NAPYDC839GT02 共通 CAN プログラミング技術資料

対応 MCU 名
TMPM350FDTFG（TX03）

＜ご注意＞

下記の利用条件をご了解の上本技術情報をご利用ください。

＜本技術情報の利用条件＞

１．　ホームページ上で公開される共通 CAN プログラミングに関する情報（以下本技術情報と呼びます）は、あくまでもマイコン導入時の評価・実験用途として開示されるものであり、生産ライン用プログラマとして応用されることを想定していません。
本技術情報を、フラッシュマイコンを組み込んだ製品等の生産用途用としてご利用になる際は、お客様サイドで本技術情報に関する妥当性を十分検討のうえご利用ください。

２．　横河ディジタルコンピュータは、正確な技術情報の開示に努力しますが、本技術情報の内容について製造責任を負うものではありません。
本技術情報を応用した結果についての責任は、お客様に帰属するものとします。

３．　弊社では、本技術情報を生産用途などに応用するお客様を対象に、本技術情報に関する技術支援サービス（有償）を行っております。
詳細は、弊社または弊社代理店までお問い合わせください。（日本国内のみ）

ご注意

NAPYDC839GT02 の適用 NETIMPRESS シリーズ本体は、NETIMPRESS air(AF930)です。
G-NETIMPRESS,NETIMPRESS next ではご使用になれません。

作成日：2016 年 05 月 31 日　正式版

エンベデッドソリューション事業統括本部

変更履歴

変更日付	変更内容
2016.03.11	暫定版 新規作成
2016.03.22	正式版
2016.05.31	誤記訂正、説明文修正

目次

1. 概要

１．１　動作条件

項目	内容	ユーザ設定※
対象マイコン	TMPM350FDTFG (TX03)	
書き込み対象アドレス	#00008000～#0007FFFF　*1	不可
インタフェイス	CAN 通信（拡張・標準 ID 対応）	
ボーレート	500Kbps、1Mbps、250Kbps、125Kbps （デフォルト 500Kbps）	可
動作クロック	88MHz （原振クロック 16MHz）	可
CAN チャネル番号	チャンネル 0	可
モード制御端子	なし	不可
バンドルファイル	BTP ファイル、YSM ファイル、KEY ファイル AMK ファイル	可
プローブ	AZ915 AZ916	不可
CCP バージョン	Version 2.1 をベースとする *2	不可
開発環境	IAR Embedded Workbench for ARM 6.21.1.3149　＊3	不可
コンパイラバージョン	IAR C/C++ Compiler for ARM 6.21.1.52794　＊3	不可
最適化	高(H) バランス　＊3	不可
プログラマ共通仕様	特定領域プロテクト機能（定義体にてアクセス禁止パラメータ対応）	
IBL 内のスタックレベル	200byte ＊4	不可

※表中の「ユーザ設定」が「可」以外の項目は絶対に設定変更しないで下さい。以降ページも同様です。

*1 block0/block1 領域変換は対象外とします。またライトプロテクトはサポートしません。

*2 CCP を拡張したプロトコルです。完全互換性はありません。

*3 開発環境及びコンパイラバージョンは弊社にて動作確認を行ったバージョンになります。
他のバージョンのものを使用された場合の動作は保証致しかねますのでご注意ください。

*4 各関数の使用スタックサイズを単純に加算した値ですので、実際にはより少なくなります。

１．２　マイコンパック内容

本マイコンパックに関する公開ドキュメント一覧（和文）

項　　目	ドキュメント名(ファイル名)	備　　考
マイコンパックマニュアル	MNJ-NAPYDC839GT02	
CAN 共通プログラミング技術資料	TR- NAPYDC839GT02	本書
マイコンパック	NAPYDC839GT02	・FDF シート内容 サンプル APL オブジェクト KEY ファイル YSM ファイル
サンプルプログラム アプリソフト例（APL） ﾕｰｻﾞｲﾆｼｬﾗｲｽﾞﾙｰﾁﾝ（IPR） ｲﾆｼｬﾙﾌﾞｰﾄﾛｰﾀﾞ（IBL） 書き込み制御ﾌﾟﾛｸﾞﾗﾑ(WCP)	EX_NAPYDC839GT02 |-HEADER |　|-y_ibl.h |　|-y_init.h |　|-user_init.h |　|-fix-iotmpm350fdtfg.h |-IBL_NAPYDC839GT02 |　|-RamFunc.c |　|-sum_data.c |　|-user_ap_start.s |　|-user_apl.c |　|-user_ipr_start.s |　|-user_ipr.c |　|-user_key.c |　|-y_ibl.c |　|-y_ibl_init.s |　|-TMPM350_Flash.icf |　|-usr_api.c |　|-usr_api.h |-WCP_NAPYDC839GT02 |-y_wcp_init.s |-y_wcp.c |-TMPM350_wcp.icf	 固有値定義ファイル 初期設定ファイル ｴﾝﾄﾘｱﾄﾞﾚｽ定義ﾌｧｲﾙ 修正済みﾚｼﾞｽﾀ定義ファイル RAM で実行するﾍﾙﾊﾟｰ関数群 APL サムサンプル ﾕｰｻﾞ APL ｽﾀｰﾄｱｯﾌﾟﾙｰﾁﾝｻﾝﾌﾟﾙ ﾕｰｻﾞ APL ｻﾝﾌﾟﾙﾌｧｲﾙ IPR ｽﾀｰﾄｱｯﾌﾟﾙｰﾁﾝｻﾝﾌﾟﾙ ﾕｰｻﾞ IPR ｻﾝﾌﾟﾙﾌｧｲﾙ ﾊﾟｽﾜｰﾄﾞ設定ﾌｧｲﾙ CAN ﾘﾌﾟﾛｸﾞ用ﾌﾞｰﾄﾛｰﾀﾞ ﾌﾞｰﾄﾛｰﾀﾞｽﾀｰﾄｱｯﾌﾟﾙｰﾁﾝ ﾘﾝｸﾃﾞｨﾚｸﾃｨﾌﾞﾌｧｲﾙ ﾉｰﾄﾞ別 CAN ID 設定 同ﾌﾟﾛﾄﾀｲﾌﾟ宣言ﾌｧｲﾙ WCP ｽﾀｰﾄｱｯﾌﾟﾙｰﾁﾝ CAN ﾘﾌﾟﾛｸﾞ用書き込み制御ﾌﾟﾛｸﾞﾗﾑ ﾘﾝｸﾃﾞｨﾚｸﾃｨﾌﾞﾌｧｲﾙ

２．用語の定義と略語

UCOP

Universal CAN Open Protocol の略です。
弊社が提唱した MCU に依存しない CAN 共通プロトコルです。

IPR

Initial Processing Routine の略です。
イニシャル・プロセッシング・ルーチン・プログラムです。
プログラミング上、初期化しなければならない処理を記述いただきます。
お客様サイドでカスタマイズしていただきます。

IBL

Initial Boot Loader の略です。
イニシャル・ブート・ローダ・プログラムです。
プログラミングエントリの判定、書き込み制御プログラム（WCP）の受信
及び内蔵 RAM への書き込みをおこないます。
基本的にはそのままご使用していただけます。

WCP

Write Control Program の略です。
書き込み制御プログラムです。
拡張子が“.BTP”のファイルです。
デバイスに対する消去・書き込み・読み出し等のプログラムが書かれています。
基本的にはそのままご使用していただけます。

APL

アプリケーション・プログラムです。
お客様のアプリケーションプログラムです。

ReProg Area

お客様のアプリケーションプログラムを書き込む ROM エリアです。

UCOP リプログモード

UCOP を利用してアプリケーションプログラムの消去／書き込みを行うモードを UCOP リプログモードと呼びます。
UCOP リプログモードへは３つあるエントリー方法のどれかを通ってエントリーします。

r Entry

レスキュー・エントリー

UCOP リプログモードに入るエントリー方法の 1 つです。

電源投入後、一定期間(※)経過後、約 10mSec 間 Connect コマンドを待ちます。

この約 10mSec 間に Connect コマンドを受信すると r Entry になります。

※この一定期間は電源投入後 Connect コマンド受信待ちを開始するまでの時間で IPR の処理時間などで時間が変わってきます。

n Entry

ノーマル･エントリー

UCOP リプログモードに入るエントリー方法の 1 つです。

IBL 内で Connect コマンドを受信するまで待ちつづけます。

u Entry

ユーザ･エントリー

UCOP リプログモードに入るエントリー方法の 1 つです。

APL 内で Connect コマンドを受信した場合のエントリー方法です。

APL 内での Connect コマンド受信方法は、お客様に作り込んで頂きます。

Primary ID

初期設定ファイル(y_init.h)の ID_P_NI と ID_P_MCU に設定されている

アイデンティファイヤです。

Secondary ID

UCOP リプログモード中に追加したアイデンティファイヤです。

ROM の一部にアイデンティファイヤ登録領域(以下「Secondary ID 領域」という)を確保し、その領域へ追加したアイデンティファイヤを登録します。

CAN メッセージ ID

CAN プロトコルのフレームにおける、アイデンティファイヤのことです。

KILL レジスタ

UCOP リプログモードを強制終了するかどうかを判定する機能です。

ROM の一部を KILL レジスタ領域とします。

KILL レジスタ領域が All FFh でない場合、KILL レジスタ ON となります。

KILL レジスタ領域が All FFh の場合、KILL レジスタ OFF となります。

KILL レジスタ ON 時は、リセット実行処理関数をコールし UCOP リプログモードから抜けます。

KILL レジスタ OFF 時は、UCOP リプログモードを続行します。

ステーションアドレス

ターゲット毎に２バイト（リトルエンディアン）で設定します。

初期設定ファイル(y_init.h)の CCP_STATION で設定します。

Connect コマンド、Disconnect コマンドのフレームにステーションアドレス情報が入っています。

(UCOP プロトコルのマニュアル参照)

Connect コマンドにおいてアイデンティファイヤ、ステーションアドレスが一致した場合のみ IBL は UCOP リプログモードにエントリーします。

Disconnect コマンドのステーションアドレスは無視します。

パスワードチェック領域

UCOP には「暗号機能※1」があります。暗号機能においてチェックを行うパスワード数はある領域内において７～２５６バイト迄で設定します。その領域を「パスワード設定領域」といいます。

この領域はお客様サイドで変更していただくことが可能です。

※１：「暗号機能」については「UCOP_CAN PROGRAMMER」のインストラクションマニュアルを参照してください。

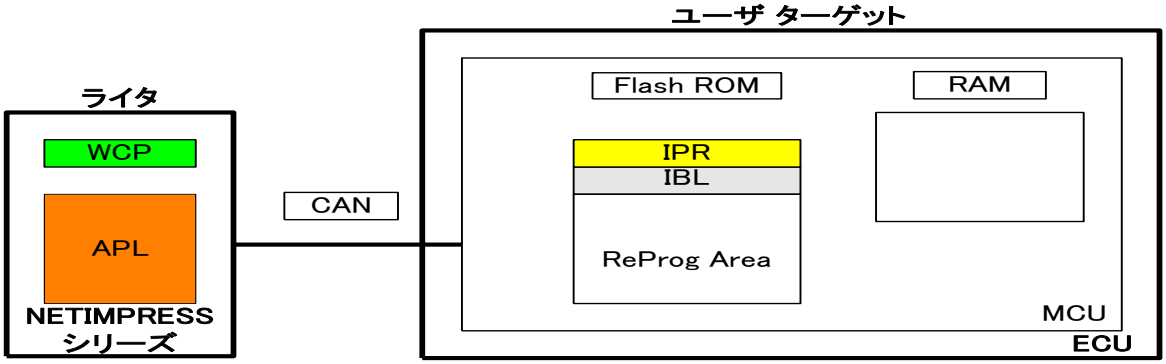
ユーザアプリ領域サム値チェック

UCOP では IBL において、お客様のアプリケーションプログラムが既に書かれているかどうかをサム値にて判断します。このサム値チェックのことを「ユーザアプリ領域サム値チェック」といいます。

サム値計算領域は”y_init.h”ファイルで変更することが可能です。

3. 機能概要

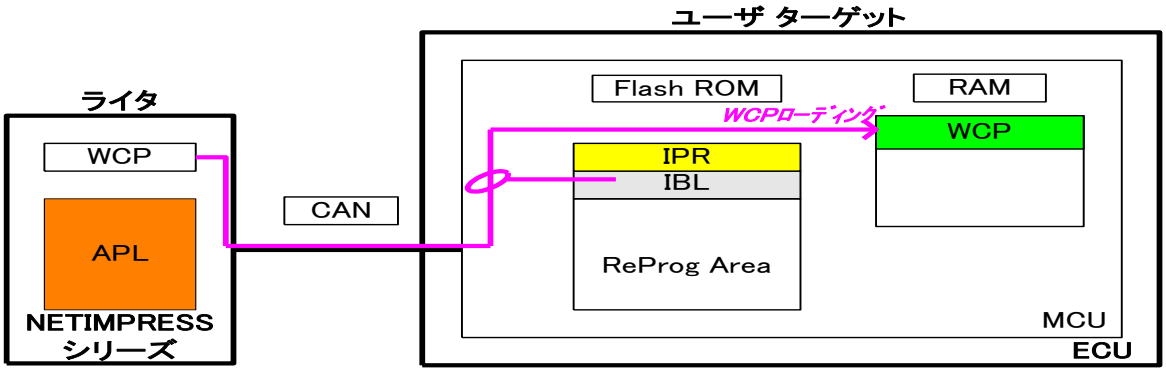
3．1　UCOP システム構成図



1．予め、IPR と IBL はターゲット MCU の Flash ROM の一部に書き込んでおきます。

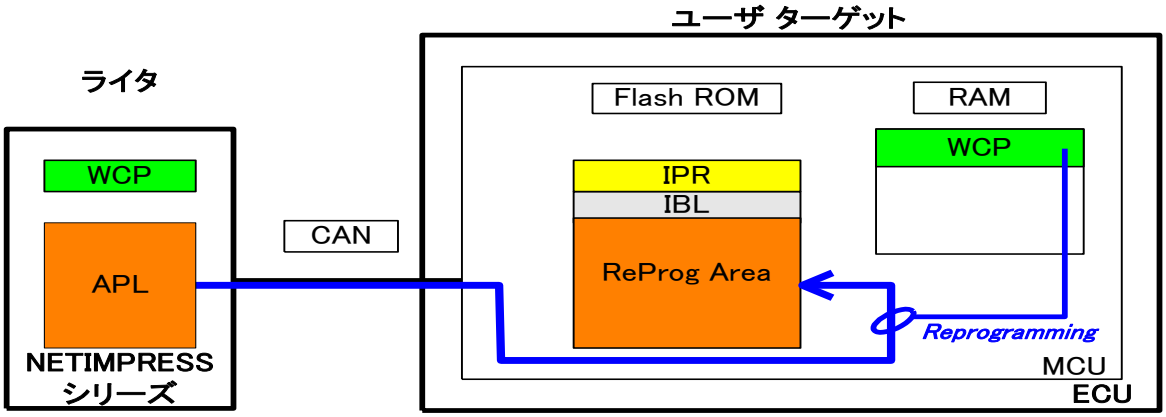
2．リセット解除後、IPR において UCOP リプログモード実行に際して最低限必要なシステムの初期化を行います。

3．IPR で初期化終了後、制御が IBL へ移行し各エントリー（r Entry, n Entry, u Entry）のどれかを介してターゲットは UCOP リプログモードに入ります。



4．ライタは IBL と通信をおこない WCP をターゲット MCU に順次送信します。IBL はライタより受信した WCP をターゲット MCU の内蔵 RAM に書き込みます。

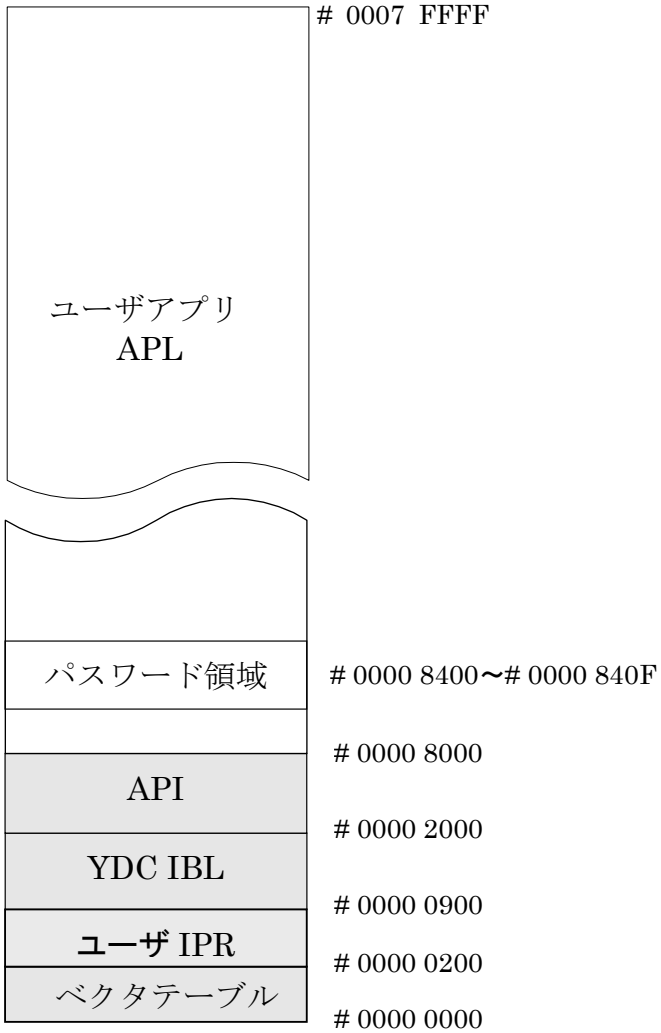
5．WCP を全て内蔵 RAM に書き込んだ後、ターゲット MCU 側の制御は WCP へ移行します。



6．ライタは WCP と通信をおこないライタにある APL を ReProg Area に書き込みます。

３．２　ROM

３．２．１　ROM構成図

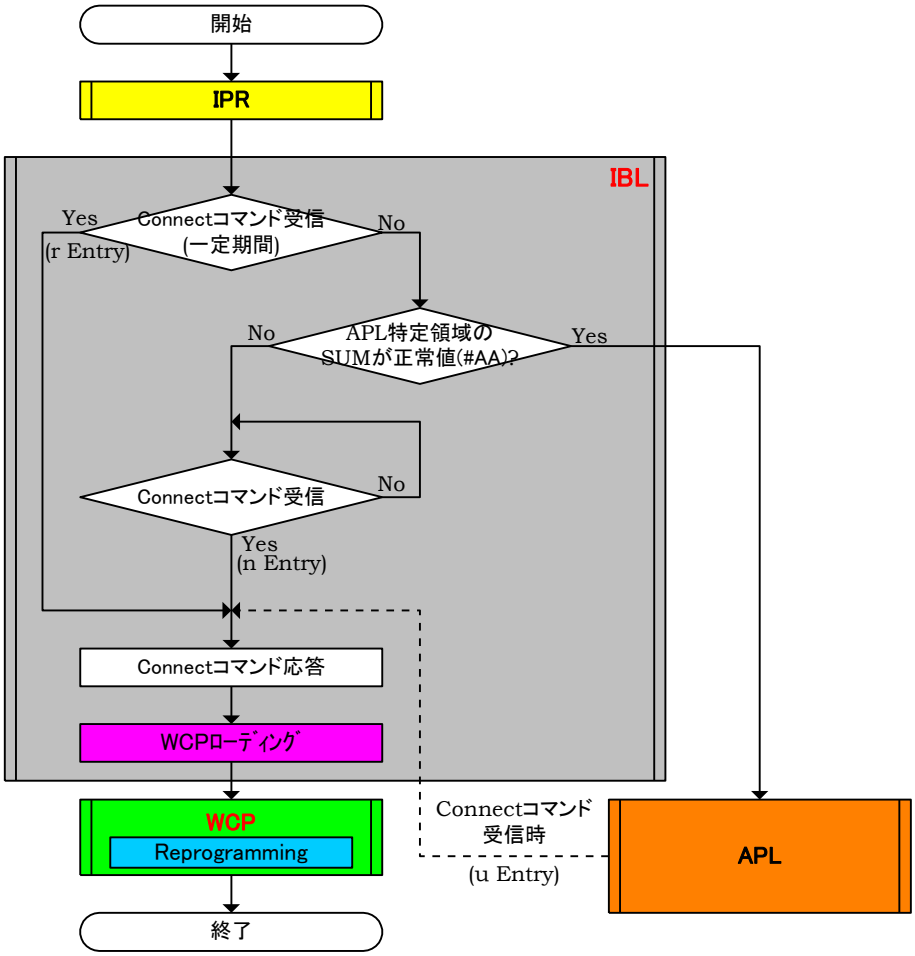


３．２．２　消去ブロックアドレス

ブロック	ブロックアドレス
ブロック 5	#00060000～#0007FFFF
ブロック 4	#00040000～#0005FFFF
ブロック 3	#00020000～#0003FFFF
ブロック 2	#00010000～#0001FFFF
ブロック 1	#00008000～#0000FFFF
ブロック 0	#00000000～#00007FFF

灰色部は IBL、IPR 域で書き換え禁止領域です。

３．３　プログラムエントリモードフローチャート



１．電源投入後、IPR 処理をおこない一定期間(約 10mS)Connect コマンドを待ちます。この期間内に Connect コマンドを受信しますと r Entry で UCOP リプログモードに遷移します。

２．一定期間(約 10mS)内に Connect コマンドを受信しなかった場合、ユーザアプリ領域サム値チェックの SUM 値を計算します。SUM 値が#AA ならば APL へジャンプし、お客様のアプリケーションプログラムが実行されます。APL 側で Connect コマンドを受信しますと u Entry で UCOP リプログモードに遷移します。

３．ユーザアプリ領域サム値チェックの SUM 値が#AA 以外ならば、Connect コマンドを受信するまで Connect コマンド受信待ちになります。この状態で Connect コマンドを受信しますと n Entry で UCOP リプログモードに遷移します。

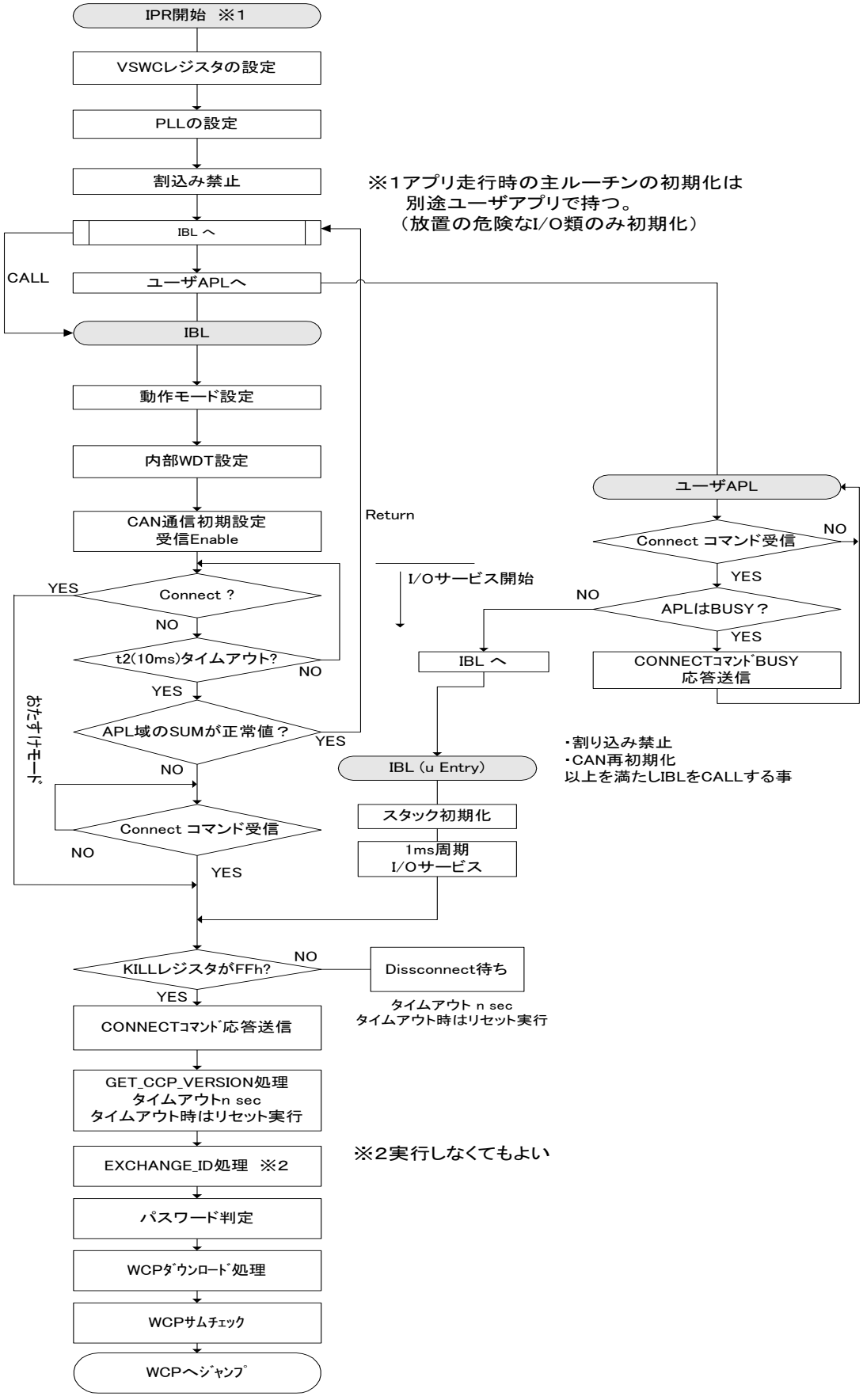
※ライタは Connect コマンド発行後規定時間(25mS)応答がない場合、Disconnect コマンドを発行し、ターゲット MCU との接続を解除します。

※ターゲット MCU は各エントリー方式で UCOP リプログモードに遷移後、Disconnect コマンドを受信するまでライタとは接続状態にあるものとします。

※Disconnect コマンドはデバイスファンクション終了時にライタが発行します。

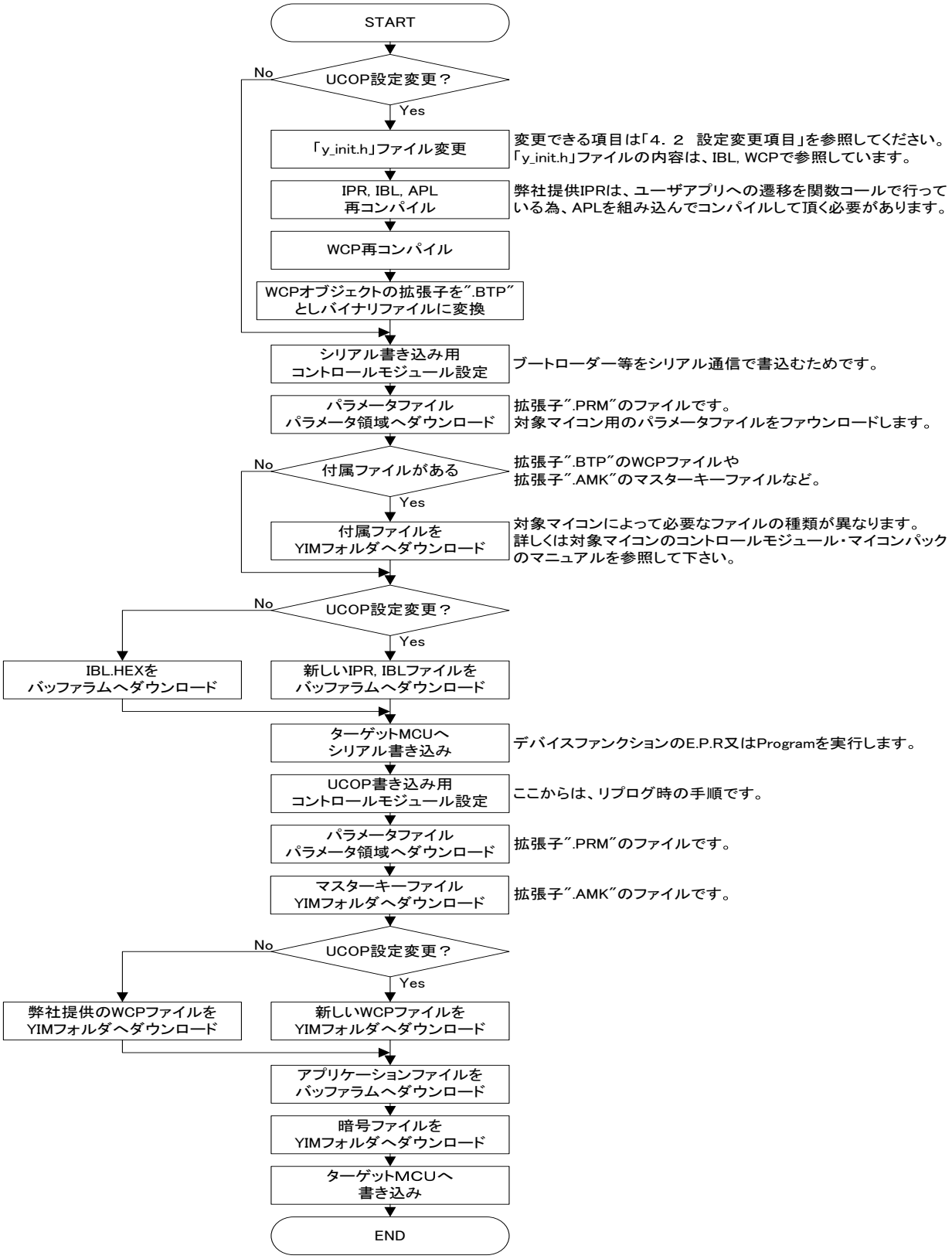
※ターゲット MCU は Disconnect コマンド受信後、リセット状態に戻るものとします。

３．４　IBL プログラム概略フローチャート



４．初期導入手順

４．１　書き込み手順フロー



４．２　設定変更項目

４．２．１　CANボーレートの変更

「5.1　CAN ボーレート」、「5.5　CAN ビットレートプリスケーラレジスタ値」、「5.6　CAN ビットコンフィグレーションレジスタ値」を参照してください。

４．２．２　動作クロックの変更

４．２．２．１　入力クロック値の変更

「5.2　入力クロック周波数」、「5.5　CAN ビットレートプリスケーラレジスタ値」、「5.6 CAN ビットコンフィグレーションレジスタ値」を参照してください。

４．２．２．２　クロック逓倍比の変更

「5.3　クロック逓倍比」、「5.5　CANビットレートプリスケーラレジスタ値」 、「5.6　CANビットコンフィグレーションレジスタ値」を参照してください。

４．２．２．３　クロック分周比の変更

「5.4　クロック分周比」、「5.5　CANビットレートプリスケーラレジスタ値」、「5.6　CANビットコンフィグレーションレジスタ値」を参照してください。

４．２．３　ビットタイミングパラメータ、ボーレートプリスケーラの変更

「5.5　CAN ビットレートプリスケーラ値」、「5.6　CAN ビットコンフィグレーションレジスタ値」を参照してください。

４．２．４　パスワードチェック領域の変更

「5.7　パスワードチェック領域開始アドレス」、「5.8　パスワードチェック領域終了アドレス」を参照してください。

４．２．５　ユーザアプリ領域サム値チェック領域の変更

「5.9　ユーザアプリ領域サム値チェック開始アドレス」、「5.10　ユーザアプリ領域サム値チェック終了アドレス」を参照してください。

４．２．６　ウォッチドッグタイマサービスの変更

４．２．６．１　ウォッチドッグタイマサービス有無の変更

「5.11　I/O ポートサービス対応フラグ」を参照してください。

４．２．６．２　ウォッチドッグタイマサービス周期の変更

「5.12　I/O ポートサービス周期」を参照してください。

４．２．６．３　ウォッチドッグタイマサービス用ポートの変更

「5.13　I/O ポートサービス用ポート変更方法」を参照してください。

４．２．７　Primary IDの変更

「5.14　Primary ID」、「5.15　CAN ID フォーマット設定」を参照してください。

４．２．８　ステーションアドレスの変更

「5.16　ステーションアドレス」を参照してください。

４．２．９　内蔵ウォッチドッグタイマの変更

「5.17　内蔵ウォッチドッグタイマ設定」、「5.18　内蔵ウォッチドッグタイマ周期設定」を参照してください。

５．UCOP設定変更方法

UCOP の一部の設定は、お客様のシステムに応じて変更していただくことが可能です。
ターゲット MCU 側の各種設定を行っている初期設定ファイル“y_init.h”とライタ側両方の変更が必要な項目もあります。
初期設定ファイル“y_init.h”を変更された場合は、「IPR, IBL, WCP」のファイルを再コンパイルしていただく必要があります。
ライタ側の変更は AZ990-air Connect を用いて行います。
一部設定につきましてはライタのファンクション機能を用いて変更することが出来ます。
air Connect の詳細な操作方法は air Connect のインストラクションマニュアルをご参照ください。

５．１　CAN ボーレート

CAN 通信のボーレートを変更するには、初期設定ファイル“y_init.h”とライタ側両方の変更が必要です。

①．初期設定ファイル“y_init.h”設定変更

“CAN_BAUD”にボーレート値に下記の表の設定値を設定してください。

ボーレート値	設定値
1 Mbps	1000
500 Kbps	500
250 Kbps	250
125 Kbps	125

※必ずマイコンのビットコンフィグレーションレジスタ値も再計算し、初期設定ファイルの” CAN_BCR1_DATA”と” CAN_BCR2_DATA”も再設定してください。
「5.5 CAN ボーレートプリスケーラレジスタ値」と「5.6 CAN ビットコンフィグレーションレジスタ値」をご参照ください。

②．ライタ側設定変更

ｉ．air Connect での変更

Specific Parameter のアドレス#0C2,0C3 を変更することでボーレートを変更できます。
ボーレート値とアドレス#0C2,0C3 の値との関係は下記の表のようになっています。

ボーレート値	#0C2 の値	#0C3 の値
1 Mbps	0x 34	0x 40
500 Kbps	0x 34	0x 41
250 Kbps	0x 34	0x 43
125 Kbps	0x 34	0x 47

※　「5.19　Specific Parameter 変更方法」を参照下さい。

ⅱ．メニューからの変更

“SUB SETTING”メニューの”CAN BAUDRATE SETING”からボーレート変更を行います。

上下キーで設定したいボーレートを選択します。

５．２　入力クロック周波数

ターゲット MCU の入力クロック周波数を変更するには、初期設定ファイル“y_init.h”ファイルの変更が必要です。

①．初期設定ファイル“y_init.h”設定変更

“CLK_EXT”に入力クロック周波数値を 10 倍した値を設定してください。

例）16MHz の場合、160 と設定

※必ずマイコンのビットコンフィグレーションレジスタ値も再計算し、初期設定ファイルの” CAN_BCR1_DATA”と” CAN_BCR2_DATA”も再設定してください。

「5.5 CAN ビットレートプリスケーラレジスタ値」と「5.6 CAN ビットコンフィグレーションレジスタ値」をご参照ください。

５．３　クロック逓倍比

ターゲットMCUに入力されたクロックをPLL逓倍回路などにより逓倍して動作周波数とする場合、その逓倍比を設定します。

逓倍比を変更するには、初期設定ファイル“y_init.h”の変更が必要です。

①．初期設定ファイル“y_init.h”設定変更

“CLK_EXT_MULT”に逓倍比を設定してください。

例）逓倍比が「×11」の場合、11 と設定

※必ずマイコンのビットコンフィグレーションレジスタ値も再計算し、初期設定ファイルの” CAN_BCR1_DATA”と” CAN_BCR2_DATA”も再設定してください。

「5.5 CAN ビットレートプリスケーラレジスタ値」と「5.6 CAN ビットコンフィグレーションレジスタ値」をご参照ください。

５．４　クロック分周比

ターゲット MCU に入力されたクロックを分周回路などにより分周して動作周波数とする場合、その分周比を設定します。

分周比を変更するには、初期設定ファイル“y_init.h”の変更が必要です。

①．初期設定ファイル“y_init.h”設定変更

“CLK_PLL_DIV”に分周比を設定してください。

例）分周比が「÷2」の場合、2 と設定

※必ずマイコンのビットコンフィグレーションレジスタ値も再計算し、初期設定ファイルの” CAN_BCR1_DATA”と” CAN_BCR2_DATA”も再設定してください。

「5.5 CAN ビットレートプリスケーラレジスタ値」と「5.6 CAN ビットコンフィグレーションレジスタ値」をご参照ください。

５．５　CAN ビットレートプリスケーラレジスタ値

ビットレートプリスケーラレジスタの値を設定します。
ターゲット MCU の動作周波数、CAN ボーレートを変更する場合に変更してください。
ビットレートプリスケーラレジスタの値を変更するには、初期設定ファイル“y_init.h”の変更が必要です。

①．初期設定ファイル“y_init.h”設定変更
“CAN_BCR1_DATA”に設定値を設定してください。
ビットレートプリスケーラレジスタ値には、0～255(0x00～0xFF)までの値を設定してください。
設定値はマイコンのマニュアルを参照して計算してください。
※必要に応じて CAN ボーレートやクロック関連の再設定を行ってください。
※「5.1　CAN ボーレート」「5.2　入力クロック周波数」「5.3　クロック逓倍比」「5.4　クロック分周比」をご参照ください。

５．６　CAN ビットコンフィグレーションレジスタ値

ビットコンフィグレーションレジスタの値を設定します。
ターゲット MCU の動作周波数、CAN ボーレートを変更する場合に変更してください。
ビットレートレジスタの値を変更するには、初期設定ファイル“y_init.h”の変更が必要です。

①．初期設定ファイル“y_init.h”設定変更
“CAN_BCR2_DATA”に設定値を設定してください。
ビットコンフィグレーションレジスタは 32 ビットレジスタです。
設定値はマイコンのマニュアルを参照して計算してください。
※必要に応じて CAN ボーレートやクロック関連の再設定を行ってください。
※「5.1　CAN ボーレート」「5.2　入力クロック周波数」「5.3　クロック逓倍比」「5.4　クロック分周比」をご参照ください。

５．７　パスワードチェック領域開始アドレス

ReProg Area 内で暗号機能に使用する領域の開始アドレスを変更する場合に設定します。
パスワードチェック領域開始アドレスを変更するには、初期設定ファイル“y_init.h”の変更が必要です。

①．初期設定ファイル“y_init.h”設定変更
“PASS_START”にパスワードチェック領域の開始アドレスを設定してください。

パスワードチェック領域開始アドレスのデータもパスワードチェックの対象とすることができます。
パスワードチェック領域中の 7byte 以上のデータをチェックします。
パスワードをチェックするデータのサイズが 7byte 未満の場合、エラーになります。
パスワードチェック領域中のすべてのデータをチェックする必要はありません。

５．８　パスワードチェック領域終了アドレス

ReProg Area 内で暗号機能に使用する領域の終了アドレスを変更する場合に設定します。
パスワードチェック領域終了アドレスを変更するには、初期設定ファイル“y_init.h”の変更が必要です。

①. 初期設定ファイル“y_init.h”設定変更
“PASS_END”にパスワードチェック領域の終了アドレスを設定してください。
パスワードチェック領域終了アドレスのデータもパスワードチェックの対象とすることができます。
パスワードチェック領域中の 7byte 以上のデータをチェックします。
パスワードをチェックするデータのサイズが 7byte 未満の場合、エラーになります。
パスワードチェック領域中のすべてのデータをチェックする必要はありません。
パスワードチェック領域終了アドレスは最低でもパスワードチェック領域開始アドレスから 7byte 分サイズを確保して設定してください。

５．９　ユーザアプリ領域サム値チェック開始アドレス

書き込みプロセス正常終了判定（６－４．参照）に使用する“ユーザアプリ領域サム値チェック”領域の開始アドレスを変更する場合に設定します。
ユーザアプリ領域サム値チェック開始アドレスを変更するには、初期設定ファイル“y_init.h”とライタ側両方の変更が必要です。

①. 初期設定ファイル“y_init.h”設定変更
“APL_SUM_START”に設定値を設定してください。
32 ビットで設定します。
ユーザアプリ領域サム値チェック開始アドレスのデータもサム値演算の対象になります。

②. ライタ側設定変更
air Connect でのみ変更可能です。
Specific Parameter のアドレス#140,141,142,143 を変更することでユーザアプリ領域サム値チェック開始アドレスを変更できます。
例）ユーザアプリ領域サム値チェック開始を「0xFFAABBCC」と設定する場合

Specific Paramete のアドレス	#140	#141	#142	#143
設定値	0xFF	0xAA	0xBB	0xCC

※　「5.19　Specific Parameter 変更方法」を参照下さい。

５．１０　ユーザアプリ領域サム値チェック終了アドレス

書き込みプロセス正常終了判定（6－4. 参照）に使用する“ユーザアプリ領域サム値チェック”領域の終了アドレスを変更する場合に設定します。
ユーザアプリ領域サム値チェック終了アドレスを変更するには、初期設定ファイル“y_init.h”とライタ側両方の変更が必要です。

注意：この領域はコードフラッシュ範囲内で設定する必要があります。

①. 初期設定ファイル“y_init.h”設定変更

“APL_SUM_END”に設定値を設定してください。

32 ビットで設定してください。

ユーザアプリ領域サム値チェック終了アドレスのデータもサム値演算の対象になります。

②. ライタ側設定変更

air Connect でのみ変更可能です。

Specific Parameteのアドレス#144,145,146,147 を変更することでユーザアプリ領域サム値チェック終了アドレスを変更できます。

例）ユーザアプリ領域サム値チェック開始を「0xFFEEDDCC」と設定する場合

Specific Parameteのアドレス	#144	#145	#146	#147
設定値	0xFF	0xEE	0xDD	0xCC

※「5.19 Specific Parameteの変更方法」を参照下さい。

５．１１　I/O ポートサービス対応フラグ

UCOP では、I/O ポートを制御することにより外部ウォッチドッグタイマの制御を行う仕組みを持っています（「6.13　ウォッチドッグタイマ」参照）。
I/O ポートサービス対応フラグを変更することで I/O ポートサービスの有無を設定します。
I/O ポートサービス対応フラグを変更するには、初期設定ファイル“y_init.h”の変更が必要です。

①. 初期設定ファイル“y_init.h”設定変更

“IOS_ON”に「0」か「1」を設定します。

「0」の場合：I/O ポートサービスなし

「1」の場合：I/O ポートサービスあり

５．１２　I/O ポートサービス周期

I/O ポートサービスを行う周期を設定します。
I/O ポートサービス周期を変更するには、初期設定ファイル“y_init.h”の変更が必要です。

①. 初期設定ファイル“y_init.h”設定変更

“IOS_PERIOD”に I/O ポートサービスの周期を設定してください。

周期の単位は msec です。

５．１３　I/O ポートサービス用ポート変更方法

I/O ポートサービス用のポートを変更するためには、” y_init.h”変更する必要があります。

・#define　　IOS_PORT　　0x40021004

I/O サービス用ポートのポートレジスタアドレスです。

I/O サービスを実行するポートのアドレスを設定してください。

・#define IOS_BIT　　0x0001

I/O サービス用のポートを示す Bit です。

I/O サービスを実行するポートのビットに 1 をセットしてください。

上記の例では、IOS_PORT で設定したレジスタのビット 0 に割り当てられているポートを使用します。

５．１４　Primary ID

本マイコンパックではノード ID により Primary ID を選択する機能が追加されています。そのため Primary ID の変更は IBL ソースファイル”usr_api.c”を編集して行います。

ノード ID から Primary ID への変換テーブル変数 can_ID_table[]には、ライタからマイコンへ送る「受信 ID」と、マイコンからライタへ送る「送信 ID」がリスト化されていますので、本テーブルを直接変更してください。

また、通信を行うためにはライタ側で設定するアイデンティファイヤの変更も必要です。

「6.5　アイデンティファイヤ（CAN メッセージ ID）」をご参照ください。

①. IBL ソースファイル“usr_api.c”設定変更

変換テーブル変数“can_ID_table[]”に設定値を設定してください。

32 ビットで設定してください。

32 ビットの割り当てはスタンダード・フォーマット、エクステンデッド・フォーマットで変わります。

エクステンデッド・フォーマットの場合

Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24
IDE:1			STD_ID10	STD_ID9	STD_ID8	STD_ID7	STD_ID6

Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16
STD_ID5	STD_ID4	STD_ID3	STD_ID2	STD_ID1	STD_ID0	EXD_ID17	EXD_ID16

Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
EXD_ID15	EXD_ID14	EXD_ID13	EXD_ID12	EXD_ID11	EXD_ID10	EXD_ID9	EXD_ID8

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
EXD_ID7	EXD_ID6	EXD_ID5	EXD_ID4	EXD_ID3	EXD_ID2	EXD_ID1	EXD_ID0

スタンダード・フォーマットの場合

Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24
IDE:O							

Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16

Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
					STD_ID10	STD_ID9	STD_ID8

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
STD_ID7	STD_ID6	STD_ID5	STD_ID4	STD_ID3	STD_ID2	STD_ID1	STD_ID0

IDE：アイデンティファイア・エクステンションの略です。

Primary ID のフォーマットがスタンダードかエクステンデッドかを識別するためのものです。

0：スタンダード・フォーマット

1：エクステンデッド・フォーマット

（マクロ定数“EXT_ID”が定義されています、ご利用ください）

EXD_ID17～EXD_ID0

エクステンデッド・アイデンティファイヤを設定します。

STD_ID10 ～STD_ID0

スタンダード・アイデンティファイヤを設定します。

Bit 30、Bit 29：予約ビット

0 を設定してください。

例１）スタンダード・フォーマットの場合

ノード ID１の“受信 ID”のスタンダード・アイデンティファイアを「7E9」、“送信 ID” のスタンダード・アイデンティファイアを「7EA」と設定する場合

```
/*  0 */                0x0007E9,                0x0007EA,      /* ID1 */
```

例２）エクステンデッド・フォーマットの場合

ノード ID２の“受信 ID” のエクステンデッド・アイデンティファイアを「3EDCB」、スタンダード・アイデンティファイアを「7E9」、“送信 ID” のエクステンデッド・アイデンティファイアを「3EDCB」、スタンダード・アイデンティファイアを「7EA」と設定する場合

```
/*  1 */      EXT_ID | 0x1FA7ECDB,   EXT_ID | 0x1FABEDCB,      /* ID2 */
```

②．ライタ側設定変更

ｉ．air Connect での変更

マイコンからライタへ送るアイデンティファイヤ及びフレームのフォーマットは Specific

Parameter のアドレス#0C4,0C5,0C6,0C7 で、ライタからマイコンへ送るアイデンティファイヤ及びフレームのフォーマットは Specific Parameter のアドレス #0C8,0C9,0CA,0CB で変更します。

※「5.19　Specific Parameter 変更方法」を参照下さい。

#0C0～#0C7(#0C8～#0CB)の 32 ビットは下記のように割り当てられています。

#0C4(#0C8)							
IDE			EXD_ID17	EXD_ID16	EXD_ID15	EXD_ID14	EXD_ID13

#0C5(#0C9)							
EXD_ID12	EXD_ID11	EXD_ID10	EXD_ID9	EXD_ID8	EXD_ID7	EXD_ID6	EXD_ID5

#0C6(#0CA)							
EXD_ID4	EXD_ID3	EXD_ID2	EXD_ID1	EXD_ID0	STD_ID10	STD_ID9	STD_ID8

#0C7(#0CB)							
STD_ID7	STD_ID6	STD_ID5	STD_ID4	STD_ID3	STD_ID2	STD_ID1	STD_ID0

IDE：アイデンティファイヤ・エクステンションの略です。

Primary ID のフォーマットがスタンダードかエクステンデッドかを識別するためのものです。

0：スタンダード・フォーマット

1：エクステンデッド・フォーマット

EXD_ID17～EXD_ID0

エクステンデッド・アイデンティファイヤを設定します。

STD_ID10 ～STD_ID0

スタンダード・アイデンティファイヤを設定します。

例１）スタンダード・フォーマットの場合

“ID_P_NI”のスタンダード・アイデンティファイヤを「7E9」

“ID_P_MCU” のスタンダード・アイデンティファイヤを「7EA」と設定する場合

Specific Paramete のアドレス	#0C4	#0C5	#0C6	#0C7
設定値	0x00	0x00	0x07	0xEA

Specific Paramete のアドレス	#0C8	#0C9	#0CA	#0CB
設定値	0x00	0x00	0x07	0xE9

例２）エクステンデッド・フォーマットの場合

“ID_P_NI” のエクステンデッド・アイデンティファイヤを「3EDCB」、スタンダード・アイデンティファイヤを「7E9」、“ID_P_MCU” のエクステンデッド・アイデンティファイヤを「3EDCB」、スタンダード・アイデンティファイヤを「7EA」と設定する場合

Specific Paramete のアドレス	#0C4	#0C5	#0C6	#0C7
設定値	0x9F	0x6E	0x5F	0xEA

Specific Paramete のアドレス	#0C8	#0C9	#0CA	#0CB
設定値	0x9F	0x6E	0x5F	0xE9

ⅱ．メニューからの変更

“SUB SETTING”メニューの”CAN ID SET”からアイデンティファイヤを設定します。

上下キーで設定するアイデンティファイヤを選択します。

左右キーでアイデンティファイヤを変更します。

５．１５　CAN ID フォーマット設定

Primary ID のフォーマットとして、スタンダード・フォーマットとエクステンデッド・フォーマットのどちらを使用するかの指定は、IBL ソースファイル”usr_api.c”と、ライタで行います。

①．IBL ソースファイル”usr_api.c”設定変更

フォーマット指定も ID 設定と共に変換テーブル変数 can_ID_table[] に記述します。

詳しくは、「5.14 Primary ID」をご参照ください。

②．ライタ側設定変更

ⅰ．air Connect での変更

Specific Parameter でスタンダード・フォーマットかエクステンデッド・フォーマットかを設定します。

詳しくは「5.14　Primary ID」を参照してください。

ⅱ．ライタでの変更

“SUB SETTING”メニューの”CAN AF -> TGT ID FMT”、”CAN TGT -> AF ID FMT”で設定を変更します。

”CAN AF -> TGT ID FMT”はライタからマイコンへ送信するフレームの CAN ID フォーマット設定します。

”CAN TGT -> AF ID FMT”はマイコンからライタへ送信するフレームの CAN ID フォーマット設定します。

上下キーでスタンダードかエクステンデッドかを選択します。

５．１６　ステーションアドレス

ステーションアドレスを変更するには、初期設定ファイル“y_init.h”とライタ側両方の変更が必要です。

「6.11　ステーションアドレス」をご参照ください。

①．初期設定ファイル“y_init.h”設定変更

“CCP_STATION”にステーションアドレスを 2byte（リトルエンディアン）で設定してください。

例）ステーションアドレスを「0x0200」の場合

“CCP_STATION”に「0x0002」と設定する。

②．ライタ側設定変更

air Connect でのみ変更可能となります。

Specific Parameter のアドレス#0D8,0D9 に 2byte（リトルエンディアン）で設定してください。

例）ステーションアドレスを「0x0200」の場合

Specific Parameter のアドレス	#0D8	#0D9
設定値	0x00	0x02

※　「5.19　Specific Parameter 変更方法」を参照下さい。

５．１７　内蔵ウォッチドッグタイマ設定

内蔵ウォッチドッグタイマを使用するかどうかを設定することができます。

設定を変更するためには、初期設定ファイル”y_init.h”を変更する必要があります。

①．初期設定ファイル”y_init.h”設定変更

“IOS_WDT_ON”に内蔵ウォッチドッグタイマ使用の有無を設定します。

0：内蔵ウォッチドッグタイマを使用しません。

1：内蔵ウォッチドッグタイマを使用します。

５．１８　内蔵ウォッチドッグタイマ周期設定

内蔵ウォッチドッグタイマのクリア周期を設定することができます。

設定を変更するためには、初期設定ファイル”y_init.h”を変更する必要があります。

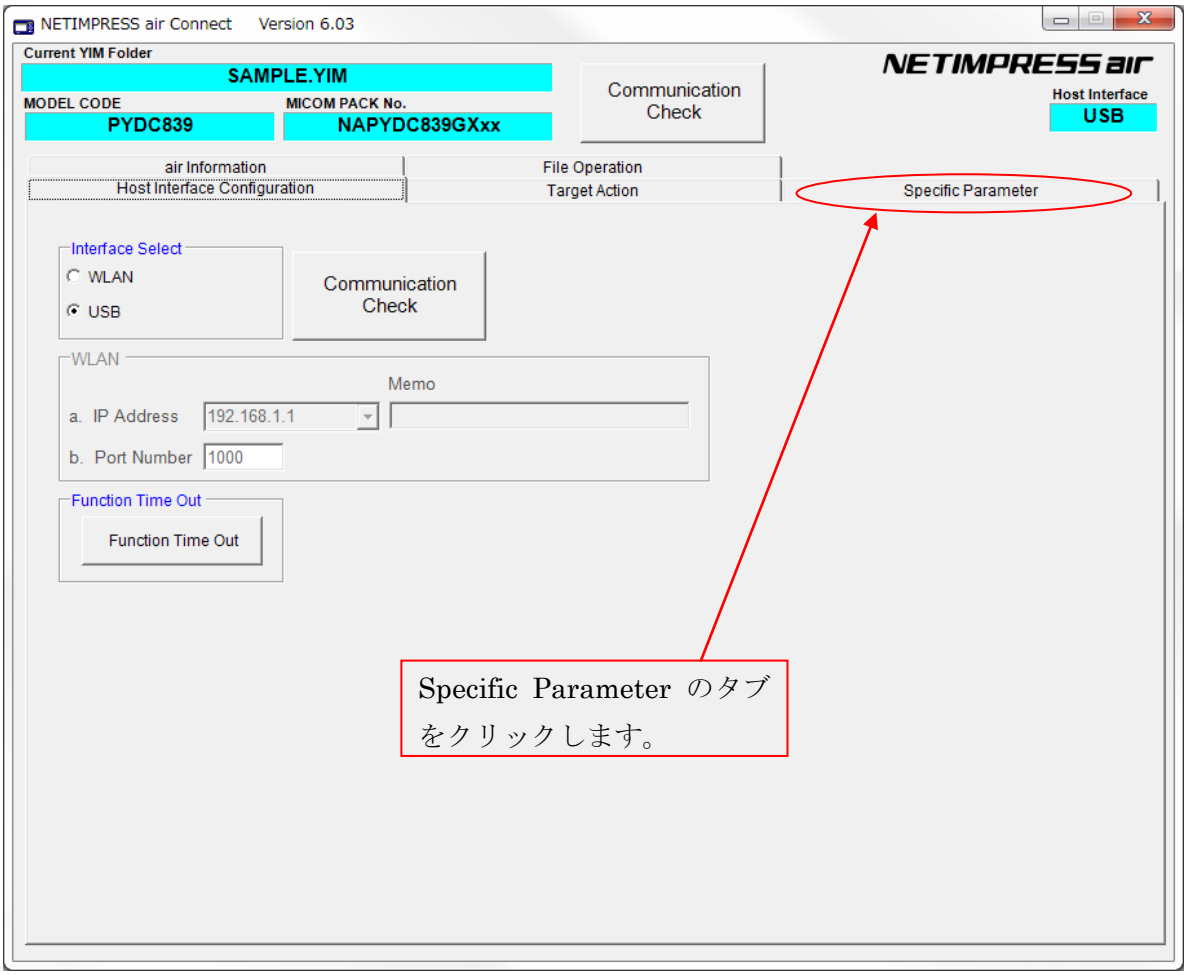
①．初期設定ファイル”y_init.h”設定変更

“IOS_WDT_PERIOD”に内蔵ウォッチドッグタイマのクリア周期を設定します。

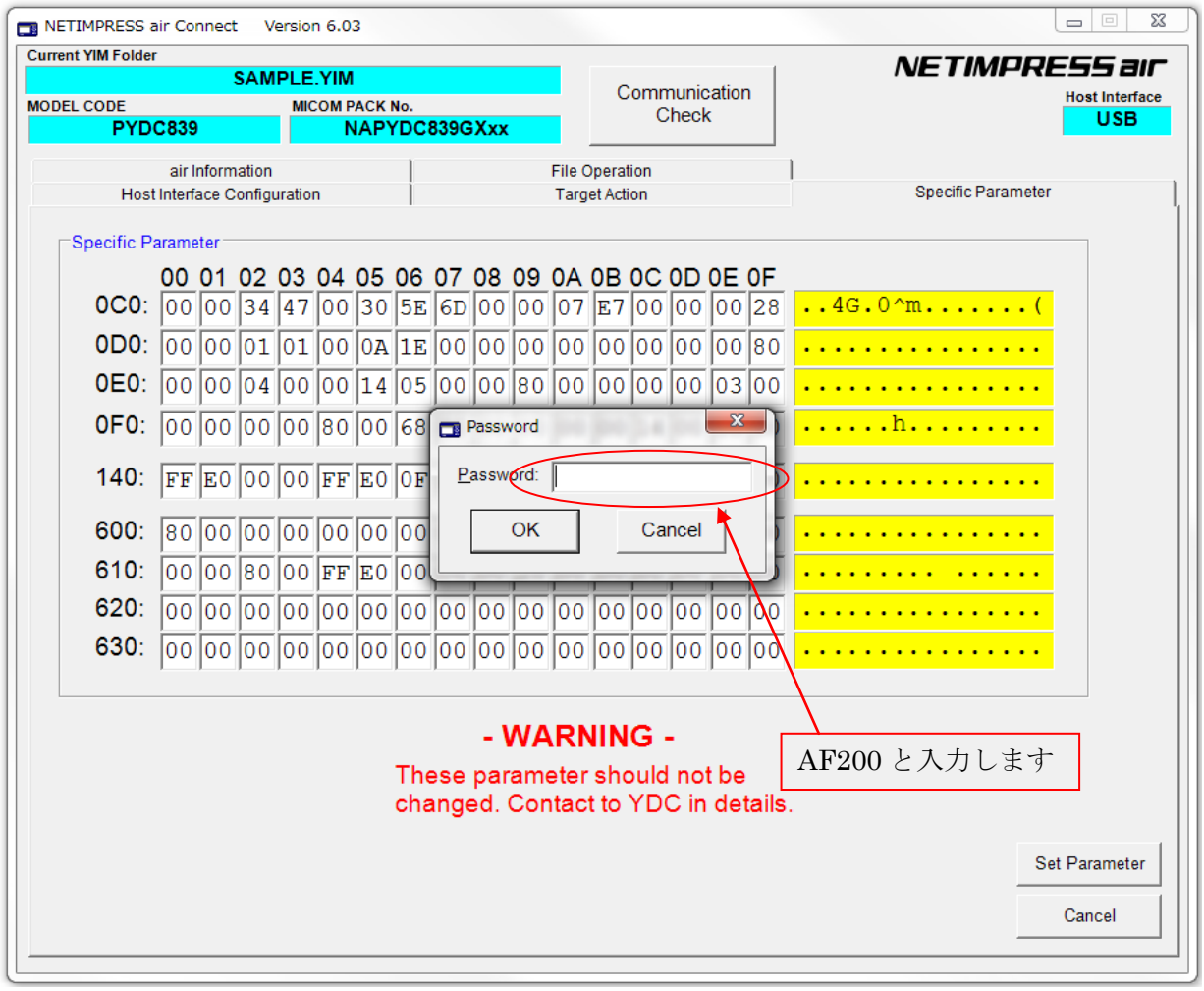
設定の単位は ms です。

５．１９　Specific Parameter 変更方法

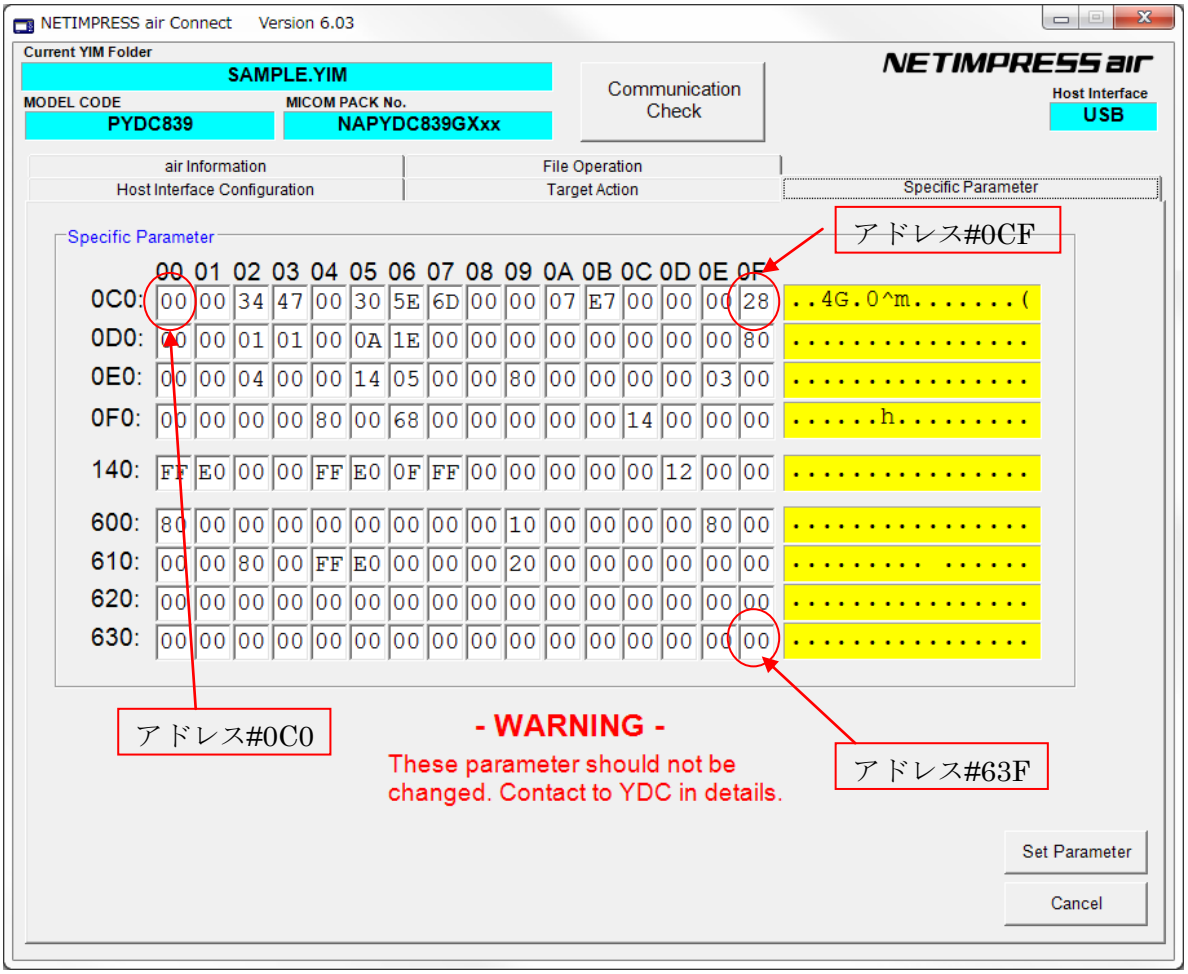
ここでは、air Connect の Specific Parameter の変更方法を説明します。
まずは、air Connect を起動し、NET IMPRESS と接続してください。
Specific Parameter のタブをクリックし、Specific Parameter を開きます。



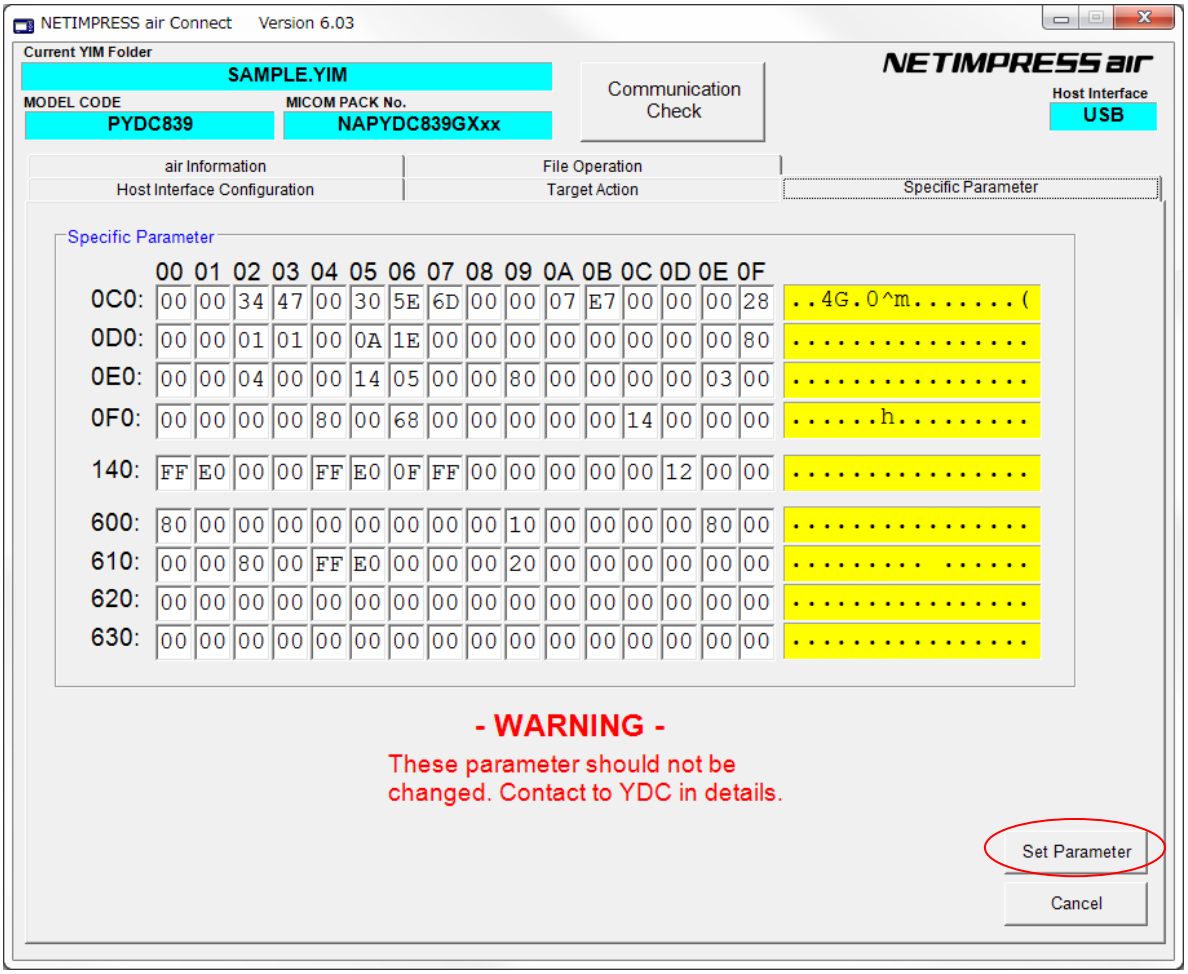
パスワードを求めるウィンドウが開きますので、”AF200”と入力し、OK ボタンを押してください。



Specific Parameter の画面が開きますので、任意のアドレスのデータを書き換えます。



パラメータを書き換えましたら、その内容を保存するために”Set Parameter”ボタンをクリックします。



以上で、Specific Parameter の値を設定することができます。

６．UCOPシステム概要

６．１　イニシャル・プロセッシング・ルーチン（IPR）
（C 言語プログラム、リロケータブルオブジェクト）

ソースファイルは「user_ipr.c」として供給されます。
お客様サイドでカスタマイズして頂きます。
ターゲット MCU 実装前にシリアルライタ等で予め ROM の所定領域に書き込んでおきます。
リセット解除後リセットベクタよりジャンプします。
UCOP リプログモード実行に際して最低限必要なシステムの初期化を行うルーチンです。
アプリケーションプログラム実行時に初めて必要となる初期化ルーチンは、これとは別にアプリケーションプログラム中に専用初期化ルーチンを設けその中に配置してください。

項目	内容	ユーザ設定
IPR 配置アドレス	#00000200～#00008FF(1792byte) #00000200　※１	不可
IPR 要初期化項目 ■：必須 □：選択	■ 割り込み禁止 ■ スタック初期化 ■ ポート初期化 □ WDT 初期化	可

※１：リセット解除後、リセットベクタからこのアドレスへ飛びます。

６．２　イニシャルブートローダ（IBL）

（C 言語プログラム、リロケータブルオブジェクト）

ソースファイルは「y_ibs_init.s」、「y_ibl.c」、「RamFunc.c」として供給されます。
基本的にそのままでお使いいただけます。
ターゲット MCU 実装前にシリアルライタ等で予め ROM の所定領域に書き込んでおきます。
IPR よりコールされます。
UCOP リプログモード用の CAN 初期化設定、UCOP リプログモードエントリー、WCP の受信及び内蔵 RAM への書き込みを行います。

IBL プログラムを予めマイコンの下記アドレスに配置します。

項目	内容	ユーザ設定
YDC 製 IBL	#00000900～#00012AF(約 2.5kbyte)	不可
IBL エントリーアドレス From IPR	#00000901(Thumb 命令のため奇数番地指定)	不可
IBL エントリーアドレス From APL	#00000911(Thumb 命令のため奇数番地指定)	不可

※ IBL ファイルとして供給されます。シリアルライタ等で所定の領域に書き込み後、MCU を実装して下さい。
※IBL 突入時スタックはイニシャライズされます。
※最適化レベル「高」を使用しております。（一部変換を除く）

６．３　書き込み制御プログラム（WCP）

（C 言語プログラム、リロケータブルオブジェクト）

ソースファイルは「y_wcp.c」として供給されます。
実行ファイルは拡張子が BTP のファイルとして供給されます。
コントロールモジュールの YIM フォルダに配置してください。
ライタはあらかじめ ROM 内に組み込まれている IBL と通信を行い、BTP ファイルを順次送信します。
IBL はターゲットの内蔵 RAM へ受信した BTP ファイルを書き込みます。

IBL プログラムとの通信によりマイコンの下記アドレスに配置します。

項目	内容	ユーザ設定
WCP 転送アドレス	#10000000～#100010FF(4.25kbyte)	不可

６．４　書き込みプロセス正常終了判定

①．アプリケーションプログラム（APL）の一部領域を“ユーザアプリ領域サム値チェック”領域として使用し、既に正常な APL が存在しているか否かの判定をします。

②．判定方法は、**“ユーザアプリ領域サム値チェック”領域の SUM 値が＃AA（８ビット単純加算８ビット比較）の場合、正常に APL が書き込まれていると判断します。**

③．“ユーザアプリ領域サム値チェック”領域の SUM 値計算は IBL 中で実行され、正常な APL が書込まれていれば APL へ JUMP し、そうでなければ IBL で Connect コマンド待ちになります。

④．書込み時にエラーが発生し、”ユーザアプリ領域 SUM 値チェック”領域のみ正常に書き込みが行われ、その他領域に正常データが書かれていない状態になるのを防ぐため、E.P.R 実行に際し下記のように動作します。

Ⅰ．１番最初に“ユーザアプリ領域サム値チェック”領域を含むブロックを消去します。

Ⅱ．その他ブロックの消去をおこないます。

Ⅲ．“ユーザアプリ領域サム値チェック”領域以外の領域にデータを書き込みます。

Ⅳ．１番最後に“ユーザアプリ領域サム値チェック”領域にデータを書き込みます。

注意：ｉ．１つの消去ブロック内に収まるように設定してください。

ⅱ．512byte の整数倍のサイズとしてください。

ⅲ．書き込み禁止領域中には設定しないで下さい。

項目	内容	ユーザ設定
“ユーザアプリ領域サム値チェック”領域	#0007FE00～#0007FFFF（デフォルト※）	可

※初期設定ファイル（y_init.h）の APL_SUM_START と APL_SUM_END で設定してある値です。

※APL 領域範囲で変更可能です。

６．５　アイデンティファイヤ(CAN メッセージ ID)

アイデンティファイヤ(以下:CAN メッセージ ID)としてデフォルトとして設定されるものを「Ｐｒｉｍａｒｙ　ＩＤ」と規定します。本マイコンパックではノード ID により Primary ID を選択する機能が追加されています。
メッセージ ID はここで規定した Primary ID 以外の他の ID へ変更することができます。
IBL ソースファイル usr_api.c 内のテーブル can_ID_table[]を変更したあと再コンパイルし、IBL を書き換えてください。(5.14　Primary ID 参照)

６．５．１　Primary ID

ノード ID	Primary ID		ユーザ設定
	ライタ→マイコン	マイコン→ライタ	
1	EXT_ID	0xFA2153	EXT_ID	0xFD5321	可
2	EXT_ID	0xFA2253	EXT_ID	0xFD5322	可
…	…	…	…
11	EXT_ID	0xFA2B53	EXT_ID	0xFD532B	可
12	EXT_ID	0xFA2C53	EXT_ID	0xFD532C	可

※Primary ID はファイル usr_api.c に記述します。

６．５．２　Secondary ID

このマイコンパックは Secondary ID 追加機能を持っておりません。ご注意下さい。

６．５．３　送受信メッセージバッファ

ＩＢＬで使用するメッセージバッファについて規定します。

基本規定

・ＩＢＬは受信用メッセージバッファとして受信バッファ 1 を使用します。
・ＩＢＬは送信用メッセージバッファとして送信バッファ 0 を使用します。
・ＩＢＬは受信用メッセージバッファのマスクは使用しません。
・ID 完全一致として使用します。
・受信バッファ 2～31 は使用しません。
・割り込みは禁止です。

ユーザ APL の使用法

・IBL から APL へ移行した際は上記「IBL 規定」の状態です。
必要であればＩＢＬで使用しているメッセージバッファの設定を変更し使用しても構いません。
ただし APL から IBL へ移行する場合(u Entry)、メッセージバッファを IBL 規定に戻し、割り込みは禁止にしてから移行してください。

項目	内容	ユーザ設定
IBL 受信用 メッセージバッファ	受信バッファ 1	不可
IBL 送信用 メッセージバッファ	送信バッファ 0	不可

６．６　ステータスレジスタ

フラッシュメモリの動作状態やイレーズ、プログラムの正常/エラー終了時の状態を示します。
ステータスレジスタ(SRD)の内容により状態を判断します。
ステータスレジスタの内容をチェックする為、SRD エリアとして WCP 中で占有します。

＜ｽﾃｰﾀｽﾚｼﾞｽﾀ(SRD)＞

SRD の各ﾋﾞｯﾄ	ｽﾃｰﾀｽ名	定義	
		“1”	“0”
SR7(bit7)	ｺﾏﾝﾄﾞﾋﾞｼﾞｨ	ﾚﾃﾞｨ	ﾋﾞｼﾞｨ
SR6(bit6)	ｲﾚｰｽﾞｽﾃｰﾀｽ	ｴﾗｰ終了	正常終了
SR5(bit5)	ﾌﾟﾛｸﾞﾗﾑｽﾃｰﾀｽ	ｴﾗｰ終了	正常終了
SR4(bit4)	ﾘｻﾞｰﾌﾞ	---	---
SR3(bit3)	ﾘｻﾞｰﾌﾞ	---	---
SR2(bit2)	ﾘｻﾞｰﾌﾞ	---	---
SR1(bit1)	ﾃﾞｰﾀ受信ﾀｲﾑｱｳﾄ	ﾀｲﾑｱｳﾄ	正常動作
SR0(bit0)	ﾘｻﾞｰﾌﾞ	---	---

(a)SR7 (コマンドビジィ)
・書き込み動作や消去動作中は”0”に、これらの動作終了とともに”1”に
セットされます。

(b)SR6 (イレーズステータス)
・消去の動作状況を示し、消去エラーが発生すると”1”にセットされます。
このビットに一旦”1”がセットされると、クリアステータスレジスタ
コマンドを行わない限りリセット(“0”に書き換わる)されません。

(c)SR5 (プログラムステータス)
・書き込みの動作状況を示し、書き込みエラーが発生すると”1”にセット
されます。
・このビットに一旦”1”がセットされると、クリアステータスレジスタ
コマンドを行わない限りリセット(“0”に書き換わる)されません。

(d)SR1　(データ受信タイムアウト)
・データの受信中にタイムアウトが発生すると”1”にセットされます。
・このビットに一旦”1”がセットされると、クリアステータスレジスタ
コマンドを行わない限りリセット(“0”に書き換わる)されません。

＜ｽﾃｰﾀｽﾚｼﾞｽﾀ 1(SRD1)＞
・8bit で構成され、受信したコマンドフレームのコマンドがセットされます。

６．７　プログラムエントリモード

UCOP には次の 3 種のエントリモードが存在します。

エントリモード	概　　要	使用方法
n Entry	“書き込みプロセス正常終了判定領域”の SUM 値が＃AA 以外の場合のエントリモード	標準
u Entry	“書き込みプロセス正常終了判定領域”の SUM 値が＃AA の場合のエントリモード ※APL上のCAN対応コマンドでエントリーします	標準
r Entry	“書き込みプロセス正常終了判定領域”の SUM 値が＃AA かつ u Entry が不可能な場合のエントリモード	非標準

※各エントリのフローは「3.3　プログラムエントリモードフローチャート」を参照してください。

６．８　u Entry 時ユーザ APL 処理項目

項目	内容	ユーザ設定
APL へジャンプ後の 処理項目 ■：必須 □：選択	■ CONNECT コマンド受信 (受信後 20ms 以内(※１)で IBL を CALL する) ※２ □ CAN メッセージボックスの設定変更 ■ CAN ボーレートの変更不可	可
Connect コマンド受信後の APL 要初期化項目 (u Entry 時) ■：必須	■ CAN 受信バッファクリア（受信完了状態にする） ■ 割り込み禁止 ■ UCOP 用メッセージボックスの設定 ※３	可

※１．CONNECT 応答規定は 25ms ですが IBL での応答までの処理が約 2ms 程要するため 20ms 以内程度で IBL を CALL して下さい

※２．APL がビジー状態で CONNECT できない場合、ビジー応答を返してください。

・ビジー応答規定

Data1	Data2	Data3	Data4	Data5	Data6	Data7	Data8
FEh	36h	CTR					

CTR=00h 固定、斜線部は don't care です

ビジー応答があった場合 IMPRESS 側は CONNECT リトライを 200ms 間隔で 5 回まで行います。

リトライオーバーした場合“resouce/function not available”エラーとなります。

尚、リトライ回数及びその間隔は設定が可能です。

項目	内容	ユーザ設定
リトライ回数	5 回	可
リトライ間隔	200ms	可

※３．ユーザアプリにて CAN のメッセージボックスを変更した場合、IBL や WCP にて CAN の送受信が正しくできるよう、UCOP で使用する設定に戻してください。

６．９ KILL レジスタ

本マイコンパックではKILLレジスタの書き換えはできません。

６．１０　誤 Entry 時無限ループ防止機能

何らかの理由で誤って Entry を行った際の無限ループを防止する目的で IBL では規定ポイントにてタイムアウト処理を行います。タイムアウトした場合リセット処理を行います。

・タイムアウト処理を行うポイント

Connect 応答後、次コマンド待ち

項目	内容	ユーザ設定
タイムアウト値 n sec	3 sec	不可

６．１１　CAN ボーレート設定時の注意

・CAN 通信におけるボーレート設定は、ＭＣＵに対するレジスタ設定によって行います。
ボーレートプリスケーラレジスタ、ビットコンフィグレーションレジスタ、CAN クロック選択ビットの値を設定してください。
詳細は「5.5 ＣＡＮビットレートプリスケーラレジスタ値」、「5.6 ＣＡＮビットコンフィグレーションレジスタ値」を参照してください。

※コンパイル時に、指定されたレジスタ情報とボーレート値、CAN クロックが矛盾する場合にはコンパイルエラーとなります。

６．１２　ステーションアドレス

・16bit 構成で CCP プロトコルで使用します。
・スレーブ側（ECU）は初期設定ファイルにて設定します。
・マスター側（ライタ）は IMPRESS モジュールのパラメータファイルにて設定します。
・スレーブ側（ECU）はステーションアドレス不一致時、エラーを返さず引き続きコネクト待ち状態となります。
（ステーションアドレスの変更方法は「5.16　ステーションアドレス」を参照してください。）

項目	内容	ユーザ設定
ステーションアドレス	#0000	可

※Disconnect コマンドについては、MCU はその受信に際してステーションアドレスを無視します。

６．１３　プログラム終了時の処理

WCP、IBL のプログラム終了時の処理について正常終了時、異常終了時、共にメッセージ送信後に Disconnect コマンド待ちとなります。

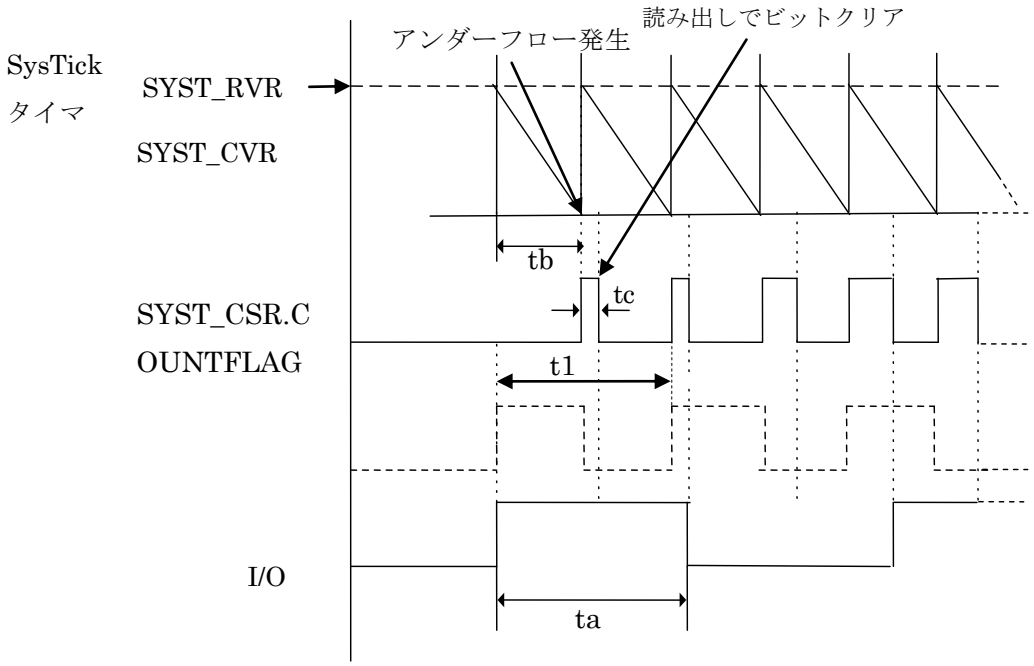
<Disconnect コマンド受信時の動作>

I/O ポートサービスの停止処理を行いその後永久ループに入ります。
（I/O ポートサービス停止に伴う、MCU 外部からのリセットにより ECU のリスタートを行うことができます）
外部からリセットが入らない場合、電源断まで永久ループに入っています。

６．１４　ウォッチドッグタイマ

・I/O ポートを制御することによりウォッチドッグタイマの制御を行います。
・ビット単位の制御可能な出力ポートに対するサイクルアクセス機能をもちます。
・アクセス周期は初期設定ファイルに設定します。
・内部タイマーを使用します。（割り込みは使用しません）
※I/O ポートサービスの有無、アクセス周期変更等は「5.11　I/O ポートサービス対応フラグ「5.12 I/O ポートサービス周期」・「5.13　I./O ポートサービス用ポート変更方法」を参照してください。

WDT 制御



SYST_CVR：タイマカウンタ
SYST_CSR.COUNTFLAG：アンダーフロー検出ビット
I/O：I/O出力パルス

仕様

項目	内容	ユーザ設定
使用内蔵タイマ	□無し ■有：　タイマ番号　SysTick タイマ	不可
t1（指定パルス幅）	min 500us で 500usec 単位とする。	可
ta （実パルス幅）	ta = t1 ± tc ms	t1 設定可
tb （タイマインターバル時間）	500uｓｅｃ	不可
tc（検出遅延時間）	Max：500usec	
I/O ポート番号	Port A　bit 0	可

※ｕＥｎｔｒｙ時直ちに１ｍｓ幅固定で１周期パルス出力し通常のパルス動作に移行します。

６．１５　IBL 処理時間

リセット後ユーザ IPR から IBL を CALL し APL SUM 域が正常だった場合の IPR へリターンするまでの処理時間です。

動作周波数 88MHz で測定

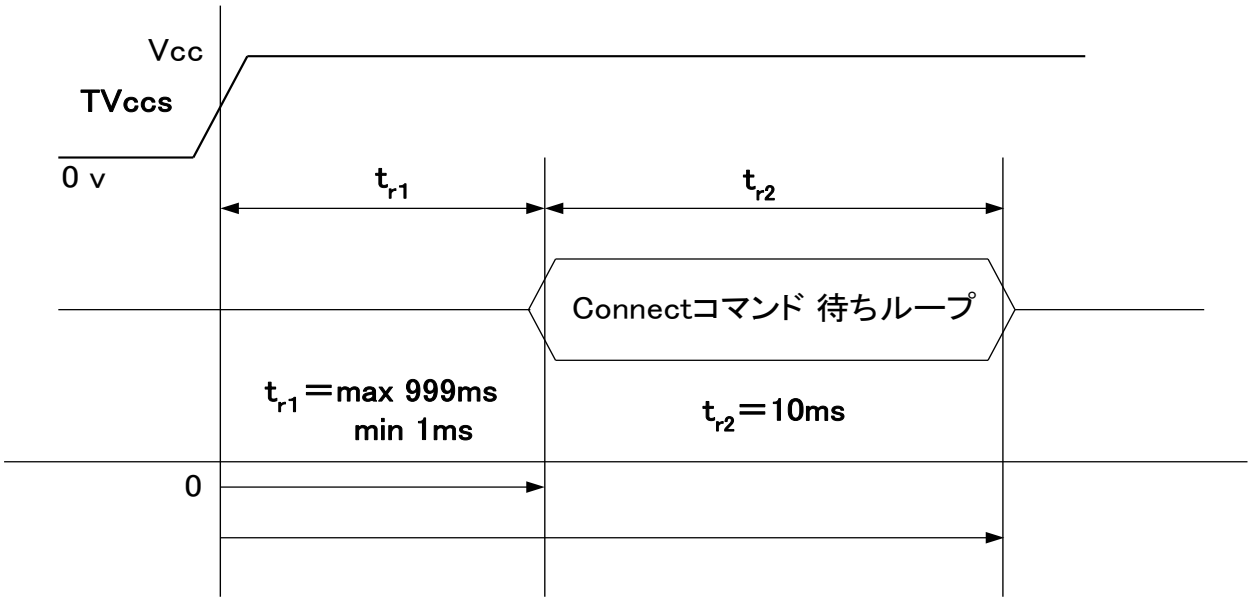
項目	内容	ユーザ設定
APL SUM 域 4Kbyte	10.8[ms]	不可
APL SUM 域 1byte	10.1[ms]	不可

７．　r Entryモード仕様

７．１　概要

r Entry は、ユーザアプリケーションが正常にかかれている状態(“書き込みプロセス正常終了判定領域”の SUM 値が＃AA)で、u Entry が不可能な場合に使用します。
電源投入後、一定期間 t_{r1} (※)経過後、約 10mSec 間 Connect コマンドを待ちます。この約 10mSec 間に Connect コマンドを受信すると r Entry になります。
ライタ側はターゲットの電源を開始することにより、タイミングを合わせ r Entry モード期間に Connect コマンドを送信するようにします。

※この一定期間はターゲットに電源投入後 Connect コマンド受信待ちを開始するまでの時間で IPR の処理時間などお客様のシステム構成によって時間が変わってきます。



ライタはゼロ基点から t_{r1} 後に CONNECT コマンドを発することで、
“SUM 一致”の場合も、リプログモードに入り込めます。

※TVcc s は電源監視用の信号線です。

項目	内容	ユーザ設定
t_{r1}	1～999ms	可
t_{r2}	10 ms	不可

７．２　r Entry モード使用方法

r Entry モードの使用には、r Entry モード用マイコンパックが必要となります。

r Entry モード使用の際は、弊社サポートセンタまで、ご連絡ください。

使用手順は以下のようになります。

１．電源監視用にターゲットマイコンの Vcc ラインに TVcc s 信号線を必ず接続してください。

２．air Connect の Specific Parameter のアドレス#14F の値が「0x01」であることを確認してください。

Specific Parameterのアドレス	#14F
設定値	0x01

3．ターゲット側の t_{r1} 時間分タイミングを合わせるために、ライタ側の Connect コマンド発行タイミングを air Connect の Specific Parameter のアドレス#14C,#14D の 2byte で設定してください。

例）t_{r1} 時間が 5mS の場合

Specific Parameterのアドレス	#14C	#14D
設定値	0x00	0x05

８．YDC製IBL、WCPの構成

① 概要

IBL 本体プログラ
(WCP 本体プログラ

＋

初期設定ファイル *1

| |

CAN　ボーレート

CAN　プリスケーラレジスタ　ビットレートレジスタ

入力クロック

逓倍値

分周値

パスワードチェック領域（スタート/エンド）　*２

ユーザアプリ領域のサム値チェックアドレス　*３
（スタート/エンド）

Station アドレス（CCP　Ver2.1 準拠）

I/O ポートサービス設定フラグ

内蔵 WDT 使用フラグ

※Primary　メッセージ ID　および
CANID(標準 ID or 拡張 ID)　は usr_api.c に記述

＊１　初期設定ファイルは、ヘッダファイルとして IBL、WCP にてインクルードされます。

＊２　パスワードチェック領域はユーザアプリ領域中の任意の領域を設定します。

項目	内容	ユーザ設定
パスワードチェック領域	#00008400～#0000840F	可

＊３　ユーザアプリ領域のサム値チェックスタートアドレスは、書き込みページの先頭（下位 8bit “00h”）とします。サイズは 512byte～４Kbyte とします。

９．RAMの使用方法

書き込み実行後 MCU 上の RAM 内容は保持されておりません。

１０．CANプロトコル

UCOP で使用する各コマンドについての詳細は弊社ホームページに掲載しています「UCOP プログラミングコマンド・プロトコル仕様書」をご参照ください。

１０．１　フレームの種類

UCOP ではデータフレームを次の 6 種類に分別して通信を行っています。

① コマンドフレーム
② ビジィフレーム
③ リードフレーム
④ ライトフレーム
⑤ レディフレーム
⑥ エラーフレーム

全てのフレームはスタンダード・フォーマットとエクステンデッド・フォーマットの２つのフレームフォーマットがあります。データフィールドは 8 バイトの固定長です。
各フレームの役割を下記に示します。

フレーム	フレームの役割
コマンドフレーム	実行するコマンドを指定するフレームです。ライタが MCU へ送信します。
ビジィフレーム	MCU がライタへ受け取ったコマンドを返すフレームです。
リードフレーム	ライタが MCU からデータを受信する為のフレームです。
ライトフレーム	ライタが MCU へデータを送信する為のフレームです。
ターミネータフレーム	ライトフレームが終了したことを知らせる為のフレームです。
レディフレーム	MCU がライタへ処理終了を知らせる為のフレームです。
エラーフレーム	１部のコマンド実行においてエラーが発生した場合や規定外のコマンドを受信した場合に MCU がライタへ送信するフレームです。

１０．２　IBL 対応コマンド

IBL では下記コマンドに対して応答する。

コマンド	内容
Connect	コネクト処理を行います。
Get CCP Version	CCP のバージョンをライタへ送信します。
Exchange ID	予め決められた ID をライタへ送信します。
Pass Word Check	パスワードチェックを行います。
Download	WCP プログラムを受信し内蔵 RAM へ書き込みます。
Extended Sum Read	内蔵 RAM へ書き込んだ WCP のサム値を計算しライタへ送信します。
Disconnect	ディスコネクト処理を行います。

上記以外のコマンドに対してはエラーフレームを発行します。Disconnect コマンドに関しては常時受信可能とし、ツールからの指示でいつでも、Disconnect・リセットスタートを可能にします。

１０．３　WCP 対応コマンド

WCP では下記コマンドに対して応答します。

コマンド	内容
Read	ROM からデータを読み出します。
Program	ROM にデータを書き込みます。
Block Erase	ブロック単位で ROM のデータを消去します。
Read SRD	ステータスレジスタの値をライタへ送信します。
Clear SRD	ステータスレジスタの値を初期化します。
Extended Blank Check	MCU 側でブランクチェックを行います。
Extended Sum Read	MCU 側で SUM 値を計算しライタへ送信します。
Disconnect	ディスコネクト処理を行います。

上記以外のコマンドに対してはエラーフレームを発行します。

Disconnect コマンドに関しては常時受信可能とし、ツールからの指示で

いつでも、Disconnect・リセットスタートを可能にします。

１１．関数一覧

この章では、IBL, WCP 内で使用されている関数について説明しています。
ただし、お客様が変更された結果についての責任は、弊社では負いかねますので予めご了承ください。

また、弊社プログラム内ではデータサイズとデータ形式は下記のように取り決めています。

弊社プログラム内データ形式	データ形式	データサイズ
DWORD	unsigned long	32bit データ
WORD	unsigned short	16bit データ
BYTE	unsigned char	8bit データ

１１．１　IBL での使用関数（y_ibl.c ファイルの関数一覧）

・ io_service(void) 関数

内容	I/O ポートサービス処理関数です。 引数を設定の上 io_sv 関数をコールします。
引数	なし
戻り値	なし

・ io_sv（volatile WORD* sp, volatile WORD* sp_wdt） 関数

内容	I/O ポートサービス処理関数です。 I/O ポートサービスを行います。 内蔵 WDT のクリアを行います。
引数	*sp IO サービス用のソフトウェアパルスカウンタです。 設定したカウンタ値になるまでインクリメントします。 設定したカウンタ値になるとゼロに初期化します。 *sp_wdt 内蔵 WDT 用のソフトウェアパルスカウンタです。 設定したカウンタ値になるまでインクリメントします。 設定したカウンタ値になるとゼロに初期化します。
戻り値	1：500usec 経過した 0：500usec 経過していない

・ reset_wdt(void) 関数

内容	WDT をリフレッシュします。
引数	なし
戻り値	なし

・ init_tpu(void) 関数

内容	I/O ポートサービス用のタイマとポートの初期化を行います。
引数	なし
戻り値	なし

・ RecData(void) 関数

内容	コマンドフレーム、ライトフレーム、ターミネータフレーム受信処理関数です。 ディスコネクトコマンドを受信した場合は応答を返し、リセット処理を行います。
引数	なし
戻り値	なし

・ TrmData(void) 関数

内容	リードフレーム送信処理関数です。 CAN データの送信を行います。
引数	なし
戻り値	なし

・ TrmReady(void) 関数

内容	レディフレーム送信処理関数です。 レディフレームを送信します。
引数	なし
戻り値	なし

・ TrmBusy(void) 関数

内容	ビジィフレーム送信処理関数です。 ビジィフレームを送信します。
引数	なし
戻り値	なし

・ TrmError(void) 関数

内容	エラーフレーム送信処理関数です。 エラーフレームを送信します。
引数	なし
戻り値	なし

・ exec_reset(void) 関数

内容	リセット実行処理関数です。 I/O ポートサービス用タイマのカウントをストップし、無限ループに入ります。
引数	なし
戻り値	なし

・ command_GetCcpVersion(void) 関数

内容	CCP バージョン取得処理関数です。 取得したバージョンが「2.1」かどうか判定します。
引数	なし
戻り値	取得したバージョンが「2.1」かどうか判定し 戻り値として、「2.1」の場合 OK(= 0)、「2.1」以外の場合 NG(= -1)を返します。

・ command_ExchangeID(void) 関数

内容	Exchange_ID 処理関数です。 予め決められた ID をライタ側に送信します。
引数	なし
戻り値	戻り値は、OK(= 0)を返します。

・ command_password_check(void) 関数

内容	パスワードチェック処理関数です。 ライターから送られてきた ID と、フラッシュメモリの内容が同じかを判定します。
引数	なし
戻り値	戻り値は、正常の場合 OK(= 0)、エラーの場合 NONE(= 1)を返します。

・ command_ExtSumcheck(void) 関数

内容	拡張サムチェック処理関数です。 8bit 単純加算したサム値と 16bit 単純加算したサム値をライター側に返します。
引数	なし
戻り値	なし

・ get_wcp(WORD* paddr) 関数

内容	W.C.P データ受信処理関数です。 W.C.P データを受信し内蔵 RAM へ書き込みます。 拡張 SUM チェックコマンドを受信したら、command_ExtSumcheck 関数を Call します。
引数	*paddr 内蔵 RAM ジャンプ先アドレスです。
戻り値	正常の場合 OK(= 0)、コマンド異常の場合 NG(= -1)を返します。

・ RecConnect(BYTE tout, WORD* piosdiv, WORD* pwdtdiv) 関数

内容	コネクトコマンド受信処理関数です。 r Entry，n Entry においてコネクトコマンドを受信します。
引数	Tout コネクトコマンド受信においてタイムアウトの有無を設定します。 =1：タイムアウト有り（10ms） =0 ：タイムアウト無し *piosdiv IO サービス用のソフトウェアパルスカウンタです。 *pwdtdiv 内蔵 WDT 用のソフトウェアパルスカウンタです。
戻り値	コネクトコマンドを受信した場合「1」、コネクトコマンドを受信しなかった場合「0」を返します。

・ ibl_main(void) 関数

内容	ibl メイン処理関数です。 IPR からコールされるメインルーチンです。 各種初期設定と、コネクトコマンド受信処理を行います。
引数	なし
戻り値	戻り値「1」の場合、r Entry または n Entry で UCOP リプログモードへ遷移します。 戻り値「0」の場合、APL へ制御が移ります。

・ ibl_entry(WORD ent) 関数

内容	エントリー処理関数です。 r Entry、n Entry、u Entry 後のメインルーチンです。
引数	ｅｎｔ UCOP リプログモードへエントリーしたエントリモードを判定します。 =1：rEntry,nEntry =0：uEntry
戻り値	なし

・ init_can(DWORD id_recv, DWORD id_send) 関数

内容	CAN 設定関数です。 CAN 用レジスタの初期化を行います。 バッファ 0 を送信用、バッファ 1 を受信用に設定します。
引数	id_recv ライタからマイコンへ送る ID id_send マイコンからライタへ送る ID
戻り値	なし

RAM に展開して使用する関数

※以下三つの関数は、FLMWSR3 レジスタ仕様により一旦 RAM にコピーしてから呼び出します。

・ RAM_AplSum (DWORD top, DWORD bot, Func2 io_sv, WORD *pIosDiv, WORD *pWdtDiv) 関数

内容	アプリ領域のサム値を求め、0xAA との比較結果を返します。 処理中に io_sv を呼び出します。
引数	top サム計算開始アドレスを指定します bot サム計算終了アドレスを指定します io_sv io_sv(...)関数のアドレスを指定します pIosDiv io_sv()で使用する I/O サービス用カウンタのアドレスを指定します pWdtDiv io_sv()で使用する WDT リセット様のカウンタのアドレスを指定します
戻り値	1：アプリサムチェックは不一致です 0：アプリサムチェック一致しました（アプリケーションは存在します）
備考	領域内に消去状態が検出された場合、不一致とみなします。

・ RAM_PeekFlash (DWORD adr)

内容	指定アドレスの 1 バイトを読み出します。 指定アドレスが消去状態の場合、0xFF を返します。
引数	adr 読み出すアドレスを指定します
戻り値	読み出したバイトの値
備考	消去状態が検出された場合、実際の読み出し値に関わりなく 0xFF を返します。 本関数はパスワードチェックの際使用します。パスワードシステムは消去状態が 0xFF であることを前提に設計されています。

・ RAM_ExtSum (DWORD top, DWORD bot, Func io_service)

内容	指定された範囲の 8bit/16bit サムを同時に求めます。 処理中に引数で指定された io_service を呼び出します。
引数	top サム計算開始アドレスを指定します bot サム計算終了アドレスを指定します io_service io_service()のアドレスを指定します。
戻り値	上位 16bit→8bit 加算 16bit サム値 下位 16bit→16bit 加算、16bit サム値
備考	消去状態のデータは 0xFF として扱います。 本関数はフラッシュ領域のみに使用できます。RAM 上のデータには使用できません。

ノード ID からＣＡＮ ID への変換は usr_api.c に設けた以下の関数で行います。

・ usr_api_get_can_ID(unsigned long *recvID, unsigned long *sendID) 関数

内容	CAN ID 取得関数です。 ノード ID よりテーブルを参照し CAN ID を戻します。
引数	*recvID ライタからマイコンへ送る ID を戻します *sendID マイコンからライタへ送る ID を戻します
戻り値	1：ノード ID が無効です 0：ノード ID が有効です

１１．２　WCP での使用関数(y_wcp.c ファイルの関数一覧)

・ io_service(void) 関数

内容	I/O ポートサービス処理関数です。 引数を設定の上 io_sv 関数をコールします。
引数	なし
戻り値	なし

・ io_sv(WORD *sp, WORD *sp_wdt) 関数

内容	I/O ポートサービス処理関数です。 I/O ポートサービスを行います。 内蔵 WDT のクリアを行います。
引数	*sp IO サービス用のソフトウェアパルスカウンタです。 設定したカウンタ値になるまでインクリメントします。 設定したカウンタ値になるとゼロに初期化します。 *sp_wdt 内蔵 WDT 用のソフトウェアパルスカウンタです。 設定したカウンタ値になるまでインクリメントします。 設定したカウンタ値になるとゼロに初期化します。
戻り値	なし

・ reset_wdt(void)

内容	内蔵 WDT をクリアします。
引数	なし
戻り値	なし

・ RecData(void) 関数

内容	コマンドフレーム、ライトフレーム、ターミネータフレーム受信処理関数です。 ディスコネクトコマンドを受信した場合は応答を返し、リセット処理を行います。
引数	なし
戻り値	なし

・ TrmData(void) 関数

内容	リードフレーム送信処理関数です。 CAN データを送信します。
引数	なし
戻り値	なし

・ TrmReady(void) 関数

内容	レディフレーム送信処理関数です。 レディフレームを送信します。
引数	なし

戻り値	**なし**

・ TrmBusy(void) 関数

内容	ビジィフレーム送信処理関数です。 ビジィフレームを送信します。
引数	なし
戻り値	なし

・ TrmError(void) 関数

内容	エラーフレーム送信処理関数です。 エラーフレームを送信します。
引数	なし
戻り値	なし

・ exec_reset(void) 関数

内容	リセット実行処理関数です。 I/O ポートサービス用タイマのカウントストップを行い、無限ループに入ります。
引数	なし
戻り値	なし

・ Command_Read(void) 関数

内容	リードコマンド処理関数です。 読み出したデータをライタ側に送信します。
引数	なし
戻り値	なし

・ Command_Program(void) 関数

内容	プログラムコマンド処理関数です。 書き込み先アドレスと書き込みデータを受信します。 ページ単位で書き込みを行います。 ブート領域書込み時にはブートスワップ機能を使用します。
引数	なし
戻り値	なし

・ Command_BlockErase(void) 関数

内容	ブロックイレーズコマンド処理関数です。 対象アドレスから計算したブロックの消去を行います。
引数	なし
戻り値	なし

・ Command_ReadSRD(void) 関数

内容	リードステータスレジスタコマンド処理関数です。 ステータスレジスタの値をライタ側に送ります。
引数	なし
戻り値	なし

・ Com_ClearSRD(void) 関数

内容	クリアステータスレジスタコマンド処理関数です。 ステータスレジスタを初期化します。
引数	なし
戻り値	なし

・ Command_ExtBlankcheck(void) 関数

内容	拡張ブランクチェックコマンド処理関数です。 指定範囲のブランクチェックを実行し、結果をライタ側に返します。
引数	なし
戻り値	なし

・ Command_ExtSumcheck(void) 関数

内容	拡張サムチェック処理関数です。 8bit 単純加算したサム値と 16bit 単純加算したサム値をライタ側に返します。
引数	なし
戻り値	なし

・ wcp_main(void) 関数

内容	WCP のメイン関数です。 ライタからのコマンドを受信し、各関数を Call します。
引数	なし
戻り値	なし

・ flash_init (void) 関数

内容	FLASH 制御の初期設定をします
引数	なし
戻り値	なし

・ flash_write(DWORD buff,DWORD addr) 関数

内容	指定されたアドレスにページ単位でデータを書込みます
引数	buff 書き込みデータ格納先アドレス addr 書き込み先アドレス
戻り値	ステータス 1：異常終了 0：正常終了

１２．使用I/Oリソース一覧

項目	リソース	備考	ユーザ設定
サイクルアクセスポート	Port A bit 0	I/O サービスを使用する場合に使用	可
MCU 内蔵タイマ	SysTick タイマ	サイクルアクセス用	不可
CAN 通信ポート	CANRX0 = PIN42 ピン CANTX0 = PIN43 ピン	CAN0 用	可
WDT	タイマ WDT		不可

１３．付録

① 初期設定ファイル

初期設定ファイル名：y_init.h

項目	初期設定 ファイル定義名	内容	デフォルト値	ユーザ 設定
CAN チャンネル選択	CAN_CHNO	0 or 1 （1 の設定による動作は未確認です）	0	可
CAN　ボーレート	CAN_BAUD	125～1000 (Kbps 単位) 1000 ： 1M 500 ： 500K 250 ： 250K 125 ： 125K	500 (500Kbps)	可
CAN ビットレート プリスケーラレジスタ	CAN_BCR1_DATA	ボーレートプリスケーラレジスタ値 8BIT	0x01	可
CAN ビットレートレジスタ	CAN_BCR2_DATA	ビットコンフィグレーションレジスタ値	0x0000014F	可
入力クロック	CLK_EXT	MHz×10 で指定 例） 10MHZ＝100	160 (16MHz)	可
逓倍値	CLK_EXT_MUL	11 （本マイコンでは固定）	11	不可
分周値	CLK_PLL_DIV	2 （本マイコンでは固定）	2	不可
CAN クロック分周値	CLK_CAN_DIV	4 （本マイコンでは固定）	4	不可
パスワードチェック領域 （スタート）	PASS_START	32bit アドレス指定	0x00008400	可
パスワードチェック領域 （エンド）	PASS_END	32bit アドレス指定	0x0000840F	可
ユーザアプリ領域の サム値チェックアドレス （スタート）	APL_SUM_START	32bit アドレス指定	0x0007FE00	可
ユーザアプリ領域の サム値チェックアドレス （エンド）	APL_SUM_END	32bit アドレス指定	0x0007FFFF	可
I/O ポートサービス 対応フラグ	IOS_ON	0 or 1 0：I/O サービス無し 1：I/O サービス有り	0	可
I/O ポートサービスポート レジスタ	IOS_PORT	ポートレジスタのアドレスを指定	0x40021004	可
I/O ポートサービスビット	IOS_BIT	I/O ポートサービスに使用するビットを 1 に設定	0x0001	可
I/O ポートサービス 周期	IOS_PERIOD	ms 単位（1ms～65535ms）	4	可
内蔵 WDT 対応フラグ	IOS_WDT_ON	0 or 1 0：内蔵 WDT 未使用 1：内蔵 WDT 使用	1	可
内蔵 WDT クリア周期	IOS_WDT_PERIOD	ms 単位	100	可
Primary　メッセージ ID （ライタ→マイコン）	ID_P_NI	32bit 値 bit0～bit10：標準 ID bit11～bit28：拡張 ID bit31：ID のフォーマット(0:標準、1:拡張)	0x00FA2053	無効
Primary　メッセージ ID （マイコン→ライタ）	ID_P_MCU	32bit 値 bit0～bit10：標準 ID bit11～bit28：拡張 ID bit31：ID のフォーマット(0:標準、1:拡張)	0x00FD5320	無効
Primary　メッセージ ID フォーマット指定	CAN_ID	0 or 1 0：スタンダード・フォーマット 1：エクステンデッド・フォーマット	1	無効
Station アドレス	CCP_STATION	16bit 値	0x0000	可
KILL レジスタ対応フラグ	USE_KILL_R	0 or 1 0：KILL レジスタ機能無し 1：KILL レジスタ機能有り	0	不可
KILL レジスタアドレス	KILL_ADDR	32bit アドレス指定	0x00000000	不可
SecondaryID 領域	ID_ADDR	32bit アドレス指定	0x00000000	不可

②リプログ時 NET IMPRESS 設定項目（本定義体固有項目）

項目	設定方法	内容	デフォルト値	ユーザ設定
CAN ID(ライタ→マイコン)フォーマット設定 *1	FUNC 81	“STANDERD”/“EXTENDED”を選択	EXTENDED	可
CAN ID(マイコン→ライタ)フォーマット設定 *1	FUNC 82	“STANDERD”/“EXTENDED”を選択	EXTENDED	可
CAN 通信ボーレート設定 *1	FUNC 83	500Kbps/1Mbps/250Kbps/125Kbps を選択	500K	可
CAN ID 設定 *1	FUNC 86	ライタ→マイコン、マイコン→ライタの標準／拡張 ID 設定	ライタ→マイコン 標準 : #03E 拡張 : #22153 マイコン→ライタ 標準 : #03F 拡張 : #15321	可
KILL レジスタ ON	FUNC 87	KILL レジスタを ON に設定します		
ブランクチェックモード設定	FUNC 88	“NORMAL BLANK” （NI で読み出しチェック） “EXTENDED BLANK” （マイコン側でチェック）	EXTENDED BLANK	可
パスワードチェック領域	.KEY ファイルにて指定	初期設定ファイルで指定した、パスワードチェック領域スタート～エンドの範囲内に 7byte～255byte で指定		可
CONNECT Station Address *1	Specific Parameter #0D8	AZ990 等で Specific Parameter #0D8 から 2BYTE 指定する	#0000	可
ユーザアプリ領域サムチェックスタートアドレス *1	Specific Parameter #140	AZ990 等で Specific Parameter #140 から 4BYTE 指定する	#0007FE00	可
ユーザアプリ領域サムチェックエンドアドレス *1	Specific Parameter #144	AZ990 等 Specific Parameter #144 から 4BYTE 指定する	#0007FFFF	可
r Entry コネクト発行タイミング(5 章 t1)	Specific Parameter #14C	AZ990 等で Specific Parameter #14C から 2BYTE 指定する 1～999ms(1ms 単位)	#15 (10ms)	可
Entry モード	Specific Parameter #14F	AZ990 等で Specific Parameter #14F から 1BYTE 指定する 00： n Entry モード 01： r Entry モード	#00	可
CONNECT ビジー(36h)応答時リトライ周期	Specific Parameter #E5	AZ990 等で Specific Parameter #E5 から 1BYTE 指定する 0～2550ms（10ms 単位）	#14 (200ms)	可
CONNECT ビジー(36h)応答時リトライ回数	Specific Paramete #E6	AZ990 等で Specific Parameter #E6 から 1BYTE 指定する 0～255 回	#05 (5 回)	可

*1 初期設定ファイルと同期をとる項目です。