XBusServo
ライブラリ解説

Ver. 1.1.2

2016 / 2 / 17

JR PROPO 技術開発部

●はじめに

JR PROPO 製 XBus サーボモータ（以下、XBus サーボ）にご興味をお持ちいただき、ありがとうございます。XBusServo ライブラリ解説（以下、本ドキュメント）は、XBus サーボをマイコンボード「Arduino」（以下、Arduino）からお使いいただく時のための XBusServo ライブラリ（以下本ライブラリ）について解説しています。

XBus サーボが採用している 1 線半二重の通信を行うのに必要な回路図についても、簡単に解説しておりますので、参考になさって頂ければ幸いです。

以下の章に記載しました仕様は、まだまだ拡張可能な要素を残しております。もし何かご要望等がございましたら、後述の専用窓口までご相談いただければと思います。

Arduino に XBus サーボを接続して、様々な作品が生み出されていくことを願っております。

●ドキュメント履歴

リリース日	バージョン	作業者	リリース内容
2014/09/29	1.0.0	澤	初版
2014/11/06	1.0.1	澤	プルアップに関する記述を追加
2014/12/12	1.1.0	澤	サンプルコード、及びその解説追加
2015/9/28	1.1.1	澤	新機種追加
2016/2/17	1.1.2	澤	タイムアウトエラー追加、新機種追加
（以下余白）			

●注意事項

- 本ドキュメントは、XBus サーボを Arduino に接続して使うためのライブラリに関する仕様概略です。このドキュメントに掲載されていない部分については、別ドキュメントの「XBus サーボ仕様概略」をご参照ください。

- 本ドキュメントは Arduino に対応するものではありますが、全ての Arduino での動作を保障するものではありません。現時点においては、Arduino UNO での動作を前提に記述されています。互換機等、他の環境での動作については確認しておりませんので、ご注意ください。

- 本ドキュメントや本ライブラリは、製品改良等及びドキュメント改良等のため予告なく変更されることがあります。

- 本ドキュメントや本ライブラリは、状況に応じて非公開になる場合があります。その際、下記お問い合わせ窓口も閉鎖になる場合があります。

- 本ドキュメントや本ライブラリの無断転載は、最新情報更新の妨げになる場合が多いですのでお控えください。但し、本ドキュメントを基にした解説等の公開や、その際に必要な正しい引用については無論問題ございません。むしろ、歓迎いたします。もしよろしければ、その際は参照用として弊社 web ページへのリンク等を掲示していただければ幸いです。

- 本ドキュメントや本ライブラリに関連するトラブル等については、弊社は一切の責任を負わないものとします。なお、弊社製品について修理等が必要な場合には、弊社サービス部宛に通常の修理品としてご依頼頂ければ随時対応いたします。

- 本ドキュメントや本ライブラリに関するお問い合わせは、下記の専用メールアドレス宛のみとします。なお弊社サービス部は、本ドキュメントや本ライブラリに関しては一切お答えできません。常に XBus サーボ担当者が応対にあたります。また、担当者不在等により頂いたメールの返答には２週間程度かかる場合があり、全てのお問い合わせにお答えできるとは限らないものとします。

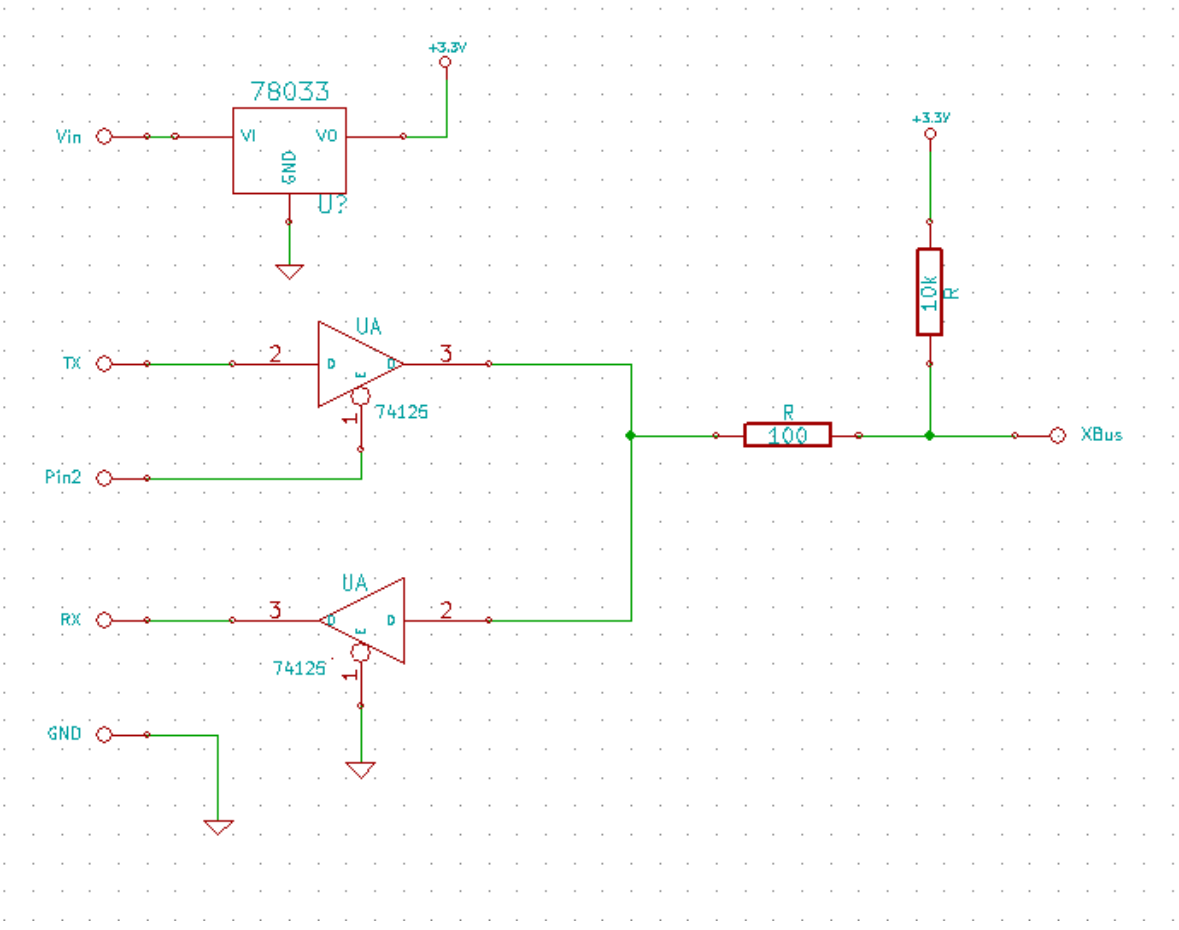
  専用メールアドレス　　xbusservo@jrpropo.co.jp

●ハードウエアについて

・通常仕様

XBusサーボをArduinoに接続する場合、Arduinoの信号線は基本的に５Ｖでありかつ通常の2線双方向シリアル信号ですので、これを3.3Vの1線双方向シリアル信号に変換する必要があります。

以下の回路はその概略です。



やっていることとしては、

- TXに3ステートバッファを入れ、受信時にTXのラインをカットする。
- 3.3V駆動のバッファ（上の例では74125と表記）を使い、レベル変換を行う。
- 保護抵抗を入れ、回路を保護する。

といった辺りです。回路図中の74125は、実際には最近販売されているタイプで構いませんが、信号レベルがTTLであること、電源電圧が3.3Vであること、入出力ともに5Vトレラントであることが条件です。

出力側が5Vトレラントでない場合、Arduino基板のRxのプルアップがRxバッファの出力側から回り込み、3.3V電源を上昇させてしまいますのでご注意ください。その場合は、Rxバッファのみ5V電源にすれば解決する場合がありますが、入力レベルに注意が必要です。

Pin2が回路図上指定されていますが、これは固定ではなく任意のデジタルピン、アナログピンを設定できます。ライブラリの初期化時に、何番のピンを使うのかを指定しています。

10KΩでプルアップしていますが、これは受信に切り替えた時になんらかの理由で一時的にバスのレベルが下がるのを防止するために入れています。XBusサーボからの情報を受信する際、一時的に双方が受信する状態になる場合があり、このときバスのレベルが下がってしまうと、信号が開始されたと勘違いをする場合があるためです。

なお、このプルアップの値は100KΩまでの間で調整するのが望ましいです。特に複数のXBusサーボを設置する場合は、もう少し大きな値にしておくことをお勧めします。

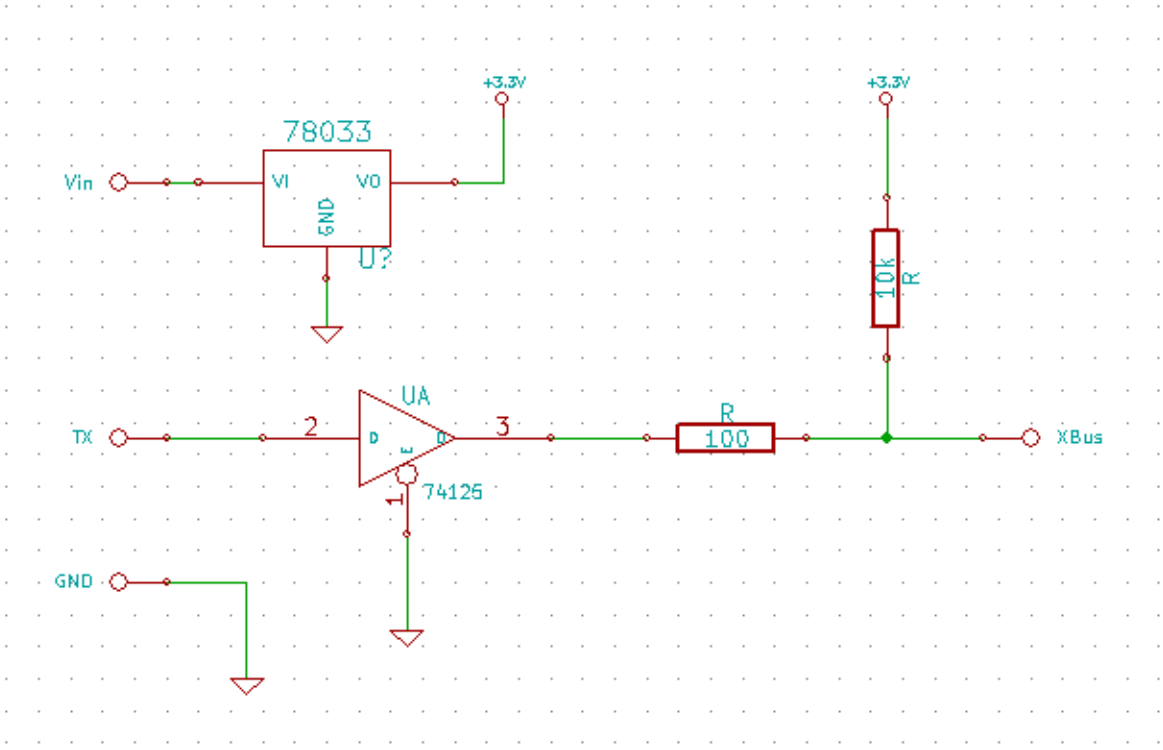
・TXOnly 仕様

XBusサーボからの情報を一切受け取らない場合、TXのみの接続で動かすことができます。使用部品は通常仕様とあまり変わりませんが、RX側の回路や切り替え信号が不要になります。

ただし、この場合以下の制限があります。

- 一切XBusサーボの情報を受け取れない。
- ID設定を含む各種設定は、その都度個別に当該XBusサーボのみを接続して行う。

気軽にXBusサーボを使う場合は、こちらの方が便利かもしれません。



通常仕様に合わせて10KΩでプルアップしていますが、こちらはあってもなくても特に大きな問題はありません。

なお、このプルアップを設置する場合は100KΩまでの間で調整するのが望ましいです。特に複数のXBusサーボを設置する場合は、もう少し大きな値にしておくことをお勧めします。

・3.3V 仕様の Arduino について

一部の Arduino（及び互換機）には、もともと信号が 3.3V で出力されているものがあります。この場合、レベル変換は不要かと思いますが、通常仕様の場合は TX を遮断するバッファが必要になります。また、マイコンボード自体の出力が弱い（出力インピーダンスが高い）場合、最悪は信号が鈍る可能性もありますので、安全のためにも 5V 仕様と同様、バッファを入れておくことをお勧めします。

●ソフトウエアについて

・概略１：初期化について

本ライブラリは、XBus サーボのパケットプロトコルを内包しています。ですので、Arduino ユーザーは API を呼ぶだけで、細かい初期設定抜きに XBus サーボ相手にコマンドを送受信できるようになっています。内部的には CRC 計算等の面倒な処理もありますが、本ライブラリを使えばそういったことに煩わされる心配はありません。

メモリの少ない Arduino での使用を踏まえ、本ライブラリ初期化時に使用する XBus サーボの最大数を設定することで、必要最小限のメモリをバッファとして割り当てるようにしています。XBus サーボの仕様上の最大数である 50 個分を確保すると、500 バイト強のメモリが割り当てられてしまいますので、ご注意ください。

本ライブラリ初期化時には、同時に TX を制御する信号線を割り当てます。前述の回路図では Pin2 に割り当てていますが、デジタル信号が出る端子であれば、どこでも構いません。また、この割り当て引数にマイナス 1 を与えると、TXOnly 仕様として動作します。

割り当てたバッファには、使用する XBus サーボを登録していきます。もちろん、登録できる最大数は、本ライブラリ初期化時に指定した数までです。登録するときは、登録する XBus サーボの初期位置とチャンネル ID を任意に設定できます。

登録するチャンネル ID は一連でなくても構いません。ですので、たとえば「0x05, 0x02, 0x07」の三つを使うといった設定も可能です。使用途中での追加登録、削除も問題ありませんが、その際は位置情報の送信メソッド呼び出しとタイミングが被らないようにだけ注意してください。被ってもトラブルにはなりませんが、登録、削除中はデータが送信されません。

なお、登録する必要があるのは位置の指示を行う場合だけです。ですので、たとえばチャンネル ID 設定機を製作する等設定を行うだけの場合は、登録を行う必要はありません。その場合、XBus サーボの最大数にも 0 を指定して構いません。

・概略２：操作について

通常、XBus サーボは位置情報を受け続けないと脱力するように設定されています。そのため、Arduino でどんな操作が行われていても位置を維持するために、タイマを用いて位置情報送信のメソッドを常に呼ぶようにしてください。サンプルコードでは、そのやり方も解説しています。

XBus サーボのストップモードをオンにすると、最後に受け付けた位置情報の位置で維持されます。脱力させることが不要で、特定のタイミングでのみ動作させたい場合は、事前に各 XBus サーボのストップモードをオンにしてください。必要なタイミングで位置情報送信のメソッドを呼べば、送信したタイミングでその通りに動作します。初心者には、こちらの方が使いやすいかもしれません。

位置の指示は16bitで行います。この値は、従来のPWM信号の概念を踏襲しており、以下のような値で指示します。XBusサーボは概ね850-2150uSecの範囲で動作するようになっています。それ以外の範囲での角度指定については、脱力します。

0x0000	800uSec	
0x1249	900uSec	（-60度、または-90度）
0x7FFF	1500uSec	（0度）
0xEDB6	2100uSec	（+60度、または+90度）
0xFFFF	2200uSec	

ただし、後述のリミット値が設定されている場合、その値よりも外側の指示についてはリミット値でクリップされるため、設定によっては脱力せず、リミット位置で停止します。現時点では、リミット値が工場出荷時に 800－2200 で設定されているため、リミット値は無効になっています。

XBus サーボの最大角は通常 120 度ですが、設定により 180 度まで広げることができます。用途に応じ、設定のコマンドを送信して切り替えて使うのがお勧めです。この場合、位置の指示は同じく 900-2100 で 180 度動作するようになっています。

・コンストラクタ

```
XBusServo      myXBusServo(kDirPinNum, kMaxServoNum);
```

kDirPinNum	通常仕様で接続する場合の、TX 切り替え端子を指し示します。前述の回路図では、2 になります。この値にマイナス 1 を指定すると、TXOnly モードとして動作します。シールドの回路に合わせて選択してください。
kMaxServoNum	接続する XBus サーボの最大数を指示します。

XBusサーボを使用する場合、最初に必ずこの宣言が必要になります。もちろん、名称は**myXBusServo**でなくても構いません。

注意：もしサブIDを使う場合、同一サーボID分は単一のXBusサーボとしてカウントしてください。サブIDに関しての詳しい情報は、「XBusサーボ仕様概略」をご覧ください。

・使用開始・終了関係のメソッド

```
void    begin(void);
```

パラメータ	なし
戻り値	なし

XBusサーボを使用する場合、最初に必ずこのメソッドを呼んでください。これ以降、シリアルポートは本ライブラリが占有します。通常はsetupメソッドの内部で呼びます。

```
void    end(void);
```

パラメータ	なし
戻り値	なし

XBusサーボの使用を終了する場合、最後に必ずこのメソッドを呼んでください。これ以降、シリアルポートは開放されます。

・XBus サーボ登録・角度指示関係のメソッド

```
XBusError     addServo(char channelID, unsigned int initValue);
```

channelID	登録する XBus サーボのチャンネル ID
initValue	登録する XBus サーボの初期角度。
戻り値	エラーコード

XBusサーボを登録します。kMaxServoNumの個数分呼ぶことができます。

```
XBusError     removeServo(char channelID);
```

channelID	削除する XBus サーボのチャンネル ID
戻り値	エラーコード

登録されたXBusサーボを削除します。

```
XBusError     setServo(char channelID, unsigned int value);
```

channelID	位置指示する XBus サーボのチャンネル ID
value	位置指示値
戻り値	エラーコード

XBusサーボへ位置指示をします。ただし、この段階ではまだXBusサーボには送信されていません。本ライブラリ内部のバッファに蓄積されただけです。

```
void     sendChannelDataPacket(void);
```

パラメータ	なし
戻り値	なし

XBusサーボへ位置指示のためのデータを送信します。この時点で、登録されている全てのXBusサーボへの位置指示が一括して行われます。

・通常仕様時のコマンド関係のメソッド

`XBusError  setChannelID (char oldChannelID, char newChannelID);`

`oldChannelID`	チャンネル ID を変更したい XBus サーボのチャンネル ID
`newChannelID`	新しいチャンネル ID
戻り値	エラーコード

XBusサーボのチャンネルIDを変更します。`oldChannelID`が設定されているXBusサーボのチャンネルIDを`newChannelID`に変更します。`oldChannelID`にゼロを指定した場合、接続されている全てのXBusサーボが`newChannelID`で指示される同一のチャンネルIDに設定されます。

`XBusError  setCommand(char channelID, char order, int* value);`

`channelID`	設定する XBus サーボのチャンネル ID
`order`	設定項目
`value`	指示内容(指示後は設定値が戻ります)
戻り値	エラーコード

XBusサーボの設定を変更します。設定項目や設定値の解説は、下記の表ををご覧ください。

`XBusError  getCommand(char channelID, char order, int* value);`

`channelID`	取得する XBus サーボのチャンネル ID
`order`	設定項目
`value`	設定値
戻り値	エラーコード

XBusサーボの設定値を取得します。設定項目や取得値の解説は、下記の表ををご覧ください。

・TXOnly 仕様時のコマンド関係のメソッド

`XBusError  setChannelID (char newChannelID);`

`newChannelID`	新しいチャンネル ID
戻り値	エラーコード

現在接続されている全てのXBusサーボのチャンネルIDを、`newChannelID`に変更します。

`XBusError  setCommand(char order, int* value);`

`order`	設定項目
`value`	指示内容(指示後は設定値が戻ります)
戻り値	エラーコード

現在接続されている全てのXBusサーボの設定を変更します。設定項目や設定値の解説は、下記の表をご覧ください。

・設定項目、設定値解説

Order	名前	意味	初期値	備考
`0x04`	`Version`	ファームウエアバージョン	機種毎	get only
`0x05`	`Product`	機種番号（別表 2 参照）	機種毎	get only
`0x07`	`Parameter Reset`	下記別表 1 の Index で指定したパラメータを初期値に戻す。	`------`	set only
`0x08`	`Parameter Write`	下記別表 1 の Index で指定したパラメータを ROM 領域へ書き込む。	`------`	set only
`0x10`	`Reverse`	`0x0000` 通常動作 `0x0001` 左右反転動作	`0x0000`	
`0x11`	`Neutral`	ニュートラル位置（1500uSec）に対するオフセットを指定する。±600 の範囲で指定できる。主に複数 XBus サーボの同期連動時補正に使用する。	`0`	

0x12	Travel High	ニュートラル位置（1500uSec）よりも上の領域において、角度指示を拡大する。通常 128、最大 192 まで指定できる。主に複数 XBus サーボの同期連動時補正に使用する。	128	
0x13	Travel Low	ニュートラル位置（1500uSec）よりも下の領域において、角度指示を拡大する。通常 128、最大 192 まで指定できる。主に複数 XBus サーボの同期連動時補正に使用する。	128	
0x14	Limit High	角度指示できる最大値。Limit Low 未満には設定できない。無理に設定しても、自動的に Limit Low に修正される。	0xFFFF	
0x15	Limit Low	角度指示できる最小値。Limit High より大きくは設定できない。無理に設定しても、自動的に Limit High に修正される。	0x0000	
0x16	P Gain	XBus サーボが持つ P ゲインに対する増減値。±50 の範囲で指定できる。	0	
0x17	I Gain	XBus サーボが持つ I ゲインに対する増減値。±50 の範囲で指定できる。	0	
0x18	D Gain	XBus サーボが持つ D ゲインに対する増減値。±50 の範囲で指定できる。	0	
0x19	Dead Band	XBus サーボが持つデッドバンドに対する増減値。±10 の範囲で指定できる。	0	
0x1A	Boost	XBus サーボが持つブースト値に対する増減値。±999 の範囲で指定できる。	0	
0x1B	Alarm Level	XBus サーボからアラームが発せられるパワー閾値。0-100%で指定できる。アラームが発せられると、モータ音が変化する。	機種毎	
0x1C	Alarm Delay	XBus サーボからアラームが発せられるまでの遅延時間を指定できる。この時間が経過しない範囲で閾値から下がれば、警告されない。0-5000mSec まで指定できる。	機種毎	
0x1D	Angle	0x00 通常動作（最大角 120 度） 0x01 最大角 180 度（一部の機種では、180 度に到達しない場合がある）	機種毎	

0x1E	Slow Start	0x00 起動後、すぐに通常動作する。 0x01 起動時、最初に取り込んだ指示位置までゆっくり移動する。ただし、移動中に指示位置が変化した場合、その時点から通常動作に戻る。	機種毎	
0x1F	Stop Mode	0x00 通常動作（角度指示が途絶えると 1 秒前後で脱力する） 0x01 ホールド動作（角度指示が途絶えると直前の位置を維持する）	0x00	
0x20	Current Position	現在の出力軸の位置を返す。静止していても、指示位置と一致しているとは限らないので注意すること。	------	get only
0x21	Current Power	現在モータへ掛けているパワーを返す。 0-100%の値を示す。	------	get only
0x22	Speed Limit	0 通常動作 1-30 速度制限モード（1 が最も遅い）	機種毎	
0x23	Max Integer	I ゲインの積分値リミッタに対する増減値。±999 の範囲で指定できる。	0	

別表 1　Parameter Reset、Parameter WriteにおけるIndex

Index	名前	適用	備考
0x0001	All Data with ID	reset only	CH-ID を 0x00 にする必要がある。ROM 領域へ自動で記録される。リセットに成功すると、チャンネル ID は 0x01 になる。
0x0002	All Data without ID	reset only	
0x0003	Servo ID	reset only	CH-ID を 0x00 にする必要がある。ROM 領域へ自動で記録される。リセットに成功すると、チャンネル ID は 0x01 になる。
0x0004	Reverse	both	
0x0005	Neutral	both	
0x0006	Travel High	both	
0x0007	Travel Low	both	
0x0008	Limit High	both	
0x0009	Limit Low	both	
0x000A	P gain	both	
0x000B	I gain	both	

0x000C	D Gain	both	
0x000D	Dead Band	both	
0x000E	Boost	both	
0x000F	Alarm Level	both	
0x0010	Alarm Delay	both	
0x0011	Angle	both	
0x0012	Slow Start	both	
0x0013	Stop Mode	both	
0x0014	Speed Limit	both	
0x0015	Max Integer	both	

別表２　Productにおける応答値／機種名一覧

応答値	機種名
0x0200	NX8921
0x0201	NX3421
0x0202	NX588
0x0203	NX8925
0x0204	NX3425
0x0205	NX6421
0x0206	NXR89
0x0207	NXR34
0x0208	NXB8921
0x0209	NXB8925
0x020A	NXB89G
0x020B	NX8921mk2

・エラーコード解説

kXBusError_NoError	エラーはありません。
kXBusError_CRCError	CRC に問題があります。多くの場合、これは受信エラーです。
kXBusError_ServoNumOverflow	これ以上、XBux サーボを登録できません。
kXBusError_ServoNumIsZero	これ以上、XBus サーボを削除できません。

kXBusError_AddWithSameID	新たに登録しようとしたのと同じチャンネル ID の XBus サーボが既に登録されています。
kXBusError_IDNotFound	そのチャンネル ID は見つかりません。
kXBusError_Unsupported	その指示はサポートされていません。
kXBusError_OnlyForTxOnlyMode	そのコマンドは TxOnly モードでしか使えません。
kXBusError_OnlyForNormalMode	そのコマンドは通常仕様でしか使えません。
kXBusError_MemoryFull	メモリが不足しています。
kXBusError_TimeOut	データ読み出し中にタイムアウトしました。

●スペシャルサンプルコード“XBusServoChecker”について

・概略

本サンプルコードは、XBus サーボのパラメータを変更できる設定器を Arduino を用いて実装する場合のサンプルです。なお、このコードは実際に業務で使用しているものをそのまま添付しておりますが、作業用ですので、あまり細かい部分までは配慮がなされていない事をご了承ください。

・ハードウエア構成

ざっとした回路図を後に記載しました。基本的に以下の部品を実装しています。コードに合わせて実装するか、自身の思う配置で実装してコードを修正するかはお任せします。なお、全ての部品が秋葉原の電子部品店等で購入可能なものになっていますので、入手は容易かと思います。参考までに、秋月電子通商さんでの番号も付記します。回路図中には描いていませんが、もちろん双方向用のバッファの電源はどちらも 3.3V にしてください。

- 前述の XBus 双方向通信用バッファ回路（I-06481、P-04800 等）
- バッファ回路用 3.3V 電源回路（I-00538、P-06165、P-05002 等）
- タクトスイッチ４つ（P-03646、P-03649、P-03651、P-03650 等）
- ロータリーエンコーダ１つ（P-06357、P-07240、P-00997 等）

- I2C 接続小型 LCD 1 つ（P-06669、P-06794、K-06795 等）
- 基板、配線関係（P-07555、C-00167、C-04397 等）

・ソフトウエア構成

基本的な使い方としては、上下左右のボタンでカーソルの位置を移動させ、その項目に対してロータリーエンコーダーを回して変化させるという形になっています。

なお、同時にリンクする LCD のライブラリによっては、カーソル移動の指示を出してもカーソルが表示されない場合がありますので、適宜ライブラリの変更をお願いします。

・注意

Arduino 側の 3.3V 電源は容量が小さいのか、バッファ回路を接続するとうまく動作しないことがありましたので、別途電源を確保しています。

サーボ用の電源を一旦シールド基板へ接続し、Arduino の Vin へ供給することで Arduino 基板を動作させる方式で実装しています。本来はイリーガルだったかと思いますが、現状 UNO では問題は発生していません。気になる方は、別途電源を直接 Arduino 側へ供給すればよろしいかと思います。なお、Arduino 側電源からサーボへの電源を確保すると、最悪の場合基板が焼き切れるかもしれませんので、ご注意ください。