Parallel Interface Design Guide

Anybus®-S Slave & Master

Doc.Id. JCM-1201-006
Rev. 2.08

HMS Industrial Networks AB

	☎		✉
Germany	+49	- 721 - 96472 - 0	ge-sales@hms-networks.com
Japan	+81	- 45 - 478 -5340	jp-sales@hms-networks.com
Sweden	+46	- 35 - 17 29 20	sales@hms-networks.com
U.S.A.	+1	- 312 - 829 - 0601	us-sales@hms-networks.com
France	+33	- 3 89 32 76 76	fr-sales@hms-networks.com
Italy	+39	- 347 - 00894 - 70	it-sales@hms-networks.com
China	+86	- 10 - 8532 - 3183	cn-sales@hms-networks.com

目次

序章 **このマニュアルについて**

このマニュアルの使い方 P-1
重要なユーザ情報 P-1
関連マニュアル P-2
マニュアル更新履歴 P-2
このマニュアルで用いる規約 P-3
サポート P-3

第 1 章 **イントロダクション**

主な機能 1-1
内部 1-2
外観 1-3

第 2 章 **アプリケーション コネクタ**

コネクタ・ピンアウト 2-1
制御シグナル 2-2
アドレス入力 (A0 ... A11) 2-2
データ入力 / 出力 (D0 ... D7) 2-2
ビジー・シグナル (BUSY) 2-2
割り込みリクエスト (IRQ) 2-2
出力イネーブル (OE) 2-2
読み込み / 書き込み (R/W) 2-2
チップ・イネーブル (CE) 2-2
リセット (RESET) 2-3
非同期シリアル・インターフェース 2-3
送信データ (TxD) 2-3
受信データ (RxD) 2-3

第 3 章 **メモリ マップ**

第 4 章 制御レジスタ領域

レジスタ 4-2
モジュール・ブートローダー・バージョン (7C0h - 7C1h, RO) 4-2
アプリケーション・インターフェース・ソフトウェア・バージョン (7C2h - 7C3h, RO) 4-2
フィールドバス・ソフトウェア・バージョン (7C4h - 7C5h, RO) 4-2
モジュール・シリアル番号 (7C6h - 7C9h, RO) 4-2
ベンダ ID (7CAh - 7CBh, RO) 4-2
フィールドバス・タイプ (7CCh - 7CDh, RO) 4-3
モジュール・ソフトウェア・バージョン (7CEh - 7CFh, RO) 4-3
Watchdog カウンタ入力 (7D2h - 7D3h, R/W) 4-3
Watchdog カウンタ出力 (7D4h - 7D5h, RO) 4-4
LED ステータス (7DAh - 7DFh, RO) 4-4
モジュール・タイプ (7E0h - 7E1h, RO) 4-4
モジュール・ステータス (7E2h - 7E3h, RO) 4-5
更新データ・フィールド (7E4h - 7EBh, R/W) 4-6
イベント通知発生要因 (7ECh - 7EDh, R/W) 4-6
イベント通知ソース (7EEh - 7EFh, RO) 4-7
入力 I/O 長 (7F0h - 7F1h, RO) 4-7
入力 DPRAM 長 (7F2h - 7F3h, RO) 4-7
入力合計長 (7F4h - 7F5h, RO) 4-8
出力 I/O 長 (7F6h - 7F7h, RO) 4-8
出力 DPRAM 長 (7F8h - 7F9h, RO) 4-8
出力合計長 (7FAh - 7FBh, RO) 4-8

第 5 章 ハンドシェーキング & インジケーション・レジスタ

アプリケーション・インジケーション・レジスタ (7FEh, R/W) 5-2
Anybus インジケーション・レジスタ (7FFh, RO) 5-3
衝突 5-4
領域割り当て / リリース 5-5
非同期データ交換 5-5
同期されたデータ交換 5-6
複数領域の同時リクエスト / リリース 5-7
アプリケーション事例、サイクリック・アクセス方法 5-8

第 6 章 割り込み

ハードウェア割り込み (IRQ) 6-1
イベント通知 (ソフトウェア割り込み) 6-2

第 7 章 フィールドバス・データ交換

基本 7-1
デュアル・ポート・メモリ vs 内部メモリ 7-2
データ・タイプ 7-2
データ構成 7-3

第 8 章 メールボックス・インターフェース

概論 8-1

メッセージ・タイプ 8-2

メールボックス通知ビット 8-2
メールボックス・メッセージを送信する 8-3
メールボックス・メッセージを受信する 8-3

メールボックス・メッセージ・ストラクチャ 8-4

メッセージ・ヘッダ 8-4
メッセージID 8-5
メッセージ・インフォメーション 8-5
コマンド番号 8-5
データ・サイズ 8-5
拡張されたワード1 ... 8 8-5

第 9 章 メールボックス・メッセージ

アプリケーション・メッセージ 9-1
初期化の開始 (START_INIT) 9-2
Anybus の初期化 (Anybus_INIT) 9-3
初期化の終了 (END_INIT) 9-6
FLASH への保存 (SAVE_TO_FLASH) 9-7
FLASH からのロード (LOAD_FROM_FLASH) 9-8
ハードウェア・チェック (HW_CHK) 9-9

フィールドバス・メッセージ 9-10

内部メモリ・メッセージ 9-11
内部入力領域からの読み込み (RD_INT_IN) 9-12
内部入力領域への書き込み (WR_INT_IN) 9-13
内部入力領域のクリア (CLR_INT_IN) 9-14
内部出力領域からの読み込み (RD_INT_OUT) 9-15

リセット・メッセージ 9-16
ソフトウェア・リセット (SW_RESET) 9-16

第 10 章 スタートアップと初期化

イントロダクション 10-1

ハードウェアの初期化 10-2
スタートアップ・シーケンス 10-2
デュアル・ポート・メモリ・チェック（オプション） 10-2
ハードウェア・チェック（オプション） 10-3

ソフトウェアの初期化 10-3
データ初期化の準備 10-3
初期化の開始 10-3
パラメータ値の初期化 10-4
フィールドバス・データの初期設定 10-4
初期化の終了 10-4
基本的な初期化シーケンス 例 1 10-5
基本的な初期化シーケンス 例 2 10-5
高度な初期化例 10-6

第 11 章 LED 表示

フィールドバス・ステータス表示 11-1
Anybus-S Watchdog LED 11-1

第 12 章 ファームウェアの更新

第 13 章 ドライバ・サンプル

割り込みハンドラ 13-2
インターフェース・
ハンドラ 13-3
メールボックス・ハンドラ 13-4

第 14 章 メカニカル仕様

PCB 寸法 14-1
高さ制限 14-1
取り付け穴 14-2
アプリケーション・コネクタ 14-2
フィールドバス・コネクタ 14-2
フィールドバス・ステータス表示 LED 14-3

アペンディックス A 拡張されたメモリ・モード (4K DPRAM)

アペンディックス B 注意

概論 B-1
ロックリリースの動作 B-1
アプリケーション・インターフェース・ハードウェアの注意 B-2

アペンディックス C 割り込み信号とハードウェア・リセット

推奨される処理 C-2

アペンディックス D 電気的特性

電源仕様 D-1
信号特性 D-2
推奨される PE とシールド処理 D-2

アペンディックス E 使用環境

温度 E-1
相対湿度 E-1
EMC 規格 E-1

アペンディックス F　事前に取得されている規格
フィールドバス認証 F-1
CE- マーク F-1
UL/cUL- 認証 F-1

アペンディックス G　トラブルシューティング

このマニュアルについて

このマニュアルの使い方

本デザイン・ガイドは、Anybus-S ならびに Anybus-M 通信モジュール製品群の共通機能について説明しています。Anybus により提供されるフィールドバス個別の機能については別途フィールドバス Appendix を参照してください。

すべてのフィールドバス・システムは個別のものです。これらの相違点を埋めるために、本ドキュメントでの説明が本来の目的から外れることがあるかもしれません。詳細については B-1 “ 注意 ” を参照してください。

また、開発ツール , サンプル・コード等の詳細については HMS のウェブサイト、‘www.anybus.com’ でご覧いただけます。

重要なユーザ情報

このドキュメントにあるデータや図は強制するものではありません。HMS Industrial Networks AB は継続して製品を開発する方針に沿って製品を変更する権利を保持します。このドキュメント内の情報は、予告なく変更されることがあります。HMS Industrial Networks AB は、このドキュメントにより生じたいかなるエラーに対しても責任を負いません。

この製品には多くのアプリケーションがあります。このデバイスの使用に責任のある者は、アプリケーションがいかなる関係法令、規則、コードそして規格も含めて、性能と安全要求事項を満たすことを確かめるために、すべての必要な手続きがとられていなければならないことを保証しなければなりません。

Anybus® は、HMS Industrial Networks AB の登録商標です。その他については各社の商標です。

関連マニュアル

マニュアル名	作成者
Anybus-S API Reference Manual	HMS (www.hms-networks.com)
Anybus-S Fieldbus Appendices (各フィールドバス個別)	
Data sheet for dual port memory (CY7C136)	Cypress (www.cypress.com)
Understanding Asynchronous Dual-Port RAMs (アプリケーション・ノート)	

マニュアル更新履歴

最近の更新 (2.04...2.05)

変更内容	ページ
わかりづらいテキスト削除	5-1
Anybus-M Devicenet Master/Scanner、拡張されたメモリをサポートを追加	A-1
フィールドバス・タイプ・テーブル更新	4-3

改訂版リスト

改訂番号	改定日	作成者	章	記述
2.00	2004-02-19	PeP	前章	2 回目のメジャー・リリース
2.01	2004-06-17	ToT	第 4 章	‘ モジュール・ステータス・レジスタ ’ のオンライン / オフライン表示
2.02	2005-07-19	PeP	第 9 章 アペンディックス D 第 2 章	Anybus_INIT レスポンス（フォルト・インフォメーション）訂正 信号レベル訂正（リセット信号） プルアップ抵抗 & デカプリング訂正（リセット信号）
2.03	2008-10-14	HeS	第 2 章 第 10 章 第 14 章 第 1,9 章	/CS, /RD, /WR を CE, OE, R/W へ名称変更 デュアル・ポート・メモリ・チェック中の例外の更新 製図の DCP 寸法訂正 その他、マイナー更新
2.04	2009-09-10	KeL	第 6, 8,D,12,4, 5, B 章	その他、マイナー更新
2.05	2009-11-13	KeL	第 4, 5, A 章	その他、マイナー更新
2.06	2010-01-12	KeL	6, 7, D	Misc. minor updates
2.07	2010-04-16	KeL	4	Minor update
2.08	2010-12-17	KeL	P, 4	Minor update

このマニュアルで用いる規約

以下の規約がこのマニュアル中で使用されます：

- 番号を付けたリストが連番で提供されます。
- 黒丸を付けたリストは情報を提供していますが、手順のステップではありません。
- ‘モジュール’とは Anybus モジュールを意味します。
- ‘アプリケーション’とは、Anybus アプリケーション・コネクタに接続されるハードウェアを意味します。
- 16 進値は NNNNh というフォーマットで書かれ、NNNN は 16 進法の値です。
- このドキュメント内のすべての寸法表記は、他に記述がない限り ±0.25mm の誤差があります。
- 16/32 ビット値は、一般的に記述がない限り Motorola フォーマット（ビッグエンディアン）で保存されます。

サポート

HMS Sweden (Head Office)

E-mail:	support@hms-networks.com
Phone:	+46 (0) 35 - 17 29 20
Fax:	+46 (0) 35 - 17 29 09
Online:	www.anybus.com

HMS North America

E-mail:	us-support@hms-networks.com
Phone:	+1-312-829-0601
Toll Free:	+1-888-8-Anybus
Fax:	+1-312-738-5873
Online:	www.anybus.com

HMS Germany

E-mail:	ge-support@hms-networks.com
Phone:	+49-721-96472-0
Fax:	+49-721-964-7210
Online:	www.anybus.com

HMS Japan

E-mail:	jp-support@hms-networks.com
Phone:	+81-45-478-5340
Fax:	+81-45-476-0315
Online:	www.anybus.com

HMS China

E-mail:	cn-support@hms-networks.com
Phone:	+86 10 8532 3023
Online:	www.anybus.com

HMS Italy

E-mail: it-support@hms-networks.com
Phone: +39 039 59662 27
Fax: +39 039 59662 31
Online: www.anybus.com

HMS France

E-mail: fr-support@hms-networks.com
Phone: +33 (0) 3 89 32 76 41
Fax: +33 (0) 3 89 32 76 31
Online: www.anybus.com

イントロダクション

Anybus-S/Anybus-M は、メモリと高性能プロセッサを搭載した、組み込み型フィールドバス通信用モジュールです。 フィールドバス通信に要求されるすべてのソフトウェアならびにハードウェア機能が組み込まれていて、エンジニアはアプリケーションや他のタスクに専念することができます。

データはデュアル・ポートを介して通信されます。そのため、高速通信を実現し、一般的にアプリケーションに対して僅かな負荷しか与えません。

Anybus-S/ Anybus-M モジュールは、ハードウェア / ソフトウェアとも共通したインターフェースのため、異なるモジュールとの互換性を実現します。また、同一基板を異なるフィールドバス・システムで使用することも可能です。

代表的なアプリケーションとして、周波数インバータ、HMI、表示機器、計測器、スケール、ロボット、PLC など、インテリジェントな計測機器があげられます。

注意：Anybus-M のアプリケーション・インターフェースは、Anybus-S と同じです。本マニュアル内のすべてのレファレンスは Anybus-S 向けですが、Anybus-M に対しても同様です。

主な機能

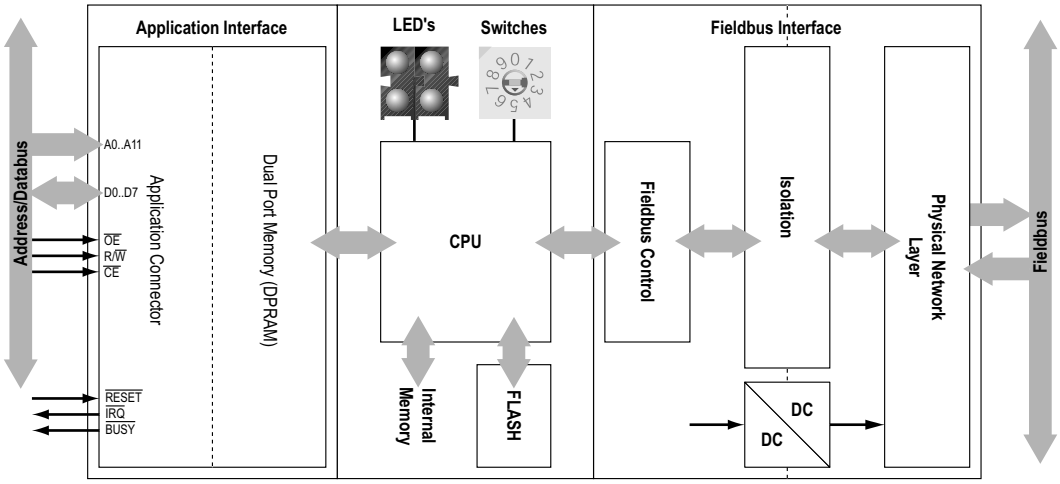
- **内部交換可能 (各フィールドバスに共通したインターフェース)**
- **スレーブ、マスター・バージョンを提供**
- **共通アプリケーション・インターフェースにより主要フィールドバス・システムをサポート**
- **CPU を搭載しているためネットワーク対応を迅速に実現**
- **すべてのフィールドバス・ネットワークの認証取得済み[1]**
- **メールボックス・インターフェース**
- **2KB デュアル・ポート Ram (DPRAM)[2] アーキテクチャ**
- **最大 2048 バイトの入力 / 出力データ対応 [2]**
- **オン・ボード・コンフィギュレーション・スイッチ**
- **オン・ボード LED 表示**
- **絶縁処理済みフィールドバス・インターフェース**
- **CE, UL & cUL 承認済み**

---

1. F-1 “ フィールドバス認証 ” を参照してください。
2. Anybus-S のバージョンによってはより多くの DPRAM や I/O データを提供します。詳細については A-1 “ 拡張されたメモリ・モード (4K DPRAM)” を参照してください。

内部

Anybus-S モジュールをブロック図で表わすと下図のようになります。アプリケーション・インターフェース、内部データパス、フィールドバス・インターフェース。



アプリケーション・インターフェース

外観からは 8 ビット並列ポート・インターフェースを装備した、アドレス / データ・バスを持つマイクロプロセッサ搭載システムに容易に組み込むことのできるアプリケーション・インターフェースが共通しています。さらにこのアプリケーション・インターフェースはまた、リセット・ピン、ビジー・シグナル、そして割り込み要求シグナル機能を提供します。

フィールドバス・インターフェース

Anybus-S モジュールのフィールドバス・インターフェースは絶縁処理がされていて、各フィールドバス標準向けにデザインされています。フィールドバス・プロトコルは Anybus-S 側で扱われ、アプリケーションによる処理を必要としません。Anybus-S モジュールは、各フィールドバスの可能性を最大限に利用できるよう、各フィールドバス向け仕様に対応しています。これらの機能を活かすのはアプリケーション次第です。

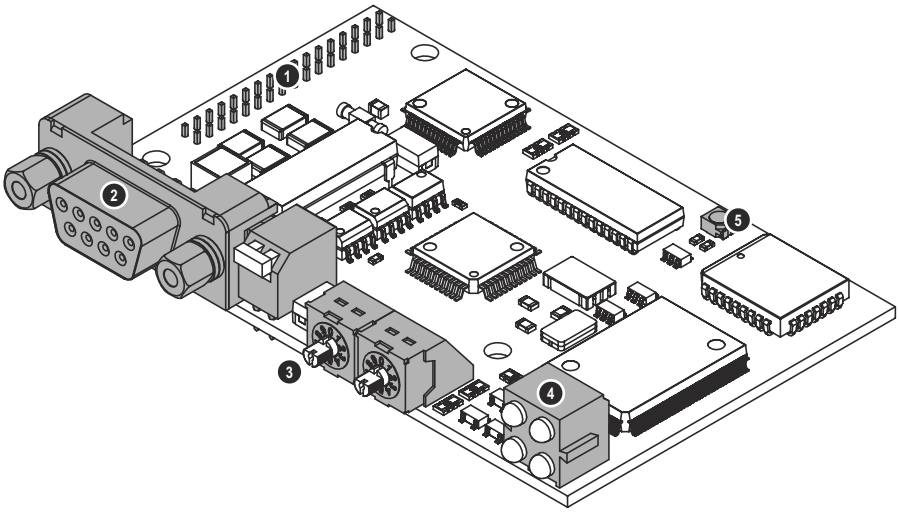
Anybus-S は単体でテストされ、各フィールドバスに対応します。詳細については F-1 “ フィールドバス認証 ” を参照してください。

データ交換

アプリケーション・インターフェースはデュアル・ポート メモリ (DPRAM) アーキテクチャに基づいています。そのためアプリケーション側は Anybus-S モジュールと共有メモリ領域を介してデータを交換します。基本的にフィールドバス上でデータを交換するために、すべてのアプリケーションは領域間でデータの読み込み / 書き込みを行う必要があります。データは搭載された CPU によりフィールドバス間で転送されます。つまりすべてのフィールドバスの作業は Anybus-S により取り扱われ、アプリケーション側との対話を必要としません。

外観

下図は Anybus-S の代表的なものです。詳細については 14-1 “ メカニカル仕様 ” を参照してください。



1. アプリケーション・コネクタ

Anybus-S は 2mm ストリップ・ヘッダーの 34-pin コネクタ で接続されます。このコネクタは制御シグナル、アドレス / データ・バス・シグナル、そして電力供給などの機能を提供します。詳細については 2-1 “ アプリケーション コネクタ ” ならびにアペンディックス、14-2 “ アプリケーション・コネクタ ” を参照してください。

2. フィールドバス・コネクタ

Anybus-S は各フィールドバス仕様 [1] に沿ったフィールドバス・コネクタを提供します。

3. コンフィギュレーション・スイッチ

Anybus-S モジュールの機能には、ボーレート、ノード・アドレス、フィールドバスの終端など、フィールドバス設定のためのコンフィギュレーション・スイッチがボード上に用意されています。

4. フィールドバス・ステータス表示 LED

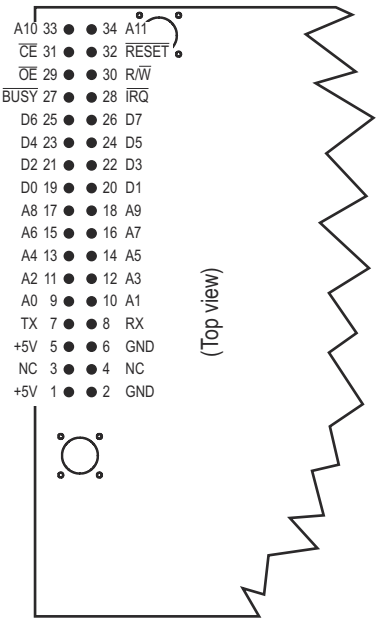
Anybus-S モジュールは、フィールドバス標準に準拠した LED 表示を行います。詳細については 11-1 “ フィールドバス・ステータス表示 ” ならびにアペンディックス 14-3 “ フィールドバス・ステータス表示 LED” を参照してください。

5. Anybus-S Watchdog Led

詳細については 11-1 “Anybus-S Watchdog LED” を参照してください。

1. 詳細については F-1 “ フィールドバス認証 ” を参照してください。

アプリケーション コネクタ



Anybus-S アプリケーション・コネクタは、パラレルスレーブ・ポート・タイプのインターフェースです。このインターフェースは8ビット幅のSRAMと類似しています。そのため、ホスト・アプリケーション側のアドレス / データ・バスに直接モジュールを組み込むことができます。

アプリケーション・コネクタは、また、非同期シリアル・インターフェースです。このインターフェースは通常ファームウェア・アップグレードなどに使用されますが、外部コンフィギュレーションやモニタリングを目的として使用されることもあります。

メカニカルの詳細、測定など、詳細については D-1 “ 電気的特性 ” を参照してください。

コネクタ・ピンアウト

注意 : すべてのシグナルは記述がない限り TTL レベルです。

ピン	名前	説明	方向	注意
1	+5V	VCC	入力	電力供給、バス・インターフェース、D-1 “ 電源仕様 ” を参照してください。
2	GND	グラウンド	-	
3, 4	NC	絶縁距離	-	(未接続)
5	+5V	VCC	入力	電力供給、モジュール・エレクトロニクス D-1 “ 電源仕様 ” を参照してください。
6	GND	グラウンド	-	
7	TxD	伝送データ [a]	出力	非同期シリアル・インターフェース伝送 [a]
8	RxD	データ受信 [a]	入力	非同期シリアル・インターフェース受信 [a]
9 - 12	A_0 - A_3	アドレス入力	入力	アドレス・ライン 0 ... 3
13	A_4	アドレス入力 [a]	入力	アドレス・ライン 4[a]
14 - 18	A_5 - A_9	アドレス入力	入力	アドレス・ライン 5 ... 9
19 - 26	D_0 - D_7	データ入力 / 出力	双方向	データ・バス、ビット 0 ... 7
27	$\overline{BUSY}$	ビジー・シグナル [a]	出力	アクティブ・ロー・オープン・コレクタ出力 [a]
28	$\overline{IRQ}$	割り込みリクエスト [a]	出力	アクティブ・ロー・オープン・コレクタ出力 [a]
29	$\overline{OE}$	出力イネーブル	入力	アクティブ・ロー入力
30	R/$\overline{W}$	読み込み / 書き込み	入力	アクティブ・ロー入力
31	$\overline{CE}$	チップ・イネーブル [a]	入力	アクティブ・ロー入力 [a]
32	$\overline{RESET}$	リセット	入力	アクティブ・ロー入力 内部的に 35 ー 75KΩ までプルアップ
33	A_{10}	アドレス入力 [a]	入力	アドレス・ライン 10[a]
34	A_{11}	アドレス入力[b]	入力	アドレス・ライン 11[b]

a. 内部は 10kΩ でプルアップ
b. 本シグナルはより大きいデュアル・ポート・メモリを収容するため Anybus モジュールに使用されます。これらのモジュール上では、10kΩ でプルアップされます。詳細については A-1 “ 拡張されたメモリ・モード (4K DPRAM)” を参照してください。本ピンの使用についてはオプションであることに注意してください。使用しない場合には接続しないか VCC にプルアップする必要があります。

注意 : Anybus-S が出荷されて以来、アプリケーション・インターフェースは少し変更されています。上記テーブルの情報は最新版の Anybus-S バージョンに該当します。詳細については B-2 “ アプリケーション・インターフェース・ハードウェアの注意 ” を参照してください。

制御シグナル

アドレス入力 (A_0 ... A_{11})

アドレス入力ピン。DPRAM 内で選択してください。A0 は最下位ビットを含み、A11 は最上位ビットを含みます。A_4、A_{10} そして A_{11} は、10kΩ でプルアップします。

A_{11} の使用はオプションですが、メモリ機能拡張に使用している Anybus モジュールもありますので、アプリケーション上で使用することを推奨します。使用しない場合、接続しないままか VCC でプルアップする必要があります。詳細については A-1 “ 拡張されたメモリ・モード (4K DPRAM)” を参照してください。また B-2 “ アプリケーション・インターフェース・ハードウェアの注意 ” も参照してください。

データ 入力 / 出力 (D_0 ... D_7)

データ出力ピンは書き込み中、入力ピンは書き込み中に動作します。D_0 は最下位ビット、D_7 は最上位ビットになります。

ターゲット・メモリ位置はアドレス入力 (A_0 ... A_{11}) で指定されます。

ビジー・シグナル ($\overline{BUSY}$)

アクティブ・ローのオープン・コレクタ出力、内部 10kΩ でプルアップされています。ローの場合、このピンは希望するアドレス が Anybus モジュールに使用されていることを示しています。また、モジュールの用意ができるまで、実行中の動作を停止するために待ち状態にすることができます。

割り込みリクエスト ($\overline{IRQ}$)

アクティブ・ローのオープン・コレクタ出力、内部 10kΩ でプルアップされています。ローの場合、このピンは Anybus インジケーションレジスタ (7FFh) で新しい情報が利用可能になることを示します。これはホスト・アプリケーション上でこのシグナルを実行するために特に推奨されます。

出力イネーブル ($\overline{OE}$)

ローのとき D_0 ... D_7 のデータ出力が可能です。

読み込み / 書き込み (R/$\overline{W}$)

ローのとき D_0 ... D_7 のデータ出力が可能です。内部で 10kΩ でプルアップされています。

チップ・イネーブル ($\overline{CE}$)

アクティブ・ロー入力 (ほとんどのモジュールでプルアップされます)。アプリケーション・インターフェースとの通信が可能です。$\overline{CE}$ は、“ アプリケーション・インターフェース・ハードウェアの注意 ” で記述されていない限り、DPRAM のアクセス中のみアクティブにする必要があります。内部で 10kΩ でプルアップされています。

リセット (RESET)

ローの場合、システム・リセットが開始されます。

内部で 10k - 75k でプルアップされています。また、10 - 100nF コンデンサで GND から分離されています。

非同期シリアル・インターフェース

これらのピンは一般的にファームウェア・アップグレード等のため使用されます。詳細については 12-1 “ ファームウェアの更新 ” を参照してください。

シグナルの特性については D-2 “ 信号特性 ” を参照してください。

送信データ (TxD)

非同期シリアル・データ送信シグナル。内部で 10kΩ でプルアップされています。Tx シグナルを 3,3 V から 5V に変換する Anybus モジュールには 10k レジスタがありません。信号はハイまたはローで動作します。詳細については B-2 “ アプリケーション・インターフェース・ハードウェアの注意 ” を参照してください。

受信データ (RxD)

非同期シリアル・データ受信シグナル。内部で 10kΩ でプルアップされています。

メモリ マップ

デュアル・ポート・メモリは使用に基づいていくつかの小さな領域に細分化されます。メモリ・マップは下図のようになります。

アドレス :	領域 :	アクセス :	注意 :
000h - 1FFh	入力データ領域	R/W	7-1 " フィールドバス・データ交換 " を参照してください
200h - 3FFh	出力データ領域	RO	7-1 " フィールドバス・データ交換 " を参照してください
400h - 51Fh	メールボックス入力領域	R/W	8-1 " メールボックス・インターフェース " を参照してください
520h - 63Fh	メールボックス出力領域	RO	8-1 " メールボックス・インターフェース " を参照してください
640h - 7BFh	フィールドバス個別領域	-	(個別 Fieldbus Appendix を参照してください。)
7C0h - 7FDh	制御レジスタ領域	R/W	4-1 " 制御レジスタ領域 " を参照してください
7FEh - 7FFh	ハンドシェーク・レジスタ	R/W	5-1 " ハンドシェーキング & インジケーション・レジスタ " を参照してください

これらの領域はアクセスまえに割り当てなければなりません。詳細については 5-1 " ハンドシェーキング & インジケーション・レジスタ " を参照してください。

これらの領域は直接アクセスすることができます。

注意 : アプリケーション側での A11 の使用はメモリ・マップに影響します。詳細については A-1 " 拡張されたメモリ・モード (4K DPRAM)" を参照してください。

制御レジスタ領域

この領域内には、Anybus モジュールの改訂、初期化パラメータ、フィールドバス・タイプとステータスなどの情報が含まれています。この領域はまた、ウォッチドッグやイベント通知のレジスタも含んでいます。

アドレス :	レジスタ :	アクセス :	注意 :
7C0h - 7C1h	ブート・ローダー・バージョン	RO	
7C2h - 7C3h	アプリケーション・インターフェース・ソフトウェア・バージョン[a]	RO	
7C4h - 7C5h	フィールドバス・ソフトウェア・バージョン[a]	RO	
7C6h - 7C9h	モジュール・シリアル番号	RO	独自シリアル番号
7CAh - 7CBh	ベンダ ID	RO	メーカー ID 番号 (HMS、他)
7CCh - 7CDh	フィールドバス・タイプ	RO	フィールドバス・タイプ識別子
7CEh - 7CFh	モジュール・ソフトウェア・バージョン	RO	ソフトウェア・レビジョン
7D0h - 7D1h	(リザーブド)	-	
7D2h - 7D3h	Watchdog カウンタ入力	R/W	アプリケーションに制御された Watchdog カウンタ
7D4h - 7D5h	Watchdog カウンタ出力	RO	カウンタ、各 1ms でインクリメント
7D6h - 7D9h	(リザーブド)	-	
7DAh - 7DDh	LED ステータス	RO	各フィールドバス・ステータス表示の現在のステータス
7DEh - 7DFh	(リザーブド)	-	
7E0h - 7E1h	モジュール・タイプ	RO	モジュール・タイプ、マスター、スレーブ、その他
7E2h - 7E3h	モジュール・ステータス	RO	ビット情報 ; フリーズ、クリア等
7E4h - 7EBh	更新データ・フィールド	RO	ビット・フィールド、出力データ内の変更表示 デュアル・ポート・メモリ内の領域
7ECh - 7EDh	イベント通知発生要因	R/W	イベント発生レジスタ
7EEh - 7EFh	イベント通知ソース	RO	イベント通知のためのコンフィギュレーション・レジスタ
7F0h - 7F1h	入力 I/O 長	RO	入力 I/O サイズ
7F2h - 7F3h	入力 DPRAM 長	RO	デュアル・ポート・メモリ内の入力 I/O バイト数
7F4h - 7F5h	入力合計長	RO	合計入力データ・サイズ
7F6h - 7F7h	出力 I/O 長	RO	出力 I/O サイズ
7F8h - 7F9h	出力 DPRAM 長	RO	デュアル・ポート・メモリ内の出力 I/O バイト数
7FAh - 7FBh	出力合計長	RO	合計出力データ・サイズ
7FCh - 7FDh	(リザーブド)	-	

a. アプリケーション・インターフェース側のバージョン 2.00 前のモジュールでは、このレジスタは予約済みで ’0’ です。

注意 : 一般的に、制御レジスタ領域はアクセス前にアプリケーションにより割り当てられる必要があります。ただ、モジュール初期化中は、ソフトウェアのバージョン、フィールドバス・タイプ、モジュール・タイプなど、ハンドシェークを伴わないスタティック・データを呼び出すことはできます。

レジスタ

モジュール・ブートローダー・バージョン (7C0h - 7C1h, RO)

このレジスタは、モジュール内のブートローダー・ファームウェアのバージョンを指定します。

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
7C0h	Major revision (BCD coded)								Minor revision (BCD coded)								7C1h

アプリケーション・インターフェース・ソフトウェア・バージョン (7C2h - 7C3h, RO)

このレジスタは、モジュールのアプリケーション・インターフェース・ファームウェアのバージョンを指定します。

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
7C2h	Major revision (BCD coded)								Minor revision (BCD coded)								7C3h

注意 : アプリケーション・インターフェース側のバージョン 2.00 前のモジュールでは、このレジスタは予約済みでゼロに設定されています。

フィールドバス・ソフトウェア・バージョン (7C4h - 7C5h, RO)

このレジスタは、モジュールのフィールドバス制御ファームウェアのバージョンを指定します。

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
7C4h	Major revision (BCD coded)								Minor revision (BCD coded)								7C5h

注意 : アプリケーション・インターフェース側のバージョン 2.00 前のモジュールでは、このレジスタは予約済みでゼロに設定されています。

モジュール・シリアル番号 (7C6h - 7C9h, RO)

このレジスタは、独自の 32 bit シリアル・ナンバーを含んでいます。

ベンダ ID (7CAh - 7CBh, RO)

このレジスタはモジュール製造ベンダを表示します。

ID 番号	ベンダ
0000h	(未使用)
0001h	HMS
0002h - FFFFh	OEM 顧客向けにリザーブド

フィールドバス・タイプ (7CCh - 7CDh, RO)

このレジスタは、各モジュールの対応するフィールドバス・インターフェースを指定します。

タイプ番号	フィールドバス
0001h	PROFIBUS-DP
0005h	PROFIBUS-DPV1
0010h	Interbus-S
0011h	Interbus 2Mbps (銅 & 光ファイバ)
0015h	LonWorks
0020h	CANopen
0025h	DeviceNet
0035h	FIP IO
0040h	Modbus Plus
0045h	Modbus RTU
0065h	ControlNet
0082h	Ethernet (Modbus/TCP + IT)
0083h	Ethernet (EtherNet/IP + Modbus/TCP + IT)
0084h	PROFINET
0086h	FL-net
0087h	EtherCAT
0089h	PROFINET IRT
0090h	CC-Link
0091h	AS-Interface
0093h	Ethernet (Modbus/TCP + IT) 2-port
0094h	Ethernet (EtherNet/IP + Modbus/TCP + IT) 2-port
009Dh	PROFINET IRT FO

モジュール・ソフトウェア・バージョン (7CEh - 7CFh, RO)

このレジスタは、モジュールのシステム・ファームウェアのバージョンを指定します。

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
7CEh	Major revision (BCD coded)								Minor revision (BCD coded)								7CFh

Watchdog カウンタ入力 (7D2h - 7D3h, R/W)

このレジスタは、アプリケーションが正常に動作している事をフィールドバスへ通知します。そのためアプリケーションにより Watchdog 出力の値が呼び出され、このレジスタに書き込まれます。

モジュール初期化中、これらのいくつかのレジスタが、Anybus_Init コマンドにおいて指定された値を超えた場合、モジュールはアプリケーションがフィールドバスに対して正しく動作していないことを意味します。設定値を '0' に設定することでこの機能を無効にすることができます。

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
7D2h	Counter (high byte)								Counter (low byte)								7D3h

注意 : このビットの実装と動作はフィールドバス・タイプに依存します。詳細については個別のフィールドバス・アペンディックスか、B-1 " 注意 " を参照してください。

Watchdog カウンタ出力 (7D4h - 7D5h, RO)

Anybus-S ファームウェアは、1 ミリ秒毎に増加される内部カウンタに特長があります。この内部カウンタは、Anybus モジュールが正しく動作しているということを示すために継続的に書かれます。

このレジスタのリフレッシュ時間は最大 50ms で、すなわち最大 50 リフレッシュサイクルで値が更新されます。

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
7D4h	Counter (high byte)								Counter (low byte)								7D5h

LED ステータス (7DAh - 7DFh, RO)

これらのレジスタは各フィールドバス・ステータス表示 LED の現在のステータスを 1 バイト毎に含みます。

	b7	b6	b5	b4	b3	b2	b1	b0
7DAh	Led 1 (上左)							
7DBh	Led 2 (上右)							
7DCh	Led 4 (下左)							
7DDh	Led 3 (下右)							

値	説明
00h	LED オフあるいは モジュール未使用
01h	LED グリーン[a]
02h	LED レッド[a]

a. フィールドバスによっては、Anybus-S のバージョン次第で他の色を使用することがあります。詳細については個別の fieldbus appendix を参照してください。

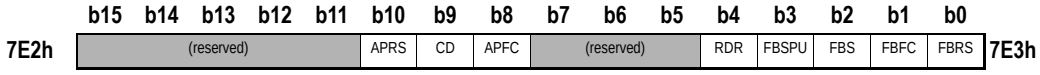
モジュール・タイプ (7E0h - 7E1h, RO)

このレジスタは、現行使用モジュールのタイプが表示されます。

タイプ番号	Module Type
0001h	Anybus DT (販売終了)
0101h	Anybus-S (Anybus-S スレーブ)
0102h	Anybus-S Drive Profile
0201h	Anybus-M (Anybus-S マスター)

モジュール・ステータス (7E2h - 7E3h, RO)

このレジスタは、様々なステータス・ビットを含みます。このレジスタの中のほとんどのビットは 'Anybus Init' メールボックス・コマンドで作成される設定に対応します。本レジスタの詳細については 9-3 "Anybus の初期化 (Anybus_INIT)" を参照してください。



- **APRS[1] (アプリケーション　ラン / ストップ)**
  このビットは Anybus_INIT において Watchdog カウンタ出力と Watchdog カウンタ入力間の差異が Watchdog タイムアウト値を超えた場合に通知されます。
  0: アプリケーション・ストップ (Watchdog タイムアウト値 超)
  1: アプリケーション動作中

- **CD (データ更新検出ステータス)**
  0: データ更新検出が無効
  1: データ更新検出が有効

- **APFC[1] (アプリケーション・ストップ　フリーズ / クリア)**
  0: アプリケーションがストップすると入力データはクリアされます
  1: アプリケーションがストップすると入力データはフリーズします

- **RDR[1] (フィールドバス・リセット機器リクエスト通知)**
  0: フィールドバス・リセット機器リクエストがありません (あるいは機能が使用されていません)
  1: フィールドバス・リセット機器リクエスト受信

- **FBS[1], FBSPU[1] & FBFC**

ビット値			フィールドバス・オフライン・アクション		注意
FBSPU[1]	FBS[1]	FBFC	出力 I/O データ	パラメータ・データ	
0	0	0	クリア	クリア	すべてのデータはフィールドバスがオフラインになるとクリアされます。
		1	フリーズ	フリーズ	すべてのデータはフィールドバスがオフラインになると現行の状態でフリーズされます。
	1	0	セット	セット	すべてのデータはフィールドバスがオフラインになると設定されます。
		1	(リザーブド)	(リザーブド)	-
1	0	0	クリア	更新	フィールドバス・システムによっては I/O データ交換が動作していなくてもフィールドバスを介してパラメータ・データが更新されることがあります。詳細については個別 fieldbus appendix を参照してください。
		1	フリーズ	更新	
	1	0	セット	更新	
		1	(リザーブド)	(リザーブド)	

- **FBRS (Fieldbus On / Off Line)**
  1: Fieldbus online
  0: Fieldbus offline

1. このビットの実装と動作は各フィールドバスに依存します。詳細については個別のフィールドバス・アペンディックスか、B-1 " 注意 " を参照してください。

更新データ・フィールド (7E4h - 7EBh, R/W)

これらのレジスタは、出力データ領域に更新データが含まれる場合、アプリケーション側により決定されるビット・フィールドより構成されます。これらのレジスタの各ビットは、出力データ領域内の 8 バイト の代数値になります。

注意 1: これらのレジスタを使用することでモジュールの全体の精度を向上させることができます。

注意 2: Anybus-M (a.k.a. Anybus-S マスター) で、このレジスタが実行されないバージョンもあります。

	b7	b6	b5	b4	b3	b2	b1	b0
7E4h	56-63	48-55	40-47	32-39	24-31	16-23	8-15	0-7
7E5h	120-127	112-119	104-111	96-103	88-95	80-87	72-79	64-71
7E6h	184-191	176-183	168-175	160-167	152-159	144-151	136-143	128-135
7E7h	248-255	240-247	232-239	224-231	216-223	208-215	200-207	192-199
7E8h	312-319	304-311	296-303	288-295	280-287	272-279	264-271	256-263
7E9h	376-383	368-375	360-367	352-359	344-351	336-343	328-335	320-327
7EAh	440-447	432-439	424-431	416-423	408-415	400-407	392-399	384-391
7EBh	504-511	496-503	488-495	480-487	472-479	464-471	456-463	448-455

イベント通知発生要因 (7ECh - 7EDh, R/W)

このレジスタは、イベント通知のソースを通知します。このレジスタ内のビットはエッジトリガ、つまりイベント発生時、モジュールにより設定されたビットになります。デフォルトでは値は '0' になります。イベント通知中、アプリケーションは対応するビットをクリアにします。

割り込みならびにイベント通知に関する詳細については 6-1 “ 割り込み ” ならびに 6-2 “ イベント通知 (ソフトウェア割り込み)” を参照してください。

- **RST**[1]
  0: -
  1: フィールドバス側からのリセット・リクエスト発行
- **FBON**
  0: -
  1: フィールドバス・オンライン
- **FBOF**
  0: -
  1: フィールドバス・オフライン
- **DC**
  0: -
  1: データ更新、4-6 “ 更新データ・フィールド (7E4h - 7EBh, R/W)” を参照してください。

1. このビットの実装と動作はフィールドバス・タイプに依存します。詳細については個別の Fieldbus appendix か、B-1 “ 注意 ” を参照してください。

イベント通知ソース (7EEh - 7EFh, RO)

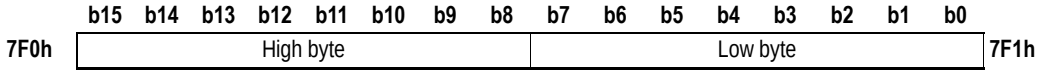
このレジスタはイベント通知のトリガとなるのはどのイベントかを通知します。このレジスタの構成は、メールボックス・コマンド 'Anybus Init' にてモジュールが初期化中に決定されます。

割り込みならびにイベント通知に関する詳細については 6-1 " 割り込み " ならびに 6-2 " イベント通知 (ソフトウェア割り込み)" を参照してください。

- **RST**[1]
  0: -
  1: イベント通知はフィールドバスから " リセット・リクエストが発行される毎に発生します。
- **FBON**
  0: -
  1: フィールドバスがオンラインになるとイベント通知が発生します。
- **FBOF**
  0: -
  1: フィールドバスがオフラインになるとイベント通知が発生します。
- **DC**
  0: -
  1: 出力領域内のデータが変更される毎にイベント通知が発生します。これにはモジュールが初期化されている間に更新データ・フィールドが有効になっていることに注意してください。

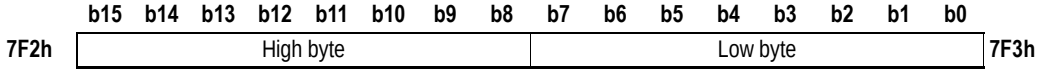
入力 I/O 長 (7F0h - 7F1h, RO)

モジュール初期化中に指定された入力 I/O 長を維持します。



入力 DPRAM 長 (7F2h - 7F3h, RO)

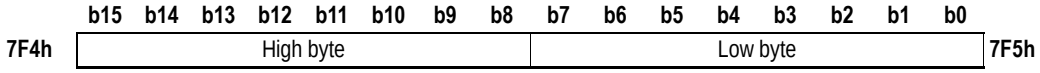
モジュール初期化中に指定された入力 DPRAM 長を維持します。



1. このビットの実装と動作はフィールドバス・タイプに依存します。詳細については個別の Fieldbus appendix か、B-1 " 注意 " を参照してください。

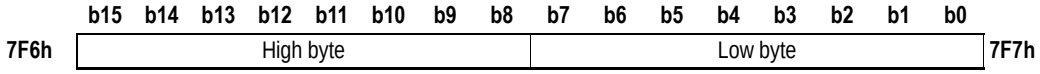
入力合計長 (7F4h - 7F5h, RO)

モジュール初期中に指定された入力合計長 を維持します。



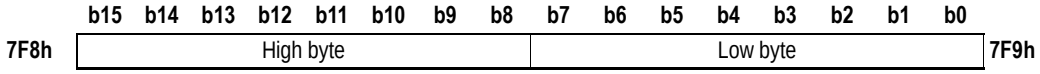
出力 I/O 長 (7F6h - 7F7h, RO)

モジュール初期化中に指定された出力 I/O 長を維持します。



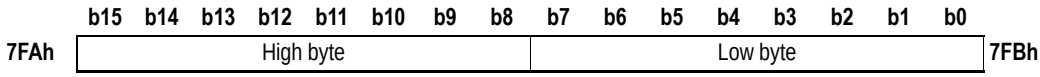
出力 DPRAM 長 (7F8h - 7F9h, RO)

モジュール初期化中に指定された出力 DPRAM 長を維持します。



出力合計長 (7FAh - 7FBh, RO)

モジュール初期化中に指定された出力合計長を維持します。

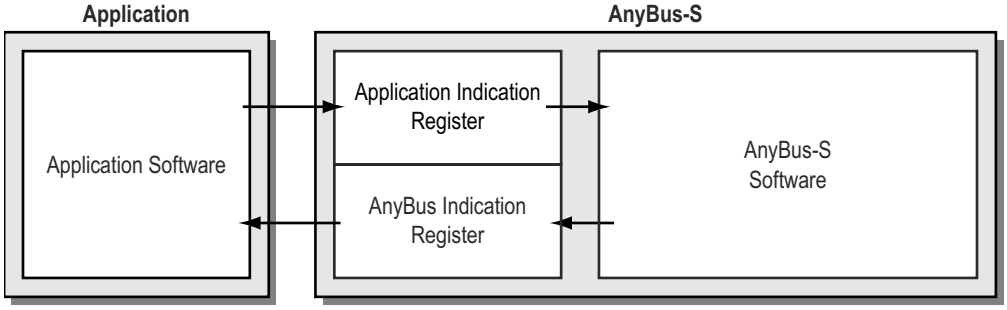


ハンドシェーキング & インジケーション・レジスタ

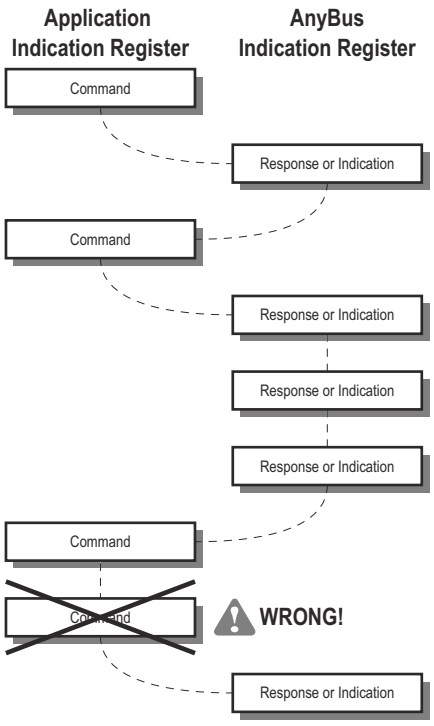
7FEh ならびに 7FFh のメモリ・ロケーションは、アプリケーション・インジケーション・レジスタと Anybus インジケーション・レジスタの、二つの重要なレジスタを維持します。これらのレジスタは 3 つの主要な役割があります。

- **領域割り当て / リリース**
  この工程は入力 -、出力 -、フィールドバス個別 - そして制御レジスタ領域にアクセスする時には必須です。詳細については 5-5 “ 領域割り当て / リリース ” を参照してください。
- **イベント通知**
  6-2 “ イベント通知 (ソフトウェア割り込み)” を参照してください。
- **メールボックス・メッセージの送受信**
  8-1 “ メールボックス・インターフェース ” を参照してください。

一般的にアプリケーション・インジケーション・レジスタはモジュール指定のローレベルな処理の実行をさせる命令（コマンド）に使用されます。Anybus インジケーション・レジスタには先に指示されたコマンドのレスポンスが含まれ、モジュールの全体のステータスを表示します。



これらのレジスタはよくコマンド - レスポンス方式によって使用されますが、レジスタは互いに独立しており、必要に応じて扱われます。しかし、アプリケーションはモジュールが前回のコマンドに対してレスポンスするまで新しいコマンドを送信することができません。右の図を参照してください。



モジュールがレスポンスあるいは指示を送信する毎に、UPDATED ビット (Anybus インジケーション・レジスタのビット 3) はトグルされ、ハードウェアの割り込みがトリガされます。アプリケーションはハードウェア割り込みの原因を検索するために Anybus インジケーション・レジスタを検査します。

ハードウェア割り込み機能がインプリメントされていなければ、アプリケーションはサイクリックに Anybus インジケーション・レジスタに変化があるかポーリングしなければなりません。しかしポーリングは不必要なオーバーヘッドを伴い一般的に実装がより難しいため、割り込み機能の使用を強く推奨します。

アプリケーション・インジケーション・レジスタ (7FEh, R/W)

このレジスタは次のタスクを実行するために使用されます。

- 領域割り当て / リリース (リクエスト / リリース)
- イベント承認 (イベント通知)
- メールボックス・メッセージの送受信

レジスタは次のレイアウトのビット・フィールドで構成されています。

b7	b6	b5	b4	b3	b2	b1	b0
AP_M_{IN}	AP_M_{OUT}	AP_EVNT	ACTION	LOCK	AP_IN	AP_OUT	AP_FBCTRL

ビット	機能	説明	デフォルト
AP_M_{IN}	メールボックス通知	メールボックス・メッセージを送信するために使用。8-2 " メールボックス通知ビット " を参照してください。	0
AP_M_{OUT}		受信したメールボックスメッセージがある事を知らせるために使用されます。8-2 " メールボックス通知ビット " を参照してください。	0
AP_EVNT	イベント承認	本ビットはイベント通知イベントが処理されたことを知らせるためにトグルすべきです。6-2 " イベント通知 (ソフトウェア割り込み)" を参照してください。	0
ACTION	領域リクエスト / リリース これらのビットはデュアル・ポート・メモリ内に 1 つあるいは複数の領域のリクエスト / リリースするために使用されます。 詳細については 5-5 " 領域割り当て / リリース " を参照してください。	このビットは現在のアクションがリクエストかリリースかを表示します。 1: リクエスト 0: リリース	0
LOCK		このビットはアクションがロックされているかいないかを表示します。 1: ロックされている 0: ロックされていない	0
AP_IN		このビットは入力データ領域を表します。 1: アクションはこの領域に影響します。 0: アクションはこの領域に影響しません。	0
AP_OUT		このビットは出力データ領域を表します。 1: アクションはこの領域に影響します。 0: アクションはこの領域に影響しません。	0
AP_FBCTRL		このビットはフィールドバス個別領域と制御レジスタを表します。 1: アクションはこの領域に影響します。 0: アクションはこの領域に影響しません。	0

注意 1: このレジスタにサイクリックにアクセスすることは推奨しません。入力されるイベントへのレスポンスとリクエストのときにのみ使用されるべきです。

注意 2: このレジスタにアクセスする場合、シングル・メモリ・アクセスを使用し希望する操作に関連する全てのビットを変更することが重要です。そうでなければ、操作はモジュールによる複数の操作として誤解される可能性があります。

Anybus インジケーション・レジスタ (7FFh, RO)

このレジスタには先に示されたコマンドへの応答を含みモジュールステータスを表示します。以下の情報がこのレジスタから抽出できます。

- デュアル・ポート・メモリ内の異なる領域のオーナーシップ
- イベント通知ステータス
- メールボックス受信 & 送信ステータス
- 初期化ステータス

このレジスタの各変更されるとハードウェア割り込みがトリガされます。アプリケーションは割り込みの原因を検索するためにレジスタの構成を検査します。

ハードウェア割り込み機能がインプリメントされていなければ、アプリケーションはサイクリックに Anybus インジケーション・レジスタを変更の検索のためにポーリングしなければなりません。しかしポーリングは不必要なオーバーヘッドを伴い一般的に実装がより難しいため、割り込み機能の使用を強く推奨します。(割り込みの詳細については 6-1 “ハードウェア割り込み (IRQ)” を参照してください。)

b7	b6	b5	b4	b3	b2	b1	b0
AB_M_{IN}	AB_M_{OUT}	AB_EVNT	INIT	UPDATED	AB_IN	AB_OUT	AB_FBCTRL

ビット	機能	説明	デフォルト
AB_M_{IN}	メールボックス通知	モジュールはメールボックス・メッセージを受信した事を知らせるためにこのビットをトグルします。8-2 “メールボックス通知ビット” を参照してください。	0
AB_M_{OUT}		モジュールはこのビットをメールボックス送信領域で新しいメッセージが送信可能な事を示すためにトグルします。8-2 “メールボックス通知ビット” を参照してください。	0
AB_EVNT	イベント通知	モジュールは新しいイベント通知が発生するとこのビットをトグルします。	0
INIT	モジュール初期化	モジュールが初期化されるとこのビットが設定されます。 0: モジュールは初期化されていません。 1: モジュールは初期化されています。[a]	0
UPDATED	レジスタ更新	このビットは本レジスタの内容が変更される度にモジュールによりトグルされます。	0
AB_IN	これらのビットはデュアル・ポート・メモリ内の各領域のオーナーを表示します。	このビットは入力データ領域を表しています。 1: 領域はアプリケーションにより所有されています 0: 領域は Anybus により所有されています	0
AB_OUT		このビットは出力データ領域を表しています。 1: 領域はアプリケーションにより所有されています 0: 領域は Anybus により所有されています	0
AB_FBCTRL		このビットはフィールドバス個別領域と制御レジスタを表します。 1: 領域はアプリケーションにより所有されています 0: 領域は Anybus により所有されています	0

a. フィールドバス・システムによっては、このフラグはフィールドバスが完全に初期化されデータ交換されるということを示さないことに注意してください。詳細については個別の fieldbus appendix を参照してください。

注意 : このレジスタはリードオンリーです。このレジスタに書き込みはしないでください。予測不可能な結果を引き起こします。

衝突

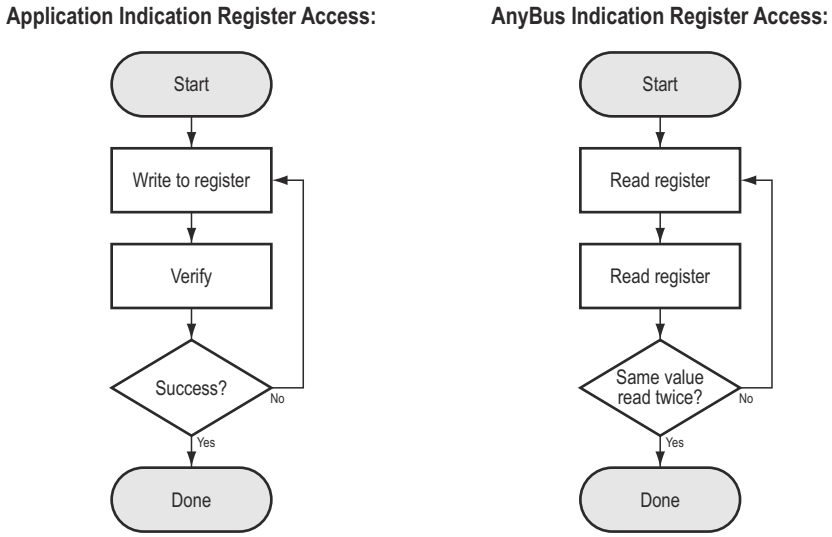
前述した通り、アプリケーション・インターフェースはデュアル・ポート・メモリ・アーキテクチャをベースにしていて同時に同じメモリ位置に両側でアクセスすることができます。

領域割り当ての仕組み (5-5 “ 領域割り当て / リリース ” を参照してください。) は動作中の衝突を避けるために使用されます。しかしながらこれは Anybus やアプリケーション・インジケーション・レジスタで生じる衝突イベントをカバーしません。もしこれが正しく扱われなければモジュールはアプリケーション・インジケーション・レジスタに書き込まれたデータを得ずにアプリケーション・インジケーション・レジスタから読まれたデータは古いか正しくないかもしれません。衝突の対処方法は 3 通りあります。(重要な順番に並んでいます。)

1. **複数の書き込み / 呼び出し (必須)**

   IRQ と BUSY シグナルが衝突を避けるために使用されている間（下図を参照してください)、実際これが*取り扱い*方法です。

   - アプリケーション・インジケーション・レジスタ に書き込みをしている時 ...
     ... 書き込まれたデータが正しいと確認できるまで書き込みと検証を続けます。
   - アプリケーション・インジケーション・レジスタから読み込んでいる時 ...
     ... 2 つの読み込みが同じになるまで読み込みと比較を続けます。



   注意 : はじめは前述の 5-1 ページならびに 5-2 ページの注意 1 に違反しているように見えますが、アクセスは高速シーケンスで実行されるのでモジュールは一回のアクセスとしか解釈しません。

2. **アプリケーションへの $\overline{\text{IRQ}}$ シグナルのインプリメント (オプション、推奨します)**

   Anybus インジケーション・レジスタが更新される毎にここの信号はローに遷移します。アプリケーションはこの割り込みを受信すると即座にこのレジスタにアクセスすることで、衝突を避けます。

3. **アプリケーションへの $\overline{\text{BUSY}}$ シグナルのインプリメント (オプション、推奨します)**

   このシグナルがローのままであればアプリケーションは、例えば待ちステータスを挿入するなど、シグナルがハイになるまでメモリ・アクセスを停止して衝突を避けます。(すべてのアーキテクチャに可能ということではありません。)

一般推奨

望ましいのはこれら3つのオプションをすべてアプリケーションにインプリメントすることです。どちらにしてもオプション 1 は必須です。少なくとも他のうち 1 つはインプリメントすることを強く推奨します。

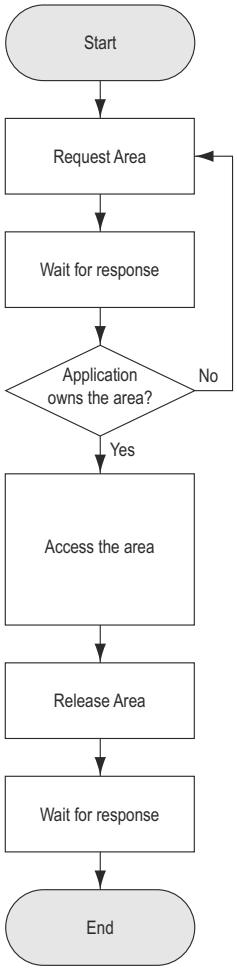
領域割り当て / リリース

前述した通り、デュアル・ポート・メモリが機能をベースにいくつかの領域に分割されています。衝突を避けるため、そしてデータの統一を保証するためアプリケーションは、アクセス前に各領域を割り当てます。領域が使用できればモジュールは、Anybus インジケーション・レジスタでレスポンス・ビットを設定することでこれをアプリケーションに通知します。領域はアプリケーションにより「所有された」とみなされ、自由にアクセスすることができます。終了時、領域は Anybus モジュールに解除すなわちリリースされなければなりません。

この割り当ては次の領域にアクセスするときこの割り当て手順が必要です。

- 入力データ領域
- 出力データ領域
- フィールドバス個別領域
- 制御レジスタ領域

アプリケーションは最大 1000ms 間領域を所有します。この時間を超えると、すなわちアプリケーションが時間内に領域をリリースしないと、割り当ては自動的に終了します。領域の所有権は Anybus モジュールに戻されます。アプリケーション内でソフトウェア・ルーチンがこれを認識し、安全にアクセスを終了することが重要です。



非同期データ交換

最も単純なフォームにおいて、モジュールに対するアクセス・シーケンスは次のステップで構成されます。（右の図を参照ください）

1. **領域へのアクセス・リクエスト**
   これを遂行するためには、アプリケーションはアプリケーション・インジケーション・レジスタ内で ACTION ビットだけでなく、その領域に該当するビットを設定しなければなりません。
2. **レスポンス待ち**[1]
3. **アクセス成功かどうかを見るためのレスポンスチェック**
   要求が成功したか知るためには、アプリケーションが Anybus インジケーション・レジスタ内でオーナーシップ領域をチェックしなければなりません。希望する領域がアプリケーションに所有されていれば、アプリケーションは自由に領域にアクセスすることができます。
4. **領域のリリース**
   これをするためにはアプリケーションは、アプリケーション・インジケーション・レジスタ内で ACTION ビットだけでなくその領域に該当するビットを設定しなければなりません。
5. **レスポンス待ち** [1]

1. 割り込み待ちか Anybus インジケーション・レジスタをポーリングするかで行うことができますが、アプリケーションの実装の仕方に依存します。詳細については 5-3 “Anybus インジケーション・レジスタ (7FFh, RO)” と 6-1 “ 割り込み ” を参照してください。

同期されたデータ交換

先に記述された手順は非同期データ交換です。しかしながらいくつかのケースではアプリケーションと Anybus-S モジュール間でデータ交換が同期した方が望ましいものもあります。

アプリケーション・インジケーション・レジスタでは LOCK ビットはこの目的のために使用されます。下のテーブルを参照してください。

アクション	LOCK	結果
リクエスト	0	使用している領域がリクエストされるとアプリケーションはアクセスが認められるまでリクエストを繰り返さなければなりません。
	1	もし、要求領域が使用されている場合、モジュールはモジュール自身がまだ使用中であることを示すレスポンスを返します。領域が自由になるとすぐに領域のオーナーシップをアプリケーションに引き渡します。（つまり領域はアプリケーションに所有されます。）
リリース	0	領域はリリースされています
	1	領域はリリースされ、Anybus モジュールのためにリザーブされています。アプリケーションはモジュールが領域へ再びアクセス する前にアクセスできません。

ロックリクエスト

ロックリクエストはアプリケーションが、領域が自由になるとすぐアクセスできるようにするのを確実にします。

1. **領域へのアクセス・リクエスト (LOCK = 1)**
2. **最初のレスポンス待ち**[1]
3. **領域のオーナーシップ・チェック**
   アプリケーションに所有された領域 - 領域は自由に使用できます。ステップ 6 へ
   Anybus に所有された領域 - 領域は現在使用中です。ステップ 4 へ
4. **追加レスポンス待ち** [1]
5. **領域のオーナーシップ・チェック**
   領域のオーナーシップはアプリケーションに渡されます。[2]
6. **終了**

1. これは割り込みを待つか Anybus インジケーション・レジスタをポーリングすることで行うことができますが、アプリケーションのインプリメントに依存します。詳細については 5-3 “Anybus インジケーション・レジスタ (7FFh, RO)” と 6-1 “ 割り込み ” を参照してください。
2. 複数の領域が同時にリクエストされた場合、モジュールは各領域に 1 つ、最多で 3 つの追加レスポンスを送信します。これらの場合、イベントのシーケンスは上に記述された内容とは多少異なります。詳細については 5-7 “ 複数領域の同時リクエスト / リリース ” を参照してください。

ロックリリース

いくつかのケースにおいては、Anybus が最初に領域にアクセスしなければアプリケーション側からその領域にアクセスする事が無意味なこともあります。例えばアプリケーションが、データが各ポーリング間で Anybus モジュールによりデータが更新されたという事が確認できなければ、アウトプット・データ領域にサイクリックにポーリングする意味がありません。

領域をリリースするときに LOCK ビットを使用することにより、アプリケーションは Anybus モジュールのために領域をリザーブすることができます。すなわちアプリケーションがデータを更新するまで領域にアクセスしません。[1]

1. **領域のリリース (LOCK = 1)**
2. **レスポンス待ち**[2]
3. **終了**
   領域は Anybus モジュールのためにリザーブされました。すなわちアプリケーションはモジュールがデータを更新するまで領域にアクセスしません。

複数領域の同時リクエスト / リリース

複数の領域は同時にリクエスト / リリースすることができます。これを行う時、Anybus インジケーション・レジスタでどの領域が実際使用できるか（そのリクエストのとき使用されている領域は 1 つかもっとか）を見るためにレスポンスをモニタすることが重要です。これはかなりアンロックリクエストに近いですが、複数の領域向けにロックリクエストを行う時には特別な注意を払わなければなりません。

- **複数の領域リクエスト (LOCK = 0)**
  モジュールは 1 つのレスポンスを送信します。
- **複数の領域リクエスト (LOCK = 1)**
  モジュールは理論的には場合によっては最大 4 レスポンスを送信するかもしれません :
  - 最初のレスポンス
  - 領域がリリースされ、リクエストされた領域のオーナーシップがアプリケーションに移行するとき、最大 3 つまで追加のレスポンスが送信されます。
- **複数の領域のリリース (LOCK = 0)**
  指定領域は即座にリリースされます。モジュールはリリースを確認するために 1 つのレスポンスを送信します。B-1 “ ロックリリースの動作 ”
- **複数領域のリリース (LOCK = 1)**
  指定の領域は即座にリリースされます。モジュールはリリースを確認するために 1 つの信号を送信します。領域は Anybus モジュールのためにリザーブされ、アプリケーションは Anybus がデータを更新するまでオーナーシップを得ることができません。

---

1. 出力データが変更された時のみ出力データ領域を更新するモジュールもあります。そのためデータ内で外部イベントが変更されるまで出力データ領域がロックされたままという状態が起こりうることに注意してください。詳細については B-1 “ ロックリリースの動作 ” を参照してください。
2. アプリケーションのインプリメントに依存して割り込みを待つか Anybus インジケーション・レジスタをポーリングするかになります。詳細については 5-3 “Anybus インジケーション・レジスタ (7FFh, RO)” と 6-1 “ 割り込み ” を参照してください。

アプリケーション事例、サイクリック・アクセス方法

この例では DPRAM 出入力データ領域へのサイクリック・アクセス・リクエストのアプリケーションの 1 つの方法を記述しています。

背景

よりよくご理解いただくためにモジュール内部で何が起こっているかについて記述します。

アプリケーションが DPRAM 入力データ領域（フィールドバスへのデータ）をリリースする時、その領域を制御するためにモジュールをトリガします。領域はそのタスクが終了するまでモジュールに所有されます。かかる時間はコンフィグレーションにより変化します。一度終了するとモジュールは次にアプリケーションが領域をリリースするまで領域をリクエストしません。アプリケーションはこのようにして領域をロックされた状態でリリースしたあと即座にその領域へのロックリリースを行います。そして次にモジュールがそれをリリース（ロックリリースで）した時即座にその領域ロックリクエストを行います。待っている間、合計時間が 1000 ms を超えない限り他のタスクを実行することができます。

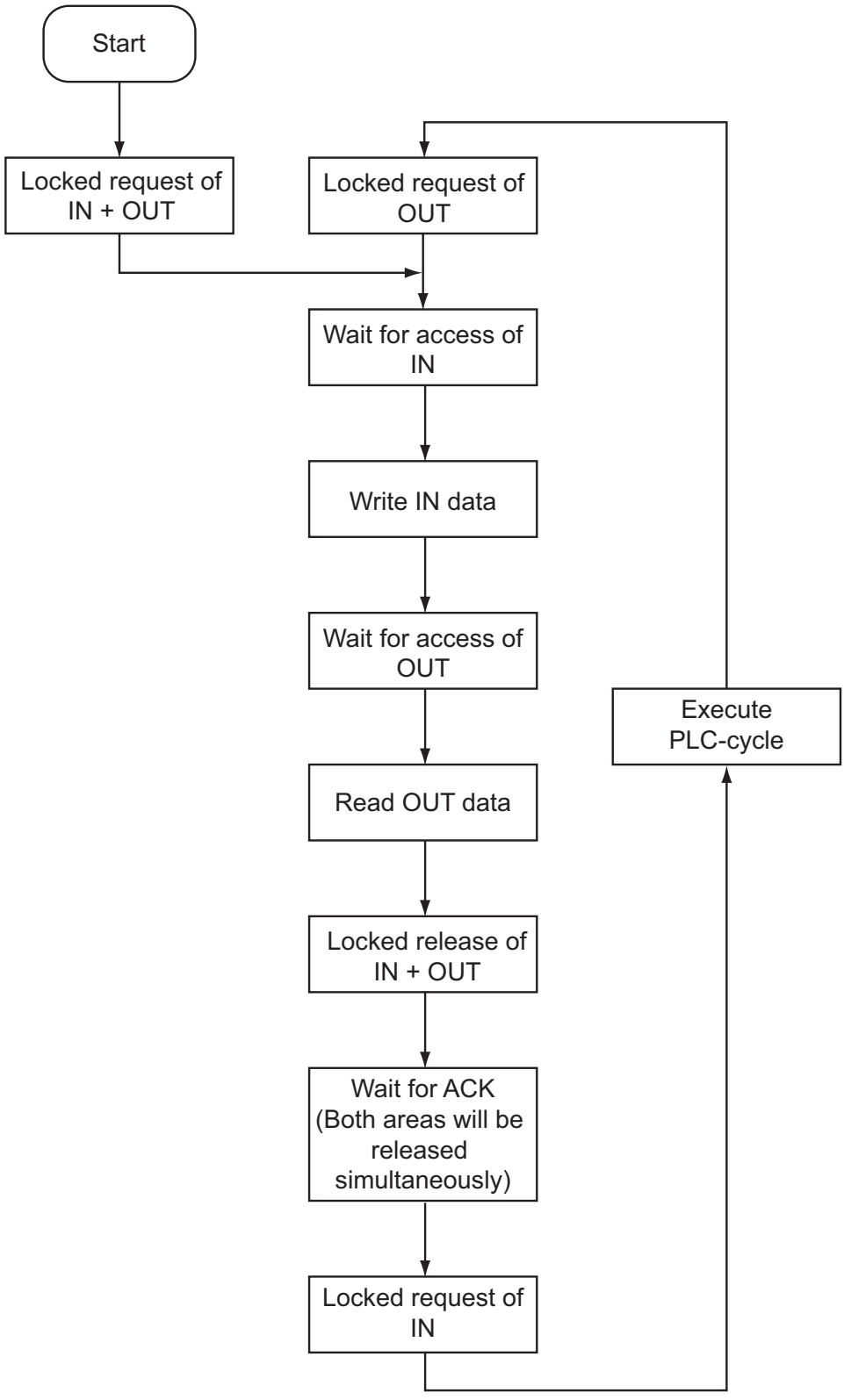
内部 OUT 出力 I/O データ・バッファはモジュールにより継続的に更新されています。すべてのスレーブからのデータが更新された時、モジュールは DPRAM 出力データ領域（フィールドバスからのデータ）へのアクセスをリクエストします。バッファから DPRAM へのデータをコピーして出力データ領域へリリースします。そのプロセスは繰り返されます。モジュールの DPRAM 出力データ領域へのアクセスはこのようにアプリケーションのその領域のアクセスに関連しません。最新呼び出しデータを確かなものにするためにアプリケーションは実際に読み出しが必要な時だけDPRAM 出力データ領域のリクエストを行わなければなりません。

提案されるサイクリック・アプリケーション向けアクセス方法

1. OUT のロックリクエスト
2. IN のアクセス待ち (前のサイクルでリクエスト)
3. IN データ書き込み
4. OUT のアクセス待ち[1]
5. OUT データ読み込み
6. IN + OUT のロックリリース
7. ACK 待ち (両領域は同時にリリースされます)
8. IN のロックリクエスト
9. PLC サイクルの実行
10. 1 の繰り返し

注意 : このループは最初の DPRAM 入出力領域のロックリクエスト後はステップ 2 に入る必要があります。次ページのフローチャートを参照してください。

1. 出力データに変更があった場合のみ出力データ領域を更新するモジュールもあります。そのためデータ内に外部データが変更されるまで出力データ領域がロックされたままということも起こり得ます。詳細については B-1 “ ロックリリースの動作 ” を参照してください。



IN : DPRAM 入力データ領域 (フィールドバスへのデータ)

OUT : DPRAM 出力データ領域 (フィールドバスからのデータ)

サイクリック・アクセス方法フローチャート

割り込み

ハードウェア割り込み (IRQ)

モジュールは割り込み要求ピンがあります (IRQ, ピン番号 28) 。インプリメントされるとAnybus インジケーション・レジスタのどんな変更でもアプリケーションに通知します。一般的にこの機能はレジスタをサイクリックにポーリングする事に比べて、顕著にオーバーヘッドを減らすことができ、推奨されています。

次のイベントによりハードウェア割り込みが生成されます。

- イベント通知 (ソフトウェア割り込み、下記を参照してください)
- メールボックス通知 (8-2 “ メールボックス通知ビット ” を参照してください)
- モジュールの初期化 (5-3 “Anybus インジケーション・レジスタ (7FFh, RO)” を参照してください)
- スタートアップ割り込み (10-2 “ スタートアップ・シーケンス ” を参照してください)
- 領域アロケーションレスポンス (5-5 “ 領域割り当て / リリース ” を参照)

一度アプリケーションが Anybus インジケーション・レジスタ構成を呼び出すと割り込みは自動的に解除されます。

イベント通知 (ソフトウェア割り込み)

イベント通知はアプリケーションへのイベント・シグナルのためのメカニズムです。次のイベントがイベント通知で生成することができます。

- フィールドバス・オン / オフライン
- フィールドバス・リセット・リクエスト
- データ変更[1]

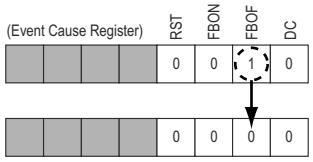
上記イベントは、モジュールが初期化されている間に 'Anybus Init' メールボックス・メッセージでのイベント通知コンフィギュレーション・パラメータで設定されなければなりません。

新しいイベントが発生するとシグナルを送るためにモジュールは Anybus インジケーション・レジスタのビット 5 をトグルします。イベント通知発生要因レジスタにおいて呼び出せるイベントの原因は 4-6 " イベント通知発生要因 (7ECh - 7EDh, R/W)" を参照してください。アプリケーションはこのレジスタの該当するビットをクリアし、アプリケーション・インジケーション・レジスタのビット 5（AP_EVNT）をトグルする事でイベントを承認します。

アプリケーションがイベントを未確認中はすべての新しいイベントはモジュール内に順番に並べられます。これはイベントが見落とされる危険を削除します。モジュールは AB_INIT と AP_INIT が等しければ新しいイベントを通知するために AB_EVNT をトグルするだけです。一度アプリケーションが応答して、AP_EVNT を AB_EVNT と同じ値にトグルすると、モジュールはキューの次のイベントを通知することができます。イベントが確認されなければアプリケーションはアプリケーション・インジケーションのビット 5（AP_EVNT）をトグルしないことが重要です。もし AB_EVNT と AP_EVNT が等しくなければモジュールは新しいイベントの通知を禁止します。

次の例は新しいイベントが発生したらどのように確認してどのように対処するかを記述したものです。

1. AB_EVNT |= AP_EVNT?
2. もし yes ならイベントが発生しました。(もし no であれば以下のステップはスキップします。)
3. イベントを発生させたものを発見するためにイベント発生レジスタを検査します。この時フィールドバスはオフラインになっています。(FBOF)
4. イベント発生レジスタの FBOF ビットをクリアにします。[2]
5. フィールドバス・オフライン・イベントに関連した処理を行います。[2]
6. イベントを確認するためにアプリケーション・インジケーション・レジスタで AP_EVNT をトグルします。

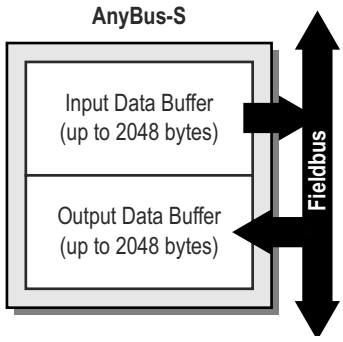


注意 : イベント通知はソフトウェア割り込みです。前述のハードウェア割り込みと混同しないでください。しかし Anybus インジケーション・レジスタを使用しているのでハードウェア割り込みをトリガします。

1. データ変更イベント通知は内部メモリにマップされた I/O データでは働きません。
2. これはフィールドバス個別 / 制御レジスタ・領域がアプリケーションで所有されている事を必要としません。

フィールドバス・データ交換

基本



モジュールは 2 つのデータ・バッファを介してフィールドバス上でデータを交換します :

- 入力データ・バッファ
  このバッファに書き込まれたデータはフィールドバスへ送信されます。
- 出力データ・バッファ
  このバッファはフィールドバスから受信したデータを含みます。

基本的にはフィールドバス上でデータ交換するためにすべてのアプリケーションがしなければならないことはこれらの2つのバッファ間のデータを読み込み/書き込むことです。

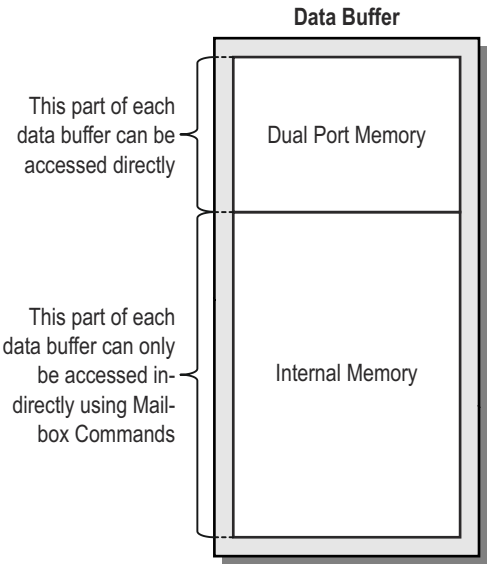
注意 : データ・バッファの容量と構成はモジュール初期化時に決定されます。従ってモジュールが初めに正しく初期化されていなければモジュールはデータ交換を行うことができません。詳細については 10-1 “ スタートアップと初期化 ” を参照してください。

デュアル・ポート・メモリ vs 内部メモリ

2 つのデータ・バッファはそれぞれ各デュアル・ポート・メモリ内に保存されたデータ部分を持つことができます。残りは内部メモリに置かれます。デュアル・ポート・メモリ内にデータが置ける利点は内部メモリ内に置かれたデータよりも高速にアクセスできることです。

内部メモリはメールボックスを介して間接的にアクセスすることができます。そのため、タイムクリティカルでないデータに適しています。

デュアル・ポート・メモリ内にどれだけのデータを置くか、また内部メモリ内にどれだけのデータを置くかは設定することができます。各データ・バッファの最大容量は 2KB で、そこから最大 512[1] バイトまでデュアル・ポート内に設定することができます。



注意 : Data change event notification (software interrupt) can not be used for I/O data mapped to the Internal Memory.

データ・タイプ

ほとんどのフィールドバス・システムは高速サイクリック・データとより低速なパラメータ・データを区別します。これは Anybus-S モジュールのデータの扱い方に影響します :

- **I/O データ**
  このタイプのデータは通常高速フィールドバス・データ（サイクリック・データ）と関連します。
- **パラメータ・データ**
  このタイプのデータは通常低速フィールドバス・データ（アサイクリック・データ）に関連します。

このデータがいかに各フィールドバス・タイプ向けに扱われるかは個別 fieldbus appendix を参照してください。

1. 今後 Anybus はより多いデータをデュアル・ポート・メモリに置くことができるかもしれません。A-1 “ 拡張されたメモリ・モード (4K DPRAM)” を参照してください。

データ構成

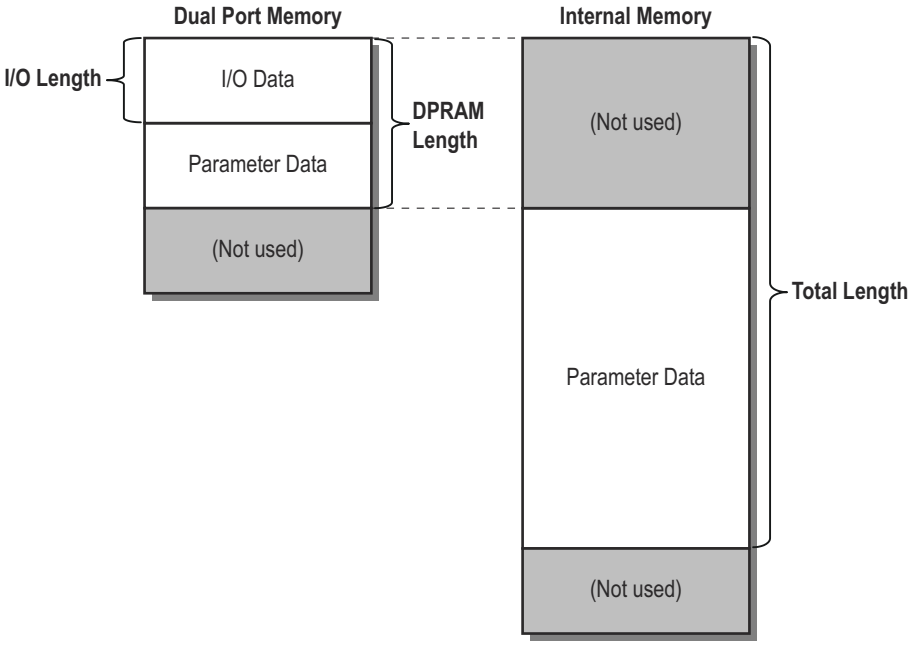
前にも記述した通り、各データ・バッファの最大容量は計 2048 バイトです。そのうち 512 [1] バイトまでをデュアル・ポート・メモリ内に置くことができます。残りは内部メモリ内に置かれ、メールボックス・コマンドを使用して非直接的にアクセスすることができます。

I/O とパラメータ・データの構成はメールボックス・コマンド 'Anybus Init' でモジュールの初期化時に決定されます。9-3 "Anybus の初期化 (Anybus_INIT)" を参照してください。

次のパラメータは各（入力と出力）データ・バッファのために設定されなければなりません。

- **合計長[2]**
  このパラメータはデータ・バッファ向けフィールドバス（I/O データ＋パラメータ・データ）の総合計を定義します。最大合計長は DPRAM と I/O 長の設定を無視しても 2048 バイトです。
- **DPRAM 長 [2]**
  このパラメータはデュアル・ポート・メモリ内にあるデータ量を定義します。DPRAM 長は 512 バイト [1] を超えることはできません。
- **I/O 長 [2]**
  このパラメータはI/Oデータとしてどれくらいのデータを取り扱うべきかを定義します。残りのデータはパラメータ・データとして扱われます。

下図は上記パラメータ間の関係を表したものです。



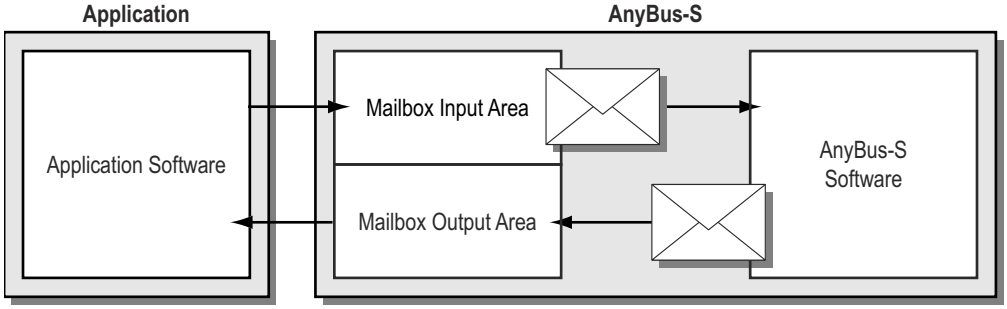
1. 将来 Anybus-S はデュアル・ポート・メモリ内により多くのデータを置くことができるかもしれません。A-1 " 拡張されたメモリ・モード (4K DPRAM)" を参照してください。
2. これらのパラメータの範囲はフィールドバスで制限されているかもしれません。詳細については個別の fieldbus appendix を参照してください。

メールボックス・インターフェース

概論

メールボックス・インターフェースはモジュールに指定のタスクの命令やデータのリクエストをするときに使用します。また、モジュールによるイベントの通知やアプリケーションに送信されたリクエストにレスポンスするために使用されます。

メールボックス通信はメールボックス入出力領域（下図を参照してください）を通して処理され、一般的にはメールボックス・メッセージ自体がフィールドバスの動作に関連しなければフィールドバス・データ交換に影響しません。

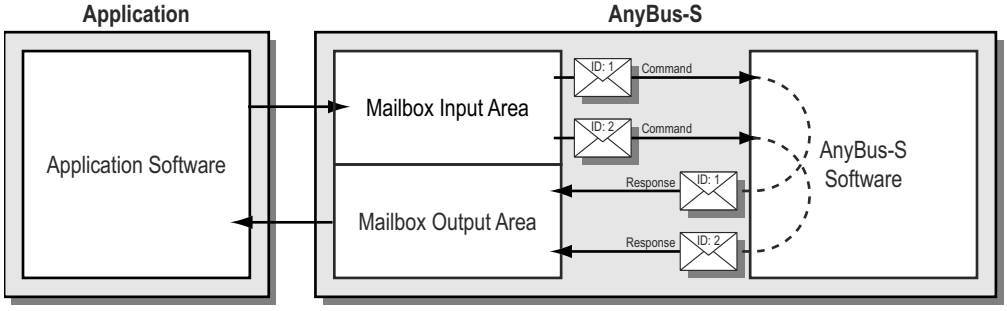


メールボックス入出力領域のハンドシェーキング手順は他の領域のものと少し異なります。詳細については 8-2 “ メールボックス通知ビット ” を参照してください。

メールボックス・インターフェースは次のタイプの通信をサポートしています。

- コマンド – レスポンス
  メッセージはメッセージ・イニシエータにより送信され、メッセージの受信にはレスポンスが要求されます。メッセージ・イニシエータはアプリケーションあるいは Anybus-S になり得ます。
- インジケーション
  メッセージはメッセージ・イニシエータにより送信されます。レスポンスは要求されません。メッセージ・イニシエータはアプリケーションあるいは Anybus-S になり得ます。

メールボックス・インターフェースは（適用可能であれば）レスポンスを受信する前に複数のメッセージをモジュールに送信できるようにデザインされています。どのメールボックスのレスポンスがどのコマンドに属するか区別するために独自の ID が各メッセージ / レスポンスに使用されています。下図を参照してください。



メッセージ・タイプ

メールボックス・メッセージはその機能性により 5 つのカテゴリにグループ化することができます。下記を参照してください。

- アプリケーション・メッセージ
  このカテゴリには内部Anybus-S機能にアクセスし制御するためのコマンドが含まれています。
- フィールドバス個別メッセージ
  このカテゴリにはフィールドバス個別のデータや機能にアクセスするためのコマンドが含まれています。詳細についてフィールドバス個別のアペンディックスを参照してください。
- 内部メモリ・メッセージ
  このカテゴリには内部メモリにアクセスするための機能が含まれています。
- リセット・メッセージ
  このカテゴリにはモジュール操作に影響する機能が含まれています。

メールボックス通知ビット

Anybus ーならびにアプリケーション・インジケーション・レジスタのメールボックス通知ビットはメールボックス・インターフェースを制御するために使用されます。両レジスタにもメールボックス・メッセージを送受信するためのビットが含まれていて、現在のメールボックスのステータスをモニタすることができます。

ビット	機能
AP_M_{IN}	メールボックス入力領域にその前に書き込まれたメッセージを送信するためにこのビットをトグルします。
AP_M_{OUT}	メールボックス・メッセージが読まれたことを確認するためにこのビットをトグルします。
AB_M_{IN}	メールボックス・メッセージが読まれたとき Anubus モジュールによりこのビットがトグルされます。
AB_M_{OUT}	メールボックス出力領域で新しいメッセージが待機する毎にこのビットがトグルされます。

メールボックス入力領域へ新しいメールボックス・メッセージを入れる前、あるいはメールボックス出力領域からのメッセージを読む前に、メールボックス・インターフェースの現在のステータスを知る必要があります。これは Anybus- やアプリケーション・インジケーション・レジスタ内のメールボックス通知ビットへのレスポンスを比較して行うことができます。下記テーブルを参照してください。

表記	結果	説明
$AP\_M_{IN} = AB\_M_{IN}$	True	メールボックス入力領域がフリーです
	False	メールボックス入力領域は現在使用中です
$AP\_M_{OUT} = AB\_M_{OUT}$	True	メールボックス出力領域では使用可能なメッセージはありません
	False	メールボックス出力領域で新しいメッセージが使用可能です

メールボックス・メッセージを送信する

モジュールにメールボックス・メッセージを送信するためには次の手順で行います。

■ スタート

1. メールボックス入力領域が空いているかどうかチェックします
（空いていなければ後で再度トライします）
2. メールボックス入力領域へのメッセージの書き込み
3. アプリケーション・インジケーション・レジスタ（７ FE h）の AP_M_{IN} ビットをトグルする

■ 終了

メールボックス・メッセージを受信する

メールボックス・メッセージを受信するためには次の手順に従います。

■ スタート

1. メールボックス出力領域にメッセージが待っていないかチェックする
(もしなければ後で再度トライする)
2. メールボックス出力領域からのメッセージの呼び出し
3. アプリケーション・インジケーション・レジスタ（7FEh）の AP_M_{OUT} ビットをトグルする

■ 終了

メールボックス・メッセージ・ストラクチャ

メールボックス・メッセージはメッセージ・ヘッダとメッセージ・データで構成されています。下記を参照してください。

オフセット：	内容：	説明：
000h - 01Fh	メッセージ・ヘッダ (32 バイト)	8-4 " メッセージ・ヘッダ " を参照してください
020h - 11Fh	メッセージ・データ (最大 256 バイトまで)	この部分にはメールボックス・メッセージに関連するデータが含まれています。

メッセージ・ヘッダ

ヘッダはメッセージのタイプとメッセージ・データの長さを指定する一連の 16 ビットレジスタで構成されています。

オフセット：	レジスタ：
000h	メッセージ ID
002h	メッセージ・インフォメーション
004h	コマンド番号
006h	データ・サイズ
008h	(リザーブド、0001h に設定)
00Ah	(リザーブド、0001h に設定)
00Ch	(リザーブド、0000h に設定)
00Eh	(リザーブド、0000h に設定)
010h	拡張ワード 1
012h	拡張ワード 2
014h	拡張ワード 3
016h	拡張ワード 4
018h	拡張ワード 5
01Ah	拡張ワード 6
01Ch	拡張ワード 7
01Eh	拡張ワード 8

メッセージ ID

メッセージ ID レジスタはコマンドのために、16 ビット整数の識別子が含まれます。レスポンスがメッセージ送信先に送信される時は同じメッセージ ID がそのメッセージで使用されます。メッセージ ID は任意に選択できますが、一連のメッセージは全てユニークな ID を持つ必要があります。

メッセージ・インフォメーション

このレジスタはメールボックス・メッセージについてのビットならびにコード情報を含んでいます。レジスタは下図に従って 5 つの領域に分割されます。

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ERR	C/R	(reserved)		Error Code				Message Type							

ビット / フィールド	説明	内容
ERR	このビットはエラーを含むコマンドを受信すると表示されます。	0: メッセージ OK 1: エラー (下記 ' エラー・コード ' も参照してください)
C/R	このビットはメッセージがコマンドかレスポンスかを指定します。	0: レスポンス・メッセージ 1: コマンド・メッセージ
Error Code	ERR ビットが設定されるとこのフィールドにはエラーについての追加情報が含まれます。	0h: 無効メッセージ ID 1h: 無効メッセージ・タイプ 2h: 無効コマンド 3h: 無効データ・サイズ 4h: メッセージ・ヘッダ生成 (オフセット 008h) 5h: メッセージ・ヘッダ生成 (オフセット 00Ah) 6h: メッセージ・ヘッダ生成 (オフセット 00Ch - 00Dh) 7h: 無効アドレス 8h: 無効レスポンス 9h: フラッシュ・コンフィグ・エラー Fh: 無効その他 (すべての他の値はリザーブド)
Message Type	このフィールドはメッセージのタイプを指定します。	1h: アプリケーション・メッセージ 2h: フィールドバス個別メッセージ 3h: メモリ・メッセージ 5h: リセット・メッセージ (すべての他の値はリザーブされています。)

コマンド番号

このレジスタは 16 ビットのコマンド識別子を含んでいます。

データ・サイズ

このレジスタはメッセージ・データのサイズをバイトで指定しています。最大メッセージ・データ・サイズは 256 バイトです。

拡張されたワード 1 ... 8

これらのレジスタは各コマンド個別です。詳細については各コマンド向けの仕様を参照してください。

メールボックス・メッセージ

アプリケーション・メッセージ

概論

このカテゴリは Anybus-S 機能の内部へのアクセスと制御をするコマンドを含みます。

このカテゴリのメッセージ

メッセージ	省略	説明	ページ
Start initialization	START_INIT	初期化プロセスを開始します	9-2
Anybus initialization	Anybus_INIT	基本的なモジュールのパラメータ設定に使用されます	9-3
End initialization	END_INIT	初期化を終了します	9-6
Save to FLASH	SAVE_TO_FLASH	メールボックス初期化シーケンスを FLASH へ保存します	9-7
Load from FLASH	LOAD_FROM_FLASH	前回のメールボックス初期化シーケンスを Flash からロードします	9-8
Hardware Check	HW_CHK	Anybus-S のハードウェア上の診断テストを行います	9-9

初期化の開始 (START_INIT)

このコマンドは初期化プロセスを開始します。

メッセージ・イニシエータ	アプリケーション
メッセージ名	START_INIT
メッセージ・タイプ	1. （アプリケーション・メッセージ）
コマンド番号	0001h
拡張ヘッダ	-
コマンド・データ	-
レスポンス・データ	-

コマンドとレスポンス・レイアウト：

	コマンド	期待されるレスポンス	
メッセージ ID	(ID)	(ID)	
メッセージ情報	4001h	0001h	*アプリケーション・メッセージ*
コマンド	0001h	0001h	*START_INIT*
データ・サイズ	0000h	0000h	*メッセージ・データなし*
	0001h	0001h	
	0001h	0001h	
	0000h	0000h	
	0000h	0000h	
拡張ワード 1	-	-	
拡張ワード 2	-	-	
拡張ワード 3	-	-	
拡張ワード 4	-	-	
拡張ワード 5	-	-	
拡張ワード 6	-	-	
拡張ワード 7	-	-	
拡張ワード 8	-	-	

Anybus の初期化 (Anybus_INIT)

このコマンドはデータ交換のためのデータ・バッファの構成や、モジュールがネットワーク上でどのように機能するかを設定します。このメールボックス・コマンドの送信は必須で、'Load from FLASH' を読み込み間接的に実行するか、直接このコマンドを送信するかのどちらかの方法で実行します。

アプリケーションはモジュールからのレスポンスをモニタし、コマンドが承認されたか検証する必要があります。

コマンド内の初期化パラメータはモジュールによって解析されます。設定範囲を超えるパラメータの値が設定された場合、レスポンス・メッセージとして推奨値が返されます。アプリケーションは正しい値でメッセージを再送する必要があります。

注意 : このコマンドはモジュールの初期化時、START_INIT と END_INIT が実行される間にのみ送信することが可能です。

メッセージ・イニシエータ	アプリケーション
メッセージ名	Anybus_INIT
メッセージ・タイプ	1. （アプリケーション・メッセージ）
コマンド番号	0002h
拡張ヘッダ	レスポンス・ヘッダはフォルト・インフォメーションを含むことがあります
コマンド・データ	初期化パラメータ・データ
レスポンス・データ	コマンド・データのコピー、または設定できる最大値

コマンドとレスポンス・レイアウト :

	コマンド	期待されるレスポンス	
メッセージ ID	(ID)	(ID)	
メッセージ情報	4001h	0001h	*アプリケーション・メッセージ*
コマンド	0002h	0002h	*Anybus_INIT*
データ・サイズ	0012h	0012h	*9 ワードのデータ (18 バイト)*
	0001h	0001h	
	0001h	0001h	
	0000h	0000h	
	0000h	0000h	
拡張ワード 1	-	-	
拡張ワード 2	-	-	
拡張ワード 3	-	-	
拡張ワード 4	-	-	
拡張ワード 5	-	-	
拡張ワード 6	-	-	
拡張ワード 7	-	-	
拡張ワード 8	-	フォルト・インフォメーション	
コマンド・データ・ワード 1	入力 I/O 長	入力 I/O 長	
コマンド・データ・ワード 2	入力 DPRAM 長	入力 DPRAM 長	
コマンド・データ・ワード 3	入力合計長	入力合計長	
コマンド・データ・ワード 4	出力 I/O 長	出力 I/O 長	
コマンド・データ・ワード 5	出力 DPRAM 長	出力 DPRAM 長	
コマンド・データ・ワード 6	出力合計長	出力合計長	
コマンド・データ・ワード 7	操作モード	操作モード	
コマンド・データ・ワード 8	イベント通知コンフィグ	イベント通知コンフィグ	
コマンド・データ・ワード 9	Watchdog タイムアウト値	Watchdog タイムアウト値	

- フォルト・インフォメーション

  エラー・コードが 'Invalid Other' (Fh) のとき、次の拡張エラー情報がこのレジスタに格納されます。

ビット	フォルト	説明
0	入力 I/O 長	正しくない入力 I/O データ長
1	入力 DPRAM 長	正しくない DPRAM 入力のデータ長
2	入力合計長	正しくない合計入力データ長
3	(リザーブド)	
4	出力 I/O 長	正しくない出力 I/O データ長
5	出力 DPRAM 長	正しくない DPRAM 出力データ長
6	出力合計長	正しくない合計出力データ長
7	(リザーブド)	
8	Module Status	正しくない Module Status レジスタのビットのコンフィグレーション
9	イベント通知	正しくない Event Notification レジスタのビットのコンフィグレーション
10	Incorrect Watchdog	正しくない Watchdog Counter の異なる値
11 - 15	(リザーブド)	

- 入力 I/O データ長、入力 DPRAM データ長と入力合計データ長

  これらのパラメータは入力データ・バッファ構成を設定します。詳細は 7-3 " データ構成 " を参照してください。

- 出力 I/O データ長、出力 DPRAM データ長と出力合計データ長

  これらのパラメータは Output Data Buffer の構成を決定します。詳細は 7-3 " データ構成 " を参照してください。

- 操作モード

  このパラメータはモジュールの様々な機能に使用されます。パラメータの内容は Control Register Area の Module Status レジスタに反映されます。

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
						CD	APFC				RDR	FBSPU	FBS	FBFC	

ビット	説明	State
FBFC	これらのビットはフィールドバスがオフラインになったときのモジュールの動作を決めます。詳細については、4-5 " モジュール・ステータス (7E2h - 7E3h, RO)" を参照してください。	詳細については、4-5 " モジュール・ステータス (7E2h - 7E3h, RO)" を参照してください。
FBS[1]		
FBSPU[1]		
RDR[1]	フィールドバス・リセット・デバイス・リクエスト通知	0: 無効 1: 有効
APFC[1]	このビットは Watchdog タイムアウトなど、アプリケーションが停止した時のモジュールの動作を決めます。	0: 入力データをクリア 1: 入力データをフリーズ
CD	このビットは Control Register Area の Changed Data Field レジスタを有効 / 無効にします。	0: 無効 1: 有効

- イベント通知コンフィグ

  このパラメータはイベント通知のトリガとなるイベントの設定に使用されます。詳細は 4-7 " イベント通知ソース (7EEh - 7EFh, RO)" を参照してください。

- **Watchdog タイムアウト値[1]**

  このパラメータは、Watchdog Counter Input/Output レジスタの値の差の最大許容値を設定します。詳細は 4-3 “Watchdog カウンタ入力 (7D2h - 7D3h, R/W)” と 4-4 “Watchdog カウンタ出力 (7D4h - 7D5h, RO)” を参照してください。

  この値が最大値を超えると、モジュールはフィールドバスにアプリケーションが正常に動作していないことを伝えます。パラメータの設定値は 100 から 3000 で、0.1-30 秒に対応します。ゼロの値はこの機能を不使用にする場合の設定です。

1. このビットの実装と動作はフィールドバス・タイプに依存します。詳細は個別の fieldbus appendix か、B-1 “ 注意 ” を参照してください。

初期化の終了 (END_INIT)

このコマンドは初期化のプロセスを終了します。

注意 : ソフトウェアまたはハードウェア・リセットをしないとモジュールの再初期化ができません。

メッセージ・イニシエータ	アプリケーション
メッセージ名	END_INIT
メッセージ・タイプ	1.（アプリケーション・メッセージ）
コマンド番号	0003h
拡張ヘッダ	-
コマンド・データ	-
レスポンス・データ	-

コマンドとレスポンス・レイアウト :

	コマンド	期待されるレスポンス	
メッセージ ID	(ID)	(ID)	
メッセージ情報	4001h	0001h	*アプリケーション・メッセージ*
コマンド	0003h	0003h	*END_INIT*
データ・サイズ	0000h	0000h	*メッセージ・データなし*
	0001h	0001h	
	0001h	0001h	
	0000h	0000h	
	0000h	0000h	
拡張ワード 1	-	-	
拡張ワード 2	-	-	
拡張ワード 3	-	-	
拡張ワード 4	-	-	
拡張ワード 5	-	-	
拡張ワード 6	-	-	
拡張ワード 7	-	第 2 フォルト・インフォメーション	
拡張ワード 8	-	第 1 フォルト・インフォメーション	

- **第 1 & 第 2 フォルト・インフォメーション**
  モジュールによっては内部初期化中にこれらのワードを介してフィールドバス・エラーコードを返信する機能を持ちます。これらのエラー・コードはフィールドバスによって異なるため、詳細は個別の fieldbus appendix を参照してください。

FLASH への保存 (SAVE_TO_FLASH)

このコマンドは Anybus-S の初期化コマンドを Flash へ保存するために送信されます。このコマンドは START_INIT コマンド送信後のみ送信することができます。このコマンドの実行によってストアされているコンフィグレーション・データは消去されます。

メッセージ・イニシエータ	アプリケーション
メッセージ名	SAVE_TO_FLASH
メッセージ・タイプ	1. （アプリケーション・メッセージ）
コマンド番号	0004h
拡張ヘッダ	レスポンス・ヘッダはフォルト・インフォメーションを含むことがあります
コマンド・データ	-
レスポンス・データ	-

コマンドとレスポンス・レイアウト：

	コマンド	期待されるレスポンス	
メッセージ ID	(ID)	(ID)	
メッセージ情報	4001h	0001h	*アプリケーション・メッセージ*
コマンド	0004h	0004h	*SAVE_TO_FLASH*
データ・サイズ	0000h	0000h	*メッセージ・データなし*
	0001h	0001h	
	0001h	0001h	
	0000h	0000h	
	0000h	0000h	
拡張ワード 1	-	-	
拡張ワード 2	-	-	
拡張ワード 3	-	-	
拡張ワード 4	-	-	
拡張ワード 5	-	-	
拡張ワード 6	-	-	
拡張ワード 7	-	-	
拡張ワード 8	-	フォルト・インフォメーション	

- フォルト・インフォメーション

  メッセージ情報 に ‘Flash Config Error’ が返ってきた場合、エラー内容をここで確認できます。

  0001h: Flash の容量オーバー – このメッセージのレスポンスがある場合保存できません。

  0002h: 保存操作エラー－コンフィグレーションのロードができません。

FLASH からのロード (LOAD_FROM_FLASH)

このコマンドは FLASH から Anybus-S の初期化コマンドのロードをするために送信されます。このコマンドは START_INIT コマンド送信後のみ送信することができ、送信時に FLASH にコンフィグレーションがセーブされていなければなりません。

メッセージ・イニシエータ	アプリケーション
メッセージ名	LOAD_FROM_FLASH
メッセージ・タイプ	1. （アプリケーション・メッセージ）
コマンド番号	0005h
拡張ヘッダ	レスポンス・ヘッダはフォルト・インフォメーションを含むことがあります
コマンド・データ	-
レスポンス・データ	-

コマンドとレスポンス・レイアウト：

	コマンド	期待されるレスポンス	
メッセージ ID	(ID)	(ID)	
メッセージ情報	4001h	0001h	*アプリケーション・メッセージ*
コマンド	0005h	0005h	*LOAD_FROM_FLASH*
データ・サイズ	0000h	0000h	*メッセージ・データなし*
	0001h	0001h	
	0001h	0001h	
	0000h	0000h	
	0000h	0000h	
拡張ワード 1	-	-	
拡張ワード 2	-	-	
拡張ワード 3	-	-	
拡張ワード 4	-	-	
拡張ワード 5	-	-	
拡張ワード 6	-	-	
拡張ワード 7	-	-	
拡張ワード 8	-	フォルト・インフォメーション	

- フォルト・インフォメーション

  メッセージ情報 に 'Flash Config Error' が返ってきた場合、エラー内容をここで確認できます。

  0003h: CRC エラーまたは Flash にデータがない - 直接レスポンスがあり、コンフィグレーションがロードされません。

  0004h: ロード中のエラー - モジュールはコンフィグレーションをロードできません。モジュールはリセットされます。

ハードウェア・チェック (HW_CHK)

このコマンドはモジュールにハードウェアの診断テストをさせます。RAM、DPRAM、FLASH （CRC テスト）とフィールドバス ASIC の診断テストをします。

エラーが発生した場合には、ハードウェア・リセットを実行しない限りモジュールはレスポンスしません。

Anybus-S の Watchdog LED がエラータイプを表示します。詳細は 11-1 “Anybus-S Watchdog LED” を参照してください。

注意 : このコマンドは START_INIT コマンドの前のみ送信できます。

メッセージ・イニシエータ	アプリケーション
メッセージ名	HW_CHK
メッセージ・タイプ	1. （アプリケーション・メッセージ）
コマンド番号	0006h
拡張ヘッダ	-
コマンド・データ	-
レスポンス・データ	-

コマンドとレスポンス・レイアウト :

	コマンド	期待されるレスポンス	
メッセージ ID	(ID)	(ID)	
メッセージ情報	4001h	0001h	*アプリケーション・メッセージ*
コマンド	0006h	0006h	*HW_CHK*
データ・サイズ	0000h	0000h	*メッセージ・データなし*
	0001h	0001h	
	0001h	0001h	
	0000h	0000h	
	0000h	0000h	
拡張ワード 1	-	-	
拡張ワード 2	-	-	
拡張ワード 3	-	-	
拡張ワード 4	-	-	
拡張ワード 5	-	-	
拡張ワード 6	-	-	
拡張ワード 7	-	-	
拡張ワード 8	-	-	

フィールドバス・メッセージ

このカテゴリはフィールドバス・データと機能にアクセスするコマンドを含みます。内容については fieldbus appendix を参照してください。

内部メモリ・メッセージ

概論

このカテゴリは内部メモリにアクセスする機能を含みます。

このカテゴリのメッセージ

メッセージ	省略	説明	ページ
Read Internal Input Area	RD_INT_IN	内部入力領域からのデータ・ブロック読み込み	9-12
Write Internal Input Area	WR_INT_IN	内部入力領域へのデータ・ブロック書き込み	9-13
Clear Internal Input Area	CLR_INT_IN	内部入力領域のデータ・ブロックをクリアする	9-14
Read Internal Output Area	RD_INT_OUT	内部出力領域のデータ・ブロックを読み込む	9-15

内部入力領域からの読み込み (RD_INT_IN)

このコマンドは Internal Input Area からデータ・ブロックを読み込むために使用されます。1 回のコマンドで最大で 256 バイトのデータを読み込むことができます。

メッセージ・イニシエータ	アプリケーション
メッセージ名	RD_INT_IN
メッセージ・タイプ	3. (内部メモリ・メッセージ)
コマンド番号	0001h
拡張ヘッダ	オフセット・アドレスのソースとデータ・ブロック・サイズを含む
コマンド・データ	-
レスポンス・データ	読み込まれたデータ・ブロックの内容

コマンドとレスポンス・レイアウト：

	コマンド	期待されるレスポンス	
メッセージ ID	(ID)	(ID)	
メッセージ情報	4003h	0003h	*内部メモリ・メッセージ*
コマンド	0001h	0001h	*RD_INT_IN*
データ・サイズ	0000h	(データ・サイズ)	*(ブロック・サイズと同じ)*
	0001h	0001h	
	0001h	0001h	
	0000h	0000h	
	0000h	0000h	
拡張ワード 1	ブロック・オフセット	ブロック・オフセット	
拡張ワード 2	ブロック・サイズ	ブロック・サイズ	
拡張ワード 3	-	-	
拡張ワード 4	-	-	
拡張ワード 5	-	-	
拡張ワード 6	-	-	
拡張ワード 7	-	-	
拡張ワード 8	-	-	
		レスポンス・データ・ワード 1	
		データ・ブロック	**...**
		レスポンス・データ・ワード n	

- **ブロック・オフセット**
  出力データ・バッファの開始からのアドレス・オフセット
- **ブロック・サイズ**
  読まれるブロック・サイズ（バイト）
- **データ・ブロック**
  実際のデータ・ブロック

内部入力領域への書き込み (WR_INT_IN)

このコマンドは Internal Input Area へデータ・ブロックを書き込むときに使用します。1 回のコマンドで最大で 256 バイトのデータを読み込むことができます。

メッセージ・イニシエータ	アプリケーション
メッセージ名	WR_INT_IN
メッセージ・タイプ	3. (内部メモリ・メッセージ)
コマンド番号	0002h
拡張ヘッダ	対象オフセット・アドレスとデータ・ブロック・サイズを含む
コマンド・データ	書き込まれるべきデータ
レスポンス・データ	書き込まれたデータのコピー

コマンドとレスポンス・レイアウト：

	コマンド	期待されるレスポンス	
メッセージ ID	(ID)	(ID)	
メッセージ情報	4003h	0003h	*内部メモリ・メッセージ*
コマンド	0002h	0002h	*WR_INT_IN*
データ・サイズ	(データ・サイズ)	(データ・サイズ)	*(ブロック・サイズと同じ)*
	0001h	0001h	
	0001h	0001h	
	0000h	0000h	
	0000h	0000h	
拡張ワード 1	ブロック・オフセット	ブロック・オフセット	
拡張ワード 2	ブロック・サイズ	ブロック・サイズ	
拡張ワード 3	-	-	
拡張ワード 4	-	-	
拡張ワード 5	-	-	
拡張ワード 6	-	-	
拡張ワード 7	-	-	
拡張ワード 8	-	-	
コマンド・データ・ワード 1	データ・ブロック	データ・ブロック	**レスポンス・データ・ワード 1**
...			**...**
コマンド・データ・ワード n			**レスポンス・データ・ワード n**

- **ブロック・オフセット**
  出力データ・バッファの開始からのアドレス・オフセット
- **ブロック・サイズ**
  読まれるブロック・サイズ（バイト）
- **データ・ブロック**
  実際のデータ・ブロック

内部入力領域のクリア (CLR_INT_IN)

このコマンドは Internal Input Area のデータ・ブロックをクリアするときに使用します。1 回のコマンドで最大で 256 バイトのデータをクリアできます。領域全体のデータをクリアするには、このコマンドの複数回の実行が必要です。

メッセージ・イニシエータ	アプリケーション
メッセージ名	CLR_INT_IN
メッセージ・タイプ	3. (内部メモリ・メッセージ)
コマンド番号	0003h
拡張ヘッダ	対象オフセット・アドレスとクリアされるデータ・ブロック・サイズを含む
コマンド・データ	-
レスポンス・データ	-

コマンドとレスポンス・レイアウト:

	コマンド	期待されるレスポンス	
メッセージ ID	(ID)	(ID)	
メッセージ情報	4003h	0003h	*内部メモリ・メッセージ*
コマンド	0003h	0003h	*CLR_INT_IN*
データ・サイズ	0000h	0000h	*メッセージ・データなし*
	0001h	0001h	
	0001h	0001h	
	0000h	0000h	
	0000h	0000h	
拡張ワード 1	ブロック・オフセット	ブロック・オフセット	
拡張ワード 2	ブロック・サイズ	ブロック・サイズ	
拡張ワード 3	-	-	
拡張ワード 4	-	-	
拡張ワード 5	-	-	
拡張ワード 6	-	-	
拡張ワード 7	-	-	
拡張ワード 8	-	-	

- **ブロック・オフセット**
  入力データ・バッファのアドレス・オフセット
- **ブロック・サイズ**
  クリアされるブロック・サイズ（バイト単位）

内部出力領域からの読み込み (RD_INT_OUT)

このコマンドは内部出力領域のデータ・ブロックを読み込むときに使用します。1 回のコマンドで最大で 256 バイトのデータを読み込むことができます。

メッセージ・イニシエータ	アプリケーション
メッセージ名	RD_INT_OUT
メッセージ・タイプ	3. (内部メモリ・メッセージ)
コマンド番号	0004h
拡張ヘッダ	オフセット・アドレスのソースとデータ・ブロック・サイズを含む
コマンド・データ	-
レスポンス・データ	読み込まれるデータ・ブロックの内容

コマンドとレスポンス・レイアウト：

	コマンド	期待されるレスポンス	
メッセージ ID	(ID)	(ID)	
メッセージ情報	4003h	0003h	*内部メモリ・メッセージ*
コマンド	0004h	0004h	*RD_INT_OUT*
データ・サイズ	0000h	(データ・サイズ)	*(ブロック・サイズと同じ)*
	0001h	0001h	
	0001h	0001h	
	0000h	0000h	
	0000h	0000h	
拡張ワード 1	ブロック・オフセット	ブロック・オフセット	
拡張ワード 2	ブロック・サイズ	ブロック・サイズ	
拡張ワード 3	-	-	
拡張ワード 4	-	-	
拡張ワード 5	-	-	
拡張ワード 6	-	-	
拡張ワード 7	-	-	
拡張ワード 8	-	-	
			レスポンス・データ・ワード 1
		データ・ブロック	...
			レスポンス・データ・ワード n

- **ブロック・オフセット**
  出力データ・バッファの開始からのアドレス・オフセット
- **ブロック・サイズ**
  読まれるブロック・サイズ（バイト）
- **データ・ブロック**
  実際のデータ・ブロック

リセット・メッセージ

概論

このカテゴリにはリセットに関する機能が含まれます。

このカテゴリのメッセージ

メッセージ	省略	説明	ページ
Software Reset	SW_RESET	Performs a software reset of the module.	9-16

ソフトウェア・リセット (SW_RESET)

このコマンドは Anybus-S モジュールをリスタートするときに使用します。例えば、ある初期化データを変更する必要があるときに使用します。アプリケーションがレスポンスを読み、リセット・コマンドがアクティブになるまで 1 秒かかります。

注意 : このコマンドはモジュールのソフトウェア・リセットを行うのみで、ハードウェア・リセットは行いません。

メッセージ・イニシエータ	アプリケーション
メッセージ名	SW_RESET
メッセージ・タイプ	5. (リセット・メッセージ)
コマンド番号	0001h
拡張ヘッダ	-
コマンド・データ	-
レスポンス・データ	-

コマンドとレスポンス・レイアウト :

	コマンド	期待されるレスポンス	
メッセージ ID	(ID)	(ID)	
メッセージ情報	4005h	0005h	*リセット・メッセージ*
コマンド	0001h	0001h	*SW_RESET*
データ・サイズ	0000h	0000h	*メッセージ・データなし*
	0001h	0001h	
	0001h	0001h	
	0000h	0000h	
	0000h	0000h	
拡張ワード 1	-	-	
拡張ワード 2	-	-	
拡張ワード 3	-	-	
拡張ワード 4	-	-	
拡張ワード 5	-	-	
拡張ワード 6	-	-	
拡張ワード 7	-	-	
拡張ワード 8	-	-	

スタートアップと初期化

イントロダクション

Anybus-S モジュールは以下の 3 つのステートの内のどれかに遷移します。

1. **ハードウェアの初期化ステート**
   パワーオン / リセット後に、モジュールが遷移する最初のステートです。次のステートに移るため、様々なハードウェアの診断テストが実行されます。10-2 “ ハードウェアの初期化 ” を参照してください。
   - このステートではデータ交換不可能です。

2. **ソフトウェア初期化ステート**
   基本的なパラメータが設定されます。次のステートに移るため、ソフトウェアの初期化シーケンスを完了する必要があります。詳細は 10-3 “ ソフトウェアの初期化 ” を参照してください。
   - このステートではデータ交換不可能です。

3. **データ交換ステート**
   フィールドバスと二つの I/O データ・バッファ間のデータ交換が可能です。このステートに遷移する唯一の方法は、他のステートを正常に通過する事です。

このチャプタでは、ステート 1 とステート 2 について説明します。ステート 3（データ交換ステート）の詳細については 7-1 “ フィールドバス・データ交換 ” を参照してください。

ハードウェアの初期化

この手順は必須で、ソフトウェアの初期化シーケンス前にモジュールが正常に動作していることを確認します。この手順は次のステップを含みます。

- スタートアップ・シーケンス
- デュアル・ポート・メモリ・チェック（オプション）
- ハードウェア・チェック・メールボックス・メッセージ（オプション）

スタートアップ・シーケンス

アプリケーションの実装の仕方によって、スタートアップ手順は異なります。

- **ハードウェア割り込みが実装されていない場合**

  パワーオン / リセットの後、アプリケーションは、モジュールによって適切に更新されていることを検出するため、Watchdog Counter Out レジスタ (7D4h -7D5h) を約 10ms 周期でポーリングします。最低 10 回このレジスタがモジュールによって更新されるとき、モジュールは動作可能です。

- **ハードウェア割り込みが実装されている場合**

  パワーオン / リセットの後、Anybus モジュールは割り込みによって動作可能であることを示します。割り込みの前、アプリケーションはデュアル・ポート・メモリにデータを書き込むことができません。割り込み信号は DPRAM の内部ロジックによって制御されます。このロジックはハードウェア・リセットではなくパワーオンリセットによって開始されます。そのため適切な機能を保証するために注意が必要です。詳細については C-1 “ 割り込み信号とハードウェア・リセット ” を参照してください。

デュアル・ポート・メモリ・チェック（オプション）

デュアル・ポート・メモリの読み込み／書き込みテストの実行を推奨します。これはアプリケーションの性質によって二つの方法で可能です。

- **ハードウェア・リセットが実装されていない場合**
  （またはアプリケーション・ソフトウェアで制御されていない場合）

  テストは上記のスタートアップ・シーケンスの後、直接実行します。テストは非破壊的に行なわなければなりません。つまり、デュアル・ポート・メモリのデータはテスト後にリストアされなければなりません。また、以下のレジスタについてはモジュールのオペレーションとのインターフェースであるためテスト対象外とされなければなりません。

  - Watchdog Counter In レジスタ (7D2h - 7D3h)
  - Watchdog Counter Out レジスタ (7D4h - 7D5h)
  - Handshake Registers (7FEh - 7FFh)
  - LED 表示・ステータス (7DAh-7DFh)
  - Module Status (7E2h-7E3h)
  - Fieldbus Specific Area (640h - 7BFh)

- **ハードウェア・リセットが実装されている場合**
  （そしてアプリケーション・ソフトウェアで制御される場合）

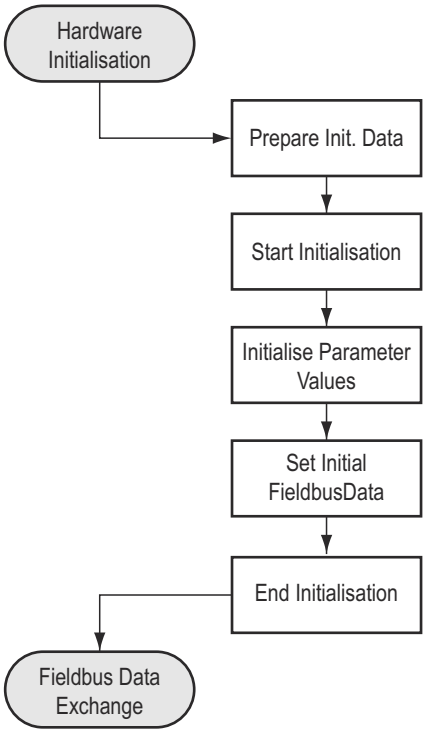
  デュアル・ポート・メモリ・テストは、上記のスタートアップ・シーケンス前に、リセット信号がアプリケーションによって low になっている間に実行しなければなりません。全てのメモリ・ロケーションは破壊的または非破壊的にテスト可能です。

ハードウェア・チェック（オプション）

注意 : この手順はスタートアップ・シーケンスとデュアル・ポート・メモリ・チェック後のみ実行できます。

'Hardware Check' のメールボックス・コマンドの送信によって、モジュールに対してハードウェアセルフテストが実行されます。テストが正常に完了すると、モジュールはレスポンスします。モジュールがレスポンスしない場合は、テストが失敗したことを示します。詳細については、9-9 " ハードウェア・チェック (HW_CHK)" と 11-1 "Anybus-S Watchdog LED" を参照してください。

ソフトウェアの初期化



フィールドバス通信の前に、Anybus-S モジュールのソフトウェアは初期化されなければなりません。

このプロセスは必須で、モジュールがネットワーク上でどのように動作するかを決定します。

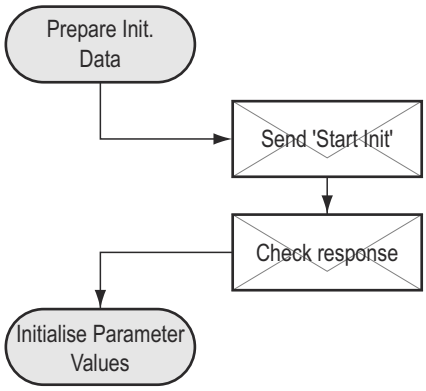
1. **初期化データの準備（オプション）**
2. **初期化の開始**
3. **パラメータ値の初期化**
4. **フィールドバス・データの設定**
5. **初期化の終了**

データ初期化の準備

このステップはオプションですが、複数のソフトウェアバージョンを使用することなく、アプリケーションがフィールドバスに特化した機能を利用できるようにします。アプリケーションによっては Anybus モジュールのタイプの情報やその他のファームウェア・リビジョンのような情報が必要です。初期化の間、この情報はハンドシェークなしで Control Register Area から読み込まれます。

アプリケーションは、より有効に Anybus モデルを利用するためソフトウェアの初期化パラメータを変更できます。

初期化の開始



このステップは必須で、ソフトウェアの初期化シーケンスを開始します。

1. **メールボックス・メッセージ 'Start Init' の送信**
   ソフトウェアの初期化プロセスを開始します。
2. **レスポンスの確認**
   コンフィグレーションメールボックスメッセージの受信ができることを確認します。

パラメータ値の初期化

このステップは必須ですが、正確な手順は条件によって異なります。次のステップを含みます。

- **メールボックス・コマンド 'Save to FLASH' (オプション) の送信**
  このメールボックス・コマンドは以下のメールボックス・コマンドを記録するテープレコーダのような機能を持ちます。モジュールが 'End Init' のメールボックス・コマンドを受信すると記録は停止します。

- **メールボックス・コマンド 'Load from FLASH' (オプション) の送信**
  このメールボックス・コマンドは、'Save to FLASH' メールボックス・コマンドで記録されたメールボックス・シーケンスを再送します。

- **メールボックス・コマンド 'Anybus Init' の送信**
  メールボックス・コマンドは data exchange buffer のデータ長の設定をするために使用されます。このメールボックスは必須で、直接または間接的にメールボックス・コマンド 'Load from FLASH' を使用します。アプリケーションはモジュールからのレスポンスをモニタし、コマンドが受信されたことを確認する必要があります。

- **フィールドバスの初期化コマンド（オプション）**
  この手順はモジュールへのフィールドバスに特化したメールボックスの送信が含まれます。詳細は fieldbus appendix を参照してください。フィールドバスに特化した初期化コマンドは、Anybus Init の前に送信しなければなりません。この処理はアプリケーション・ソフトウェアに含まれなければなりません。

  注意 : フィールドバスに特化した初期化コマンドはフィールドバスの事前認証を無効にする可能性があります。F-1 " フィールドバス認証 " を参照してください。

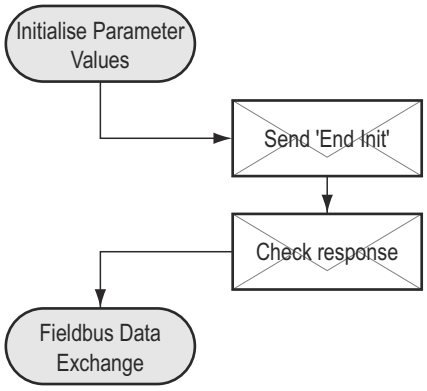
フィールドバス・データの初期設定

このステップはオプションですが、このステップによってアプリケーションはバスサイクルの前に Input Data Buffer の内容を制御することができます。データが書き込まれるロケーションによって、アプリケーションは DPRAM の Input Data Area にデータを書き込むか、内部メモリ・コマンドを使用しこのコマンドを送信しなければなりません。9-11 " 内部メモリ・メッセージ " を参照してください。

初期化の終了

このステップはモジュールに初期化が完了しフィールドバス上でデータ交換を開始することを伝えます。

1. **メールボックス・メッセージ 'End Init' を送信します。**
2. **レスポンスを確認します。**
   このレスポンスが OK の場合、モジュールはフィールドバスのデータ交換を開始します。



基本的な初期化シーケンス 例 1

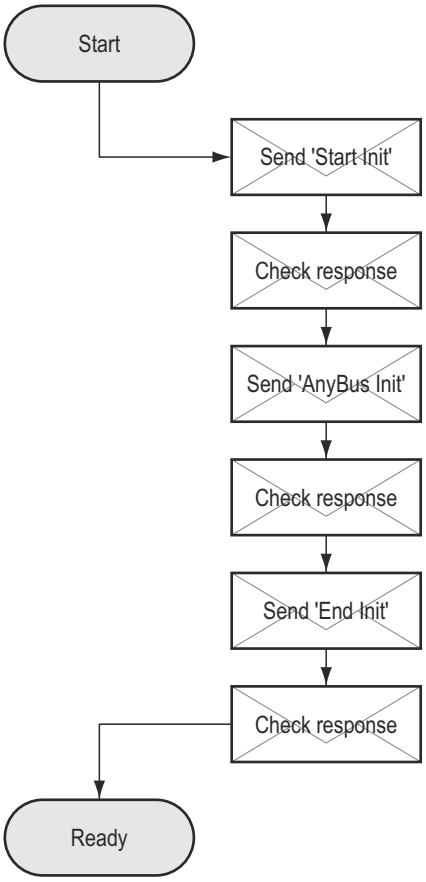
(次の手順は、‘Anybus Init’ コマンドのパラメータが有効であることが前提です。)

基本的な機能のみを必要とする場合、次のような基本的な初期化シーケンスが使用できます。

■ パワーオン（リセット）

1. **‘Start Init’ メールボックス・コマンドの送信**
2. **レスポンスの確認**
3. **‘Anybus Init’ メールボックス・コマンドの送信**
4. **レスポンスの確認**
5. **‘End Init’ メールボックス・コマンドの送信**
6. **レスポンスの確認**

■ 初期化完了



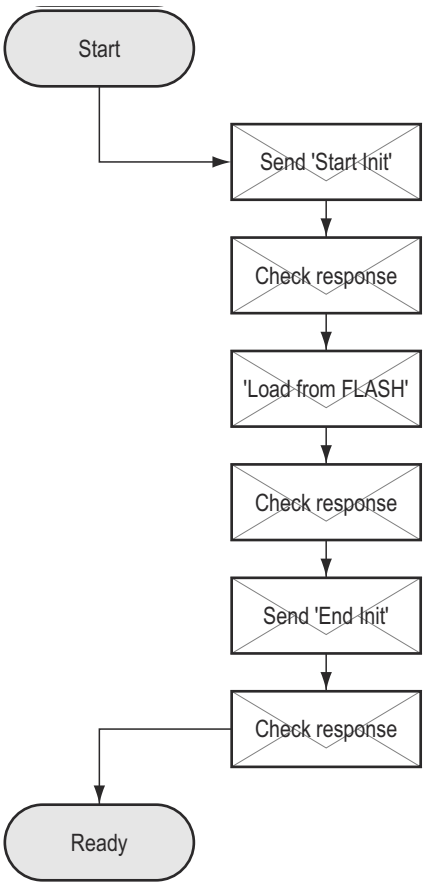
基本的な初期化シーケンス 例 2

この例では初期化メールボックス・シーケンスが Flash からロードされます。初期化シーケンスは、‘Save to FLASH’ メールボックス・コマンドを使用し Flash にリストアされていることが必要です。

■ パワーオン（リセット）

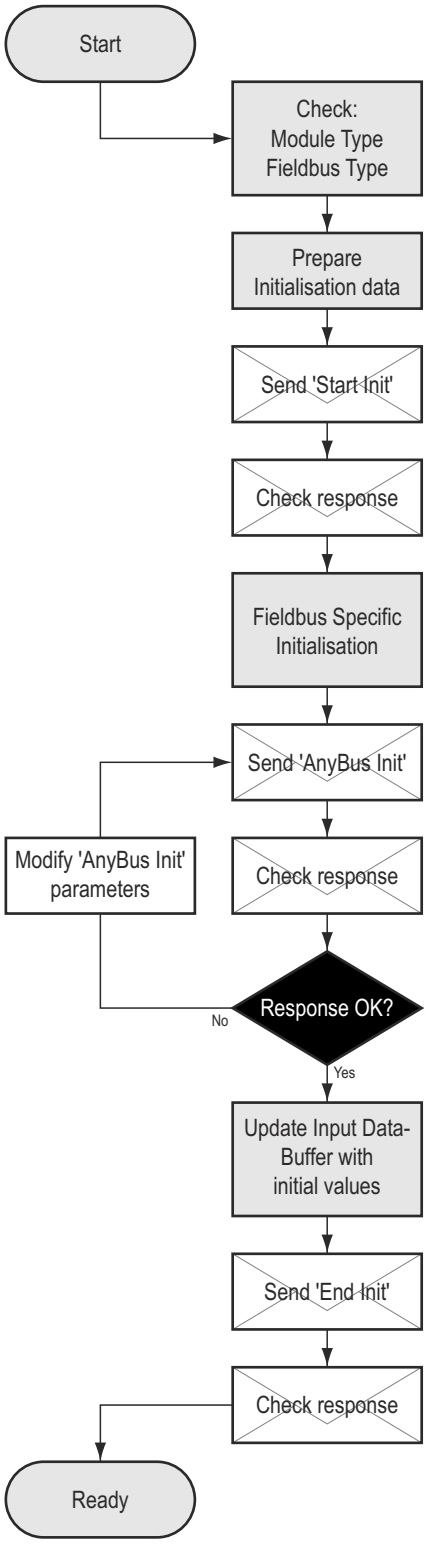
1. **Start Init’ メールボックス・コマンドの送信**
2. **レスポンスの確認**
3. **‘Load from flash’ メールボックス・コマンドの送信**
4. **レスポンスの確認**
5. **‘End Init’ メールボックス・コマンドの送信**
6. **レスポンスの確認**

■ 初期化完了



高度な初期化例

この例では、初期化シーケンスは使用されているモジュール / フィールドバス・タイプに適合され、Input Data Buffer は最初のバスサイクルの前に更新されます。



■ パワーオン（リセット）

1. **モジュール / フィールドバス・タイプの確認**
   この情報は Control Register Area からハンドシェークなしで読み込めます。
2. **初期化データの準備**
   Control Register Area からの情報に基づき、アプリケーションは残りの初期化プロセスの中でどの初期化パラメータを使用するかを決定します。
3. **'Start Init' コマンドの送信**
   このステップは初期化シーケンスを開始します。
4. **レスポンスの確認**
5. **フィールドバスに特化した初期化**
   このステップの手順はフィールドバスごとに異なり、各個別 fieldbus appendix に記載されています。
6. **Anybus Init メールボックスの送信**
   'Anybus Init' は I/O サイズと様々なコンフィグレーション・ビットの設定に使用されます。
7. **レスポンスの確認**
8. **レスポンスが OK かどうかを確認**
   OK でない場合、Anybus Init メールボックス・メッセージのパラメータを変更し、ステップ 4 に進みます。
9. **初期値で Input Area を更新します。**
10. **'End Init' メールボックス・コマンドを送信します。**
    このステップは初期化シーケンスを終了します。
11. **レスポンスの確認**

■ 初期化完了

LED 表示

フィールドバス・ステータス表示

各フィールドバス規格に基づいた 4 つの LED がモジュール前面に搭載されています。これらの LED の機能はフィールドバスによって異なり、各 fieldbus appendix　に記載されています。

Anybus-S Watchdog LED

全ての Anybus-S モジュールは表面実装部品の LED によってモジュールのステータスを表示します。

色	周波数	説明
赤	-	予期せぬ内部エラー、またはブートローダーモードでの動作
	1Hz	RAM 故障
	2Hz	ASIC または FLASH 故障
	4Hz	DPRAM 故障
緑	2Hz	モジュール未初期化
	1Hz	モジュール初期化終了、動作 OK

ファームウェアの更新

Anybus-S モジュールで継続的な製品開発を継続するため、また更に改良された機能をお客様へ提供するため FLASH メモリが全ての Anybus-S モジュールで使用されます。

これにより製造後に市場にリリースされたモジュールのファームウェアの更新が可能です。

モジュールは二つのファームウェアのダウンロード方法をサポートしています。

- **DPRAM を使用したパラレル・ダウンロード**
  この方法はアプリケーションに特別なドライバの実装が必要です。ドライバの仕様については HMS にお問い合わせください。
- **非同期シリアル・インターフェースによるシリアルダウンロード**
  この方法は Anybus のアプリケーション・コネクタの非同期シリアル・インターフェースピンが、RS232 ラインドライバに接続されている必要があります。ファームウェアは HMS から提供されている Microsoft Windows アプリケーションを使用しダウンロードすることができます。
  この方法はモジュールによって、回路へのアクセスが必要になる場合があります。詳細については HMS へお問い合わせください。

アプリケーションがファームウェアのダウンロードをサポートする必要がある場合、HMS はシリアル・ダウンロード・オプションの使用を推奨します。シンプルですが、外部 RS232-TTL レベル・コンバータに接続された VCC、GND、TX と RX ピンの構成の 4 ピンコネクタよって全機能の実装が可能です。TX/RX ピンは通信に使用され、VCC/GND ピンはコンバータに電源を供給するため外部電源が不要です。

モジュールはリセット後にダウンロードコマンドのみ受信可能なため、アプリケーション・ソフトウェアはある程度の準備が必要です。一度アプリケーションが初期化を開始すると、ダウンロードを可能にするために、リセットまたはパワーサイクルが必要です。

新しいモジュールでは、新しいファームウェアをダウンロードするため、モジュール上でジャンパ結線が必要です。詳細については HMS にお問い合わせください。

ドライバ・サンプル

サンプルドライバはハンドラという三つのルーチンで構成されています。

- **割り込みハンドラ**
  このルーチンは Anybus-S からの割り込みの処理をします。詳細は 13-2 “ 割り込みハンドラ ” を参照してください。
- **インターフェース・ハンドラ**
  このルーチンはメイン・プログラムから周期的に呼び出されます。詳細は 13-3 “ インターフェース・ハンドラ ” を参照してください。
- **メールボックス・ハンドラ**
  このルーチンは、メイン・プログラムから周期的または必要に応じて呼び出されます。詳細は 13-4 “ メールボックス・ハンドラ ” を参照してください。

インプリメンテーション・ノート

サンプル内のコードはマルチタスク環境で実行されることが前提とされています。アプリケーションがマルチタスク環境に対応していない場合、コードはステートマシンまたはリニアコードではない同様の実装が必要です。

重要！

これらのサンプルルーチンは最適化されておらずガイダンスとして説明されています。モジュールを適切に動作させるための初期化、タイムアウト、Event Notification やエラー処理に必要な機能は別途実装が必要です。

グローバル変数

以下のグローバル変数がサンプルルーチンで使用されています。

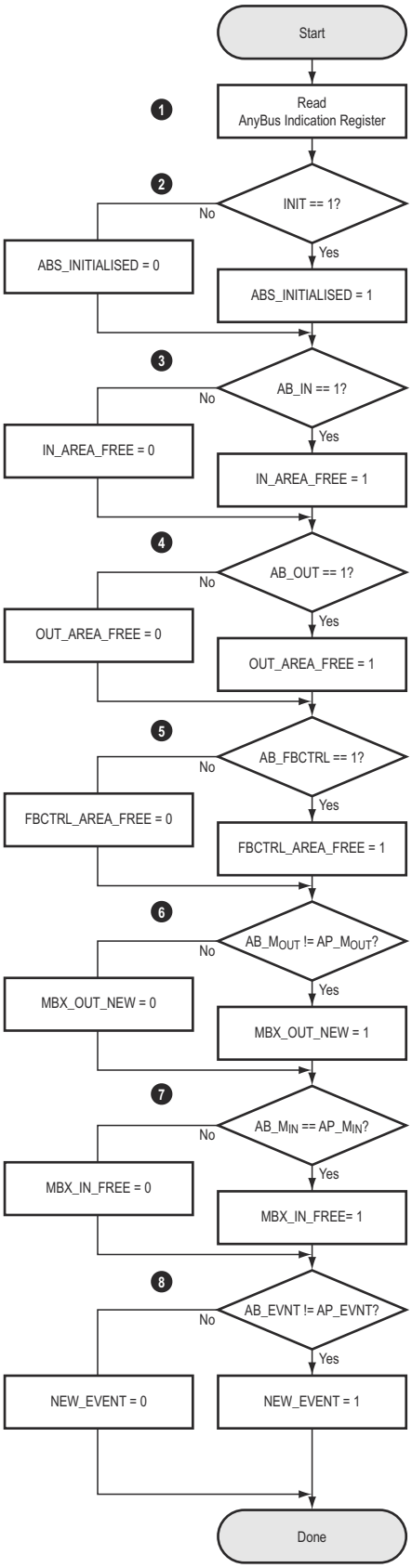
値	タイプ	説明
INIT	Boolean	これらの値は Anybus インジケーション・レジスタの個々のビットに対応します。5-3 “Anybus インジケーション・レジスタ (7FFh, RO)” を参照ください。
AB_IN	Boolean	
AB_OUT	Boolean	
AB_FBCTRL	Boolean	
AB_M_{OUT}	Boolean	
AB_M_{IN}	Boolean	
AB_EVNT	Boolean	
AP_M_{IN}	Boolean	これらの値はアプリケーション・インジケーション・レジスタの個々のビットに対応します。5-2 “ アプリケーション・インジケーション・レジスタ (7FEh, R/W)” を参照ください。
AP_M_{OUT}	Boolean	
AP_EVNT	Boolean	
ABS_INITIALISED	Boolean	この値はモジュールの初期化が終了したことを表します。
IN_AREA_FREE	Boolean	これらの値は DPRAM が使用できることを表します。（Set = free）
OUT_AREA_FREE	Boolean	
FBCTRL_AREA_FREE	Boolean	
MBX_OUT_NEW	Boolean	この値は Mailbox Output Area が新しいメッセージを含むことを表します。
MBX_IN_FREE	Boolean	この値は Mailbox Input Area が使用できることを表します。

割り込みハンドラ

このルーチンの目的は割り込みを受けたときの Anybus Indication Register の内容の変換と、インターフェース／メールボックス・ハンドラとアプリケーション・プログラムがその情報を処理できるようにすることです。

最初の割り込み処理を実行するため、さらにプロセスは必要ありません。全てのプロセスはインターフェース・ハンドラが代わりに実行します。

注意: このルーチンは 'Interrupt Handler' と呼ばれるルーチンですが、割り込み信号ピンがアプリケーションに実装されていなくても使用することができます。この場合、そして / またはステータス変数をリフレッシュするため、ルーチンは定期的に呼び出す必要があります。細かい実装方法についてはこの章では説明されていません。割り込み信号の使用をお奨めします。



手順

■ スタート

1. **Anybus インジケーション・レジスタの読み込み**
2. **モジュールの初期化完了の確認**
   (ABS_INITIALISED のステータスの保存)
3. **Input Area が空であることを確認**
   (IN_AREA_FREE のステータスの保存)
4. **Output Area が空であることを確認**
   (OUT_AREA_FREE のステータスの保存)
5. **Fieldbus Specific Area & Control レジスタが空であることを確認**
   (FBCTRL_AREA_FREE のステータスの保存)
6. **Mailbox Output Area に新しいメッセージが格納されているか確認**
   (MBX_OUT_NEW のステータスの保存)
7. **Mailbox Input Area が空であることを確認**
   (MBX_IN_FREE のステータスの保存)
8. **Event Notification が発生しているか確認**
   (NEW_EVENT のステータスを保存)

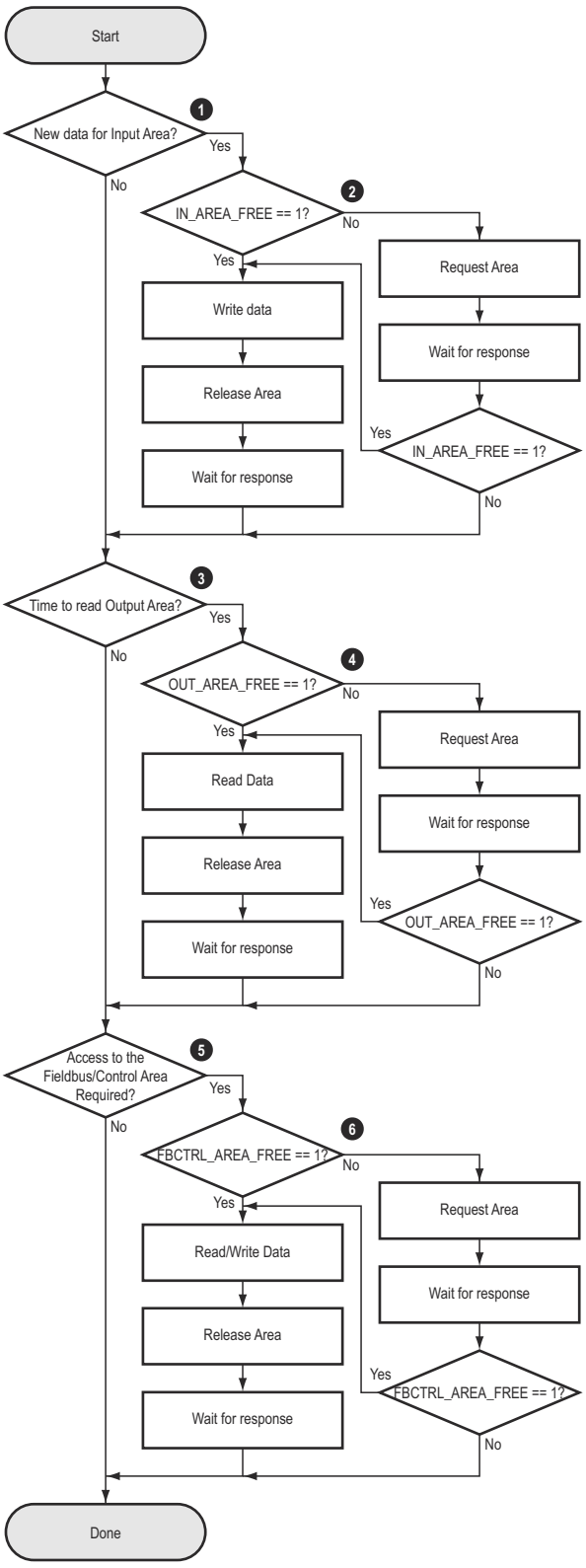
■ 終了

インターフェース・ハンドラ

インターフェース・ハンドラはメイン・プログラムから周期的に更新される必要があります。

このルーチンの目的はデータの転送と、Input/Output Data Area と Fieldbus Specific/Control Register Areas への全ての必要なハンドシェークの実行です。

デュアル・ポート・メモリ内の領域が使用中の場合、領域にアクセスし、次にルーチンが呼び出されるように、リクエストが生成されます。



手順

■ スタート

1. **Input Area に新しいデータがありますか？**
2. **領域は空ですか？**

   はいの場合、データにアクセスし、領域をリリースし、レスポンスを待ちます。

   いいえの場合、リクエストし、可能であれば領域にアクセスします。

   もし領域が使用中の場合、新たにリクエストが開始され、次にルーチンが呼ばれます。

3. **Output Area を読み出すときですか？**
4. **領域は空ですか？**

   はいの場合、領域をリリースし、レスポンスを待ちます。

   いいえの場合、リクエストし、可能であれば領域にアクセスします。もし領域が使用中の場合、新たにリクエストが開始され、次にルーチンが呼ばれます。

5. **Fieldbus Specific Area & Control Register Area へのアクセスが要求されましたか？**
6. **領域は空ですか？**

   はいの場合、データにアクセスし、領域をリリースしレスポンスを待ちます。

   いいえの場合、リクエストをし、可能であれば領域にアクセスします。もし使用中であれば、新たにリクエストが開始され、次にルーチンが呼ばれます。

■ 終了

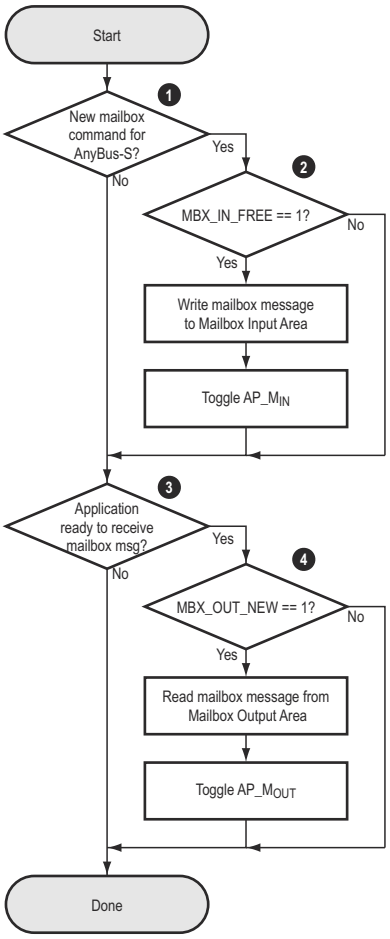
メールボックス・ハンドラ

メールボックス・ハンドラがメイン・ルーチン・プログラムから周期的に呼ばれます。

このルーチンの目的はモジュールのメールボックス・メッセージのデータ交換です。ルーチンは入力、出力メールボックス・データ交換を処理します。

- **アプリケーションから Anybus モジュール**
  アプリケーションからモジュールへメールボックスを送信するとき、ルーチンは最初に Mailbox Input Area が空であること (i.e. MBX_IN_FREE == 1) を確認し、メッセージを転送します。Mailbox Input Area にメッセージが書き込まれたことをモジュールに知らせるため、ルーチンは AP_M_{IN} ビットをトグルします。Mailbox Input Area が使用中の場合 (i.e. MBX_IN_FREE == 0)、次にルーチンが呼ばれるとメッセージを送信します。

- **モジュールからアプリケーション**
  アプリケーションが新しいメールボックス・メッセージを受信でき、新しいメッセージが検出されるとき (i.e.MBX_OUT_NEW == 1)、ルーチンは、アプリケーション・インジケーション・レジスタの AP_M_{OUT} ビットをトグルし、メッセージとレスポンスをモジュールへ転送します。



手順

■ スタート

1. **メールボックス・メッセージを Anybus モジュールへ送る必要がありますか？**
   (ない場合、ルーチンはステップ 3 へジャンプします。)

2. **Mailbox Input Area は空ですか？**
   はいの場合、Mailbox Input Area へメッセージを書き込みます。書き込み完了後、AP_M_{IN} ビットをトグルし、新しいメッセージが書き込まれたことをモジュールに知らせます。
   いいえの場合、何も処理をせず、新しい要求をし、次にルーチンが呼ばれます。

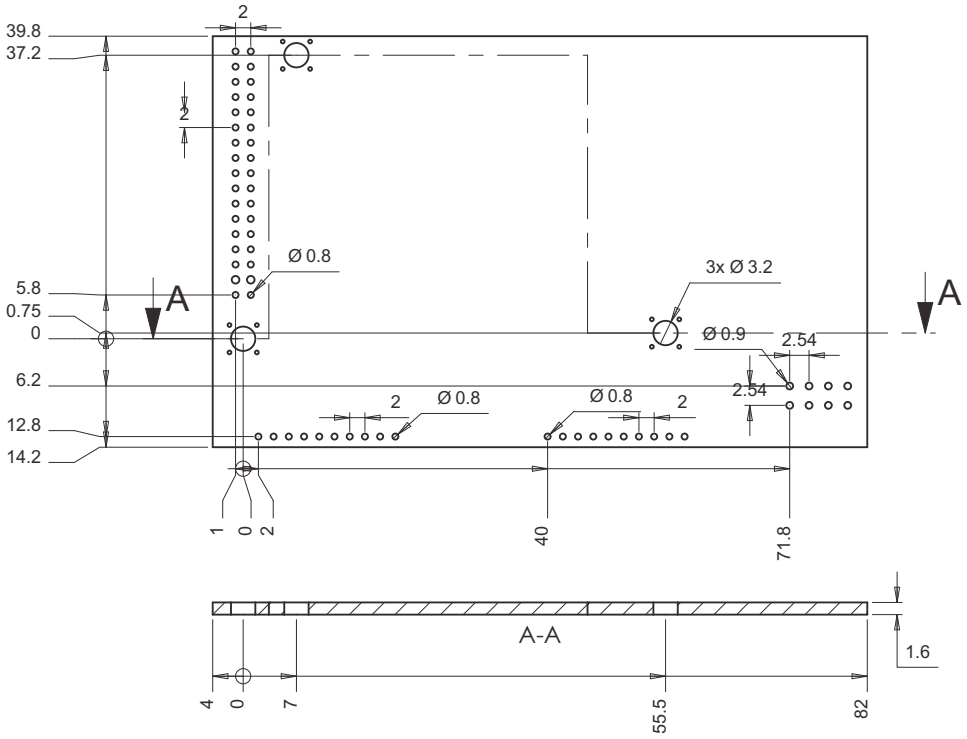
3. **アプリケーションは新しいメールボックス・メッセージを受信できる準備ができていますか？**
   (いいえの場合、ルーチンは存在しています。)

4. **Mailbox Output Area に新しいメッセージがありますか？**
   はいの場合、Mailbox Output Area からメッセージを読み込みます。読み込み後、AP_M_{OUT} ビットをトグルし、メッセージが読み込まれたことを知らせます。
   いいえの場合、何も処理をせず、新しい要求をし、次にルーチンが呼ばれます。

■ 終了

メカニカル仕様

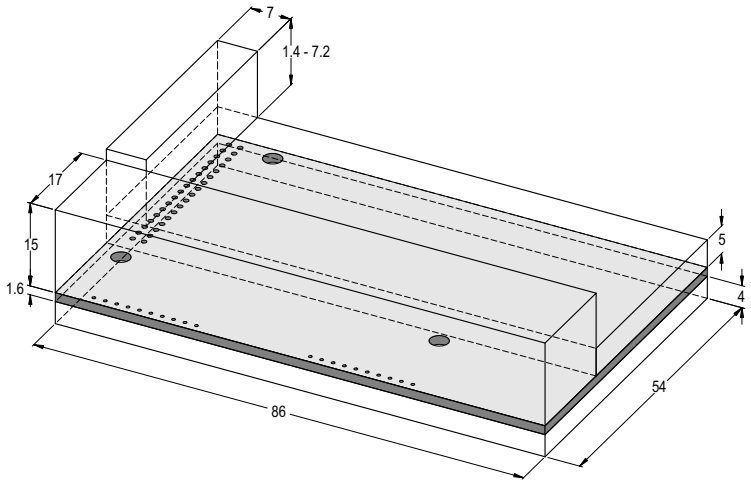
PCB 寸法



高さ制限

Anybus-S の基板／部品とアプリケーションの周辺部品の間のスペースを確保するためいくつかの制約を考慮する必要があります。

下の図面は Anybus-S モジュール上の部品の高さ制限です。一般的に、部品等の高さは下の境界外に越えることができません。

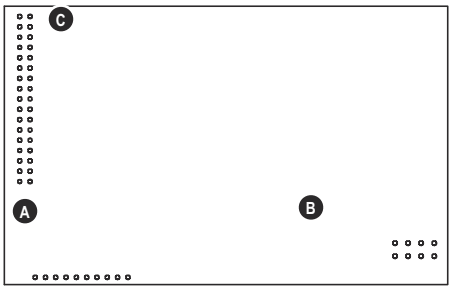


注意 : 現在使用している部品の制限により、Anybus-S ControlNet は上記の基準に適合しません。このモジュールの表面の高さ制限は 15mm です。

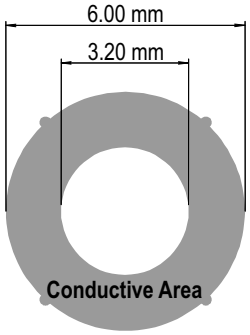
取り付け穴

Anybus-S モジュールには 3 つの取り付け穴があります。この取り付け穴の内の二つは、アプリケーションとモジュール間の電気的接続を改善するためのものです。

(寸法については、14-1 “PCB 寸法 ” を参照してください。)

#	説明	Position
A	伝導性のある取り付け穴でフィールドバスの電源（シールド）と PE(プロテクティブ・アース) を接続するために使用されます。（この取り付け穴はフィールドバスの仕様に従ってフィルタを介してフィールドバス・ケーブル・シールドに接続されます。）	
B	電気的に絶縁された取り付け穴で伝導、非伝導性スクリューのために使用されます。	
C	伝導性取り付け穴で改善された GND 接続のために使用されます。	

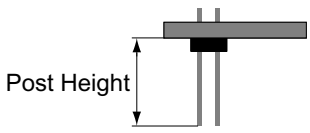
伝導、非伝導性の取り付け穴の周囲には Anybus モジュール上の PCB のパターンが無い領域があります。この領域の直径は 8mm です。



アプリケーション・コネクタ

アプリケーションコネクタはピン長が 2mm でモジュールの両面に実装可能です。必要に応じ追加の変更のための代わりのソリューションもご提供できます。モジュールに標準または非標準の他のピン長のコネクタを実装することもできます。

標準高さ
6.4 mm
8.1 mm
10.1 mm
12.2 mm



詳細は 14-1 “PCB 寸法 ” を参照してください。

フィールドバス・コネクタ

全ての主要なフィールドバス・システムのコネクタ仕様に対応するため、Anybus-S モジュールは様々なコネクタタイプでの実装が可能です。ピン長 2mm のコネクタも本来フィールドバス・コネクタが実装される位置に実装することも可能です。

一つ目のフィールドバス・コネクタとアプリケーション・コネクタは固定されています。2 つ目のフィールドバス・コネクタも必要とするフィールドバス・システムもありますが、二つ目のコネクタの位置は固定されていません。

詳細は 14-1 “PCB 寸法 ” を参照してください。

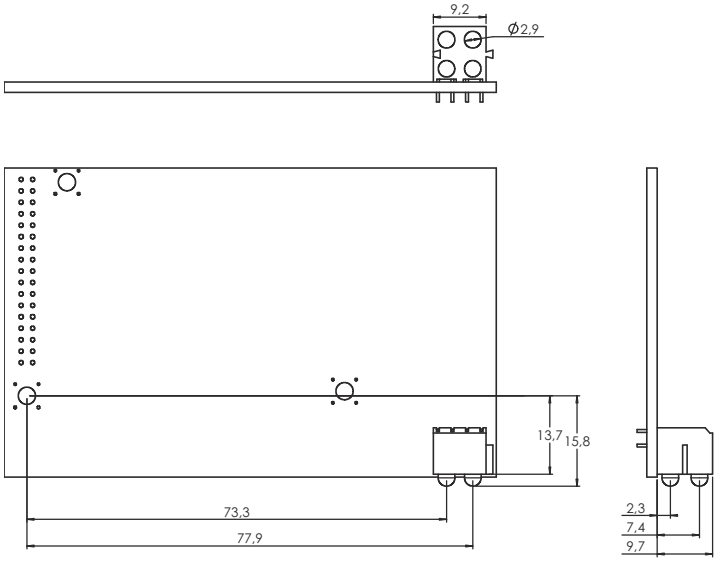
フィールドバス・ステータス表示 LED

全ての Anybus-S モジュールはフィールドバス・ステータス LED が使用されます。LED の実装方向は 180° または 90° が可能です。LED の実装位置は全てのモジュールで標準化されております。

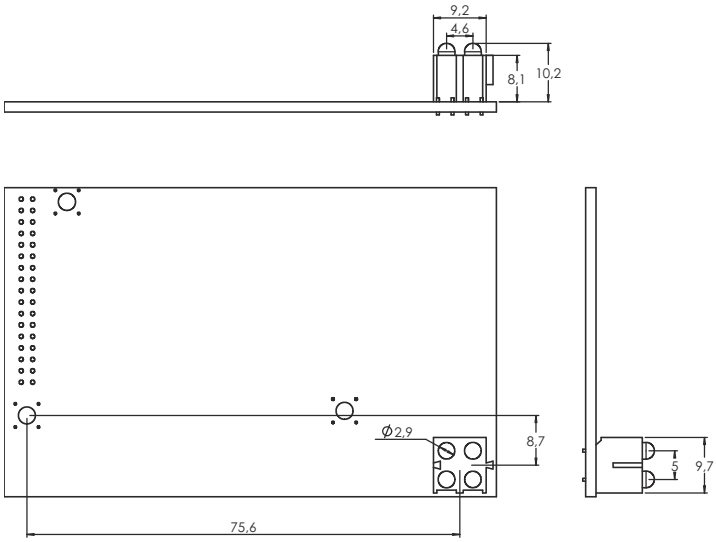
LED がアプリケーション基板に実装される場合、モジュール上の 2.54mm ピッチの 2x4 の LED 端子が使用可能です。

（詳細は 14-1 “PCB 寸法 ” を参照してください。）

Angled LED’s



Straight LED’s



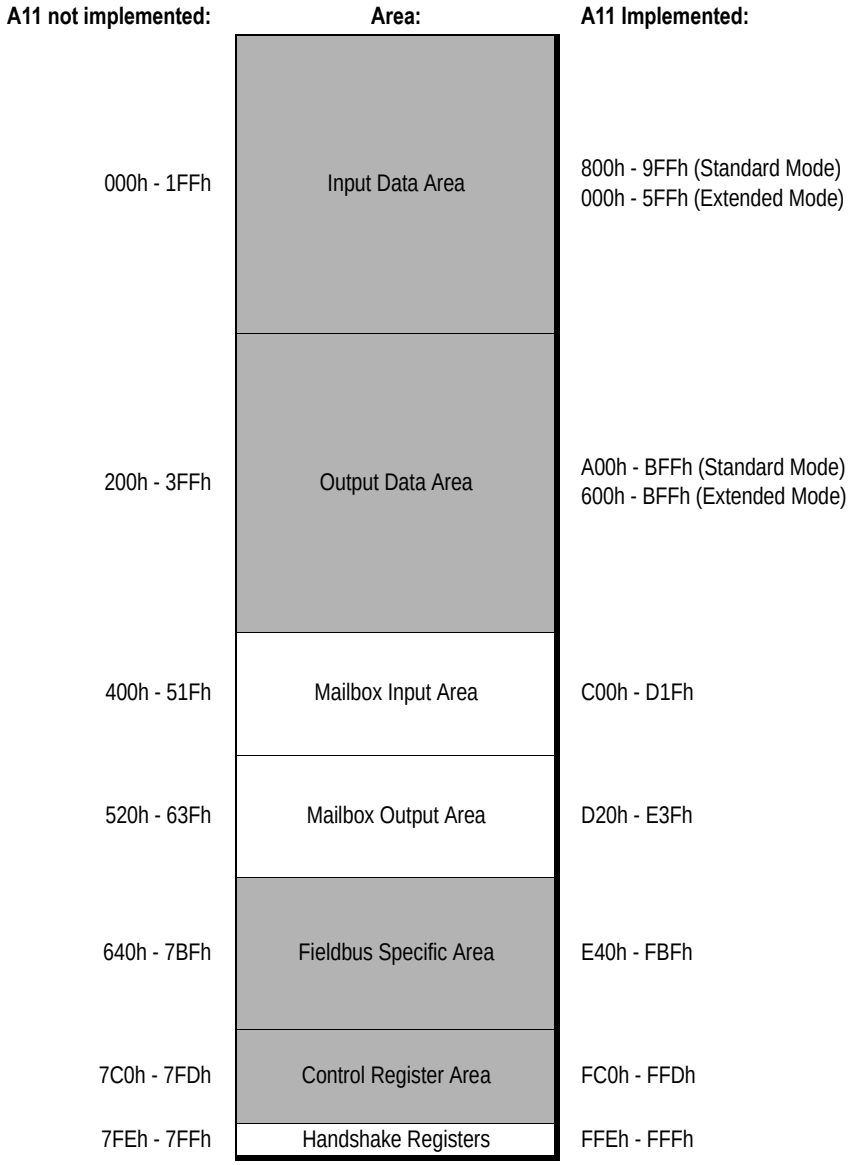
拡張されたメモリ・モード (4K DPRAM)

Anybus-S プラットフォームはフィールドバス・データへのより速いアクセスのために拡張されました。この拡張にはアプリケーション・コネクタの 34 ピンのアドレス・ライン 11（A11）を使用し 2k バイトの代わりに 4k バイトの効果的なアドレス範囲が与えられます。

現時点ではこの機能は Anybus-M Profibus DPV1 マスターと Anybus-M Devicenet マスター / スキャナのみで使用され、将来は Anybus-S でも使用できるかもしれません。

DPRAM の大きさに関わらず、Anybus-S の全てのバージョンは、A11 が実装されていないアプリケーションと下位互換を保証しています。

この機能の実装はメモリ・マップに影響します。下図を参照ください。



この機能を使用するために、アドレス・ライン 11 をアプリケーションに実装する必要があり、ソフトウェアで使用されるアドレス・オフセットは上記のメモリ・マップに関連付けられる必要があります。

注意

概論

Anybus-S は仮想的にどのようなネットワークタイプもサポートするように設計されており、拡張性のあるネットワークイベントと機能を一定の方法でアプリケーションに提供します。

このため Anybus-S は多くのビットコード・ステータス情報を含みます。

実際にどのビットを使用するか、これらのビットの正確な処理はフィールドバスによって異なります。このため、指定のビットまたは機能は指定のステータスや機能に大きく依存し、フィールドバス・システムの変更のときに、多少変更が必要になるかもしれません。これらの差異が考慮されない限り完全に透過的な Anybus-S の実装を行うことはできません。

Anybus-S イーサネットモジュールのソケット・インターフェースのようなフィールドバス個別機能をサポートするため、アプリケーション・ソフトウェアに特別な実装を準備する必要があります。

これらの例外は各フィールドバス・アペンディックスに記載されています。

ロックリリースの動作

出力データ領域の Locked Release が要求されたとき、Anybus モジュールが領域を更新するまで領域には再度アクセスすることができません。例外はありますが、ほとんどのモジュールはデータに変更があるかどうかこの領域を周期的に更新します。モジュールによっては出力領域を出力データに変更があるときだけ更新するモジュールもあります。これはアプリケーションをコンフィグレーションするときに考慮される必要があり、領域を開放するために外部イベントを使用する必要があるかもしれません。

以下のモジュールはデータが変更されたときのみ出力データ領域を更新します：

モジュール	コメント
ABS PROFIBUS DP-V0	-
ABS PROFIBUS DP-V1	-
ABS PROFIBUS DP-V1 I&M	-
ABS EtherNet/IP	-
ABS CANopen	COS[a] データが使用されるときのみ
ABS DeviceNet	COS データが使用されるときのみ

a. COS = Change of state

アプリケーション・インターフェース・ハードウェアの注意

製品のライフサイクルの間、信号特性などを改善するためアプリケーション・インターフェース・ハードウェアにマイナー更新がありました。以下の表中はその変更内容です。一般的に新しい製品は2-1“アプリケーション コネクタ”の記載内容で実装されています。

これを書いている時点で、以下に挙げられていない製品または以下よりも高いリビジョン番号の製品は、2-1“アプリケーション コネクタ”の記載内容より早く実装されるかもしれません。(低いリビジョン番号の製品は既に使用されていないか、少し異なるレジスタ値などをもつ可能性があります。ご使用の製品がこれに該当する場合、HMS に詳細をご確認ください。)

Anybus モジュール	PCB		A4/DE	$\overline{CE}$	$\overline{IRQ}$	$\overline{BUSY}$	A10	A11
	ID #	Revision						
Anybus-M DeviceNet	2102	2.3.1	-	-	*10k*	*10k*	*10k*	*10k*
Anybus-M DPV	2101	1.1.1	-	*10k*	*10k*	*10k*	-	*10k*
Anybus-S Interbus (500k)	3265	1.20 - 1.30	-	-	*10k*	*10k*	*10k*	*x*
Anybus-S Interbus 光ファイバ (500k)	3273	1.01 - 1.10	-	-	*100k*	*100k*	*100k*	*x*
Anybus-S Modbus Plus	4028	1.11	-	-	*10k*	*10k*	*10k*	*x*

-	プルアップ抵抗なし
10k	プルアップ抵抗
x	信号なし

割り込み信号とハードウェア・リセット

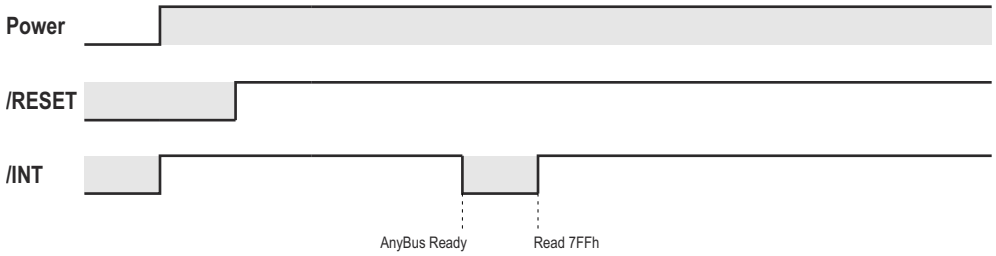
Anybus-S はパワー・オンリセット後に動作開始可能であることを示すため割り込みを発生させます。この割り込みが発生するまでアプリケーションは DPRAM にデータを書き込むことができません。

割り込み信号はデュアル・ポート RAM の内部ロジックによって管理されます; このロジックはハードウェア・リセットではなくパワー・オンリセットによって動作します。例えばリセットボタンなどアプリケーションがハードウェア・リセットを使用している場合、割り込み信号がスタートからローレベルのまま変わらない危険性があります。

割り込みとハードウェア・リセットについて以下のようなケースがあります。[1]

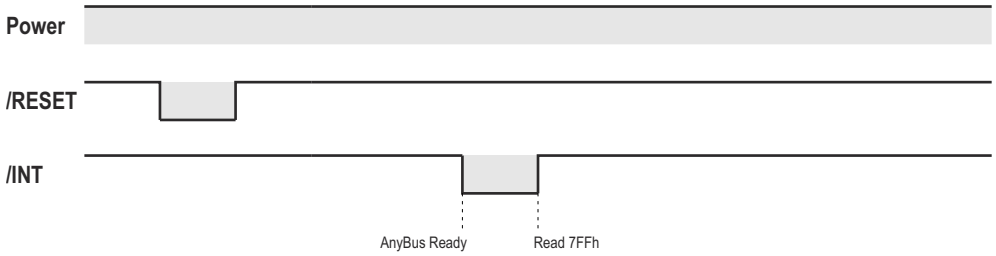
標準のパワー・オンリセット

割り込み信号（/INT）はパワー・オンリセット後ハイレベルに初期化されます。



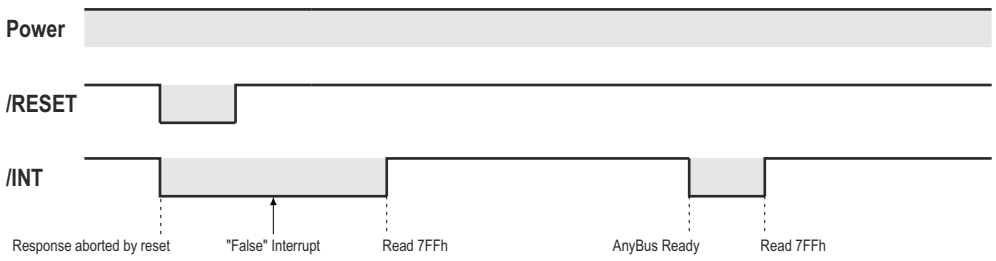
ハードウェア・リセット – ケース 1

標準のスタートアップの場合割り込み信号がハイのときハードウェア・リセットが発生します。



ハードウェア・リセット – ケース 2

ハードウェア・リセットはアプリケーションがハンドシェーク・レジスタを読む前ではなく Anybus-S が要求に対してレスポンスした後にかかります。このケースでは Anybuys-S の準備が完了する前にアプリケーションがハンドシェーク処理を開始する潜在的な危険性があります。

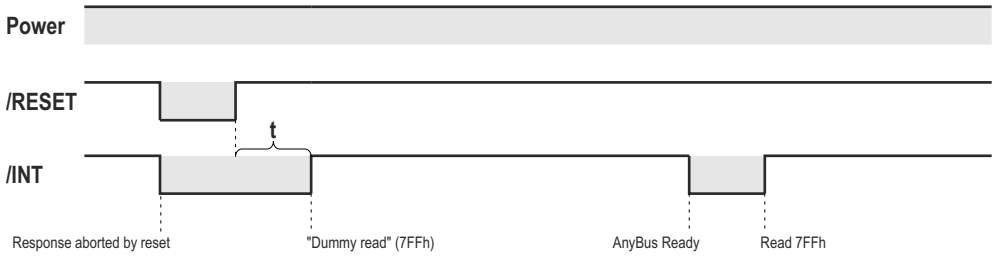


1. これらのダイアグラムは厳密なタイミングではなく、イベントのシーケンスと関係を示しています。

推奨される処理

上記のような場合に安全な初期化をするために、リセット信号が開放されてからローまたは t<120ms 以内の間に、Anybus Indication Register (7FFh) への ” ダミー ” の呼び出しをアプリケーションに追加することを推奨します。

ハードウェア・リセット – ケース 2 の正しい処理



電気的特性

電源仕様

モジュールはデュアル電源ピンを使用します。これらのピンはともにつながれるかホスト・アプリケーションの電源によって別々に電源が供給されます。どちらの方法も、モジュールが適切に動作するために両方のピンに電源を与える必要があります。一般的に両方の電源をつなぐことを推奨します。

シンボル	内容		最小	平均	最大	単位
I_{IN}	消費電流	バス・インターフェース（Pin 1）	-	-	300	mA
		モジュール回路（Pin 5）	-	-	300	mA
		両方（Pin 1 と Pin 5）	-	-	450[a]	mA
V_{CC}	電源電圧 (DC)	バス・インターフェース（Pin 1）	4.75	5.00	5.25	V
		モジュール回路 (Pin 5)	4.75	5.00	5.25	V
	最大リプル (AC)	バス・インターフェース（Pin 1）	-	-	± 100 pp	mV
		モジュール回路 (Pin 5)	-	-	± 100 pp	mV
t_A	電源立ち上がり時間 (0.1 V_{CC} から 0.9 V_{CC})		-	-	50	ms

a. 最大入力電流値はバス・インターフェースとモジュールの電流値の合計です。

注意 : 上記テーブル値はほとんどのモジュールで有効です。これを書いている時点での 1 つの例外は Anybus-S Profinet IRT FO です。HMS に詳細をご確認ください。

モジュールの電源の近くにバルク・キャパシタを追加することもできます。

キャパシタタイプ	値
セラミック	22uF / 6.3V
電解	100uF / 16V

信号特性

シンボル	内容		最小	平均	最大	単位	テスト条件
V_{OH}	出力 High 電圧	D_0-D_7	2.4	-	-	V	I_{OH} = -4.0 mA
		Tx	3.5	-	-	V	I_{OH} = -1.0 mA
V_{OL}	出力 Low 電圧	D_0-D_7	-	-	0.4	V	I_{OL} = 4.0 mA
		$\overline{BUSY}$, $\overline{IRQ}$	-	-	0.5	V	I_{OL} = 16.0 mA
		Tx	-	-	0.4	V	I_{OL} = 1.60 mA
V_{IH}	入力 High 電圧	A_0-A_{11}, D_0-D_7, $\overline{CE}$, $\overline{OE}$, R/W	2.2	-	-	V	-
		Rx	2.0	-	-	V	-
		$\overline{RES}$	3.5	-	-	V	-
V_{IL}	入力 Low 電圧	A_0-A_{11}, D_0-D_7, $\overline{CE}$, $\overline{OE}$, R/W	-	-	0.8	V	-
		Rx	-	-	0.8	V	-
		$\overline{RES}$	-	-	0.8	V	-
I_{IX}	入力負荷電流	A_0-A_{11}, D_0-D_7, $\overline{CE}$, $\overline{OE}$, R/W	-5	-	+5	µA	$GND \le V_I \le V_{CC}$
I_{OZ}	出力リーク電流	D_0-D_7	-5	-	+5	µA	$GND \le V_O \le V_{CC}$ (Output Disabled)
-	パルス幅	$\overline{RES}$	1.0	-	-	µs	-

注意 : プルアップ・レジスタはこれらの特性に含まれません。

推奨される PE とシールド処理

適切な EMC に対する動作をさせるため、フィールドバスの仕様によってモジュールはプロテクティブ・アースに接続される必要があります。

Anybus-S には三つのメタル加工された取り付け穴があり、そのうち二つはアプリケーションとモジュールの電気的接続を拡張する場合に使用します。二つの取り付け穴のうちの一つはプロテクティブ・アース（PE）に接続します。この穴をここでは 'PE- ホール ' と呼びます。

アプリケーションの筐体が絶縁性である場合、PE- ホールの使用を推奨します。しかし、筐体が導電性である場合には以下の二つの方法があります :

- **PE- ホールを介したプロテクティブ・アースへの接続**
  この場合、グランドループなどを避けるためフィールドバス・コネクタはアプリケーションの筐体から完全に絶縁されている必要があります。
- **フィールドバス・コネクタ / アプリケーション筐体を介したプロテクティブ・アースへの接続**
  この場合、グランドループなどを避けるため PE- ホールはアプリケーションから絶縁されている必要があります。

アプリケーションはフィールドバス・システムの互換性を保証するためこれらの方法をサポートする必要があります。取り扱いが異なる二つ以上のフィールドバス・コネクタを使用する場合この問題はさらに複雑になります。これらの場合、フィールドバス・コネクタをアプリケーションの筐体から絶縁し PE- ホールを使用することを推奨します。

各フィールドバスの仕様ごとの PE/ シールド制約についての詳細については HMS にお問い合わせください。

使用環境

温度

動作中

+0 ～ +70 ℃
（試験は IEC-68-2-1 と IEC 68-2-2 に準拠しています。）

停止中

-15 から +85 ℃
（試験は IEC-68-2-1 と IEC 68-2-2 に準拠しています。）

相対湿度

製品は非凝結で 5% から 95% の相対湿度に対応するように設計されています。

試験は IEC 68-2-30 に準拠しています。

EMC 規格

EMC 事前規格試験が *Electromagnetic Compatibility Directive 2004/108/EC* に従い行われています。

準拠している規格の詳細は個々の EMC 事前規格の文書にてご確認頂けます。この文書については www.anybus.com からダウンロード可能です。

事前に取得されている規格

フィールドバス認証

Anybus-S モジュールは事前に認証が取得され各フィールドバス規格に準拠していることが確認されています。モジュールは事前に認証を取得していますが、最終的な製品は再度フィールドバス規格の認証が必要です。

事前の認証は以下の条件で有効です。

- 標準のフィールドバス・コネクタ
- フィールドバス特有の初期化パラメータを使わない
- デバイス・ファイル、‘.GSD’ あるいは ‘.EDS’ の内容の変更をしない

これらの条件を全て満たさない場合、モジュールは事前に認証を取得したとはみなされず再度認証を取得する必要があります。

詳細についてはフィールドバス規格団体または HMS にお問い合わせください。

CE- マーク

一般的に Anybus-S モジュールはヨーロッパ CE 規格に準拠しています。Anybus-S そのものは認証を取得していますが、アプリケーションによっては最終的な製品の認証の取得が必要な場合があります。

UL/cUL- 認証

Anybus-S モジュールは米国 (NRAQ2) とカナダ (NRAQ8) 向けに、UL508, “Programmable Controller” として UL/cUL の認証を受けています。

トラブルシューティング

モジュールがデータ交換しない

- Anybus-S の Watchdog LED を確認してください。LED が緑 1hz 点滅していない場合、モジュールは初期化されていないためデータ交換できません。

初期化中の問題

- 初期化のメールボックス・メッセージのレスポンスにエラー情報が含まれていない。
- 最大 I/O/ パラメータ・データ・サイズはフィールドバス・システムごとに異なります。初期化パラメータが現在使用されている Anybus-S バージョンに合っていることを確認してください。
- HW_CHK メールボックス・メッセージが使用されている場合、モジュールはレスポンスしますか？レスポンスしない場合モジュールはハードウェアに問題があります。問題の原因は Anybus-S Watchdog LED に表示されます。
- 'Load from FLASH' メールボックス・コマンドを使用している場合 - Flash に有効なメールボックス・シーケンスが含まれていることを確認してください。

本ドキュメントのサンプルドライバが動作しない

- 多くの場合ドライバ例は使用するためには指定の条件に合わせるための変更が必要です。サンプルドライバは指定のアプリケーションへの実装の仕方ではなく、一般的な動作の仕方が示されています。
- サンプルドライバは教育的目的で提供されており完全ではありません。初期化のための複数の基本的な機能、エラー処理などは意図的に実装されていないためお客様にて実装して頂く必要があります。

共通のハンドシェークの問題、ソリューション

- モジュールが Anybus インジケーション・レジスタの前のコマンドにレスポンスしない限りアプリケーション・インジケーション・レジスタに新しいコマンドは書くことができません。
- 要求があったときのみアプリケーション・インジケーション・レジスタへの書き込みをしてください。レジスタへの不必要なアクセスは処理に負荷を与えます。アクセスする領域にはできるだけ Locked リクエストを使用してください。

リセット関連の問題

- アプリケーションが例えばリセットボタンのようなハードウェア・リセットを使用する場合、スタートから割り込み信号がローレベルから変わらない危険性があります。このため適切な機能を保証する必要があります。C-1 " 割り込み信号とハードウェア・リセット " を参照してください。