NAPYDC839GH15　CAN プログラミング技術資料

対応 MCU 名
SH72AY3

＜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

NAPYDC839GH15 の適用 NETIMPRESS シリーズ本体は、NETIMPRESS air(AF930)です。

C”arNETIMPRESS,G-NETIMPRESS,NETIMPRESS next ではご使用になれません。

作成日：2015 年 8 月 11 日　初版

エンベデッドソリューション事業本部

変更履歴

変更日付	変更内容
2015.08.11	新規作成

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	SH72AY3	
書き込み対象アドレス	#00002000～#000BFFFF(760Kbyte) *1 #00000000～#000BFFFF(768Kbyte) *2	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	入力=20MHz、動作=160MHz、逓倍値=8、分周値=1 周辺クロック A=40MHz(4 分周)	可
CAN チャネル番号	チャネル 1	可
モード制御端子	なし	
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	不可
プローブ	AZ915 or AZ916	不可
CCP バージョン	Version 2.1 をベースとする *3	不可
開発環境	High-performance Embedded Workshop 4.09.01.007 *4	可
コンパイラバージョン	SuperH RISC engine family Ver 9.04 Release01 *4	可
最適化	サイズ&スピード	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	192byte	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 IBL をユーザマットに配置した場合

＊2 IBL をユーザブートマットに配置した場合

＊3 CCP を拡張したプロトコルです。完全互換性はありません。

＊4 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いたしません。

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
ﾏｲｺﾝﾊﾟｯｸﾏﾆｭｱﾙ	MNJ-NAPYDC839H15	
CAN 共通ﾌﾟﾛｸﾞﾗﾐﾝｸﾞ 技術資料	TR-NAPYDC839GH15	本書
ﾏｲｺﾝﾊﾟｯｸ	NAPYDC839GH15	・FDF シート内容 BTP ﾌｧｲﾙ *1 AMK ﾌｧｲﾙ KEY ﾌｧｲﾙ YSM ﾌｧｲﾙ ｻﾝﾌﾟﾙ APL ｵﾌﾞｼﾞｪｸﾄ *2
ｻﾝﾌﾟﾙﾌﾟﾛｸﾞﾗﾑ ｱﾌﾟﾘｿﾌﾄ例（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_NAPYDC839GH15 |-Header |　|-y_ibl.h |　|-y_init.h | |-IBL_NAPYDC839GH15 |　|-user_init.h |　|-user_apl.c |　|-user_ipr_init.src |　|-user_ipr.c |　|-y_ibl_init.src |　|-y_ibl.c |　|-Release¥BOOT.mot |-WCP_NAPYDC839GH15 |　|-y_wcp_init.src |　|-y_wcp.c	 固有値定義ﾌｧｲﾙ 初期設定ﾌｧｲﾙ ﾕｰｻﾞ APL/IPR ｻﾝﾌﾟﾙﾌｧｲﾙ 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 IBL ｽﾀｰﾄｱｯﾌﾟﾙｰﾁﾝ CAN ﾘﾌﾟﾛｸﾞ用ﾌﾞｰﾄﾛｰﾀﾞ IBL/IPR（例）ｵﾌﾞｼﾞｪｸﾄ *3 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

＊1 BTP ﾌｧｲﾙは「Header , WCP_NAPYDC839GH15」ﾌｫﾙﾀﾞ内のﾌｧｲﾙをｺﾝﾊﾟｲﾙすることにより作成されるﾌｧｲﾙです。

＊ 2　ｻﾝﾌﾟﾙ APL ｵﾌﾞｼﾞｪｸﾄは「Header , IBL_NAPYDC839GH15」ﾌｫﾙﾀﾞ内のﾌｧｲﾙをｺﾝﾊﾟｲﾙすることにより作成される APL.mot ﾌｧｲﾙです。

＊3 IBL/IPR （例） ｵﾌﾞｼﾞｪｸﾄは「Header , IBL_NAPYDC839GH15」ﾌｫﾙﾀﾞ内のﾌｧｲﾙをｺﾝﾊﾟｲﾙすることにより作成される Boot.mot ﾌｧｲﾙです。

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、お客様次第です。

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」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では「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ID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。

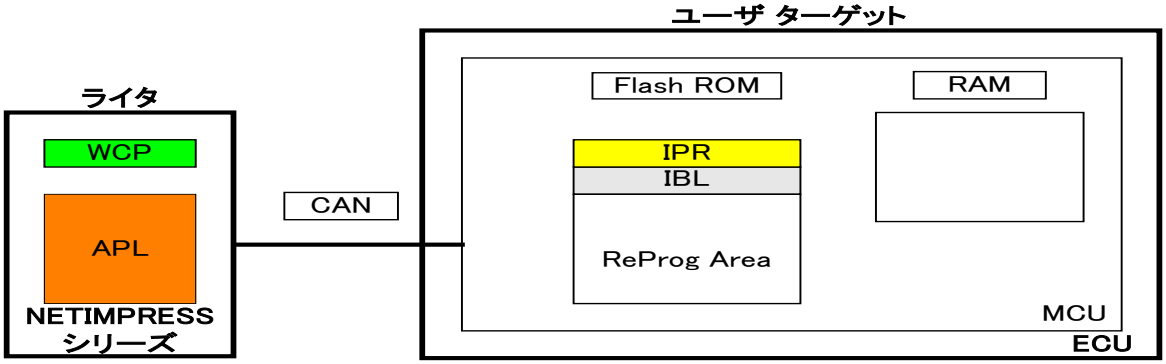
※1：「暗号機能」については「UCOP_CAN Programmer」のインストラクションマニュアルを
参照してください。

ユーザアプリ領域サム値チェック

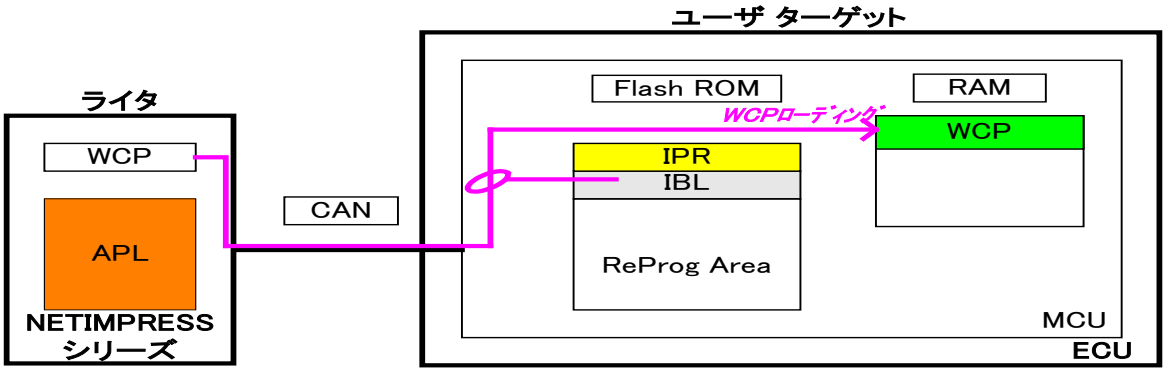
UCOP では IBL において、お客様のアプリケーションプログラムが既に書かれているかどうか
をサム値にて判断します。このサム値チェックのことを「ユーザアプリ領域サム値チェック」といい
ます。
サム値計算領域は”y_init.h”ファイルで変更することが可能です。

3. 機能概要

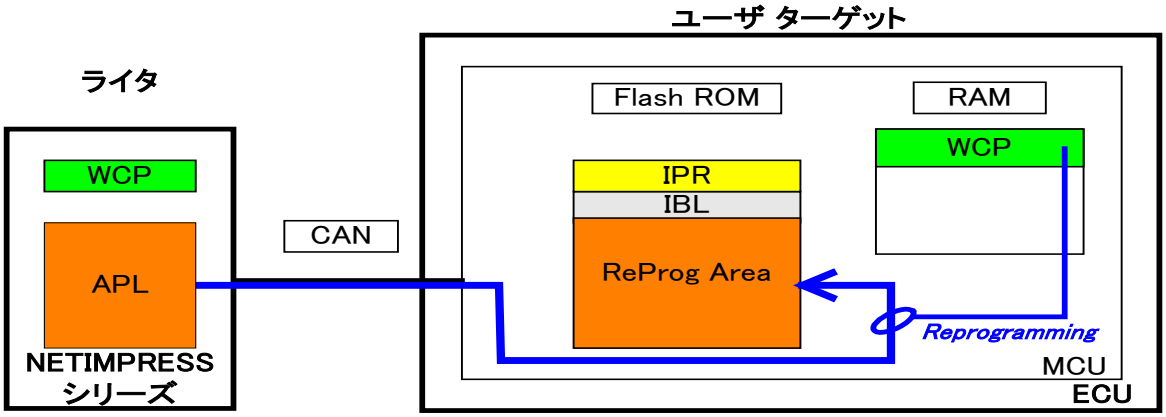
3. 1　UCOP システム構成図



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。

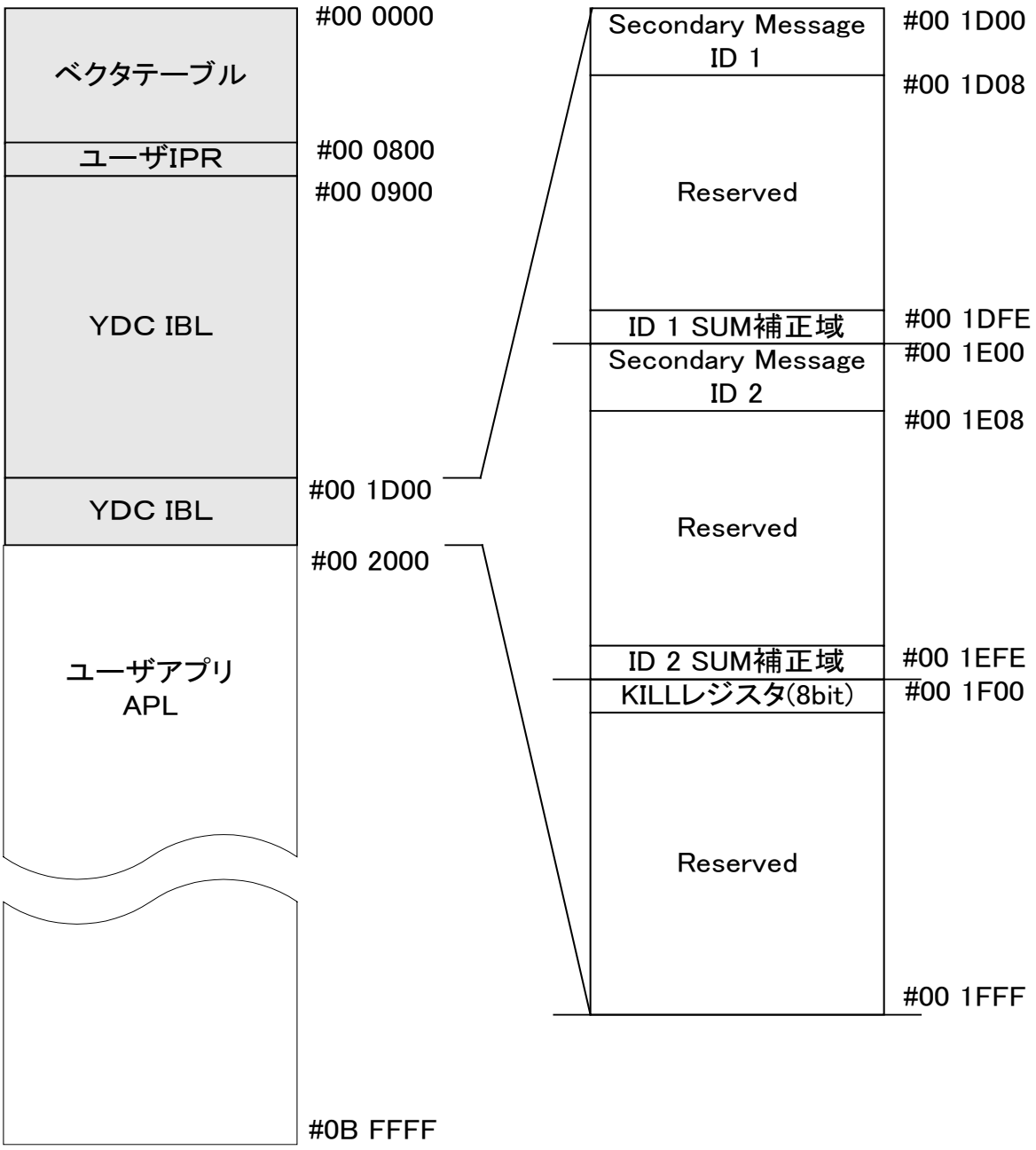


6. ライタは WCP と通信をおこないライタにある APL を ReProg Area に書き込みます。

3．2 ROM

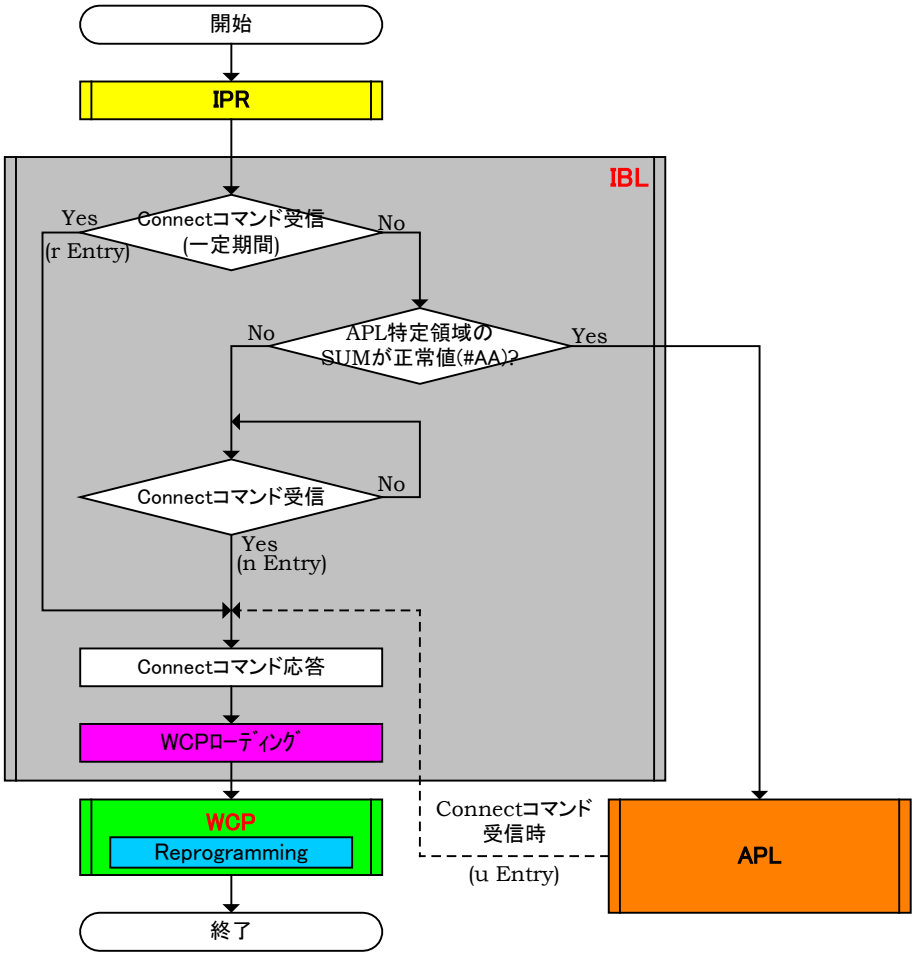
3．2．1 ROM構成図

SH72AY3



灰色部は書き換え禁止領域。ただし#1D00～#1FFF までの一部パラメータ領域を特殊コマンドにて書き換えます。また、IBL をユーザブートマットに配置した場合は、灰色部も書き換え可能です。

3．3　プログラムエントリー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(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(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。

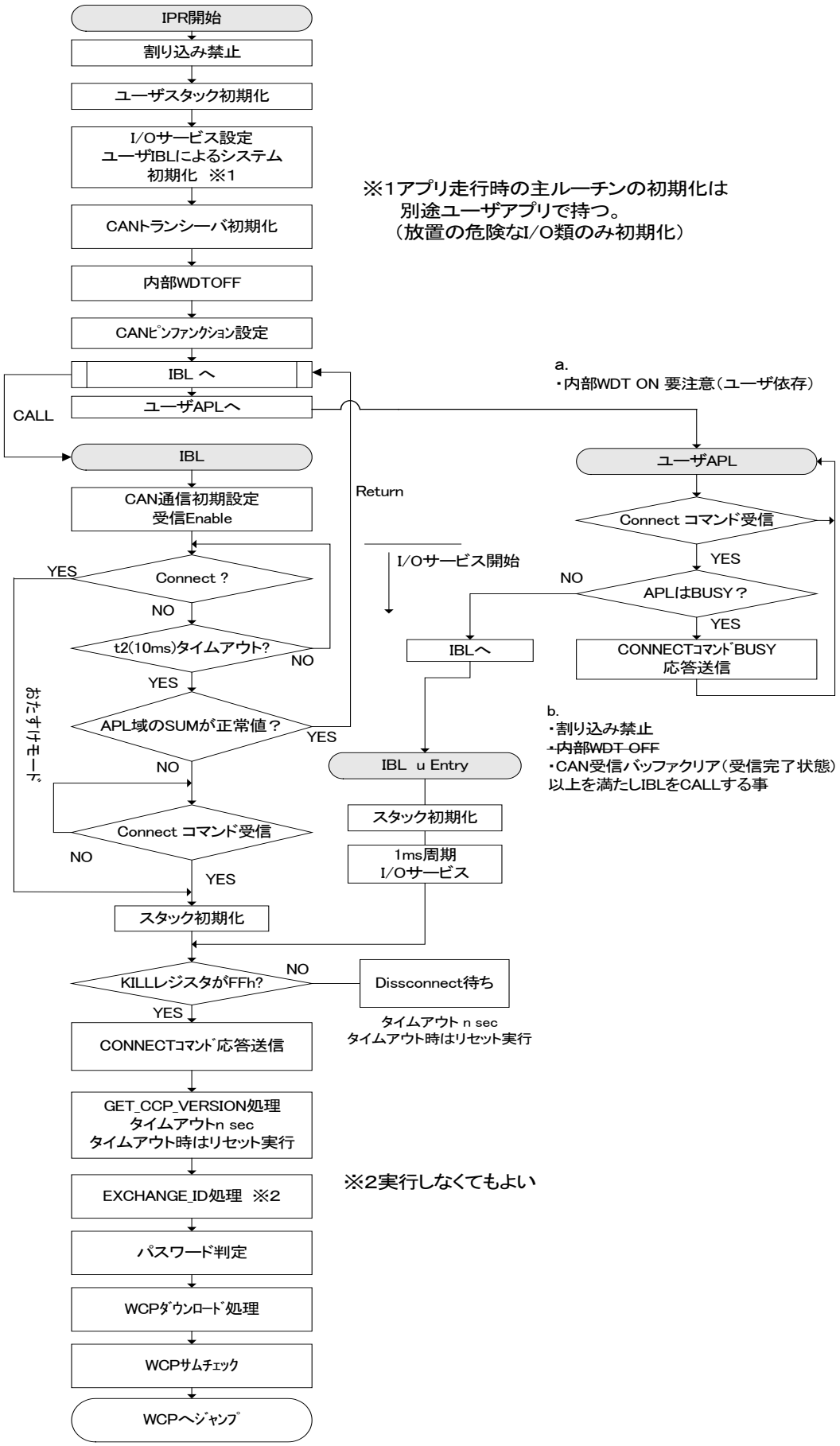
※ライタは Connect コマンド発行後規定時間(25ms)応答がない場合、Disconnect コマンドを発行し、ターゲット MCU との接続を解除します。

※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。

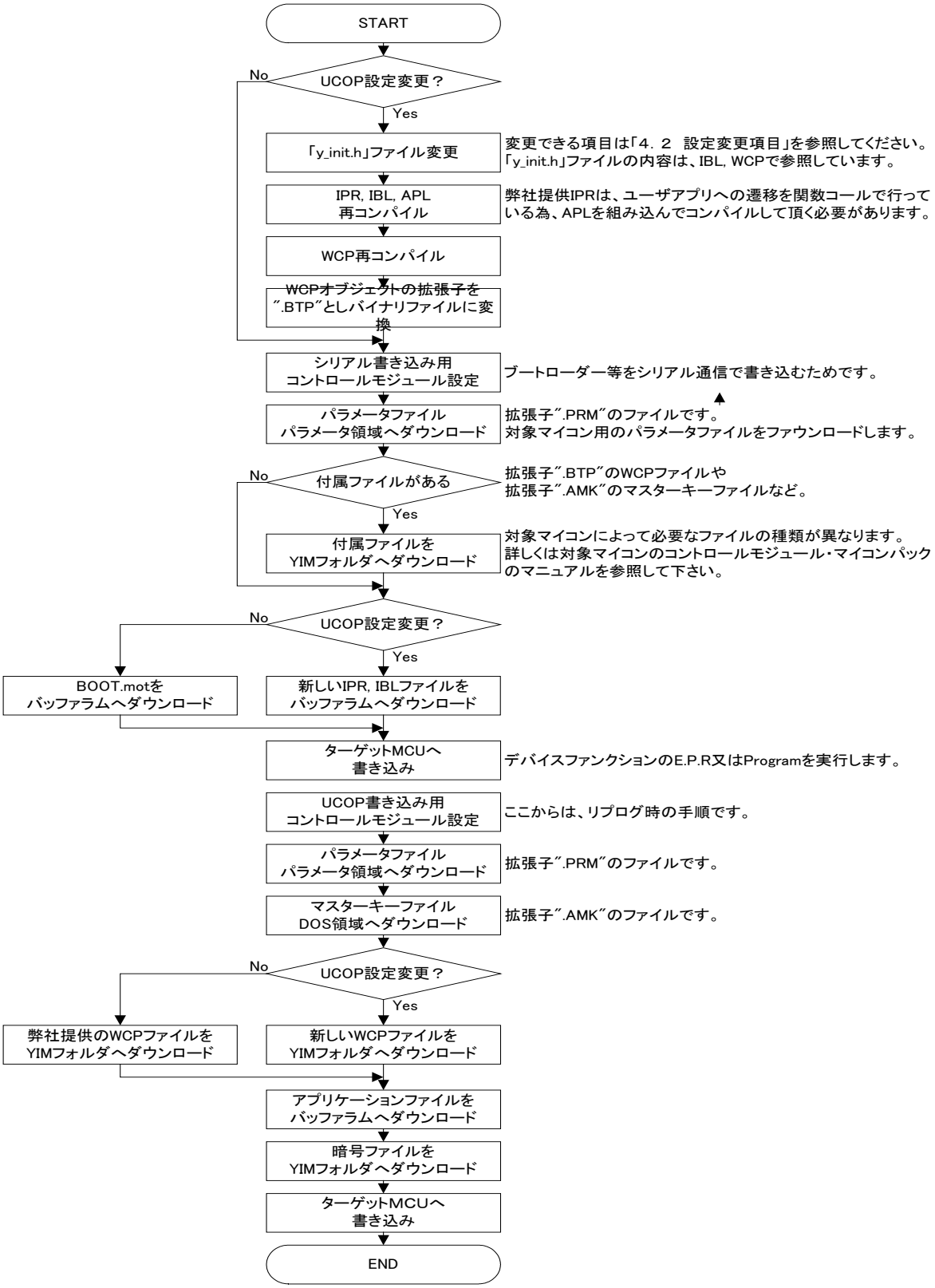
※ターゲット MCU は Disconnect コマンド受信後、リセット状態に戻るものとします。

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」、「５．６　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」、「５．６　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」、「５．６　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」、「５．６　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。

４．２．２．４　周辺機能クロックAの変更

「５．５　周辺機能クロック A」、「５．６　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の変更

「５．６　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　CANチャネルの変更

「5．17　CAN チャネルの変更」を参照してください。

4．2．10　内蔵WDT設定

「5．18　内蔵ウォッチドッグタイマ設定」「5．19　内蔵ウォッチドッグタイマアンダーフロー周期設定」を参照してください。

4．2．11　KILLレジスタアドレス設定

「5．20　KILL レジスタアドレス設定」を参照してください。

4．2．12　ロックビット設定

「5．21　ロックビット設定」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。
AZ990 の詳細な操作方法は AF930 の操作マニュアル(プログラマ編)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_CFG_DATA”も再設定してください。
「5. 6 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 25	0x 40
500 Kbps	0x 16	0x 01
250 Kbps	0x 16	0x 03
125 Kbps	0x 16	0x 07

※ 「5. 21 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”と”user_ipr.c”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）20MHz の場合、200 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_CFG_DATA”も再設定してください。

「５．６ 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。

※　このマイコンでは入力クロック周波数は「20MHz」固定です。

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。

①. 初期設定ファイル“y_init.h”設定変更

” CLK_EXT_MULT”に設定値を設定してください。

f(PLL)の動作周波数を、下記 2 種類から設定ができます。

「6」の場合：120MHz

「8」の場合：160MHz

※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_CFG_DATA”を再設定してください。

「５．６ 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

PLL 周波数シンセサイザの分周値から、システムクロックの動作周波数を設定します。

①. 初期設定ファイル“y_init.h”設定変更

” CLK_PLL_DIV”に設定値を設定してください。

f(SYS)の動作周波数を、下記 5 種類から設定ができます。

1=分周なし、2=2 分周、4=4 分周、8=8 分周、16=16 分周

※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_CFG_DATA”を再設定してください。

「５．６ 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　周辺機能クロック A

システムクロックの分周値から、周辺機能クロックの動作周波数を設定します。

①. 初期設定ファイル“y_init.h”設定変更

” CLK_PBA_DIV”に設定値を設定してください。

f(PBA)の動作周波数を、下記 2 種類から設定ができます。

2=2 分周、4=4 分周

※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_CFG_DATA”を再設定してください。

「５．６ 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。

５．６　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。

ビットコンフィグレーション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_CFG_DATA”に、CAN1 ビットコンフィグレーションレジスタへの設定値を設定してください。

Bit31-Bit26　:予約(0)

Bit25-Bit24　:再同期ジャンプ幅制御ビット

Bit23　　:予約(0)

Bit22-Bit20　:タイムセグメント 2 制御ビット

Bit19-Bit16　:タイムセグメント 1 制御ビット

Bit15-Bit10　:予約(0)

Bit9-Bit0　:プリスケーラ分周比ビット

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　CAN クロック分周比」「５．５　周辺機能クロック A」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になります。

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になります。
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 でのみの変更となります。
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「0xFFAABB00」と設定する場合

Specific Parameter のアドレス	#140	#141	#142	#143
設定値	0xFF	0xAA	0xBB	0x00

※ 「５．２１　Specific Parameter 変更方法」を参照下さい。

開始アドレスは 256byte 境界となるよう設定して下さい（詳細は ６－４. 参照）

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。

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 でのみの変更となります。

Specific Parameter 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「0xFFEEDDFF」と設定する場合

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

※ 「５．２１　Specific Parameter 変更方法」を参照下さい。

終了アドレスはサイズが 256byte 区切りとなるよう設定して下さい（詳細は ６－４．参照）

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（６－１４．参照）。
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、初期設定ファイル”y_init.h”ファイルを変更し、”y_ibl.c”、”y_wcp.c”を再コンパイル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_PORT”,” IOS_CR”,” IOS_BIT”に I/O ポートサービスを行うポートの情報をセットします。

IOS_PORT：I/O ポートサービスを行うポートのポート出力レジスタのアドレスをセットして下さい。

IOS_CR：I/O ポートサービスを行うポートのポート制御レジスタをセットして下さい。

IOS_BIT：I/O ポートサービスを行うボートのビットを選択してください。

ポート x0 であれば 1 を、ポート x3 であれば 8 というように、ビットを選択してください。

５．１４ Primary ID

Primary ID 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。

ライタからマイコンへ送る Primary ID を設定する“ID_P_NI”と、マイコンからライタへ送る Primary ID を設定する“ID_P_MCU”があります。

必要に応じて“ID_P_NI”や“ID_P_MCU”の変更を行ってください。

また、通信を行うためにはライタ側のアイデンティファイヤ設定の変更も必要です。

Primary ID の詳細は「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. 初期設定ファイル“y_init.h”設定変更

“ID_P_NI”、“ID_P_MCU”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 ビットは下記のように割り当てられています。

Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24
			EXD_ID17	EXD_ID16	EXD_ID15	EXD_ID14	EXD_ID13

Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16
EXD_ID12	EXD_ID11	EXD_ID10	EXD_ID9	EXD_ID8	EXD_ID7	EXD_ID6	EXD_ID5

Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
EXD_ID4	EXD_ID3	EXD_ID2	EXD_ID1	EXD_ID0	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

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 31、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

“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

“ID_P_NI”：0x000007E9

“ID_P_MCU”：0x000007EA

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

“ID_P_NI”：0x1F6E5FE9

“ID_P_MCU”：0x1F6E5FEA

※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は CAN ID フォーマット設定で指定する必要があります。

「5．15 CAN ID フォーマット設定」を参照してください。

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。

※ 「5．21 Specific Parameter 変更方法」を参照下さい。

#0C4～#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 Parameter のアドレス	#0C4	#0C5	#0C6	#0C7
設定値	0x00	0x00	0x07	0xEA

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

Specific Parameter 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、初期設定ファイル"y_init.h"と、ライタ側で行います。

①. 初期設定ファイル"y_init.h"設定変更

“CAN_ID”にスタンダード・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を設定します。

0:スタンダード・フォーマット使用

1:エクステンデッド・フォーマット使用

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。

ステーションアドレスの詳細は「6. 12 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」と設定します。

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. 21 Specific Parameter 変更方法」を参照下さい。

5．17　CAN チャネル番号

CAN モジュールのチャネルが複数存在する場合、どのチャネルを使用するかを設定します。
CAN のチャネルを変更するには、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_CHNO“に CAN チャネル番号を設定してください。

5．18　内蔵ウォッチドッグタイマ設定

内蔵ウォッチドックタイマ対応フラグを変更することで内蔵ウォッチドックタイマの有効/無効を設定します。
内蔵ウォッチドックタイマ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
　“WDT_ON”に「0」か「1」を設定します。
　「1」の場合：ウォッチドックタイマ有効
　「0」の場合：ウォッチドックタイマ無効

5．19　内蔵ウォッチドッグタイマクロック設定

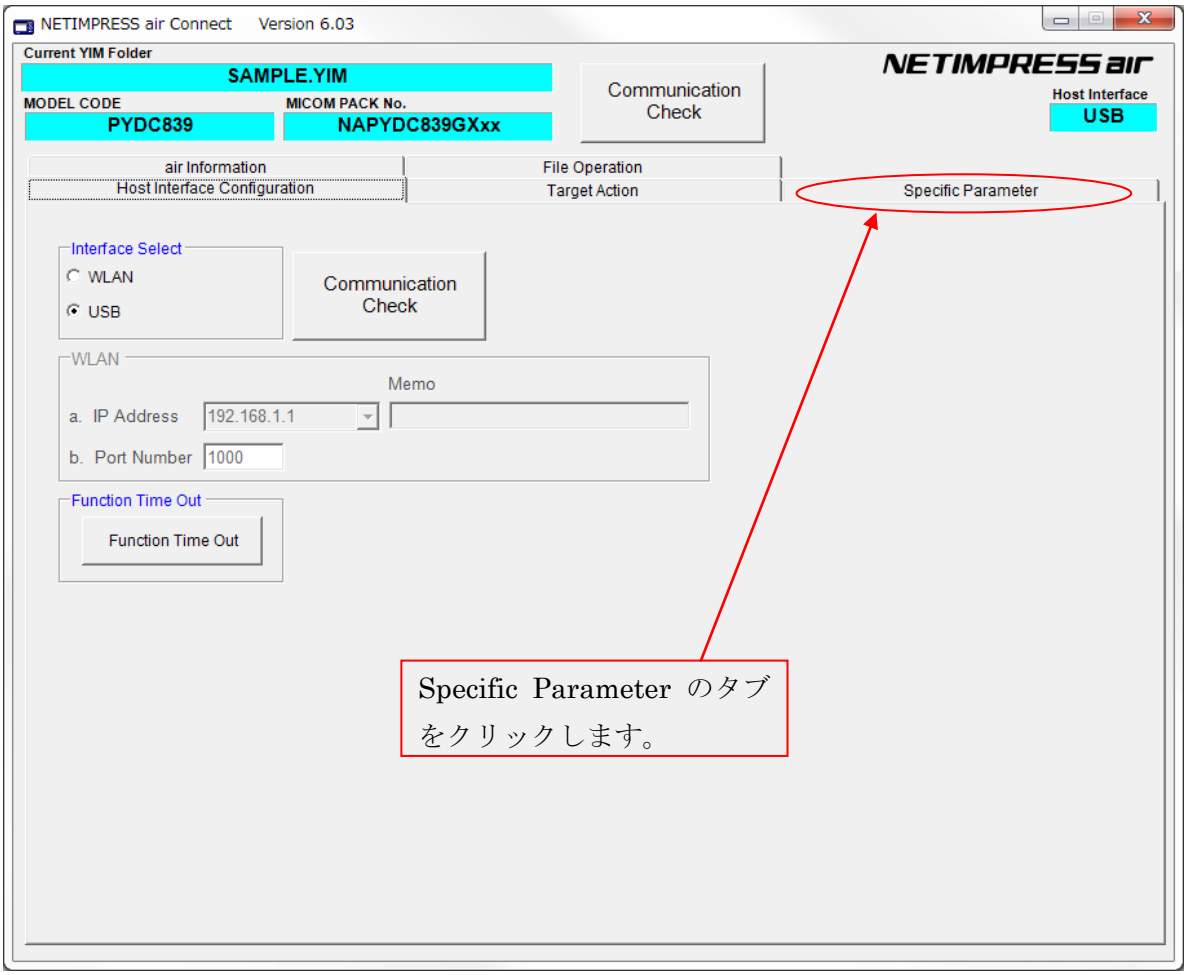
内蔵ウォッチドックタイマのアンダーフロー周期を設定します。
変更するには初期設定ファイル“y_init.h”の変更が必要です。
①. 初期設定ファイル“y_init.h”設定変更
　“WDT_PIRIOD”に値を設定してください。
　下記0～7の8種の設定ができます。（タイマ時間は入力クロック 20MHz の場合）
　　周辺機能クロック A の 16 分周
　　0x00=0.4ms、0x01=1.6ms、0x02=3.3ms、0x03=6.6ms
　　周辺機能クロック A の 128 分周
　　0x80=3.3ms、0x81=13.1ms、0x82=26.2ms、0x83=52.4ms

5．20　KILL レジスタアドレス設定

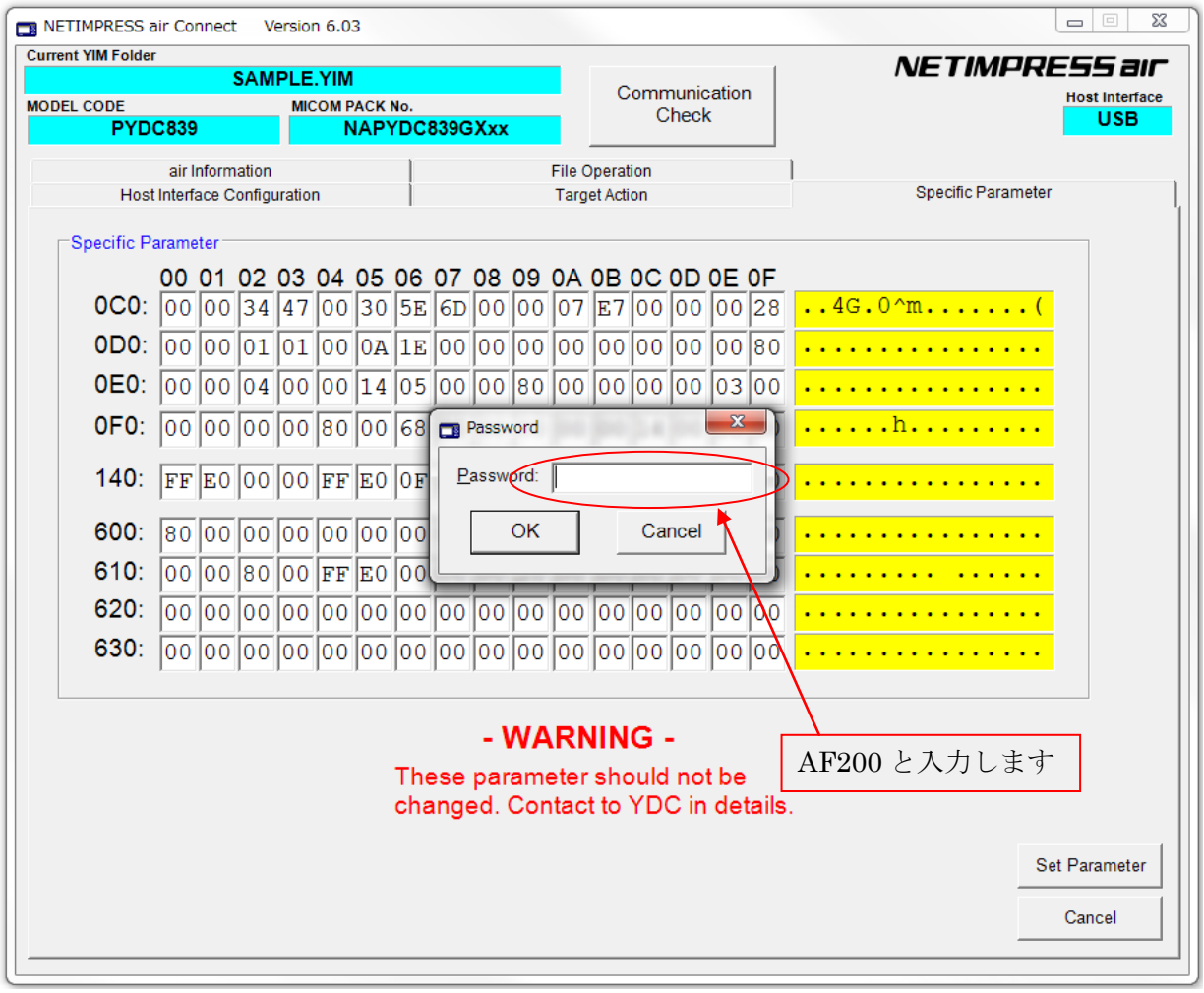
KILL レジスタアドレスを変更するには、初期設定ファイル”y_init.h”を変更する必要があります。
KILL 機能の使用方法は「6.9　KILL レジスタ」を参照下さい。
①. 初期設定ファイル”y_init.h”設定変更
　“KILL_ADDR”に KILL レジスタのアドレスを設定してください。

5. 21 Specific Parameter 変更方法

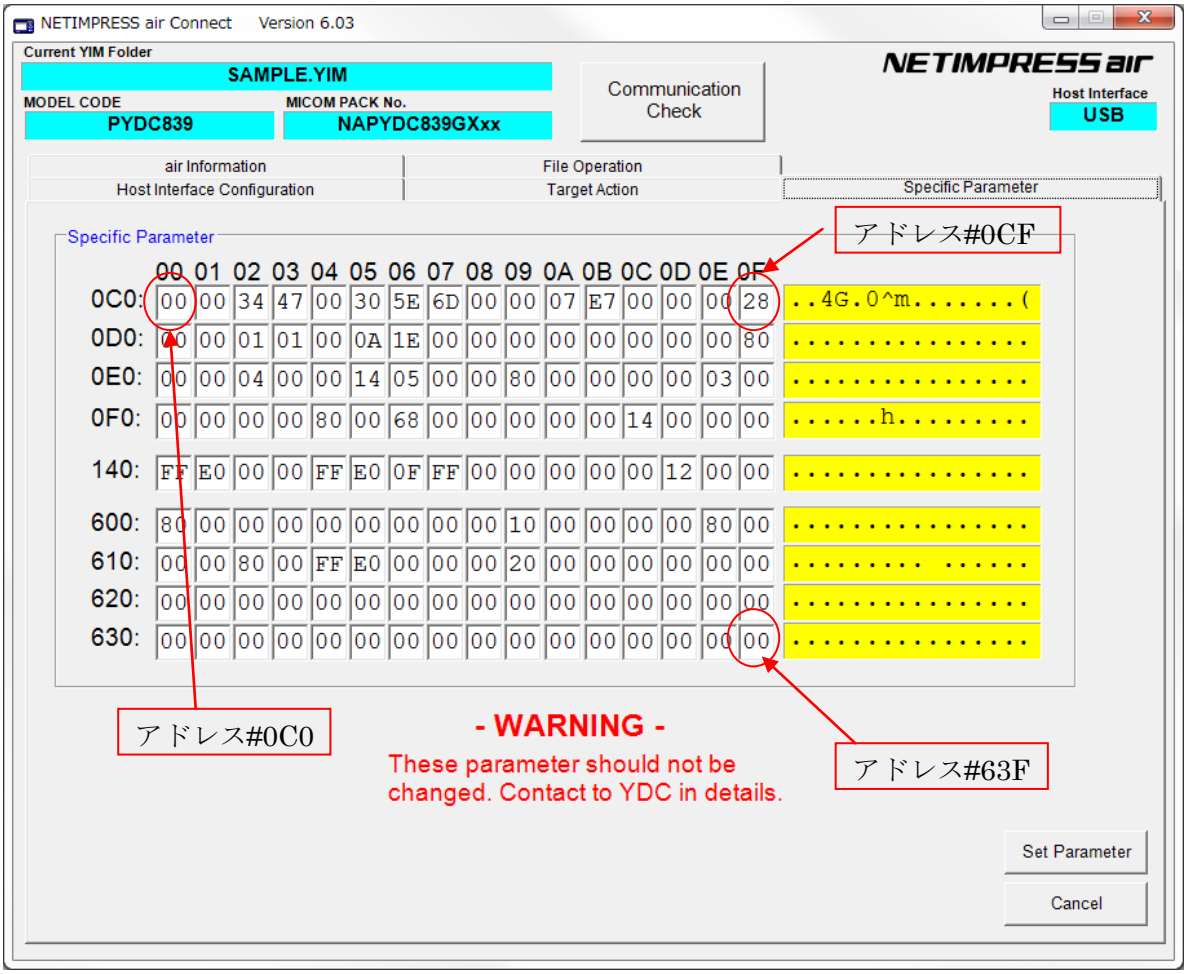
ここでは、air Connect の Specific Parameter の変更方法を説明します。
まずは、air Connect を起動し、NET IMPRESS と接続してください。
Specific Parameter のタブをクリックし、Specific Parameter を開きます。



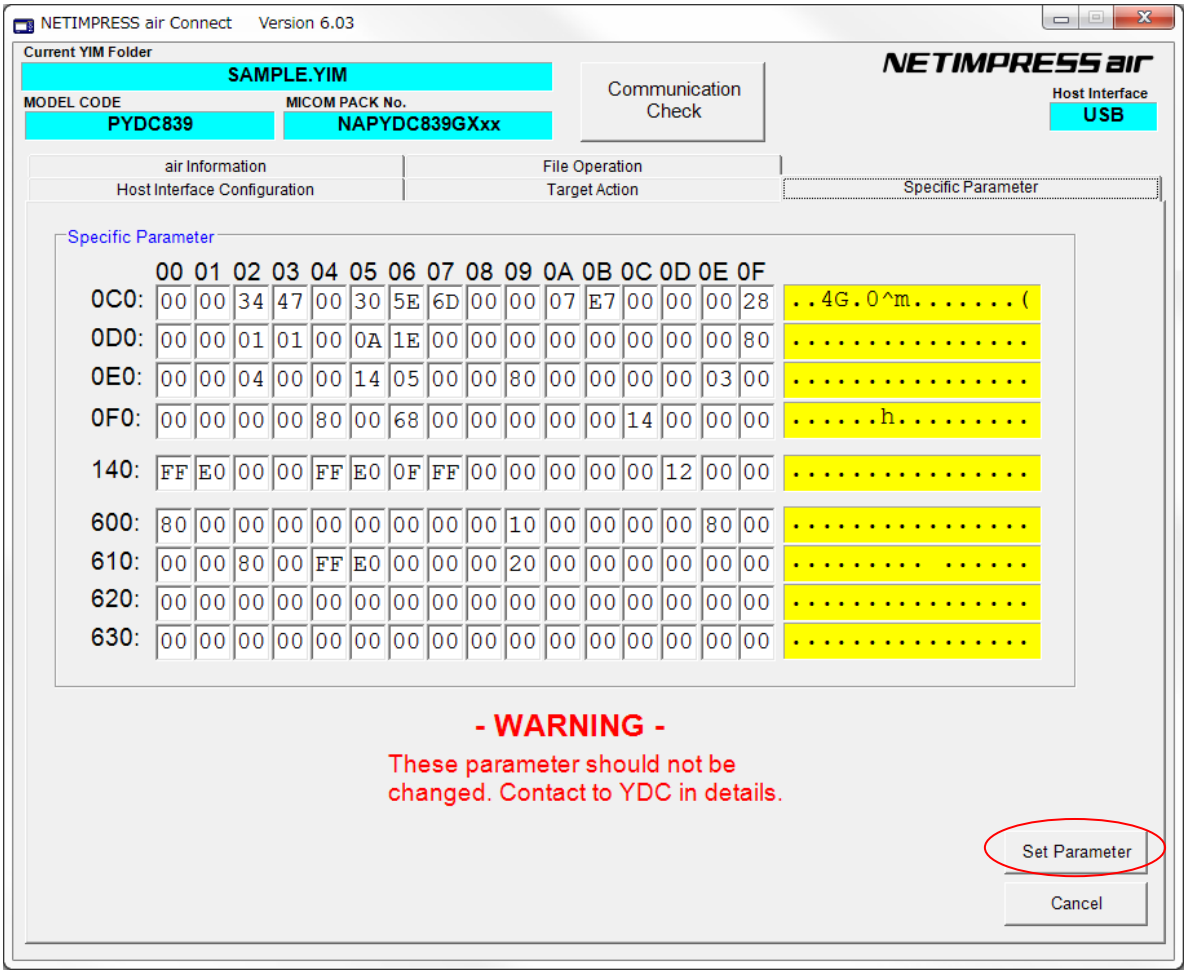
パスワードを求めるウィンドウが開きますので、”AF200”と入力し、OK ボタンを押してください。



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”Save 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	#000800h～#0008FFh （スタートアドレス:#000800h） *1	不可
IPR 要初期化項目	■ 割り込み禁止 ■ スタック初期化 ■ クロック逓倍分周設定 ■ CAN ポート設定	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_ibl.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	#000900～#001FFF	不可
IBL エントリーアドレス From IPR	#000900h	不可
IBL エントリーアドレス From APL	#000906h	不可

※ 対象マイコン共通とします。
※ 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：サイズ ＆ スピード。

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）

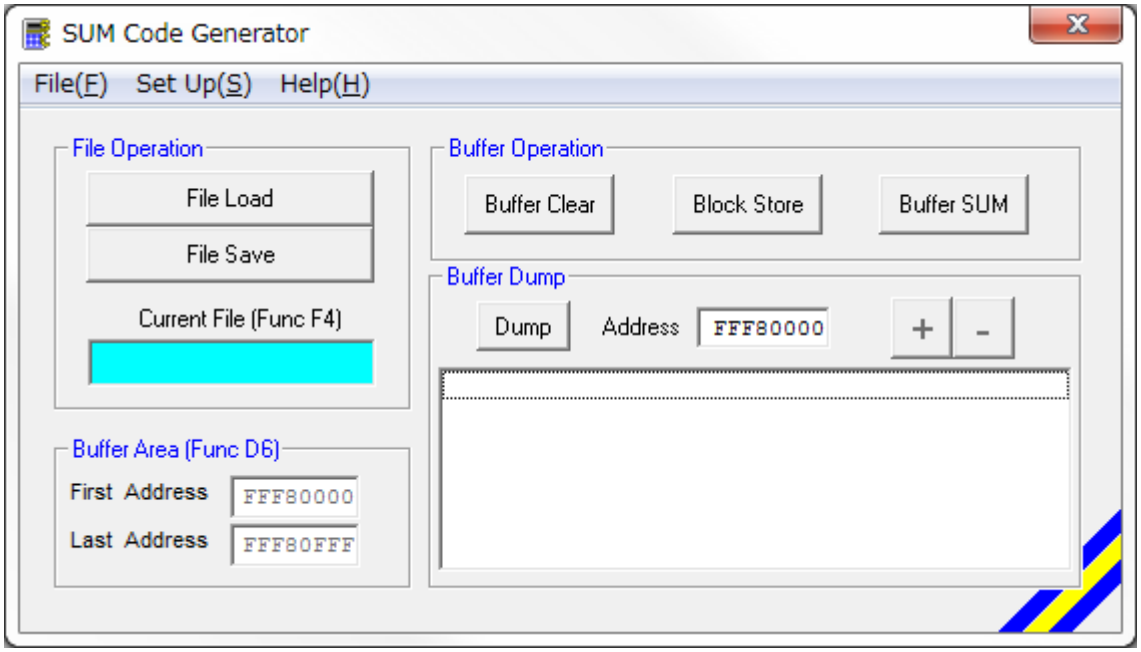
ソースファイルは「y_wcp.c」として供給されます。
実行ファイルは拡張子が BTP のファイルとして供給されます。
YIM フォルダ内に配置してください。
ライタはあらかじめ ROM 内に組み込まれている IBL と通信を行い、BTP ファイルを順次送信します。
IBL はターゲットの内蔵 RAM へ受信した BTP ファイルを書き込みます。

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
SH72AY3	#FFF80000～#FFF80FFF	不可

※ WCP のコンパイルはサイズ＆スピード最適化オプション指定。

ソースをコンパイルすると”wcp.mot”ファイルが作成されます。
上記ファイルのフォーマットは MOTOROLA S であるため、BTP ファイルのフォーマット形式(バイナリ)に変換する必要があります。
データのフォーマット変換には弊社ソフトの AZ286 をお使いください。
(AZ286 に関しては、弊社サポートにご連絡ください。)



AZ286 にて、バッファのアドレスを WCP 配置アドレスにします。
(今回の場合は#FFF80000～#FFF80FFF になります。)
その後、”wcp.mot”ファイルをロードし、セーブ時にバイナリ形式を選択して保存すれば、バイナリ形式のファイルが作成されます。

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 が書き込まれていれば 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. 書き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．この領域は“256byte～4Kbyte”の範囲で規定するものとします。

ⅱ．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。

ⅲ．先頭のアドレスは、書き込みページの先頭(下位 8bit “00h”)とします。

ページ：データ転送のアラインメント単位をここではページと呼びます。

ⅳ．サイズは 256byte 区切りとしてください。

Ⅴ．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	#00010000～#00010FFF （デフォルト値※）	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。

※ 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。
初期設定ファイルにて ID_P_NI,ID_P_MCU の値を変更したあと、コンパイルし、シリアルライタ等で IBL を書き換えてください。(５．１４ Primary ID 参照)

６．５．１ Primary ID

項目	内容	ユーザ設定
ライタ→マイコン	7ED	可
マイコン→ライタ	7EE	可

※Primary ID は初期設定ファイル(y_init.h)に登録します。

６．５．２ Secondary ID

・Secondary Message ID 情報域を設け、この領域中の最新 ID を P-on 時に Secondary Message ID として CAN モジュールに設定します。(YDC－IBL 中で)

・Secondary ID の設定確定条件

１．Secondary ID 域が ALL #FF(デフォルト)である場合 Secondary ID は設定されていないとみなします。

２．Secondary ID 域が ALL #FF(デフォルト)でない場合
設定された Secondary ID の正当性のチェックを行います。
Secondary ID 域と別に 16BIT の SUM 補正領域を持ち、Secondary ID 域の SUM 値(8bit 加算 16bit オーバーフロー無視)と SUM 補正領域を 16bit 加算し#AAAA になる場合に Secondary ID を有効とします。

・ライタメニューの SUB SETTING->CAN EXT ID SET にて送受信１組を一度に設定します。送信のみや受信のみ設定する場合、設定しない方の ID は「スタンダード ID の#7FF」として下さい。

・Secondary ID を追加できる回数は規定されております。(１回の追加は組で行います。)
この規定回数を超える追加はできません。

項目	内容	ユーザ設定
Secondary Message ID 追加可能回数	2 回	可

・追加更新不可時にはエラー応答を返します。

項目	内容	ユーザ設定
Secondary Message ID 情報域	#001D00～#001EFF	不可

※Secondary ID が有効になるのは再リセット(Disconnect)後
または次回 P-ON 時です。

※注意：Secondary ID を設定する際は動作中のリセットや電源 OFF 等ないよう

十分注意し不正書き込みが行われないようにして下さい。

動作説明

・Secondary ID が設定されていない場合、メッセージIDは Primary ID を使用します。
Secondary ID が設定されている場合、メッセージIDは優先的に Secondary ID を
使用します。従いまして Secondary ID を設定した場合、Primary ID は使用できません。

・Secondary ID は最後に追加した ID が有効になります。
最後に追加した ID の SUM 補正エリアによる正当性チェックが不正だった場合
Primary ID を使用します。

Secondary ID を 1 回追加した例）

	Primary ID	Secondary ID (ﾗｲﾀ→ﾏｲｺﾝ)	Secondary ID (ﾏｲｺﾝ→ﾗｲﾀ)	動作 ID
例 1	7ED･7EE	A	－	A･7EE
例 2	7ED･7EE	－	B	7ED･B
例 3	7ED･7EE	A	B	A･B
例 4	7ED･7EE	－	－	7ED･7EE

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オブジェクトとして CAN1 受信バッファ 0 を使用します。

･IBL は送信用メッセージオブジェクトとして CAN1 送信バッファ 0 を使用します。

・ID 完全一致として使用します。

・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	CAN1 受信バッファ 0	不可
IBL 送信用 メッセージバッファ	CAN1 送信バッファ 0	不可

６．５．４　チャネル

チャネルが複数存在するマイコンでは、使用する CAN のチャネルを設定することが可能です。
初期設定ファイルの CAN_CHNO の値を設定した後、コンパイルして IBL を書き換えてください。
（「５．１７　CAN チャネル番号」参照）

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種のエントリーモードが存在します。

エントリーモード	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ーモード	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 の場合のエントリーモード ※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ーモード	非標準

※各エントリーのフローは「３．３　プログラムエントリー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	■ CAN 設定条件変更不可 *1 □ CONNECT コマンド受信 (受信後 20ms 以内(※2)で IBL を CALL する) *3	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	可

※1. IBL で使用するメッセージバッファはの設定は変更しないで下さい。

（「６．５．３ 送受信メッセージバッファ」参照）

※2. 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

※3. 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	Data 2	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

６．９　KILL レジスタ

・8bitで構成され、初期設定ファイルにアドレス情報が格納されています。
・KILLレジスタがOFFの時（レジスタ値“#FF”）は、コネクトコマンド受信後、コネクト応答をライタに返し、UCOPリプログモードを続行します。
・KILLレジスタがONの時（レジスタ値“#FF”以外）は、コネクトコマンド受信後、コネクト応答せず、コマンド受信待ち状態になります。
コマンド受信後、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。

注）1度KILLレジスタをONに設定すると、2度とOFFにすることは出来ませんのでご注意ください。

項目	内容	ユーザ設定
KILL レジスタアドレス	0x00001F00	可

6．10　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 に対するレジスタ情報を設定します。
ビットレートレジスタの値を設定してください。
　詳細は「5．6　ビットレートタイミングレジスタ値」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。

6．11　ステーションアドレス

・16bit　構成で CCP プロトコルにて使用します。
・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	#0000h	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。

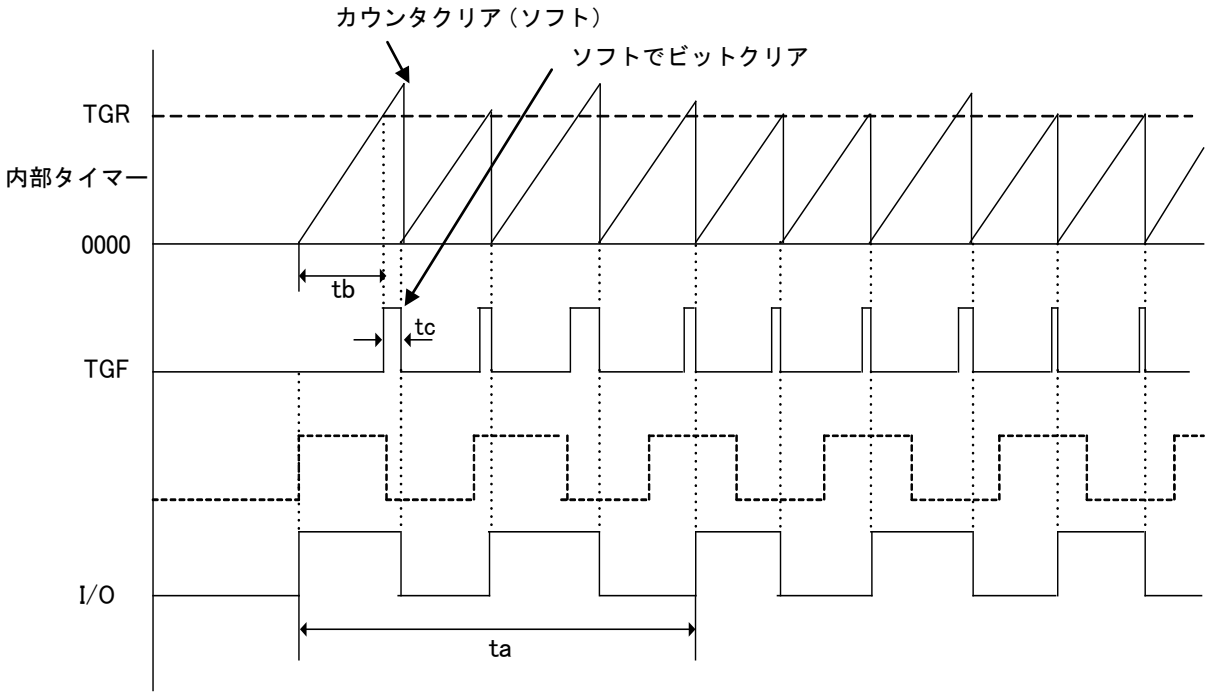
6．12　プログラム終了時の処理

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、または内蔵 WDT のリセットにより ECU のリスタートを行います）

6．13　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



TGR:タイマカウンタ
TGF:タイマオーバーフロービット
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　CMT0	
ta	ta = 2×(tb+tc)×t1 ms　(デフォルト t1=10ms) t1･･･min 1ms t1 の設定は 1ms 単位とする。	t1 設定可
tb	500μsec	
tc	Max:500μsec	
I/O ポート番号	PA06	可

※I/O ポートはビット単位のアクセス可能なポートであること。
※uEntry 時は直ちに1ms 周期固定で1回パルス出力し通常のパルス動作に移行する。

6．14 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 160 MHz で測定

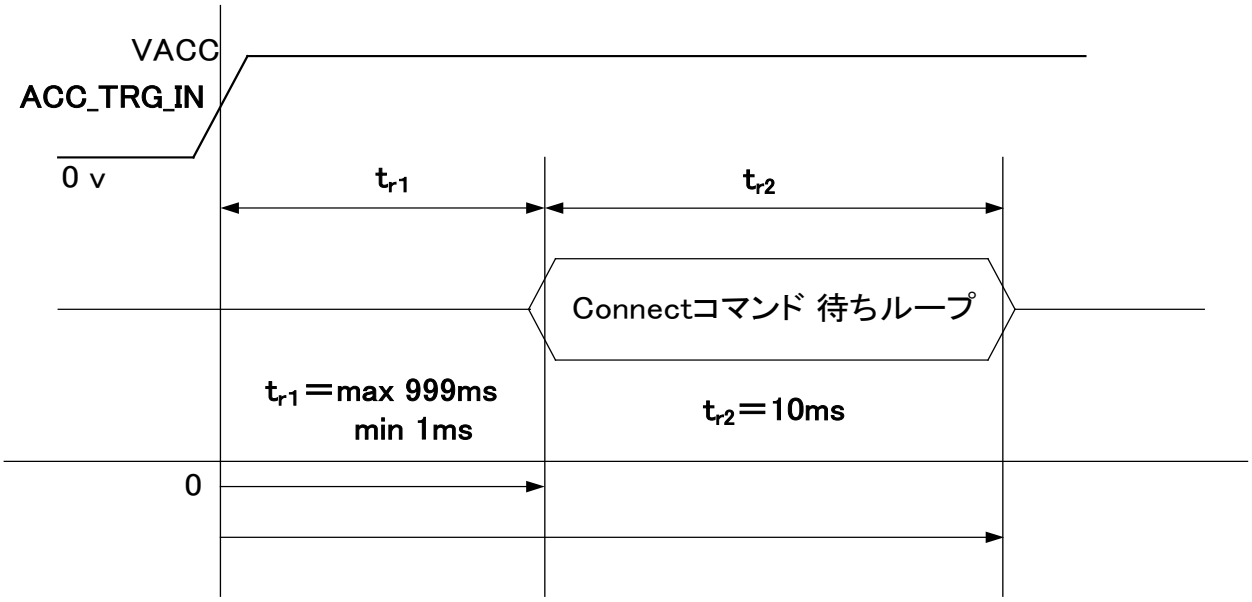
項目	内容	ユーザ設定
APL SUM 域 4Kbyte	12msec	不可
APL SUM 域 256byte	11msec	不可

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。



NET IMPRESS が VACC の Hi センス後の 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。

※ACC_TRG_IN 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駆動用電源(VACC)に ACC_TRG_IN 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

| |

KILL レジスタアドレス

CAN ボーレート

CAN コンフィグレーションレジスタ

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）
Primary メッセージ ID（#7EE、#7ED）

Station アドレス（CCP Ver2. 1 準拠）

CAN 通信チャネル番号

CANID(標準 ID or 拡張 ID)

I/O ポートサービス設定フラグ

内蔵 WDT 設定フラグ

＊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	#00002000h～#000023FFh	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1byte～4Kbyte とします。

９. RAMの使用方法

書き込み実行後 MCU 上の RAM 内容は保持されていません。

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	一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_init.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。 I/O ポートサービスを行います。 WDT のクリアを行います。
引数	なし
戻り値	なし

・ io_sv(WORD* sp) 関数

内容	I/O ポートサービス実行関数です。 I/O ポートサービスを行います。 WDT のクリアを行います。
引数	*sp 　　カウンタを指すポインタ
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の初期化を行います。 WDT の初期化を行います。
引数	なし
戻り値	なし

・ init_can(void) 関数

内容	CAN モジュールの初期化を行います。
引数	なし
戻り値	なし

・ id_get(DWORD* pNI,DWORD* pMcu) 関数

内容	CAN ID の取得を行います。
引数	* pNI 　　ライタ送信 CAN ID * pMcu 　　マイコン送信 CAN ID
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(= 2)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側に返します。
引数	なし
戻り値	なし

・ get_wcp(DWORD* 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 WCP 転送アドレス
戻り値	正常の場合 OK(= 0)、コマンド異常の場合 NG(= 2)を返します。

・ RecConnect(BYTE tout, WORD* piosdiv) 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 * piosvdiv I/O サービスカウンタ
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	r Entry または n Entry で UCOP リプログモードへ遷移する場合は戻り値「1」になります。 APL へ制御が移る場合は戻り値「0」になります。

・ 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ーモードを判定します。 ＝1：rEntry,nEntry ＝0：uEntry
戻り値	なし

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。 I/O ポートサービスを行います。 WDT のクリアを行います。
引数	なし
戻り値	なし

・ io_sv(WORD* sp) 関数

内容	I/O ポートサービス実行関数です。 I/O ポートサービスを行います。 WDT のクリアを行います。
引数	*sp 　　カウンタを指すポインタ
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。 無限ループに入ることで WDT によるリセットを行ないます。
引数	なし
戻り値	なし

・ line_read(DWORD addr) 関数

内容	ページリード処理関数です。 指定されたアドレスから 128byte データを IBL_RAMBUFF へ読み出します。
引数	addr 読み出し開始アドレスです。
戻り値	データが ALL の場合 0、それ以外は 1

・ 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。 受信データの書き込みを行います。
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。 flash_erase_block 関数を Call し、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

・ Command_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。 受信したチェック範囲に対し 1byte 単位で読出し 0xFF であるかチェックします。
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

・ Command_MessageID89write(void) 関数

内容	メッセージ ID ライトコマンド処理関数です。 メッセージ ID を書き込みます。
引数	なし
戻り値	なし

・ Command_KILLwrite(void) 関数

内容	KILL レジスタライトコマンド処理関数です。 KILL レジスタに書き込みます。
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 書き込みに対する初期化を行います。
引数	なし
戻り値	なし

・ flash_mode_rom_read(void) 関数

内容	FLASH のモードをリードモードにします。
引数	なし
戻り値	なし

・ flash_mode_pe(void) 関数

内容	FLASH のモードを P/E モードにします。
引数	なし
戻り値	なし

・ flash_write(DWORD buff,DWORD addr) 関数

内容	FLASH の書き込みを行います。
引数	buff 書き込みデータ格納バッファアドレス addr 書き込みアドレス
戻り値	正常終了時:0 エラー発生時:1

・ flash_erase_block(DWORD addr) 関数

内容	FLASH のブロック消去を行います。
引数	addr 書き込みアドレス
戻り値	正常終了時:0 エラー発生時:1

・ flash_lock_bit_check(void) 関数

内容	各ブロックのロックビット状態を取得します。
引数	なし
戻り値	正常終了時:0 エラー発生時:≠0

・ flash_lock_bit_read(DWORD addr,BYTE* mode) 関数

内容	指定ブロックのロックビット状態を読み出します。
引数	addr 読み出し対象ブロックのアドレス * mode ロックビット状態(=0:ロック状態、≠0:非ロック状態)
戻り値	正常終了時:0 エラー発生時:≠0

・ flash_lock_bit_set(void) 関数

内容	各ブロックに対しロックビットをセットします。
引数	なし
戻り値	正常終了時:0 エラー発生時:≠0

・ flash_lock_bit_on(DWORD addr) 関数

内容	指定ブロックに対しロックビットをセットします。
引数	addr 　　ロックビットをセットするブロックのアドレス
戻り値	正常終了時:0 エラー発生時:≠0

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	PA06	I/O サービスに使用	可
MCU 内蔵タイマ	SH72AY3 CMT チャネル 0	サイクルアクセス用。	
CAN 通信ポート	SH72AY3 CTx1 = 96 ピン CRx1 = 94 ピン	CAN 通信用の端子です。	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
KILL レジスタアドレス	KILL_ADDR	32bit アドレス指定	0x0001F00	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 ビットレートレジスタ	CAN_CFG_DATA	ビットタイミングレジスタ値	0x005C0003	可
入力クロック	CLK_EXT	MHz×10 で指定 例) 10MHZ = 100	200 (20MHz)	可
クロック逓倍比 f(PLL)	CLK_EXT_MULT	6:120MHz、8:160MHz	8	可
システムクロック f(PLL)の分周値	CLK_PLL_DIV	1:分周なし、2:2 分周、4:4 分周、8:8 分周、 16:16 分周	1	可
周辺機能クロックA f(SYS)の分周値	CLK_PBA_DIV	2:2 分周、4:4 分周	4	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	0x00002000	可
パスワードチェック領域 (エンド)	PASS_END	32bit アドレス指定	0x000023FF	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	0x00010000	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	0x00010FFF	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 周期	IOS_PERIOD	ms 単位(1ms～65535ms)	10	可
I/O ポートサービス用ポート 出力レジスタアドレス	IOS_PORT	32bit アドレス指定	0xFF464060	可
I/O ポートサービス用ポート 制御レジスタアドレス	IOS_CR	32bit アドレス指定	0xFF46404C	可
I/O ポートサービス用ポート ビット指定	IOS_BIT	16bit bit 指定	0x0040	可
内蔵 WDT サービス 対応フラグ	WDT_ON	0 or 1 0:WDT サービス無し 1:WDT サービス有り	0	可
内蔵 WDT サービス 周期	WDT_PERIOD	周辺機能クロック A の 16 分周 0x00=0.4ms、0x01=1.6ms、 0x02=3.3ms、0x03=6.6ms 周辺機能クロック A の 128 分周 0x80=3.3ms、0x81=13.1ms、 0x82=26.2ms、0x83=52.4ms	0x83	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)	0x000007ED	可
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)	0x000007EE	可
Station アドレス	CCP_STATION	16bit 値	0x0000	可
CAN チャネル番号	CAN_CHAN	チャネル 1	1	可

CANID フォーマット	CAN_ID	Standard=0 or Extended=1	0	可
ロックビット制御フラグ	LOCK_BIT_KEEP	ロックビットキープ：1 ロックビットクリア：0	0	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	SUB SETTING-> CAN AF->TGT ID FMT	“STANDARD”/“EXTENDED”を選択	STANDARD	可
CAN ID(マイコン→ライタ)フォーマット設定 *1	SUB SETTING-> CAN TGT->AF ID FMT	“STANDARD”/“EXTENDED”を選択	STANDARD	可
CAN 通信ボーレート設定 *1	SUB SETTING-> CAN BAUDRATE SETTING	500Kbps/1Mbps/250Kbps/125Kbps から選択	500K	可
CAN ID 設定 *1	SUB SETTING-> CAN ID SET	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:#7ED 拡張:# 00000 マイコン→ライタ 標準:#7EE 拡張:# 00000	可
KILL レジスタ ON	SUB SETTING-> KILL WRITE	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	SUB SETTING-> BLANK MODE	“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 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 で Specific Parameter #140 から 4BYTE 指定する	0x00010000	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 で Specific Parameter #144 から 4BYTE 指定する	0x00010FFF	可
r Entry コネクト発行タイミング(5 章 t1)	Specific Parameter #14C	AZ990 で Specific Parameter #14C から 2BYTE 指定する 1～999ms(1ms 単位)	- 要調整	可
Entry モード	Specific Parameter #14F	AZ990 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 で Specific Parameter #E5 から 1BYTE 指定する 0～2550ms（10ms 単位）	#14	可
CONNECT ビジー(36h)応答時リトライ回数	Specific Parameter #E6	AZ990 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。