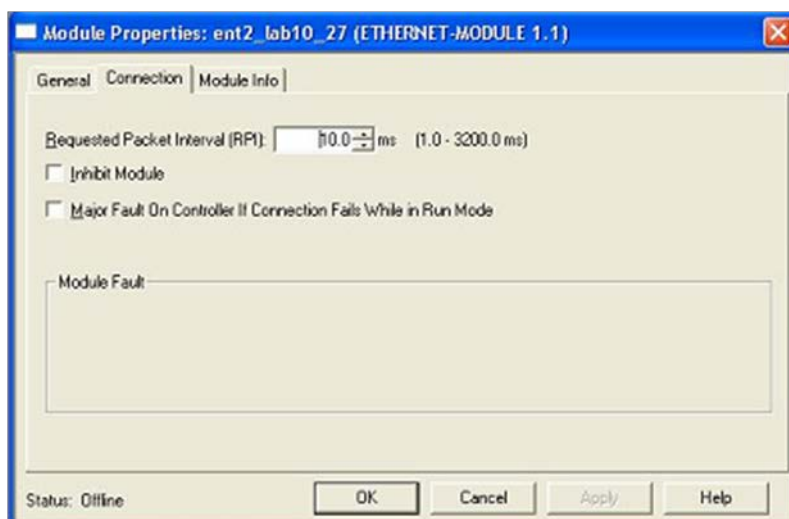
- モジュール名を指定します。
- Checker の IP アドレスを入力します。
- 通信フォーマットを[データ - INT] に設定します。これで、モジュールはデータを 16 ビット整数の配列として扱います。
- 入力アセンブリ：インスタンスを 11 に設定します。サイズを、PLC に受信させたい入力アセンブリ・データの量に設定します。サイズは、最低でも 5 ワード ($5 \times 16 = 80$ ビット) が必要です。同様にセンサーからのデータも受信したい場合は、サイズをセンサー当たり 16 バイト単位で増やします (8×16 -ビット単位 = 8 ワード)。最大で 26 センサーからのデータを受信できます (214 ビット、下の図を参照)。
- 出力アセンブリ：インスタンスを 21 に設定します。サイズを 3 つの整数に設定します。このサイズは、すべての必要な「制御」データ/コマンドを Checker へ送信できるサイズです。
- アセンブリの設定：インスタンスを 1 に設定します。サイズをゼロ (使用しない) に設定します。



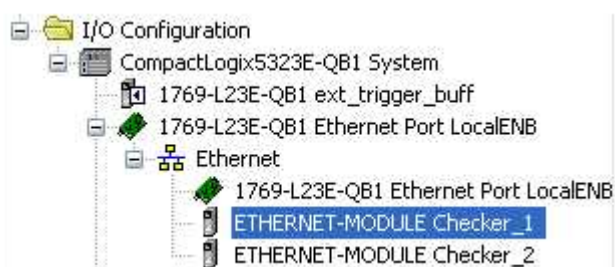
5. 最後のステップは、接続レートの設定です。データを送信/受信するレートは、要求/パケット間隔 (RPI) として定義します。RPI は、接続を通じてデータが送信/受信される頻度を定義します。ネットワークのパフォーマンスを最適化するために、このレートは該当するアプリケーションが絶対的に必要とする値より低く設定してください。いかなる場合でも、PLC ラダー・プログラムの平均スキャンレートの $\frac{1}{2}$ より低くは設定しないでください。それより低く設定すると、帯域が無駄になり、処理パフォーマンスは改善されません。

注

Checker とこれらの PLC タグ値の間には時間のディレイがあります (設定済みの RPI に基づく)。すべての PLC ラダーに書き込まれるには、この時間のディレイを考慮する必要があります。



6. ControlLogix に汎用モジュールを追加すると、以下のように I/O ツリーが表示されるはずです。



7. 汎用モジュールが I/O ツリーに追加されると、RSLogix 5000 は Checker センサーの入力および出力のデータをマッピングするためのタグを追加します (つまり、Checker センサーの入力および出力のアセンブリ・オブジェクト)。これらのタグは、プロジェクト・ツリーの [コントローラ・タグ] ノードにあります。

注

これらのタグのベース名は、前に I/O 構成に追加した汎用モジュールに与えた名前になります。

The screenshot displays the 'Controller Organizer' interface. The left pane shows the project structure, including 'Controller ext_trigger_buff', 'Tasks', 'Motion Groups', 'Data Types', and 'I/O Configuration'. The 'I/O Configuration' section shows the 'CompactLogix5323E-QB1 System' with '1769-L23E-QB1 Ethernet Port LocalENB' selected. The 'ETHERNET-MODULE Checker_1' is highlighted. The right pane shows the 'Scope: ext_trigger_buff' and 'Show: All Tags'. The tag table lists various tags for 'Checker_1:C', 'Checker_1:I', and 'Checker_1:O'. The bottom pane shows the 'Module Defined Tags' for 'Checker_1:I', 'Checker_1:O', and 'Checker_1:C'.

Name	Value	Description
Checker_1:C	{...}	
Checker_1:I	{...}	
Checker_1:I.Data	{...}	
Checker_1:I.Data[0]	0	
Checker_1:I.Data[0].0	0	Trigger Ready
Checker_1:I.Data[0].1	0	Trigger ACK
Checker_1:I.Data[0].2	0	Acquiring
Checker_1:I.Data[0].3	0	Missed Acq
Checker_1:I.Data[0].4	0	Offline Reason1
Checker_1:I.Data[0].5	0	Offline Reason2
Checker_1:I.Data[0].6	0	Offline Reason3
Checker_1:I.Data[0].7	0	IN_0.7_Online
Checker_1:I.Data[0].8	0	Inspecting
Checker_1:I.Data[0].9	0	Inspection Complete Toggle
Checker_1:I.Data[0].10	0	Result Buffer Overrun
Checker_1:I.Data[0].11	0	Result Available
Checker_1:I.Data[0].12	0	Job Loading
Checker_1:I.Data[0].13	0	Job Load Complete
Checker_1:I.Data[0].14	0	Job Load Failed
Checker_1:I.Data[0].15	0	General Fault
Checker_1:I.Data[1]	0	
Checker_1:I.Data[2]	0	
Checker_1:I.Data[3]	0	Acq ID
Checker_1:I.Data[4]	0	Inspection ID
Checker_1:O	{...}	
Checker_1:O.Data	{...}	
Checker_1:O.Data[0]	0	
Checker_1:O.Data[0].0	0	Trigger Enable
Checker_1:O.Data[0].1	0	Trigger
Checker_1:O.Data[0].2	0	Buffer Result Enable
Checker_1:O.Data[0].3	0	Result Ack
Checker_1:O.Data[0].4	0	Job Change
Checker_1:O.Data[0].5	0	-
Checker_1:O.Data[0].6	0	-
Checker_1:O.Data[0].7	0	Set Offline
Checker_1:O.Data[0].8	0	Input0
Checker_1:O.Data[0].9	0	
Checker_1:O.Data[0].10	0	
Checker_1:O.Data[0].11	0	
Checker_1:O.Data[0].12	0	
Checker_1:O.Data[0].13	0	
Checker_1:O.Data[0].14	0	
Checker_1:O.Data[0].15	0	
Checker_1:O.Data[1]	0	
Checker_1:O.Data[2]	0	Job Number (8-bit)

Module Defined Tags

Tag Name	Description	Status	Module Fault
Checker_1:I	by default it is a 4G7	Offline	
Checker_1:O			
Checker_1:C			

タグは、次の 3 つのグループで編成されます：構成の “Checker_1:C”、入力の “Checker_1:I”、出力の “Checker_1:O” です。構成タグは無視できます（未使用）。

Checker® 4G 通信およびプログラムガイド

9/3/2014 | バージョン 4.1

ページ | 49

入力タグは、受信するすべてのデータを表します (Checker から受信)。出力タグは、送信するすべてのデータを表します (Checker へ送信)。

これらのタグは、Checker アセンブリ・オブジェクトのコンテンツを表すデータテーブルです。PLC ラダーに書き込まれて、これらのタグの値にアクセスできるようになります。これらのタグの値を監視または変更することで、実際には PLC ラダーが Checker アセンブリ・オブジェクトのコンテンツを監視または変更します。

汎用暗示的メッセージング接続データへのアクセス

Checker のアドオン・プロファイルとは違って、汎用プロファイルは、アセンブリ・オブジェクト内の個別のデータ項目を表す汎用的な名前付きタグを自動的に生成しません。代わりに、定義した接続のサイズに従って、単にデータの配列を生成するだけです。

アセンブリ・オブジェクト内の個別のデータ項目にアクセスするには、汎用プロファイルが提供するタグ配列内で、正しいタグのオフセットやデータのサブタイプ (必要な場合) を手動で選択する必要があります。これは面倒な作業であり、アセンブリ・オブジェクトを定義するベンダーのドキュメントを手動で参照する必要があるため、間違いを起こしやすいです。

注

入力タグ “Checker_1:I.Data[0]” の開始は、直接 Checker 入力アセンブリにマッピングされます。同様に、出力タグ “Checker_1:O.Data[0]” の開始は、直接 Checker 出力アセンブリにマッピングされます。

例

出力アセンブリ・トリガー有効：出力アセンブリのワード 0 のビット 0 です。Checker の出力タグ配列から、ワード 0 のビット 0 を選択します。

- Checker_1:0	{...}	
- Checker_1:0.Data	{...}	
- Checker_1:0.Data[0]	0	
Checker_1:0.Data[0].0	[1]	Trigger Enable
Checker_1:0.Data[0].1	0	Trigger
Checker_1:0.Data[0].2	0	Buffer Result Enable
Checker_1:0.Data[0].3	0	Result Ack
Checker_1:0.Data[0].4	0	Job Change
Checker_1:0.Data[0].5	0	-
Checker_1:0.Data[0].6	0	-
Checker_1:0.Data[0].7	0	Set Offline
Checker_1:0.Data[0].8	0	Input0
Checker_1:0.Data[0].9	0	
Checker_1:0.Data[0].10	0	
Checker_1:0.Data[0].11	0	
Checker_1:0.Data[0].12	0	
Checker_1:0.Data[0].13	0	
Checker_1:0.Data[0].14	0	
Checker_1:0.Data[0].15	0	
+ Checker_1:0.Data[1]	0	
+ Checker_1:0.Data[2]	0	Job Number (8-bit)

デバイス・アクセス・コード

以下のテーブルには、Checker センサーのアクセス・コードをまとめています。

	EtherNet/IP ベンダー ID	EtherNet/IP デバイス・タイプ	EtherNet/IP 製品コード
Checker 4G1	0x2a6	0x306	0x601
Checker 4G7	0x2a6	0x306	0x607

サンプルのプログラムは、次のリンクから入手できます：

EIP サンプル：<http://www.cognex.com/support/downloads/File.aspx?d=2469>

PROFINET

PROFINET

は、産業用自動化アプリケーションで使用されるアプリケーション・レベルのプロトコルです。このプロトコルは、標準 Ethernet のハードウェアおよびソフトウェアを使用して、I/O データ、アラーム、診断を交換します。

Checker は PROFINET I/O をサポートしています。これは、PROFINET 通信基準に含まれている 2 つの「ビュー」のうちの 1 つです。PROFINET I/O は、循環的なデータ送信を行って、Ethernet 経由でプログラマブル・ロジック・コントローラ (PLC) とデータ交換を行います。標準 PROFINET CBA (コンポーネント・ベースの自動化) の 2 番目の「ビュー」はサポートされていません。

慎重な努力を行って、Checker PROFINET 通信モデルを Cognex In-Sight ファミリーとマッチさせています。In-Sight の顧客は、Checker ファミリーと快適に連携できることを体験できます。

PROFINET との連携を有効にするには、以下のステップを実行します：

- Siemens Step 7 プログラミング・ソフトウェア (SIMATIC) がインストールされていることを確認します。
- Checker デバイスが Siemens Software ツールによって認識されるように設定します。

Generic Station Description (GSD) ファイルをインストールします。

- Checker GUI で PROFINET を有効にします。

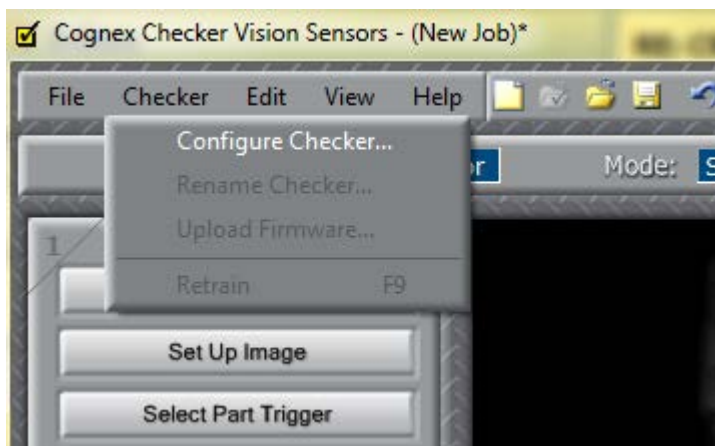
Checker GUI で PROFINET を有効にする

デフォルトでは、Checker では PROFINET プロトコルは無効になっています。このプロトコルを Checker GUI で有効にすることができます。以下のステップを実行します：

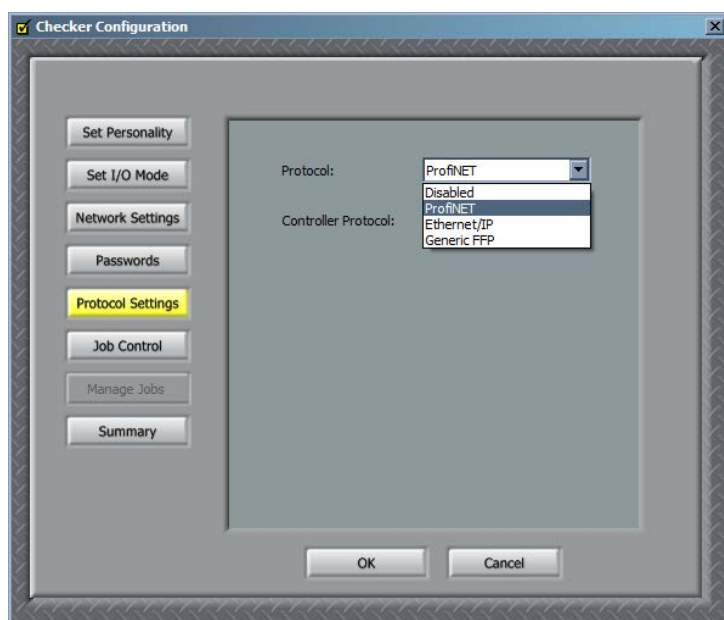
Checker® 4G 通信およびプログラムガイド

9/3/2014 | バージョン 4.1

1. 上のメニュー・バーで、[Checker] をクリックします。
2. [Checker の設定] を選択します。



3. ポップアップ表示される Checker 設定ウィンドウで、[プロトコル設定] をクリックし、[PROFINET] を選択します。



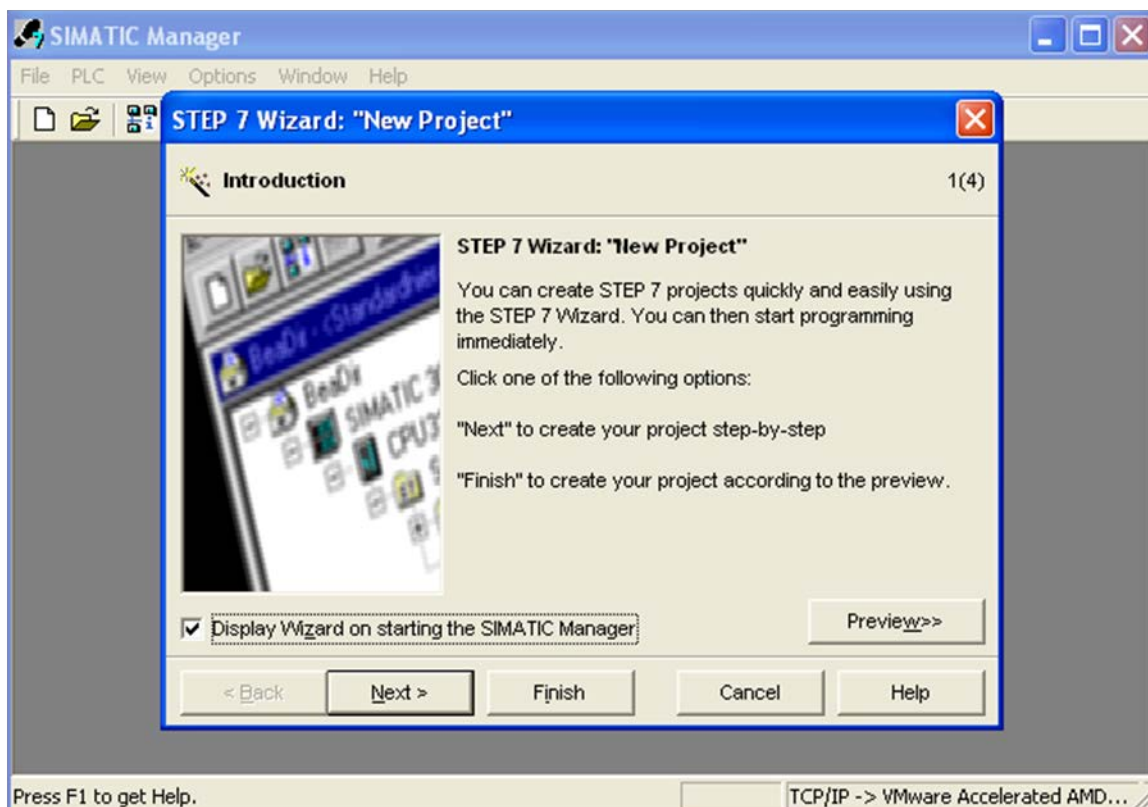
4. [OK] をクリックします。

変更を有効にするために、センサーは自動的に再起動します。

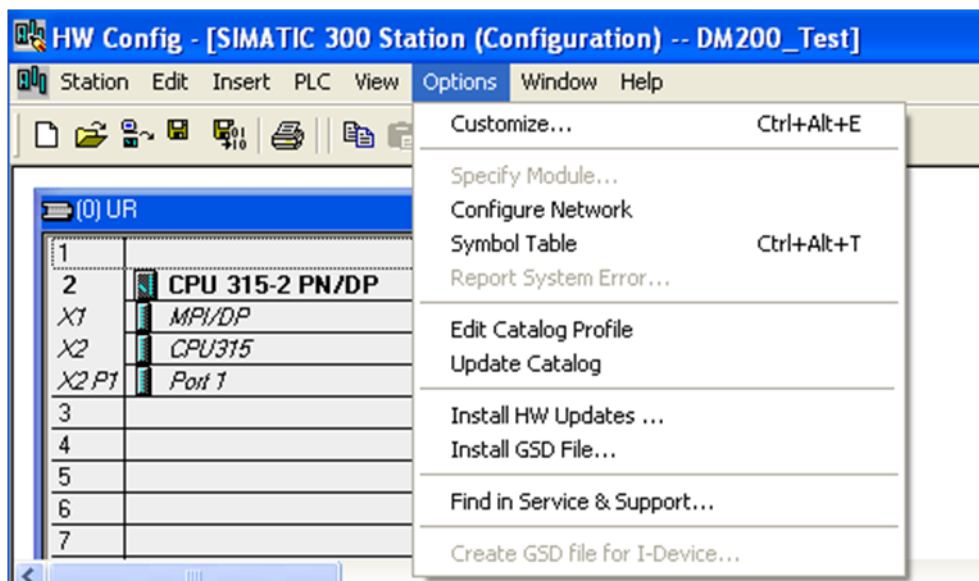
PROFINET の設定

以下のステップを実行して、PROFINET をセットアップします：

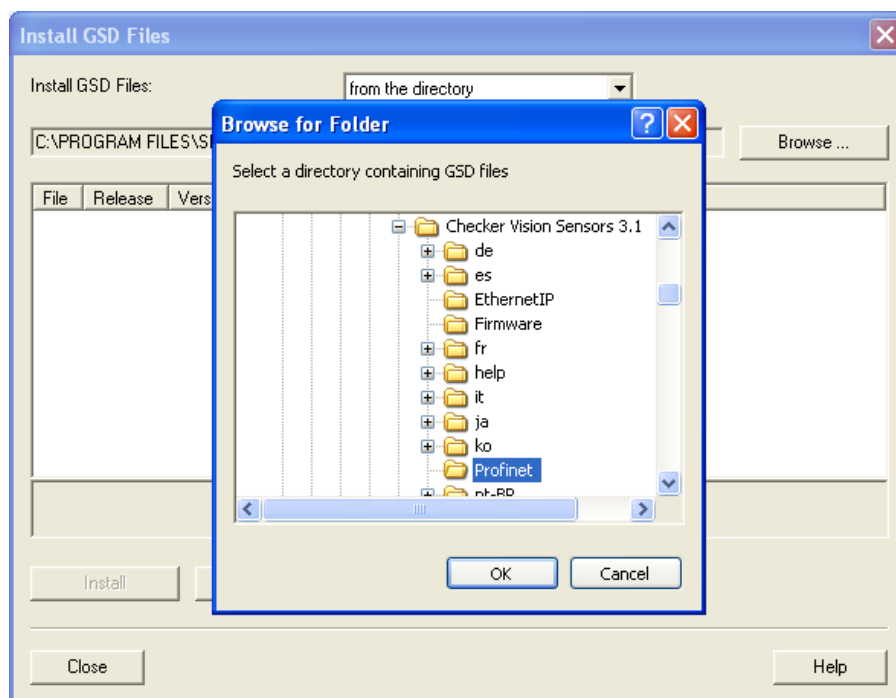
1. SIMATIC がマシンにインストールされていることを確認します。
2. Windows のスタート・メニューで、SIMATIC Manager を起動します。



3. 既にプロジェクトがある場合は、[Cancel] をクリックして、新規プロジェクト・ウィザードを省略します。それ以外の場合は、ウィザードによって新規プロジェクトの作成を進めてください。
4. Manager によってプロジェクトが開かれたら、[Hardware] アイコンをダブルクリックして、[HW Config] ダイアログ画面を開きます。メインメニューで、[Options] → [Install GSD File...] と選択します。



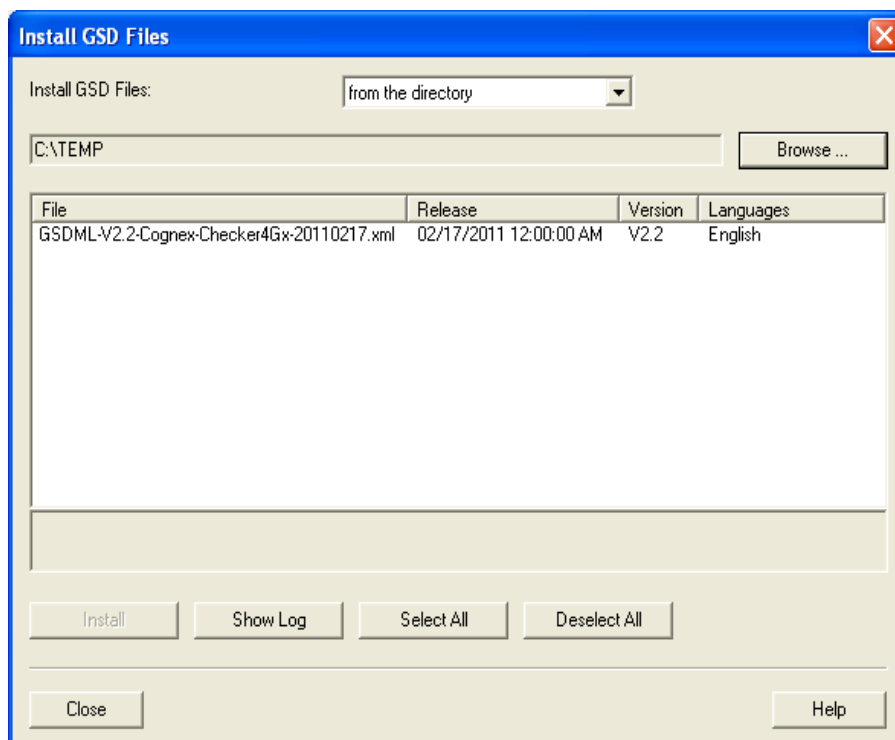
5. GSD ファイルがインストールされた場所を参照します (ウェブからダウンロードした場合は、GSD ファイルを保存した場所)。



6. インストールしたい GSD ファイルを選択し、表示される指示に従って、インストールを完了させます。

注

リストには、複数の GSD ファイルが表示されている可能性があります。
 どれをインストールすればよいかわからない場合は、最も日付が新しいファイルを選んでください。

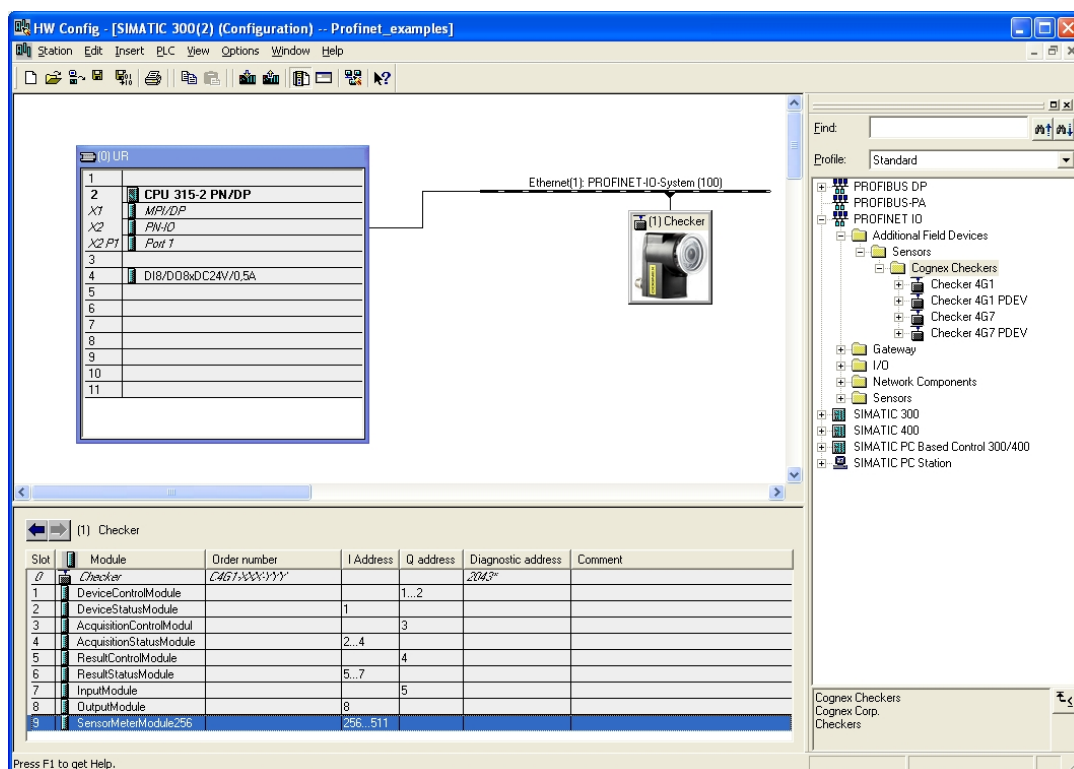


7. Checker センサーをプロジェクトに追加します。この操作で、Checker がハードウェア・カタログで使えるようになります。SIMATIC Hardware Config ツールを起動します。
8. メインメニューで、[→ Catalog の表示] を選択します。
9. カタログが表示されます。[PROFINET IO] ツリーを展開して、[Cognex Checkers] ノードを表示させます。

10. 左マウスボタンで、Checker センサーをドラッグして、左のペインにある PROFINET IO ネットワークの記号にドロップします。

注

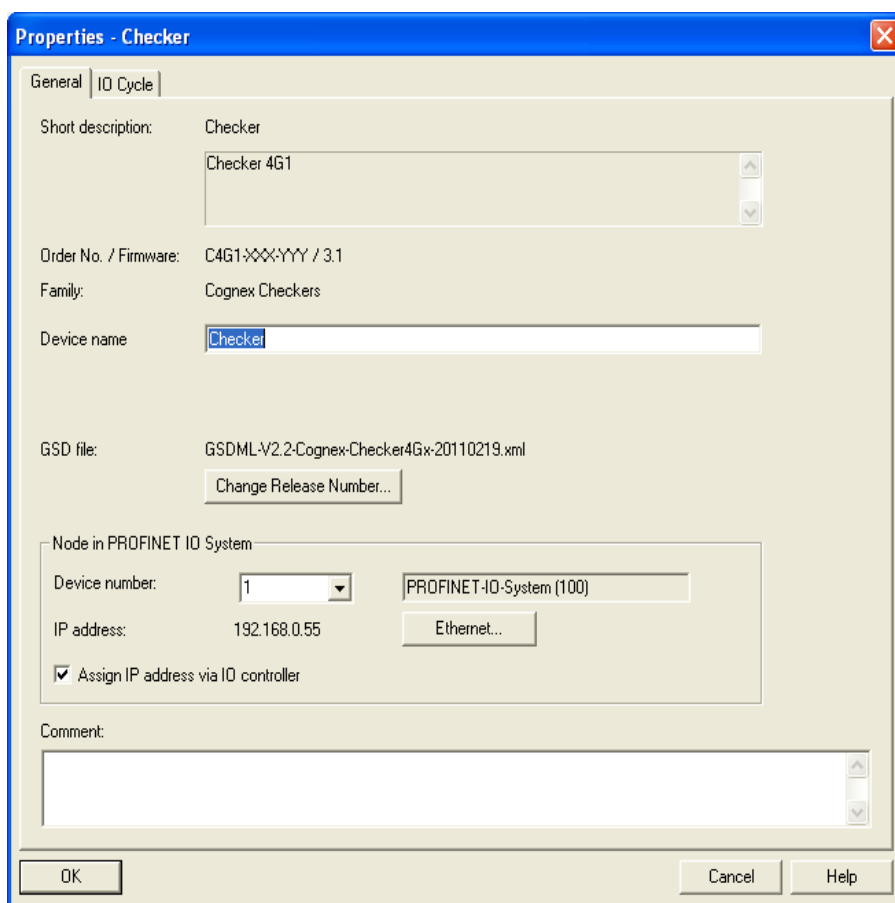
PDEV オブジェクトは現在はサポートされておらず、将来の参考として予約されています。



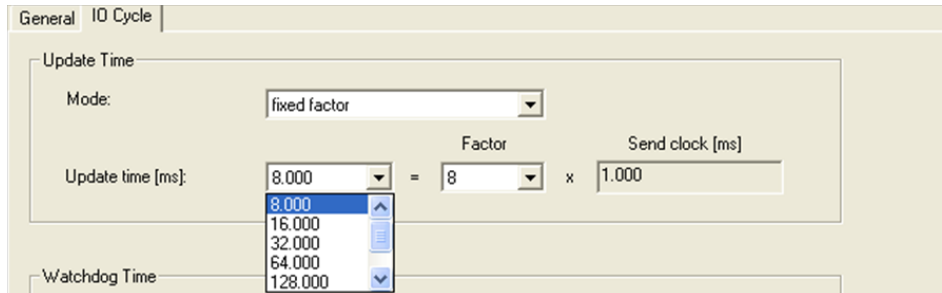
HW Config ツールは、自動的に Checker I/O モジュールをメモリ空間にマッピングします。

11. Checker アイコンを右クリックして、[オブジェクトのプロパティ...] を選択します。

12. センサー名を指定します。これは、実際の Checker センサーの名前と一致している必要があります。また、名前は一意であり、DNS 命名規則に適合している必要があります。詳細は、SIMATIC ソフトウェアのヘルプを参照してください。
13. Checker センサーが独自の IP を使用するように設定されている場合は、[**Assign IP address via IO controller**] ボックスを選択解除します。それ以外の場合、PLC によって IP アドレスを割り当てたければ、[Ethernet] ボタンを選択して、該当するアドレスを設定します。

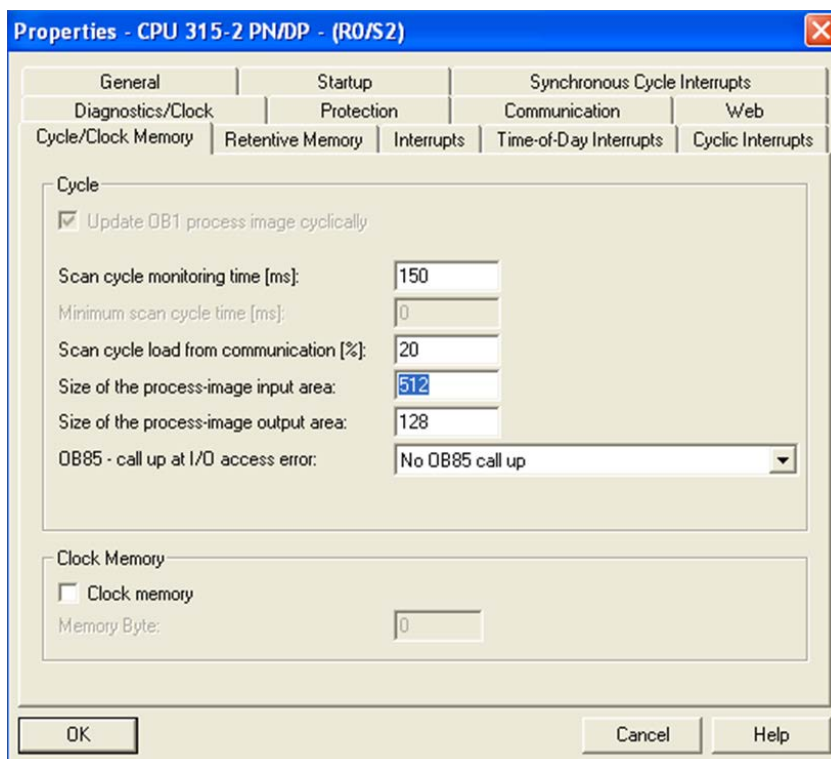


14. [IO サイクル] タブで、アプリケーションに適切なサイクル更新レートを選択します。



15. デフォルトでは、SIMATIC ソフトウェアは User Data および Result Data のモジュールをオフセット 256 ヘマッピングします。これは、デフォルトのプロセスイメージ領域のサイズである 128 を超えています。つまり、デフォルトでは、これらのモジュールのデータは、BLKMOV などの一部の SFC からはアクセスできないことになります。解決策として、プロセスイメージ領域のより低いオフセットへモジュールをマッピングし直すか、プロセスイメージ領域を拡張してこれらのモジュールを含めるようにしてください。

プロセスイメージ領域の拡張を選んだ場合は、モジュールのサイズに加えて、デフォルトで 256 のオフセットとなるように十分大きいサイズにしてください。



注

プロセスイメージを拡張すると、PLC スキャンのサイクル時間のパフォーマンスに影響することがあります。スキャン時間が重要ではない場合は、許容される最小のモジュールサイズを使用して、手動でプロセスイメージまで下げてください。

モジュール

Checker での PROFINET の実装は、以下の I/O モジュールによって構成されます：

1. デバイス制御モジュール
2. デバイス・ステータス・モジュール
3. 取得制御モジュール
4. 取得ステータス・モジュール
5. 結果制御モジュール

Checker® 4G 通信およびプログラムガイド

9/3/2014 | バージョン 4.1

6. 結果ステータス・モジュール
7. 入力
8. 出力
9. センサー・メーター・モジュール

HW Config - [SIMATIC 300 Station (Configuration) -- Simultaneous]

Station Edit Insert PLC View Options Window Help

Find: checker Profile: Standard

Ethernet(1): PROFINET-I/O-System (100)

(1) Checker4g7c

Slot	Module	Order number	I address	Q address	Diagnostic address	Comment
0	Checker4g7c	C4G7344X1111		1...2	2042	
1	DeviceControlModule			1...2		
2	DeviceStatusModule		1...2			
3	AcquisitionControlModule			3		
4	AcquisitionStatusModule		3...5			
5	ResultControlModule			4		
6	ResultStatusModule		6...8			
7	InputModule			5...7		
8	OutputModule		17...19			
9	SensorMeterModule1		256...271			
10	SensorMeterModule5		272...351			
11						

Insertion possible

Cognex
SensorMeterInfo5
GSDML-V2.2.Cognex-Checker4Gx-20111014.xml

デバイス制御モジュール

Checker デバイスを制御します。このモジュールは、PLC から Checker センサーへ送信されるデータで構成されます。

スロット数： 1

合計モジュールサイズ： 2 バイト

ビット	名前	説明
0	Set Offline (オフライン 設定)	このビットを設定すると、Checker はオフライン・モードになります。これは、Checker がすべての活動を中断して、一切の制御コマンドに応答しなくなることを意味します。
1	Job Change (ジョブの 変更)	このビットは、ジョブを変更するために使用されます。PLC によってこのビットを 1 へ変更する前 (あるいは同時) に、ジョブ番号が入力されていることを確認してください。Checker から、Job Load Failed ビットまたは Job Load Complete ビットを受信した場合は、この Job Change (ジョブの変更) ビットを PLC で 0 に戻してください。0 に戻すと、Checker が Job Load Failed または Job Load Complete のビットをクリアします。 注:いかなる産業用プロトコルでも、Checker 自体はジョブの変更のためにオフラインにする必要はありません。
2	Retrain (再登録)	このビットは、センサーを再登録するために使用します。
3	Lights Off (照明オフ)	Lights Off ビットを 1 に設定すると、ジョブの設定に関わらず、外部照明がオフになります。

4	Volatile (変動)	Volatile ビットを 0 に設定すると、ジョブに対して変更されないスタティックな結果データを選択します。これを 1 に設定すると、製品によって異なる変動データが選択されます。
8 – 15	Job Number (ジョブ番号)	Job Change (ジョブの変更) を参照

デバイス・ステータス・モジュール

Checker デバイスのステータスを表示します。このモジュールは、Checker デバイスから PLC へ送信されるデータで構成されます。

スロット数： 2

合計モジュールサイズ： 2 バイト

ビット	名前	説明
0	Online (オンライン)	Checker がオンラインであれば、この値は 1 であり、Checker はコマンドに応答します。それ以外は、値は 0 です。Set Offline (オフラインの設定) も参照してください。
1	Offline Reason 1 (オフライン の理由 1)	これが 1 なら、センサーはオフラインです。これが 0 なら、センサーはオンラインです。セットアップモード。
2	Offline Reason 2 (オフライン の理由 1)	未使用です。

ビット	名前	説明
3	Offline Reason 3 (オフラインの理由 1)	未使用です。
4	General Fault (一般的なフ ォールト)	常に 0 です。
5	Job Load Complete (ジョブのロ ード完了)	ジョブの変更が成功すると、このビットは 1 に変わります。 このビットは Checker によって、自動的にクリアされ ません。PLC で Job Change ビットを 0 に戻すと、Checker は Job Load Complete ビットをクリアします。
6	Job Load Failed (ジョブのロ ード失敗)	ジョブの変更が成功しなければ、このビットは 1 に変 わります。このビットは Checker によって、自動的にク リアされません。PLC で Job Change ビットを 0 に戻すと、 Checker は Job Load Failed ビットをクリアします。
7	Retrain Completed (再登録完了)	これが 1 であれば、センサーの再登録操作は正常に完 了です。
8	Retrain Failed (再登録失敗)	これが 1 であれば、センサーの再登録操作は失敗です。

ビット	名前	説明
9	Observer (オブザーバ ー)	これが 1 であれば、PROFINET コントローラは Checker のステータスを変更することができず、その状態を観察することができるだけです。Observer ビットは PC アプリケーションでのみ設定できます ([Checker] -> [Checker の設定] -> [プロトコル設定])。詳細情報は、付録 A - プロトコル設定、オブザーバ・ビットの変更を参照してください。

取得制御モジュール

画像の取得を制御します。このモジュールは、PLC から Checker デバイスへ送信されるデータで構成されます。

スロット数： 3

合計モジュールサイズ： 1 バイト

ビット	名前	説明
0	Trigger (トリガ)	このビットを設定すると、以下の条件が満たされた場合に、取得がトリガされます： • Trigger Enable (トリガ有効) が設定されている • 現在、取得が行われていない • デバイスは、トリガの準備ができている
1	Trigger Enable (トリガ有効)	このビットを設定すると、PROFINET 経由のトリガが有効になります。このビットをクリアすると、トリガは無効になります。
2 - 7	予約済み	将来の使用のため予約済み

取得ステータス・モジュール

実際の取得ステータスを示します。このモジュールは、Checker デバイスから PLC へ送信されるデータで構成されます。

スロット数： 4

合計モジュールサイズ： 3 バイト

ビット	名前	説明
0	Trigger Ready (トリガ・レディ)	デバイスが新しいトリガを受信できる準備ができていることを示します。Trigger Enable が設定されていると、このビットが true になり、デバイスが新しいトリガを受信できる準備ができています。
1	Trigger Ack (トリガ確認)	Checker が新しいトリガを受信し、トリガ処理が始まっていることを示します。Trigger ビットが true であれば、このビットも true のままです (つまり、Trigger ビットとのインターロック)。
2	Acquiring (取得)	Checker が、現在取得を実行していることを示します。
3	Missed Acq (確認失敗)	Checker が正常に取得をトリガできなかったことを示します。次に正常に取得が行われると、このビットはクリアされます。
4-7	予約済み	将来の使用のため予約済み
8-23	Acquisition ID	次に発行されるトリガの ID 値 (16 ビット整数)。発行されたトリガと、後で受信した対応する結果データの一致で使用。この同じ値が、結果データの Result ID で返されます。

結果制御モジュール

結果データの処理を制御します。このモジュールは、PLC から Checker センサーへ送信されるデータで構成されます。

スロット数： 5

合計モジュールサイズ： 1 バイト

ビット	名前	説明
0	Results Buffer Enable (結果バッファ有効)	結果データのキューを有効にします。有効であれば、実際の結果データは、確認済みのままになります (新しい結果が届いても)。新しい結果はキューに入ります。実際の結果が確認されるたびに、結果の次のセットがキューから取り出されます (Result Data モジュールで使用可能)。Checker は、Results Available ビットをクリアして、確認に応答します。Results Ack ビットがクリアされると、次のセットの読み取り結果が登録され、Results Available が true に設定されます。結果のバッファリングが有効でなければ、単に新しく受信した読み取り結果で、Result Data モジュールのコンテンツが上書きされます。
1	Results Ack (結果確認)	このビットは、最後の結果データを PLC が正常に読み込んだことを示します。true に設定すると、Result Available ビットがクリアされます。使用できるのは、結果のバッファリングが有効である場合だけです。
2-7	予約済み	将来の使用のため予約済み

結果ステータス・モジュール

取得および結果のステータスを表示します。このモジュールは、Checker デバイスから PLC へ送信されるデータで構成されます。

スロット数： 6

合計モジュールサイズ： 3 バイト

ビット	名前	説明
0	Part Detect (製品検出)	製品検出センサーが正常なら、この値は 1 のままです。成功しなかった場合、値は 0 に設定されます。Results Available が 1 であれば、有効です。
1	Inspecting (チェック)	Checker がチェック中であれば、この値は 1 です。それ以外、この値は 0 です。
2	Inspection Complete Toggle (チェック完了トグル)	チェックが完了すると、値が変化します。
3	Results Buffer Overrun (結果バッファ・オーバーラン)	このビットが有効であるのは、結果バッファリング (Buffer Result Enable = 1) の場合だけです。最後のチェック結果に対するバッファがない場合、値が 1 に変わります。
4	Results Available (結果使用可能)	チェックが完了すると、値が 1 に変化します。Inspection Complete Toggle (チェック完了トグル) も参照
5	Any Fail (いずれか不合格)	実際のジョブで定義されているセンサーのうち、少なくとも 1 つに関するチェックが成功しなかった場合、値は 1 になります。これの反対は、All Pass (すべて合格) です。すべてのセンサーのチェックが合格すると、値は

Checker® 4G 通信およびプログラムガイド

9/3/2014 | バージョン 4.1

ビット	名前	説明
		0になります。 このビットが有効であるのは、Result Available が 1 である場合です。
6	All Pass (すべて合格)	実際のジョブで定義されているセンサーのすべてについてチェックが成功した場合、値は 1 になります。 いずれかのセンサーで検チェックが成功しなかった場合、値は 0 になります。この反対は、Any Fail (いずれか不合格) です。 このビットが有効であるのは、Result Available が 1 である場合です。
7 - 22	Result ID Register (結果 ID レジスター)	このビットは、Acquisition ID とペアです。

入力モジュール

スロット数： 7

合計モジュールサイズ： 3 バイト

ビット	名前	説明
0 - 23	入力 0 ~ 23	これらのビットは、Checker GUI のラダーロジックとして使用されます。

出力モジュール

スロット数： 8

合計モジュールサイズ： 3 バイト

ビット	名前	説明
0-23	出力 0 ~ 23	これらのビットの機能は、Checker GUI で変化します。 以下の仮想出力のいずれかになります： • Part Detect (製品検出) • All Pass (すべて合格) • All Fail (いずれか不合格) • Coil (コイル) • External Trigger (外部トリガ) • External Retrain Pass (外部再登録の成功) • External Retrain Fail (外部再登録不合格) • Job Change Pass (ジョブ変更の成功) • Job Change Fail (ジョブ変更不合格) これらのビットが有効であるのは、Result Available が 1 である場合だけです。

センサー・メーター・モジュール

スロット数： 9, 10, 11

合計モジュールサイズ： 256 バイト

Checker デバイスを HW Config に追加しても、センサー・メーター・モジュールは自動入力されません。手動で追加する必要があります。PROFINET パケットのサイズを最小にするために、自動入力されません (PLC メモリの使用量とネットワークのトラフィックを最適化)。ジョブで最小のモジュール数を選択することを強く推奨いたします。

センサー・メーターの情報については、*スタティック・データおよび変動データの概要* セクションを参照してください。

注

PROFINET でセンサーメーター値が使用できる順番は、センサー・メーター・テーブルで設定したセンサーの順番で決まります。モジュールの場所は厳密です：

- データ・カテゴリー 1 からのモジュールは、スロット 9 にだけ配置できます。
- データ・カテゴリー 2 からのモジュールは、スロット 10 にだけ配置できます。
- データ・カテゴリー 3 からのモジュールは、スロット 11 にだけ配置できます。

操作

取得シーケンス

Checker は、アセンブリ・オブジェクトを通じて、画像取得を明示的にトリガすることができます。

起動時には、TriggerEnable 属性は false です。トリガを有効にするには、true に設定する必要があります。デバイスがトリガを受信できるようになると、AcqStatusRegister の Trigger Ready ビットが true に設定されます。

Checker® 4G 通信およびプログラムガイド

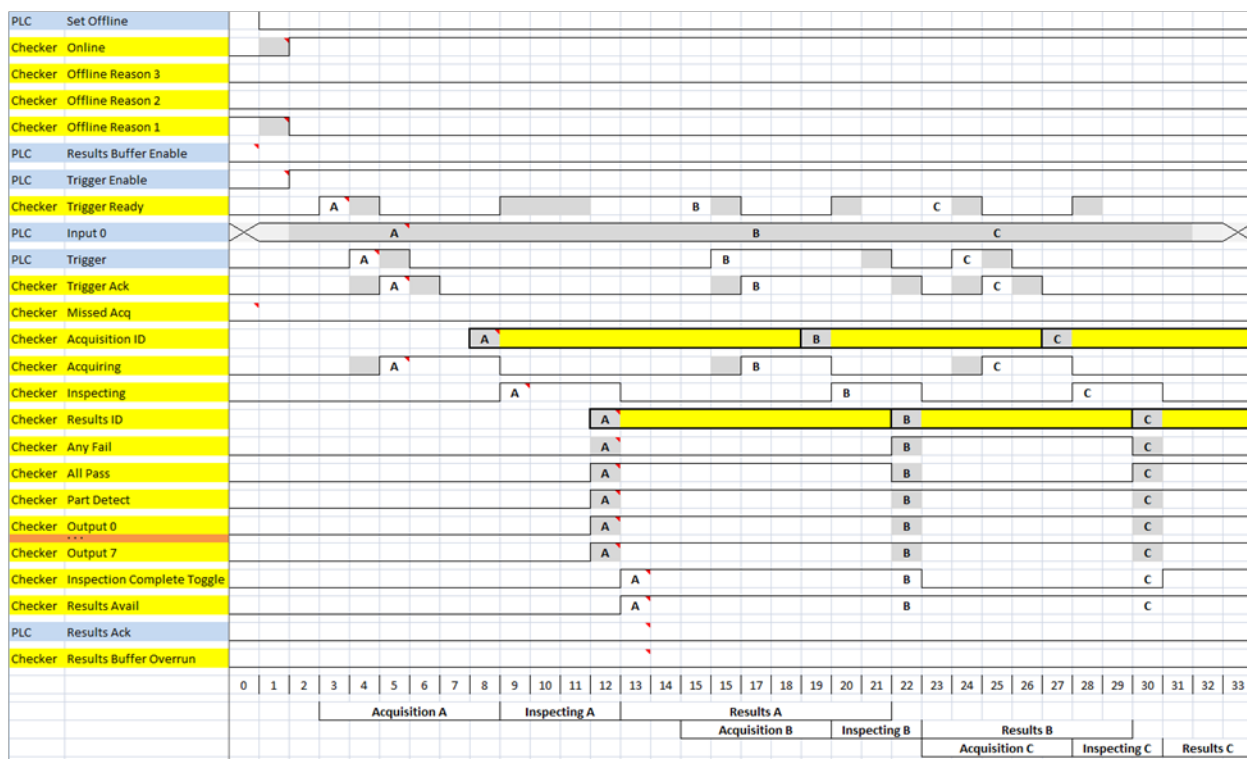
9/3/2014 | バージョン 4.1

TriggerEnable 属性が true で、Trigger Ready ビットが true であると、Trigger 属性が 0 から 1 に変わったことを Checker オブジェクトが確認するたびに、画像取得が始動されます。アセンブリ・オブジェクトでこれを設定すると、AcqStatusRegister の Trigger Ack ビットで同じ値が確認されるまで、属性は新しい状態を保持します（これは、その変更が Checker オブジェクトで確認されることを保証するハンドシェークです）。

取得中に、AcqStatusRegister で Trigger Ready ビットがクリアされ、Acquiring ビットが true に設定されます。取得が完了すると、Acquiring ビットがクリアされ、取得システムの新しい画像取得の準備ができると Trigger Ready ビットが true に設定されます。

トリガのメカニズムを強制的にリセットするには、AcqStatusRegister が 0 に設定されなくなるまで、TriggerEnable 属性を false に設定します。その後、TriggerEnable を true に設定して、取得を再度有効にすることができます。

トリガが使用できるのは、Checker が外部トリガモードである時だけです。結果のバッファリングが無効である場合の一般的な取得シーケンスの概要は、下の図を参照してください。



注

「いずれか不合格」、「すべて合格」、「製品検出」、「出力 0 - 7 ライン」の状態は、チェックの結果によります。示されている状態は、あくまでもサンプルです。

Results Buffer Enable=false であると、「Results Avail Ack」および「Results Buffer Overrun」が無効になります。最初に結果が設定された後、「Results Avail」は true のままになります。Inspection Complete Toggle を使用すると、IO バッファから PLC メモリへの結果をゲートすることができます。

Trigger Enable = true かつ Trigger Ready == false で、トリガの立ち上がりエッジが発生すると、Missed Ack は true になります。Missed Ack は、取得が成功するまで true のままです (Trigger Ready == true で、トリガの立ち上がりエッジが発生している場合)。

Set Offline が true であるか、Checker アプリケーションがセットアップ・モードになると、Online が false になり、Offline Reason 1 が true になります。

Trigger Enable が true になるのは、Online が最初に true になる時だけです。Trigger Ready に必要な信号のいずれかが true になると、Trigger Enable = true になります。

Trigger Ready が true になるのは、Online = true で、Trigger Enable = true で、Acquiring = false で、取得バッファが使用可能である場合です。

Trigger Ready が true になると、Trigger が true になります。Trigger Ack が true になると、Trigger は false になるはずですが、Trigger の立ち上がりエッジが true になると、Trigger Ready が false になり、その結果として Acquiring が true になります。

シーケンス中、入力 0 ラインは任意の状態になります。Acquiring 信号の立ち上がりエッジの値が、使用される値です。

Trigger が true になると、Trigger Ack が true になります。Trigger Ack が false になると、Trigger が false になります。

Trigger Ready == true であると、Trigger 信号の立ち上がりエッジ後は、Acquiring は true になります。取得とは、カメラが写真を撮り、画像が取得バッファに移動することを意味します。取得の立下りエッジは、Acquisition ID のゲート信号であり、これによってチェックが true になります。

[トリガ] 取得信号が false になると、Acquisition ID は有効のままです。この数は、[チェック] Results ID の同じ数を使って、取得した画像とチェック出力を一致させるのに使われます。

次のプロセスがパイプラインでつながっているため、取得が false になると、すぐにチェックが true になります。チェックの立下りエッジは、Results ID およびすべての結果データのゲート信号です。Inspection Complete Toggle および Results Avail は、チェックの立下りエッジで有効になります。

Results Available、Results ID、「いずれか不合格」、「すべて合格」、「製品検出」、出力 0 – 7の立下りエッジ前は、有効のままです。チェックの立下りエッジでは、Inspection Complete Toggle の状態が変化し、Results Available が true になります。Results ID の値は、一致するカメラ画像の [トリガ] Acquisition ID の値と同じになります。

Inspection Complete Toggle は、チェックの立下りエッジと同時に変化します。Results ID は、Results Available が true になるたび (つまり、Inspection Complete Toggle の状態が変化) に増えます。

[チェック] Results Available はチェックの立下りエッジにすぐに従い、true になるのは、結果データが安定して有効である間だけです。

注

最初の結果の後、Results Buffer Enable = false となると、Results Available 信号は必ず true です。Inspection Complete Toggle を有効信号として使用し、IO 結果バッファからの新しい結果データを PLC メモリにコピーできます。

Results Buffer Enable = false であれば、Results Ack は使用されません。

Results Buffer Enable = false であれば、Results Buffer Overrun は使用されません。

チェック/結果シーケンス

画像を取得した後、チェックを行います。チェック中は、結果ステータス・モジュールの Inspection ビットがセットされます。チェックが完了すると、Inspecting ビットがクリアされ、Inspection Completed がトグルされます。

Results Buffer Enable ビットによって、センサーによってチェック結果がどのように処理されるかが決まります。Results Buffer Enable ビットが false に設定されると、Checker® 4G 通信およびプログラムガイド

チェック結果はただちに結果モジュールに配置され、Results Available が true に設定されます。

Results Buffer Enable ビットを true に設定すると、新しい結果はキューに入ります。クライアントが Results Ack ビットを true に設定して確認するまで、以前のチェック結果は結果モジュールに残ります。Results Available ビットをクリアした後、クライアントは Results Ack ビットを false に戻して、次にキューにある結果が結果モジュールに入れるようにする必要があります。これは、Checker クライアント (PLC) によって結果が受信されることを保証するために必要なハンドシェークです。

チェックステータスレジスタの挙動

ビット	ビット名	バッファリング 無効時の結果	バッファリング有効時の結果
1	Inspecting (チェック)	画像のチェック 時にセット	画像のチェック時にセット
2	Inspection Complete (チェック 完了)	画像のチェック 完了時にトグル	画像のチェック完了時にトグル
3	Results Buffer Overflow (結果バッファ・オーバーフロー)	ゼロにセットされたまま	クライアントが前の結果を確認できなかったためにチェック結果をキューに入れることができなかった場合にセットされます。チェック結果が正常にキューに入ればクリアされます。

ビット	ビット名	バッファリング 無効時の結果	バッファリング有効時の結果
4	Results Available (結果使用 可能)	最初のチェック の後 true になり、その後 も true のままです。	新しい結果が、結果モジュールに配置され ると設定されます。Results Ack によっ て、バッファにあるすべての結果が確認さ れるまで(1 つづつ)、設定されたままに なります。

結果のバッファリング

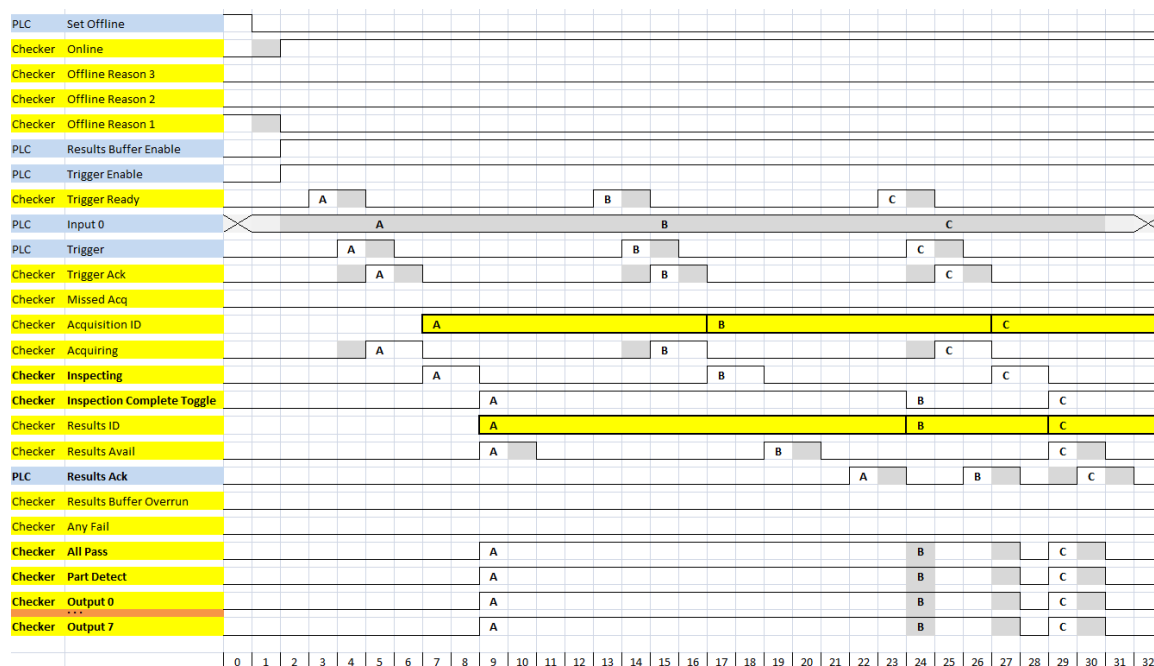
チェック結果に対してキューを有効にするオプションがあります。有効にすると、クライアント (PLC) に読み取るための時間ができるまで、有限の数のチェック結果データがキューに入ります。これは、短い期間の間、クライアント (PLC) がスローダウンする場合、あるいは読み取り操作の急激な増加がある場合に、データの流を円滑にするのに有益です。

また、結果バッファリングが有効であれば、デバイスは取得やチェックの操作の重複を許可します。アプリケーションによっては、これを使用して、全体としてより高速なトリガレートを実現することも可能です。詳細な説明は、前述の取得シーケンスの説明を参照してください。

一般に、結果が送信されるより速く書き込み (新しい結果) が発生する場合、バッファリングの有無による主な差異はどの結果が無視されるかです。バッファリングが有効でなければ、最も新しい結果が保持され、以前の結果 (PLC が十分速く読み取れなかった) は失われます。実質的には、単に最も新しい結果が以前の結果を上書きすることになります。バッファリングが有効であれば (および、キューが満杯)、結果キューに十分な空きができるまで、最も新しい結果が無視されます。

下の図に示されているように、A Acq に対応する最初の結果が、結果バッファおよび出力ラインに移動します。これに続いて、B の結果が結果バッファへ移動しますが、

出力ラインには引き続き A の結果が残ります。PLC が A の結果を読み取ると、バッファおよび出力ラインから消えます。



Siemens の例 (S7-300)

このセクションでは、Checker を Siemens S7-300 PLC と共に使用するいくつかの例を説明します。S7-300 および SIMATIC プログラミング・ソフトウェアを知っていると想定しています。対応するサンプルのプログラムは、次のリンクから入手できます：

- PROFINET

のサンプル：<http://www.cognex.com/support/downloads/File.aspx?d=2470>

シンボル・テーブル

必須ではありませんが、Checker I/O モジュール要素のシンボルを定義しておくと、非常に便利です。こうすることで、コードが非常に読みやすくなり、間違いも減ります。このサンプル・テーブルは、Checker センサーの一般的なインスタンスに対して定義されたシンボルを示しています。Checker I/O モジュールは、お客様が使用している

Checker® 4G 通信およびプログラムガイド

9/3/2014 | バージョン 4.1

ページ | 78

プロジェクトとは別のアドレスである可能性もあることに注意してください。必ず、I/O モジュールに固有のオフセットに基づいて、シンボルの定義を調整してください。

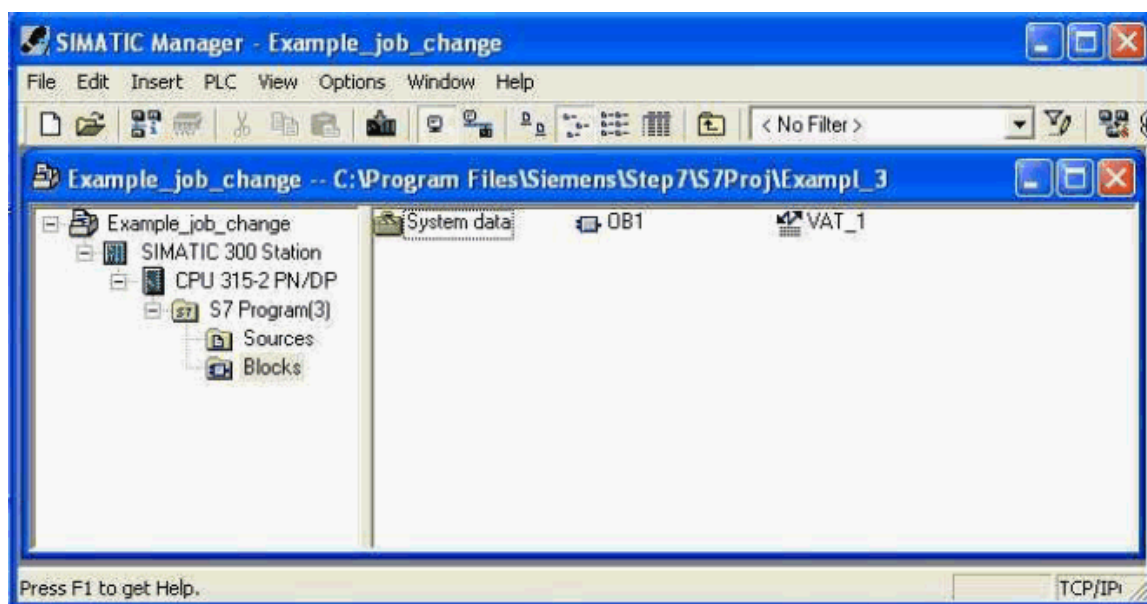
名前	データタイプ	アドレス	コメント
TEMP		0.0	
OB1_EV_CLASS	バイト	0.0	ビット 0-3 = 1 (受信イベント)、 ビット 4-7 = 1 (イベント・クラス 1)
OB1_SCAN_1	バイト	1.0	1 (コールドスタート スキャン 1 / OB 1), 3 (スキャン 2-n / OB 1)
OB1_PRIORITY	バイト	2.0	OB 実行の優先度
OB1_OB_NUMBR	バイト	3.0	1 (組織ブロック 1、OB1)
OB_RESERVED_1	バイト	4.0	システム予約済み
OB_RESERVED_2	バイト	5.0	システム予約済み
OB1_PREV_CYCLE	整数	6.0	前の OB1 スキャンのサイクル時間 (ミリ秒)
OB1_MIN_CYCLE	整数	8.0	OB1 の最小サイクル時間 (ミリ秒)
OB1_MAX_CYCLE	整数	10.0	OB1 の最大サイクル時間 (ミリ秒)
OB1_DATE_TIME	Date_And_Time	12.0	OB1 開始の日時
L_InitVars	ブール型	20.0	
L_AfterFB4	ブール型	20.1	
L_MultiTriggerPrev	ブール型	20.2	

名前	データタイプ	アドレス	コメント
L_T0Prev	ブール型	20.3	
L_Sink	ブール型	20.4	

変数テーブル

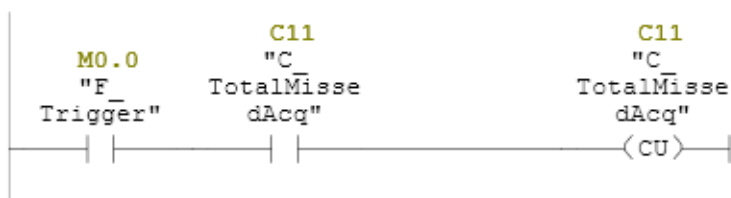
それぞれのサンプル・プログラムには、少なくとも1つの変数テーブルが含まれていて、プログラム実行中にサンプル・プログラムに存在する変数の値を監視、変更するのに使用できます。

たとえば、Example_job_change プログラムの変数テーブルを開くには、プログラムのブロックを開いて、VAT_1 (変数テーブルのファイル) をダブルクリックします。ブロックについては、下のスクリーンショットを参照してください。



これと類似の編集可能なテーブルがポップアップ表示されます：

VAT_1 -- Example_Trigger_non_buf\SIMATIC 300 Station\CPU 315-2 ...					
	Address	Symbol	Display format	Status value	Modify value
1	//start automated triggering with the PLC (1 - Triggering, 0 - Not triggering)				
2	M 0.0	"F_Trigger"	BOOL		
3	//Handling of the triggers on the PLC side				
4	Q 3.0	"Q_Trigger"	BOOL		
5	Q 3.1	"Q_TriggerEnable"	BOOL		
6	//Handling other stuff on the PLC side				
7	Q 1.0	"Q_SetOffline"	BOOL		
8	Q 4.0	"Q_BufferEnable"	BOOL		
9	Q 4.1	"Q_ResultsAvailAck"	BOOL		
10	Q 5.0	"Q_Input0"	BOOL		
11	Q 1.1	"Q_JobChange"	BOOL		
12	QB 2	"Q_JobNumber"	HEX		
13	//Basic inputs about the checker state (Sent by the Checker)				
14	I 1.0	"I_Online"	BOOL		
15	I 1.1	"I_OfflineReason1"	BOOL		
16	I 1.2	"I_OfflineReason2"	BOOL		
17	I 1.3	"I_OfflineReason3"	BOOL		
18	I 1.4	"I_GeneralFault"	BOOL		
19	//Trigger related Inputs (Sent by the Checker)				
20	I 2.0	"I_TriggerReady"	BOOL		
21	I 2.1	"I_TriggerAck"	BOOL		
22	// Other Inputs from the Checker				
23	I 2.2	"I_Acquiring"	BOOL		
24	I 2.3	"I_MissedAquisition"	BOOL		
25	MV 3	"I_AcquisitionID"	HEX		
26	I 5.0	"I_PartDetect"	BOOL		
27	I 5.1	"I_Inspecting"	BOOL		
28	I 5.2	"I_InspCompleteToggle"	BOOL		
29	I 5.3	"I_ResultsBufferOverrun"	BOOL		
30	I 5.4	"I_ResultsAvailable"	BOOL		
31	I 5.5	"I_AnyFail"	BOOL		
32	I 5.6	"I_AllPass"	BOOL		
33	MV 6	"I_ResultID"	HEX		
34	//Virtual Outputs sent by the Checker (Inputs in the PLC side!)				
35	I 8.0	"I_Output0"	BOOL		
36	I 8.1	"I_Output1"	BOOL		
37	I 8.2	"I_Output2"	BOOL		
38	I 8.3	"I_Output3"	BOOL		
39	I 8.4	"I_Output4"	BOOL		
40	I 8.5	"I_Output5"	BOOL		



注

PROFINET プログラムを実行するには、PROFINET が有効で、Checker が実行モードである必要があります。「バッファリングなしのトリガ (Triggering without Buffering)」や「バッファリングありのトリガ (Triggering with Buffering)」のサンプル・プログラムを実行する場合、外部トリガモードが有効であることを確認してください。

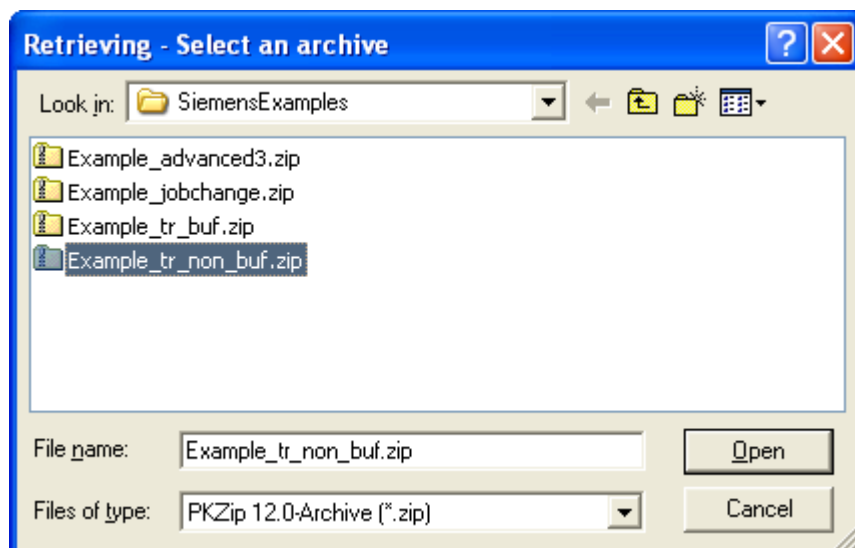
バッファリングなしのトリガ

以下のステップを実行して、“Example_tr_non_buf” サンプル・プログラムをインストールします：

1. SIMATIC Manager ソフトウェアを起動します。
2. 開いているアプリケーションがあれば、閉じます。
3. メインメニューで、[ファイル] → [抽出…] と選択します。
4. 参照して、PC でサンプルファイルがある場所を探します。



5. Siemens フォルダを探し、“Example_Tr_Non_Buf.zip” ファイルを選択します。



6. プロジェクトを PC 上に保存するディレクトリを選択します。



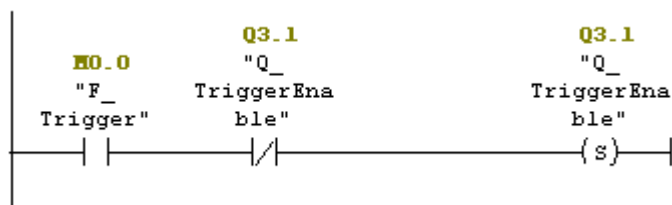
Siemens ソフトウェアが、サンプルのアーカイブを抽出し、使用可能にします。
 この場合、言語の設定が原因で、Siemens S7 PLC プロジェクトを開く際に問題が発生しますので、以下のステップを実行してください：

1. C:\Program Files\Siemens\Step7\S7Proj フォルダ（デフォルト）へ移動するか、プロジェクトを保持している場所へ移動します。
2. グローバル・フォルダを開きます。
3. 自分の地域の設定を含んでいる言語ファイルを開きます。
4. コンテンツをコピーします。
5. 受け取ったプロジェクトへ移動し、言語ファイルを開いて、コンテンツを上書きします。
6. プロジェクトを開けることを確認してください。

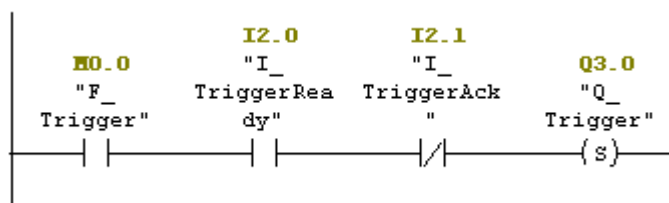
開いた後に、サンプル・プログラムを実行するには、PLC へダウンロードしてから、F_Trigger の値を 1 に設定します。

基本に戻ると、バッファを使用しない外部トリガの処理は、以下のようになります：

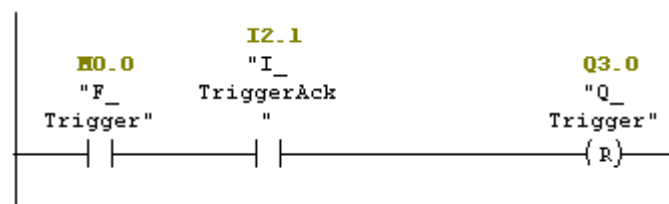
1. センサーで、外部トリガを有効にします。



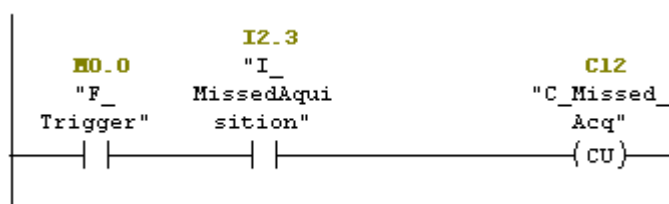
2. トリガ信号を設定して、取得を開始します。



3. トリガ信号が確認されたら、すぐにトリガ信号をクリアします。



4. 取得失敗をチェックします (失敗はないはずです)。



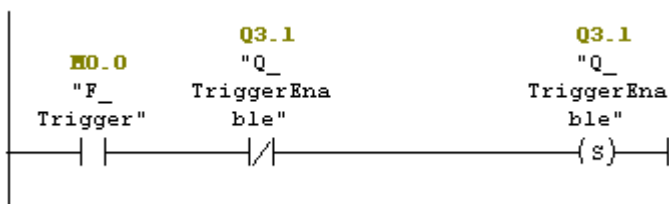
バッファリングありのトリガ

バッファリングを行う主な理由は、PLC が結果を読み取る速度より、Checker のトリガが速い場合に、一部の結果を Checker に保存しておくことができるからです。この場合、Acquisition ID と Result ID が同期していない可能性があります。

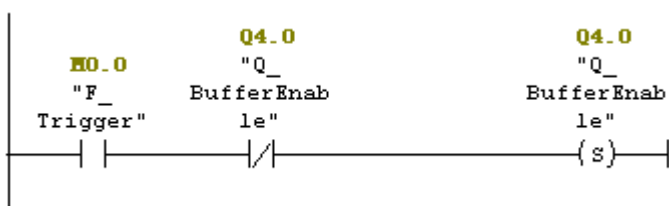
[バッファリングなしのトリガ](#)セクションでの説明と同様に、“Example_tr_buf.zip” ファイルを選択することで、“Example_Tr_Buf” サンプル・プログラムをインストールして、実行します。

基本に戻ると、バッファを使用する外部トリガの処理は、以下のようになります：

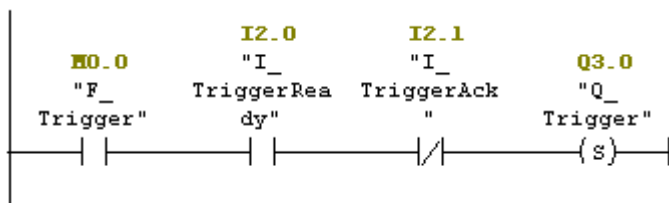
1. センサーで、外部トリガを有効にします。



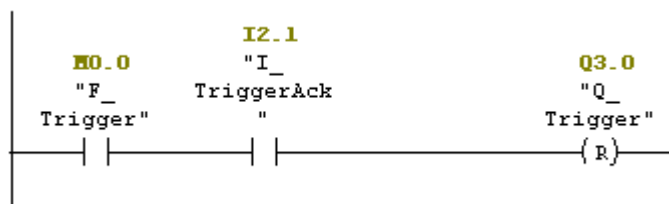
2. バッファリングをオンにします。



3. トリガ信号を設定して、取得を開始します。



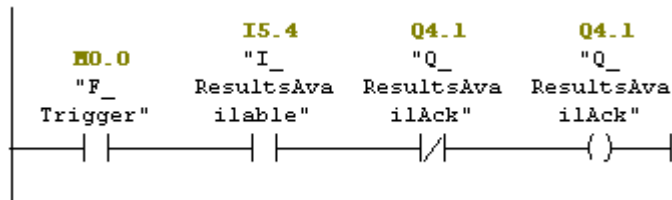
4. トリガ信号が確認されたら、すぐにトリガ信号をクリアします。



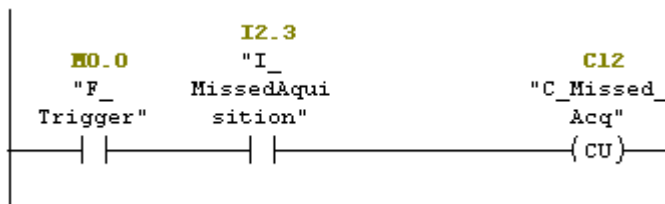
5. Results Available (Checker が送信) が高であれば、

一部の結果がまだ PLC によって読み込まれていないことを意味します。
ResultAvailableAck を設定して、読み込みます。

ResultAvailable が低であれば、resultAvailableAck を低に設定して、
次に利用可能になった時に PLC が読み取り準備可能となるようにします。



6. 取得失敗をチェックします (失敗はないはずです)。



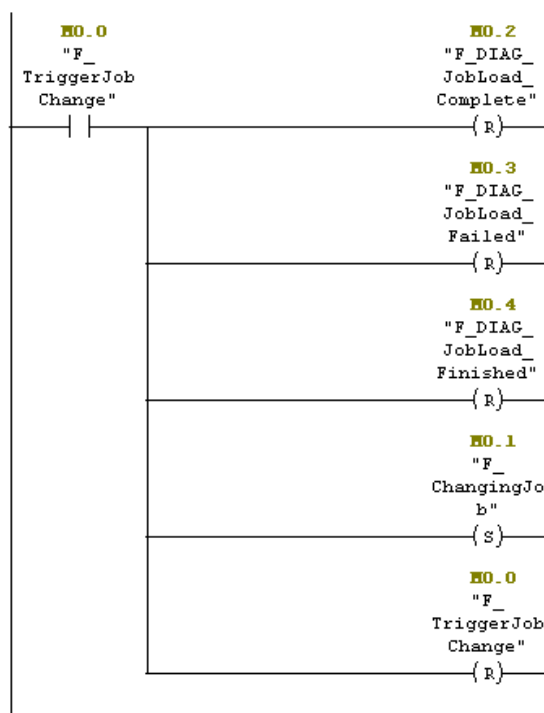
ジョブの変更の例

このプログラムは、ジョブ制御テーブルから Checker へ、ジョブをロードします。
(これは、PC アプリケーションの Checker 設定メニューにあります。)

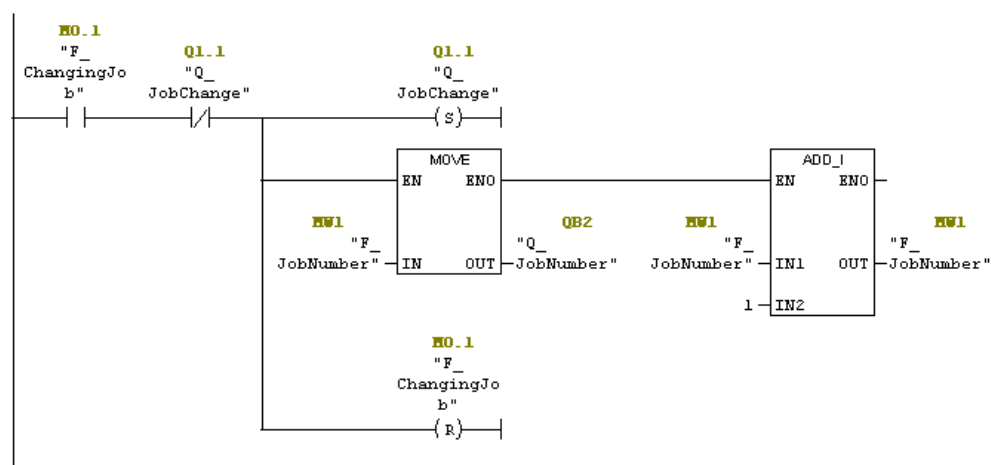
[バッファリングなしのトリガ](#)セクションでの説明と同様に、“Example_Jobchange.zip”
ファイルを選択することで、“Example_job_change” サンプル・プログラムをインストールして、実行します。

基本に戻ると、ジョブの変更の処理は、以下のようになります：

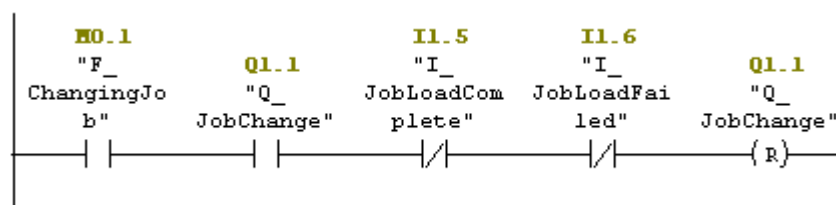
1. M0.0 を手動でトリガして、JobChange を開始します。JobNumber は 0 から増えます。



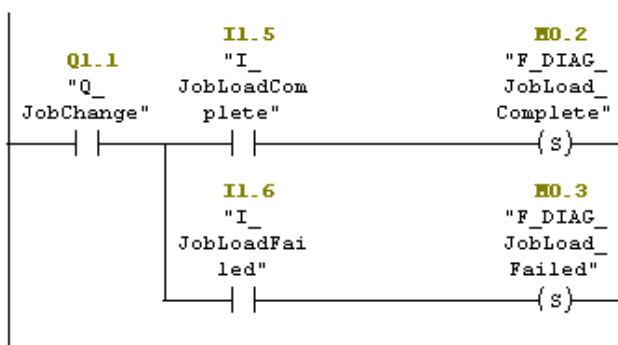
2. 最新の F_JobNumber を Checker にロードすると、次のジョブの変更プロセスでこの値が 1 だけ増えます。



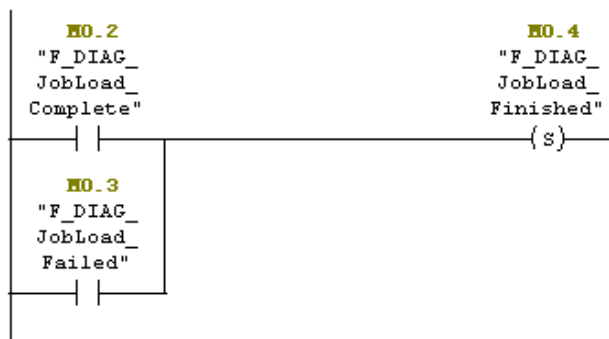
3. 安全なネットワーク。Q_JobChange が高のままであるが、アクティブなジョブの変更がない場合は、0に戻します。



4. ジョブの変更について、Checker から PLC が情報を受信したら、DIAG ビットを正しく設定します。この背後にある理由は、Checker と PLC が対応するビットを非常に素早く戻すので、目で結果を確認することができないからです。DIAG ビットは、次のジョブの変更が始まるまで、この値を保持します。



5. ジョブの変更について、Checker から PLC が情報を受信したら、DIAG ビットを正しく設定します。JobLoad_Complete または JobLoad_Failed が高であれば、JobChange も完了です。

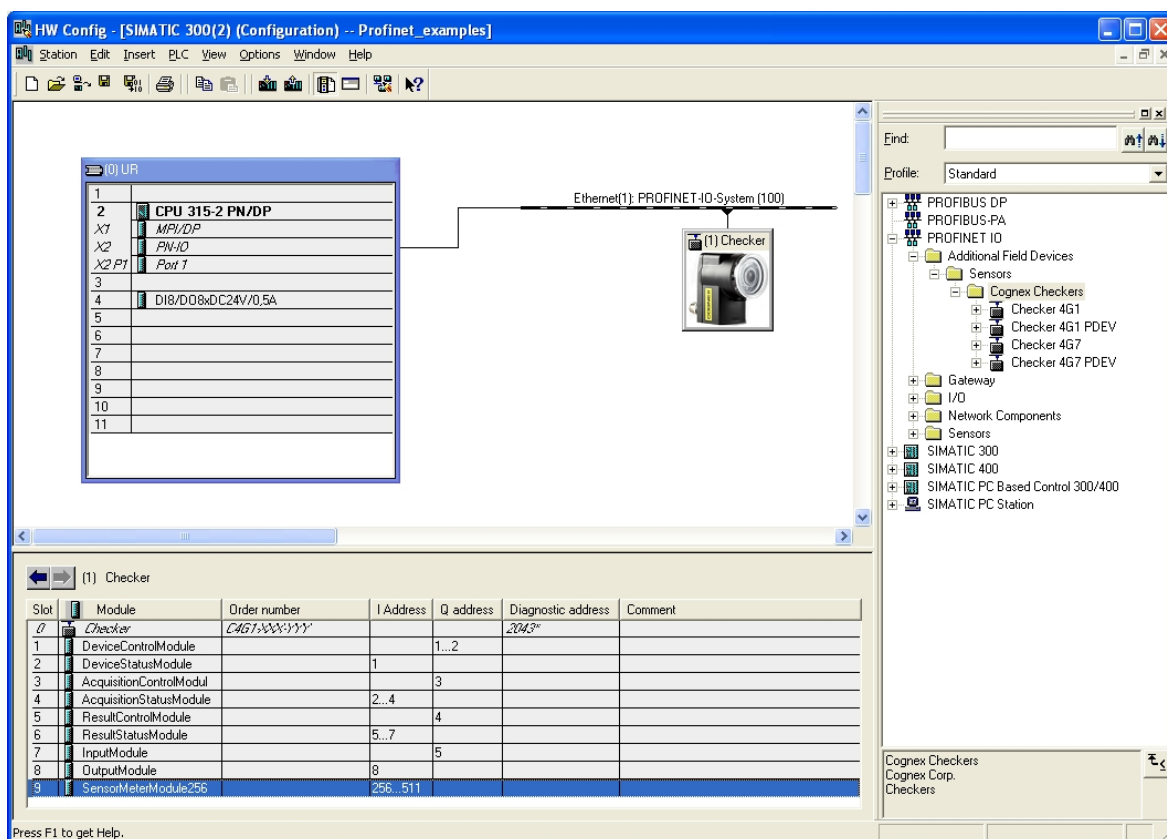


6. ジョブの変更が完了して、Q1.1 を 0 に戻すと、PLC は次のジョブの変更の準備完了です。

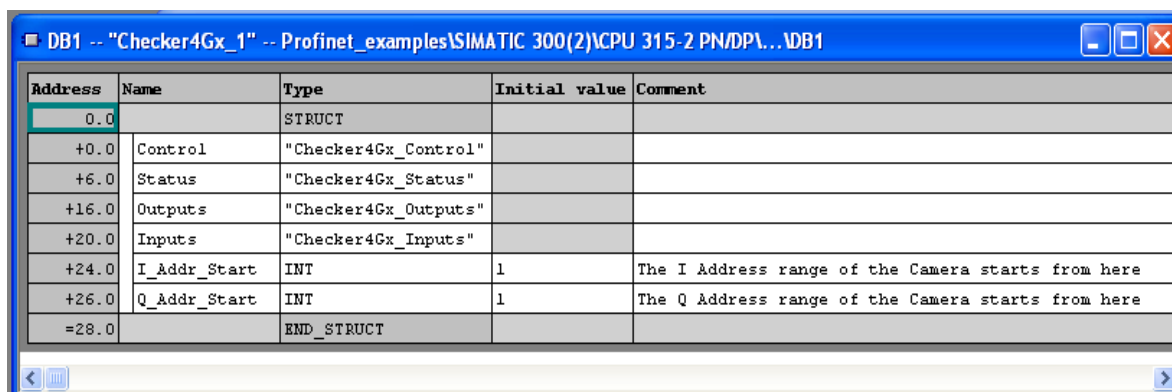


UDT の使用

提供されるサンプルで、I/O アドレスの開始値は事前に定義されており、これは HW Config ビューで確認できます。HW Config ビューについては、下のスクリーンショットを参照してください。



別のハードウェア構成（たとえば、新たに Checker を追加して、既存のハードウェア構成を変更）を使用したい場合は、DB1 ブロックで I_Addr_Start および Q_Addr_Start の値を適切に設定する必要があります。



The screenshot shows the 'DB1' block configuration in the HW Config software. The title bar reads 'DB1 -- "Checker4Gx_1" -- Profinet_examples\SIMATIC 300(2)\CPU 315-2 PN/DPA...DB1'. The table below lists the memory addresses and their corresponding data types and comments.

Address	Name	Type	Initial value	Comment
0.0		STRUCT		
+0.0	Control	"Checker4Gx_Control"		
+6.0	Status	"Checker4Gx_Status"		
+16.0	Outputs	"Checker4Gx_Outputs"		
+20.0	Inputs	"Checker4Gx_Inputs"		
+24.0	I_Addr_Start	INT	1	The I Address range of the Camera starts from here
+26.0	Q_Addr_Start	INT	1	The Q Address range of the Camera starts from here
=28.0		END_STRUCT		

また、Checker4G1 の代わりに Checker4G7 を使用したい場合もあります。この場合、まず、Checker4G1 を削除してから、Checker4G7 を追加して、HW Config ビューで Checker4G1 と Checker4G7 を交換する必要があります。最初に新しい Checker を追加してから、古いものを削除すると、ハードウェア構成での新しい Checker のハードウェアメモリが、DB1 ブロックでのメモリと対応しなくなります。

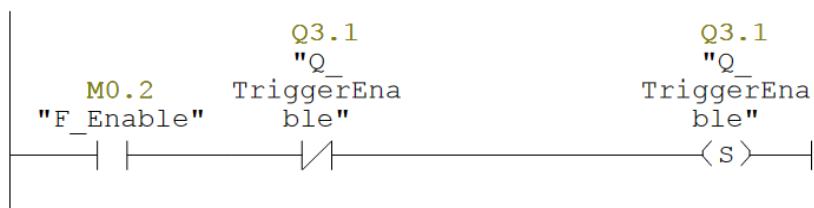
トリガとジョブの変更を交互に連続させる

この機能は、トリガとジョブの変更を連続的に実行し、2 つのイベントが重複しないことを保証する方法を示しています。(これは、多数ある可能な解決策の1つです。)

注:この例で、トリガ-ジョブの変更のサイクルは、連続的に実行されます。

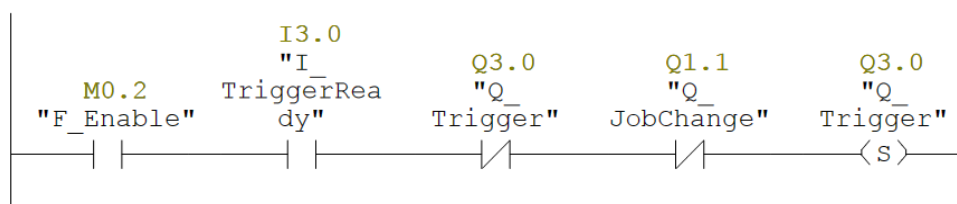
1. Trigger Enable (トリガ有効) : [トリガ有効] 信号を true (false であった場合) に設定し、Checker 側でトリガを有効にします。

注:ローカルの F_Enable ビットを使用して、この機能を起動または停止します。



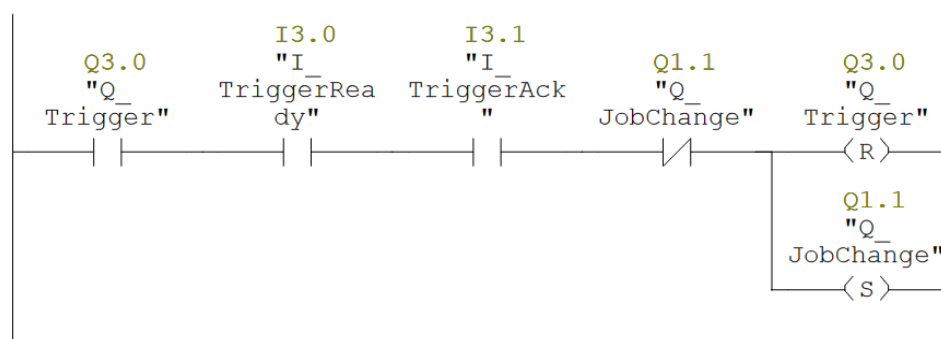
2. トリガ

Trigger Ready の範囲が拡張され、Checker の新しい取得が準備完了である場合にだけ高になります (つまり、進行中の取得、ジョブの変更、再登録はない)。これに基づくと、Checker が準備完了であり、ローカルの **Q_Trigger** および **Q_JobChange** で重複する信号をチェックできるには、**Trigger Ready**の状態を知れば十分です。



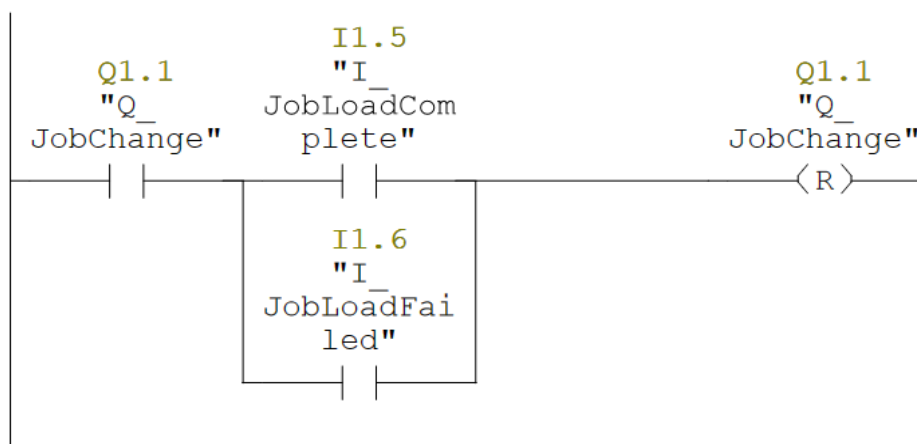
3. トリガ後にジョブの変更を開始する

トリガをリセットし、PLC に Checker から Trigger へ適切なフィードバックがあった時 (**Trigger Ack** が高で、**Trigger Ready** も高) に、ジョブの変更を同時に開始します。



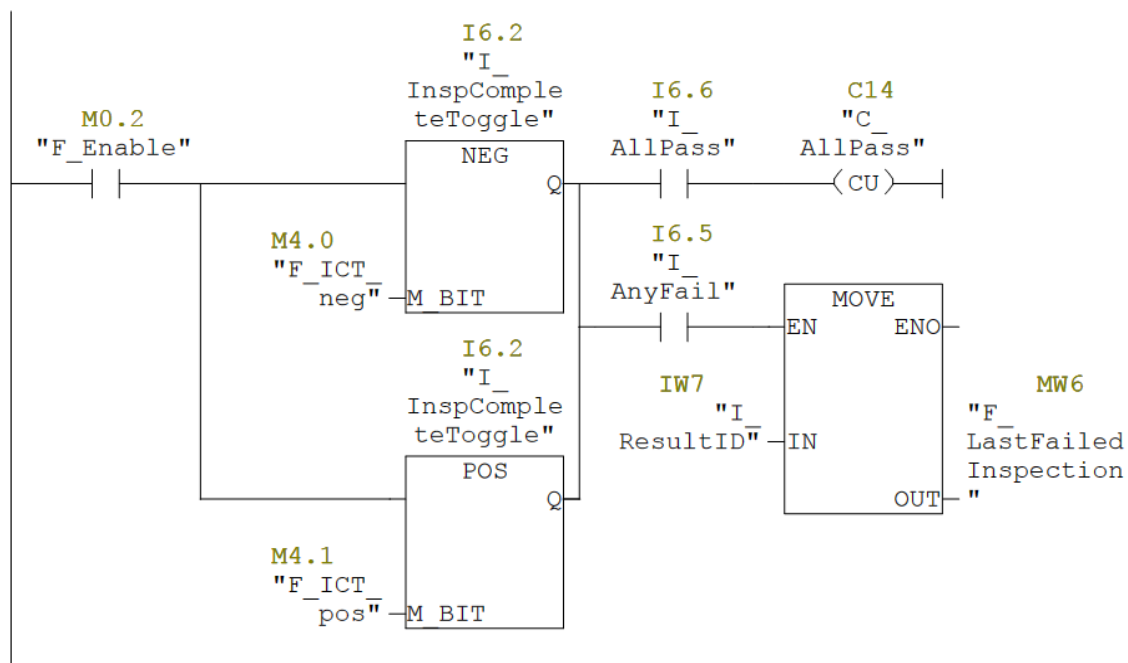
4. ジョブのロード完了、またはジョブのロード失敗の処理：ジョブの変更が完了するのを待ちます。

注:ステップ 2 で、条件の 1 つは Q_JobChange が false であり、Checker が応答を Job change へ返した時 (完了または失敗) にだけ、PLC が false に設定することです。



5. チェック結果のチェック

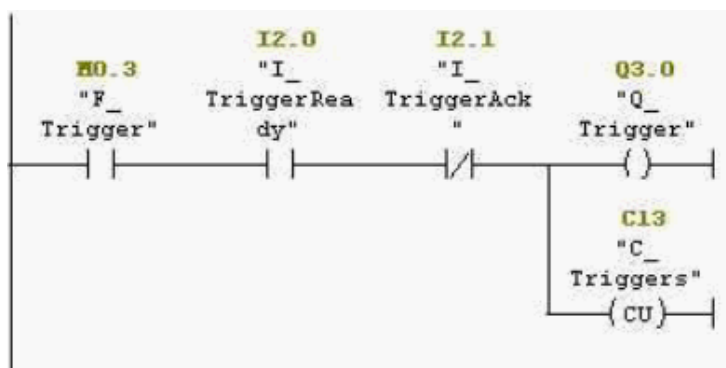
Inspection Complete Toggle ビットの変化によって、新しく使用可能な値があることが分かります。PLC は、立ち上がりと立下りの両方のエッジを検出する必要があります。**Inspection Complete Toggle** が変化するたびに、すべての結果関連データ (製品検出、すべて合格、いずれか不合格、V0..V23 など) が最新の状態に更新されます。この段階で、PLC は **すべて合格** および **いずれか不合格** のビットの状態だけをチェックし、これに基づいて、カウンターの値を増加させるか、**結果 ID** をローカル変数へ移動させます。



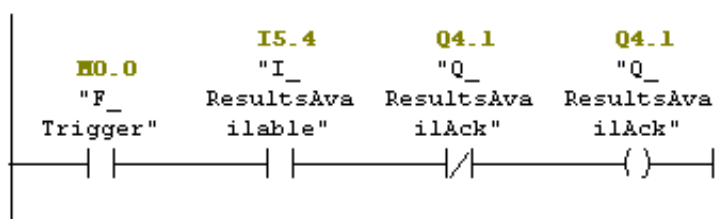
例に関する注

PLC は短い間隔で値を定期的に上書きするので、特定の変数の値を効率的に変更することとは困難であることに注意してください。

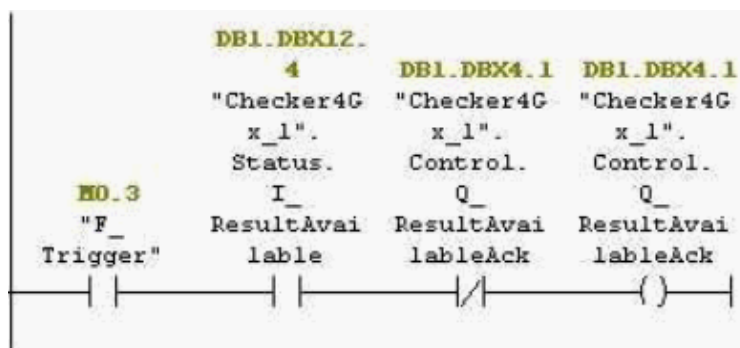
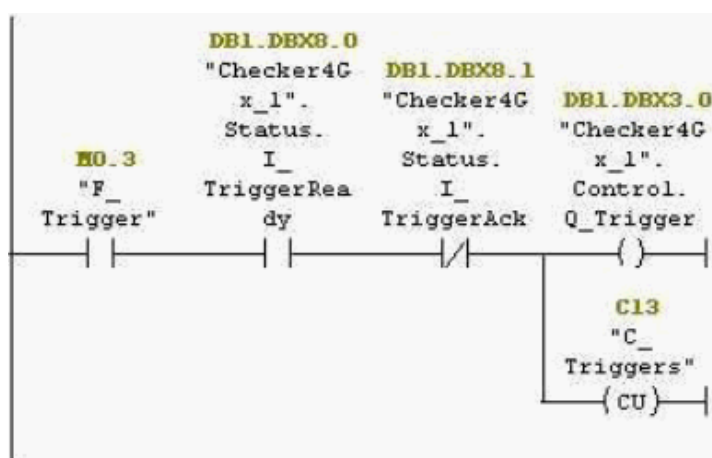
たとえば、以下のネットワークに存在する、トリガ出力 (Q3.0) を他の場所では変更することはできません：



たとえば、以下のネットワークに存在する、結果使用可能の確認 (Q4.1) を他の場所
で変更することはできません：



以下の例は、UDT データベースを使用した場合の上と同じ例を示しています：



Checker と Siemens S7-1200 PLC の間での PROFINET 通信の作成

このセクションでは、Checker を Siemens S7-1200 PLC と共に使用する方法を説明します。S7-1200 および Totally Integrated Automation (TIA) ポータルについては、知識があり、アクセス権もあると想定しています。

Checker® 4G 通信およびプログラムガイド

9/3/2014 | バージョン 4.1

1. TIA ポータルで新しいプロジェクトを作成します。
2. [デバイスとネットワーク] | [デバイスの設定] へ移動します。
3. [新しいデバイスの追加] をクリックして、S7 PLC を設定します。
4. [完了] をクリックすると、ポータルおよび内部の [DeviceView] へ移動できます。
5. プロジェクトのメイン・ウィンドウで、[ネットワーク・ビュー] を選択します。
6. [カタログ | ハードウェア・カタログ] へ移動します。
7. other field devices/PROFINET IO/Sensors/Cognex Corp/Cognex ID Readers/Checker xxx で探して、そのオブジェクトをプロジェクトのメイン・ウィンドウ (なるべく PLC 図面の近く) へドラッグ・アンド・ドロップします。

注:そこでまだリーダーが使用可能でなければ、[オプション | General station description (GSD) ファイルのインストール] へ移動して、デバイスに付属している GSD ファイルをインストールします。これが完了したら、ステップ 7 を繰り返します。

8. Checker の画像 (画像の周囲に表示される青いボックス) をクリックしてから、緑色の小さい長方形 (画像付近のテキストの周囲ではなく) をクリックします。
9. 下にある [プロパティ] タブにある、[全般] タブで、[Profinet インターフェース [X1]->Ethernet アドレス] を選択します。[Interface networked with] で、[新しいサブネットの追加] をクリックします。その後、PN/IE_1 が [サブネット] ドロップダウンボックスに表示されます。
10. [IP プロトコル] の下の [IP プロトコルの使用] チェックボックスを選択します。[別の方法を使用して、IP アドレスを設定する] を選択します (この設定で、Checker は Checker アプリケーションで設定された IP アドレスを保持しますので、スタティック・アドレスを使ってください)。

11. 最も重要：[PROFINET] で、[PROFINET デバイス名] および [変換した名前] を実際の Profinet 名に変更します。これは、実際の Checker デバイスの名前 (Checkmate で設定) と一致している必要があります。また、名前は一意であり、DNS 命名規則に適合している必要があります。
12. [新しいサブネットの追加] をクリックしたので、Checker オブジェクトの小さい緑色の長方形から緑色の線が伸びています (常に、ネットワーク・ビューにあります)。ここで、PN/IE_1 とラベルを付けます。そのラベルをドラッグして、PLC オブジェクトの小さい緑色の長方形へドロップします。
13. ここで、Checker の近くにある、青色の [未割当] のテキストをクリックし、リストから IO (PLC) コントローラを選択します。

Generic FFP

Generic FFP は、EtherNet/IP または PROFINET をサポートすることなく、Checker が PLC と通信できるように開発された、シンプルな UDP/IP ベースのプロトコルです。

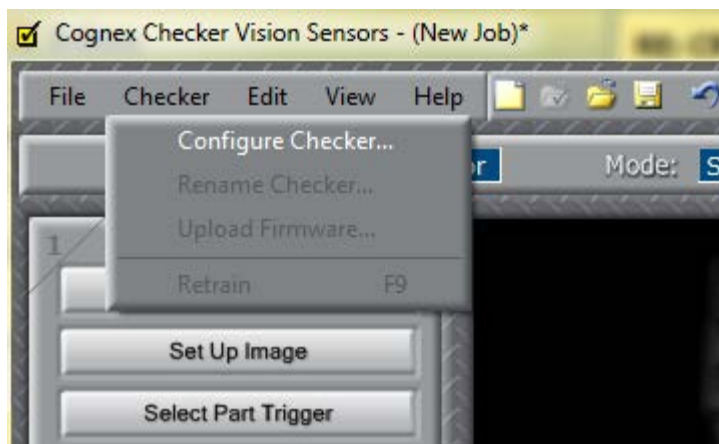
Generic FFP の基本的な通信スキームは、EtherNet/IP や PROFINET のスキームと同じです：線は、バイトの配列注のビットで表現され、ピアが解釈し、必要であれば変更に応じます。

Checker は、Generic FFP を通じて、24 の仮想の入力および出力をサポートしています。

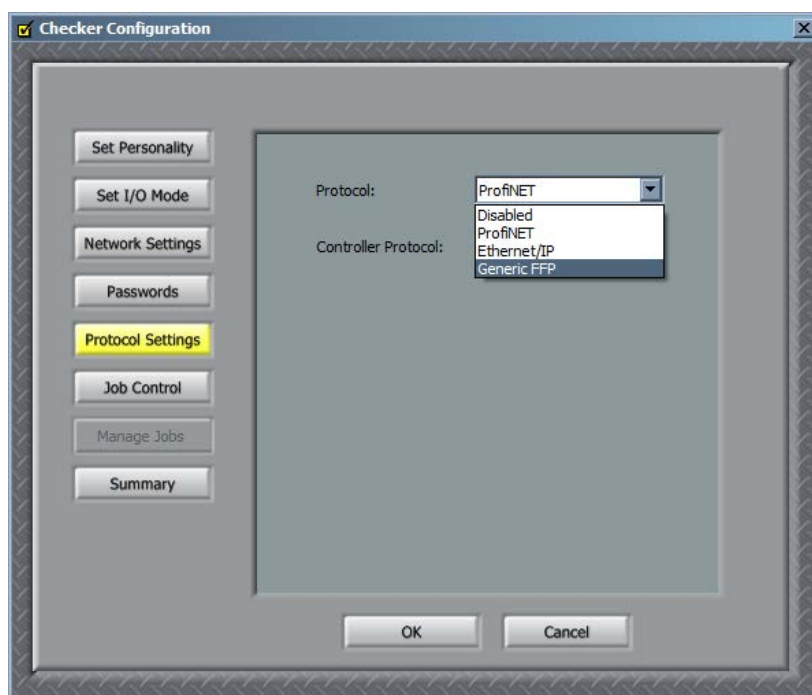
Checker GUI で Generic FFP を有効にして、設定する

デフォルトでは、Checker では Generic FFP は無効になっています。Checker GUIで、このプロトコルを有効にするには、以下のステップを実行してください：

1. 上のメニュー・バーで、[Checker] をクリックします。
2. [Checker の設定] を選択します。



3. ポップアップ表示される Checker 設定ウィンドウで、[プロトコル設定] をクリックし、[Generic FFP] を選択します。

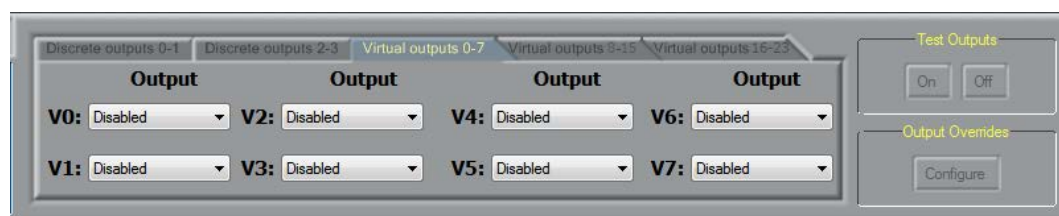


4. [OK] をクリックします。

変更を有効にするために、センサーは自動的に再起動します。

24 仮想入出力

Generic FFP を選択した場合、出力の設定のステップで 24 仮想出力スキャンを設定できます。詳細は、下のスクリーンショットを参照してください。これらの出力は、Checker から送信される入力アセンブリの PLC で使用できます。



24 仮想入力は、ラダーロジック・プログラムで選択できます。これらの入力は、出力アセンブリで PLC によって設定され、Checker へ送信されます。

レポート・メーター値およびスレッシュホールド設定ポイント

GenericFFP が有効であれば、入力アセンブリには実際の値とセンサーに対するスレッシュホールド設定ポイントが含まれます。センサーを設定して、Checkerから送信される入力アセンブリで、その結果を送信することができます。

センサーが作成されると、自動的にセンサー・メーター・テーブルに追加されます。それぞれのセンサーには、結果のテーブルに自動的に追加される値があります。最初のエントリには、製品検出値が含まれています。製品検出が存在しない場合、これらのエントリは－(1) に設定されます。

出力の設定のステップで利用できるセンサー・メーター・テーブルには、Checker ジョブに追加されているすべてのセンサーが表示されます。

1. ☒ Part Finding Sensor	2. ☒ Edge1	3. ☒ Contrast1	4. ☐	5. ☒ Diameter1	6. ☒ EdgePosition1	7. ☒ PatternPosition 1	8. ☐
9. ☐	10. ☐	11. ☐	12. ☒ Contrast2	13. ☐	14. ☐	15. ☐	16. ☐
17. ☐	18. ☐	19. ☐	20. ☐	21. ☐	22. ☐	23. ☐	24. ☐
25. ☐	26. ☐	27. ☐	28. ☐	29. ☐	30. ☐	31. ☐	32. ☐
33. ☐	34. ☐	35. ☐	36. ☐	37. ☐	38. ☐	39. ☐	40. ☐
41. ☐	42. ☐	43. ☐	44. ☐	45. ☐	46. ☐	47. ☐	48. ☐
49. ☐	50. ☐	51. ☐	52. ☐	53. ☐	54. ☐	55. ☐	56. ☐
57. ☐	58. ☐	59. ☐	60. ☐	61. ☐	62. ☐	63. ☐	64. ☐
65. ☐	66. ☐	67. ☐	68. ☐	69. ☐	70. ☐	71. ☐	72. ☐
73. ☐	74. ☐	75. ☐	76. ☐	77. ☐	78. ☐	79. ☐	80. ☐
81. ☐	82. ☐	83. ☐	84. ☐	85. ☐	86. ☐	87. ☐	88. ☐
89. ☐	90. ☐	91. ☐	92. ☐	93. ☐	94. ☐	95. ☐	96. ☐

Drag and Drop the name of the sensor to the corresponding slot.

Check the check box if you want a sensor meter value to be transmitted.

Cancel OK

セルをクリックして、別のセルへドラッグすることで、センサーを並び替えることができます。セルは、連続的に入力されている必要はありません。送信されないセンサーは、未チェックにできます。その場合は、表示は無効になります。

パターンの再登録

現在は、製品検出または PLC からのパターンセンサーを再登録することができます。PLC は、出力アセンブリでビットを設定し、ビットを設定すると、すべての再登録可能なセンサーが再登録されます。

再登録が完了すると、入力アセンブリのビットは、再登録アクションが発生していることを示すように設定されます。

入力アセンブリの Retrain Ack ビットは、再登録の完了時に設定されます。PLC によって Retrain Ack ビットが確認されると、PLC は出力アセンブリで Retrain ビットをクリアできます。Retrain ビットが PLC によってクリアされると、Retrain Ack ビットが Checker によってクリアされます。

注: パターンの再登録は、PROFINET および EtherNet/IP のプロトコルで使用できます。

内部照明の制御

PLC は、出力アセンブリの Illumination Off ビットを設定することで、Checker の外部照明を制御できます。設定すると、Checker は外部照明をオフにして、ストロボの出力を無効にします。照明がオフであり、PC アプリケーションが Checker を実行モードにしている間に、PC アプリケーションが Checker に接続すると、LED はジョブで設定された状態に設定されます。

チェックからの FTP 結果データ

FTP 接続では、FTP 経由でチェック結果データの送信を有効にすることができます。データ転送が有効であれば、それぞれのチェックの結果が FTP サーバーに送信され、データ転送ファイル名のフィールドで指定したファイルに追加されます。下のスクリーンショットを参照してください。

FTP Client Settings

FTP Connection 1

Server: [] Port #: []

User name: []

Password: [] ☐ Show password

Directory: []

File name fields: [Fields]

Template: 4G7X Simulator_[Acquisition Number].bmp

Data Transfer: [On] [Off] Data Transfer File Name: []

Draw Results: ☐ Draw Results on Image

Enabled: [On] [Off]

FTP Connection 2

Server: [] Port #: [0]

User name: []

Password: [] ☐ Show password

Directory: []

File name fields: [Fields]

Template: 4G7X Simulator_[Acquisition Number].bmp

Data Transfer: [On] [Off] Data Transfer File Name: []

Draw Results: ☐ Draw Results on Image

Enabled: [On] [Off]

[Apply] [Cancel]

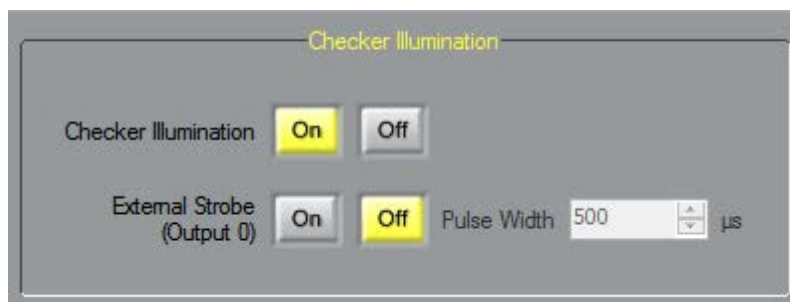
FTP データは、XML フォーマットのファイルを使用して送信されます。

ストロボの制御

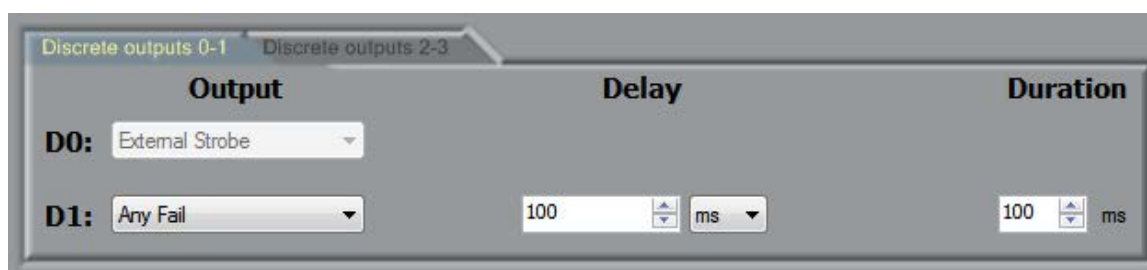
外部ストロボを使用するように Checker を設定することができます。外部ストロボに対して設定すると、1 つの出力ラインが失われ、出力の数が 2 (エンコーダー・モード) から 1 へ、または 4 から 3 へ減ります (ジョブ変更モード)。

Checker® 4G 通信およびプログラムガイド

9/3/2014 | バージョン 4.1



外部ストロボを有効にした場合、出力 0 を使ってストロボをトリガします。この出力は、出力の設定のセクションでは使用できません。Pulse Width の値は、ストロボがオンである期間です。この値は、Checker への入力となり、照明スライダーの露出/ゲイン設定の制御で使用されます。この値は、出力の期間として Checker が使用することはできません。出力の期間は、照明スライダーで制御されます。照明スライダーを変更すると、取得の期間の露出が変化します。Checker が取得している間、出力 D0 が有効になります。これによって、使用される外部ストロボ・コントローラに加えて、外部ライト・コントローラを使用できます。



操作

PLC (または PC) は、単純なバイナリ UDP メッセージを Checker IP アドレスのポート 7733 へ送ります。メッセージは、出力アセンブリで入力されます。Checker はデータを受信し、すぐに入力アセンブリに送信して返信します。2 つの出力アセンブリの UDP パケットの間には通信はないので、期間はマスター (PLC または PC) によって動的に設定されます。

単純な使用サイクルの擬似コード：

ループ

パラメータ AF_INET および SOCK_DGRAM でソケットを作成

出力アセンブリのバッファを 10 バイトに設定する：

- 2 バイトのプロトコル識別子：
 1. 0xAE – 破損したメッセージが偶発的に Checker をヒット
することを防ぐための通信のヒント。
 2. 0x01 – プロトコルのバージョン番号、ここでは 1。
- 8 バイトのペイロード

バッファを Checker の IP アドレス、ポート 7733 へ送信。

同じソケットから 436 バイトを受信。(UDP パケットが送信プログラムと受信プログラムの間のどこかで失われる可能性があるので、受信のタイムアウトを設定することを強く推奨いたします。)

ソケットを閉じる。

必要に応じて待機 (この一時停止が、期間を設定)。

ループ終了

入力アセンブリ

下のテーブルは、入力アセンブリ (Checker から送信されるデータ) を示しています。ここで、予約済み¹とは将来の使用のために予約されていることを意味します。Observer ビットが 1 に設定されると、Generic FFP コントローラは Checker のステータスを変更することができず、その状態を観察することができるだけです。Observer ビットは PC アプリケーションでのみ設定できます ([Checker] -> [Checker の設定] ->

Checker® 4G 通信およびプログラムガイド

9/3/2014 | バージョン 4.1

ページ | 104

[プロトコル設定)。詳細情報は、付録 A – プロトコル設定、オブザーバ・ビットの変更を参照してください。

インスタンス	バイト	ビット 7	ビット 6	ビット 5	ビット 4	ビット 3	ビット 2	ビット 1	ビット 0
11	0	Online (オンライン)	オフラインの理由			Missed Acq (確認失敗)	Acquiring (取得)	Trigger Ack (トリガ確認)	Trigger Ready (トリガ・レディ)
	1	General Fault (一般的なフ ォールト)	Job Load Failed (ジョブのロード 失敗)	Job Load Complete (ジョブのロ ード完了)	Observer (オブザーバー)	Results Available (結果使用 可能)	Results Buffer Overrun (結果バッフ ア・オーバ ーラン)	Inspection Complete Toggle (チェック完 了トグル)	Inspecting (チェック)
	2	予約済み ¹	予約済み ¹	予約済み ¹	Retrain Failed (再登録失敗)	Retrain Completed (再登録完了)	Any Fail (いずれか 不合格)	All Pass (すべて合格)	Part Detect (製品検出)
	3	出力7	出力6	出力5	出力4	出力3	出力2	出力1	出力0
	4	出力15	出力14	出力13	出力12	出力11	出力10	出力9	出力8
	5	出力23	出力22	出力21	出力20	出力19	出力18	出力17	出力16
	6	Acquisition ID (16-ビットの整数)							
	7								
	8								
	9	Inspection Result ID (16-ビットの整数)							
	10	センサー 1 結果データ 0							
	...	...							
	25	センサー 1 結果データ 15							
	26	センサー 2 結果データ 0							
	...	...							
	41	センサー 2 結果データ 15							
	42	センサー 3 結果データ 0							

インスタンス	バイト	ビット 7	ビット 6	ビット 5	ビット 4	ビット 3	ビット 2	ビット 1	ビット 0
	...	...							
	409	センサー 25 結果データ 15							
	410	センサー 26 結果データ 0							
	...	...							
	425	センサー 26 結果データ 15							

センサー・メーター・テーブルで返される値に関する情報は、スタティック・データおよび変動データの概要セクションを参照してください。

出力アセンブリ

次のテーブルは、出力アセンブリのインスタンスを示しています (Checker へ送信されるデータ)。ここで、予約済み¹とは将来の使用のために予約されていることを意味します。Lights Off ビットを 1 に設定すると、ジョブの設定に関わらず、外部照明がオフになります。SM Volatile ビットを 0 に設定すると、ジョブに対して変更されないスタティックな結果データを選択します。これを 1 に設定すると、製品によって異なる変動データが選択されます。このデータは、センサー・メーター・テーブルの値で返されます。

インスタンス	バイト	ビット 7	ビット 6	ビット 5	ビット 4	ビット 3	ビット 2	ビット 1	ビット 0
21	0	Set Offline (オフライン設定)	Lights Off (照明オフ)	Retrain (再登録)	Job Change (ジョブの変更)	Results Available Ack (結果使用可能確認)	Buffer Results Enable (バッファ結果有効)	Trigger (トリガ)	Trigger Enable (トリガ有効)
	1	入力7	入力6	入力5	入力4	入力3	入力2	入力1	入力0
	2	入力15	入力14	入力13	入力12	入力11	入力10	入力9	入力8

Checker の一般的な入力コマンドの処理

インスタンス	バイト	ビット 7	ビット 6	ビット 5	ビット 4	ビット 3	ビット 2	ビット 1	ビット 0
	3	入力23	入力22	入力21	入力20	入力19	入力18	入力17	入力16
	4	Job Number (8-ビットの整数)							
	5	予約済み ¹							SM Volatile (SM 変動)
	6	調整用バイト							
	7	調整用バイト							

Checker の一般的な入力コマンドの処理

定義

入力コマンド

Checker は、互いに連携する、以下の 3 つの入力コマンドを受け入れます：

1. Trigger (トリガ)

このコマンドは、取得と、それに続くチェックを開始します。

2. Job Change (ジョブの変更)

Checker が現在実行しているジョブを、選択したジョブに変更します。ジョブの
スロットから、任意のジョブを選べます。

3. Retrain (再登録)

最後に取得した画像で、該当するセンサーを再登録します。

入力コマンドのソース

Checker は、以下のソースから入力コマンドを受けることができます：

1. 個別のソース (ケーブル)
2. FFP プロトコル：PROFINET/Ethernet IP/Generic FFP
3. PC アプリケーション (実行モードでは、Retrain コマンドのみ)

同時複数のコマンド

これは、FFP 通信の場合だけです！ 3.6 以降、複数の入力コマンドを同一のネットワーク・パッケージで送信し、事前に定義した順番で実行することができます。

可能な組み合わせ：

1. Trigger および Job Change => 1. Trigger、2. Job Change
2. Trigger および Retrain => 1. Trigger、2. Retrain
3. Job Change および Retrain => 1. Job Change (Retrain はドロップ)
4. Trigger および Job Change および Retrain => 1. Trigger、2. Job Change (Retrain はドロップ)

重要な注意： 同時に PLC によって送信された 2 つの別個の信号は、複数のコマンドではなく、重複するコマンドとして処理されます (下記を参照)。ここで、信号を処理する順番は、非決定的であり、いずれかのコマンドがドロップします！

コマンドの重複

コマンドが重複する場合、以下のケースが発生します：

- 1) 入力コマンドの実行中に、新しい入力コマンドが到着。
- 2) 入力コマンドの実行中に、複数の入力コマンドを持つ FFP パケットが到着。

3) コマンドの実行中に、ディレイしたコマンドが開始。

いずれのケースでも、単に新しいコマンドが拒否されて、存在する場合には、該当する相手にエラーの結果が伝えられます。

エラーの結果としては、以下があります： Missed Acquisition、 Job Load Failed、 Retrain Failed。

ポートの利用

以下のテーブルは、通信用に Checker が使用するポートを一覧しています。
ファイアーウォールを使用している場合、これらのポートをオープンにしておく必要があります。

プロトコル	ポート
Checker の一般的な通信のための TCP および UDP	• UDP: 1069 • TCP: 50053
Ethernet/IP	• UDP: 2222 • TCP: 44818
PROFINET	UDP:4096、 4097、 4098、 34964 注： PROFINET は、 IP/TCP/UDP ヘッダーなしで、未処理の Ethernet フレームを使用します。
Generic FFP	• UDP: 7733 • TCP: 7733
FTP	デフォルトは、ポート 21 パッシブですが、FTP セットアップで変更できます。

付録 A – プロトコル設定、オブザーバ・ビットの変更

Checker PC アプリケーションでは、データ通信やデバイスの制御のために、さまざまな産業用プロトコルを有効にすることができます。これらのプロトコルは、[Checker | Checker の設定 | プロトコル設定] で設定できます。それでデバイスを制御したい場合、Generic FFP プロトコルは必ず存在します。混乱する（重複、矛盾）制御コマンドを Checker に送信することを防ぐために、制御モードからオブザーバーモードを区別する、Observer ビットが導入されています。オブザーバーモードでは、コントローラは Checker を制御することができず、その状態を観察することができるだけです。オブザーバーの場合、Observer ビットは 1 であり、コントローラの場合、0 になります。

Protocol および **Controller Protocol** で同じ産業用プロトコルを選択すると、設定された産業用プロトコルでは Observer ビットが 0 になり、Generic FFP では 1 になります。Generic FFP を明示的にコントローラのプロトコルに設定すると、それに対しては Observer ビットが 0 になり、コントローラとなります。使用可能なバリエーションについては、以下のテーブルを参照してください：

プロトコル設定	Generic FFP (Observer ビットの値)	Ethernet/IP (Observer ビットの値)	Profinet (Observer ビットの値)
Protocol: ProfiNET Controller Protocol: ProfiNET	Observer (1)	該当なし	Controller (0)
Protocol: ProfiNET Controller Protocol: Generic FFP	Controller (0)	該当なし	Observer (1)
Protocol: Ethernet/IP Controller Protocol: Ethernet/IP	Observer (1)	Controller (0)	該当なし

プロトコル設定	Generic FFP (Observer ビットの値)	Ethernet/IP (Observer ビットの値)	Profinet (Observer ビットの値)
Protocol: Ethernet/IP ▼ Controller Protocol: Generic FFP ▼	Controller (0)	Observer (1)	該当なし
Protocol: Generic FFP ▼ Controller Protocol: Generic FFP ▼	Controller (0)	該当なし	該当なし
Protocol: Disabled ▼ Controller Protocol: Generic FFP ▼	Controller (0)	該当なし	該当なし