PICを使ったラジコン開発

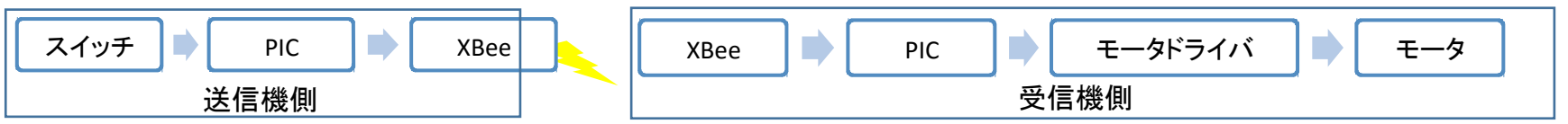
Yanorei32(小学5年時の資料そのまま/一部デザイン崩れ)

動機

僕は電気が好きだ。始まりは、5歳の時に作った、ラジオキット（AM）のはんだ付けだった。毎月第3金曜日の市の小型家電収集日は、僕にとって夢の日だ。近所の収集場所から、小型家電製品を拾い集めて分解して、部品取り。そして部品1つ1つレベルまで分解され、僕の手により、違う機械となり日々「Recycle」されている。たとえば、非安定マルチバイブレータ（LEDの点滅用の回路）を、チャリの後ろの点滅ランプに使った。また、モータに重りをつけて（自作）振動させ、歯ブラシの先につけ、床を磨いた。これは、お母さんに受けがよかった。僕が作ったもの（年代順）：ラジオ（AM）[キット]、ムーンウォーカー[キット]、ブザー[キット]、非安定マルチバイブレータ[自作]、机から落ちないロボ[キット]、ビュートレーサー[キット]、FMワイヤーレスマイク[自作]、スタンガン1号[自作]、－改造→、コイルガン1号[自作]、スタンガン2号（コッククロフトウォルトン回路[3年 理科展出品]搭載）、[自作]コイルガン2号[4年 理科展出品]、PICを使ったLED点滅[自作]、オーディオアンプ[自作]、電子オルゴール [キット]、スタンガン3号機[自作]。
パソコン歴。5歳の時からパソコンを始め、7年目になった。普段はニコ動で電子工作の動画（PICタグ、技術部タグ、Arduinoタグ、電子工作タグ）を巡回したり、電子工作をやっている人のブログを閲覧している。また、4年生よりBschVer3（回路図エディタ）とそのライブラリーを使い、回路図を描き始める。*理解し辛い為Bschv3というアプリケーション名をBschVer3に書き換えた。**師匠は、僕のお父さん。**お父さんは、会社で主に携帯電話などの高周波無線(2.5Ghz帯)を研究している。僕にとってお父さんは、知識と技術、そして色々な機材を持っている、大切な師匠である。僕の研究の時にいつも、頼りになって、わからないところをサポートしてくれる。そして今回、僕のお父さんや、ネットで知り合った小型の無線モジュールを扱っている人たちの影響により、**『PICを使ったラジコン開発』**（無線のRadio control開発）をやろうと思った！

原理

送信側：PICがスイッチがONかOFFを確かめ、それに合った信号をRS-232Cの方式でXBeeに送る。Xbeeは、その信号を電波にして飛ばす。
受信側：Xbeeからの信号で、PICがモータドライバにそれに合ったデジタル信号を送る。モータドライバは、そのデジタル信号に合わせてモータを動かす。
モーターは2個で左右の車輪の回転を変えることにより方向転換を行う事にする。



（*~~マイコン~~無線）
XBeeは、シリアル通信で信号（データ）を無線で飛ばしたり、受けっとったり出来るものである。
PICマイコンは、シリアル通信や1と0の出力や、その他のICや回路を制御するためのICである。

設計と制作

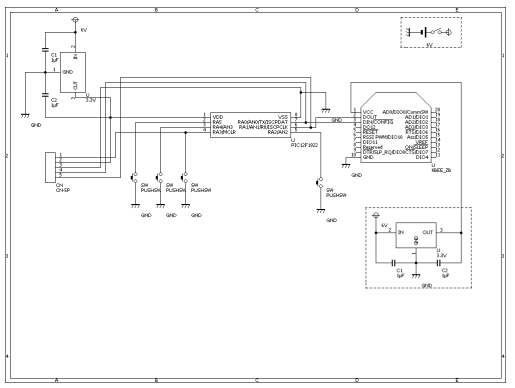
送信機側

4.5Vバッテリーだが XBeeが3.3V駆動の為、PICマイコンも3端子レギュレータを用い、3.3Vで駆動させた。前進・後退・右旋回・左旋回がそれぞれ必要なため、スイッチを4つ付ける必要があり、更にXBeeとの通信の為に、USARTが入っている物で無いといけないのでどちらも内蔵されているPIC12F1822を使った。スイッチは本来プルアップ抵抗（スイッチがOFFの時に信号を1にする為の抵抗器）が必要なのだが、それがPIC12F1822に、内蔵されているため内蔵のものを使った。プログラムの書き換えを楽にする為、L字型ピンヘッダ（6PIN）をつけた。プログラムは、前進スイッチが押された時はF（frontの頭文字）を後退スイッチが押された時はB（backの頭文字）を左旋回スイッチが押された時はL（leftの頭文字）を右旋回スイッチが押された時にはR（rightの頭文字）を送信することにする。

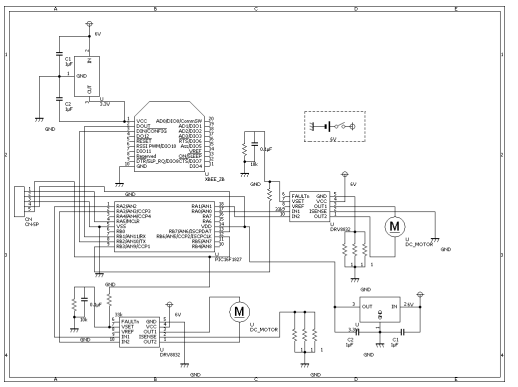
受信機側

モーターを駆動させる為に電源に余裕を持たせ単三電池4本の6Vにした。6Vバッテリーだが XBeeが3.3V駆動の為、今回もPICマイコンを3.3Vで駆動させた。ただ、PICマイコンから直接モータに繋げるなどということを誤ってしてしまうと、モータが遅いと言う事と電流の流し過ぎで最悪PICマイコン自体が壊れてしまうので対策が必要である。対策の案としてはトランジスターやFETやモータドライバーがある。トランジスターやFETは、回路が複雑になりモータを逆転するための回路安全回路（モータに何らかの異常が発生した場合緊急停止するなどの回路）を考え・導入する必要があり、そうすると回路や配線が複雑になり、もし配線ミスなどで動かなかったときにどこが原因かを調べるのにかなり時間がかかってしまうなどというデメリットが多いので、今回はモータドライバーを使うことにする。モータドライバーなら逆転はおろか、安全回路も搭載されているので安心だ。それに回路も単純化でき、配線ミスの場合など、原因がそれなりにすぐにわかる。送信機側と同じでXBeeとの通信の為に、USARTが入っている物で無いといけないと言う事、プログラムの書き換えを楽にする為、L字型ピンヘッダ（6PIN）をつける事は変わらないが、今回はI/Oポートをかなり使うため8ピンでは足りない。更に、将来的に可変抵抗でスピード調整を出来るようにするため、PWMが2ch必要である。そこで18ピンの16F1827を使った。
送信機側、受信機側の回路図を下に示す。 プログラムは付録にある。

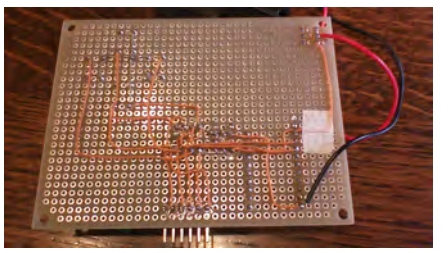
送信機側の回路図

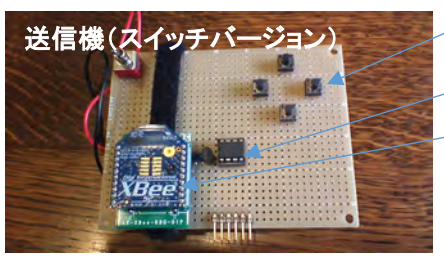


受信機側の回路図

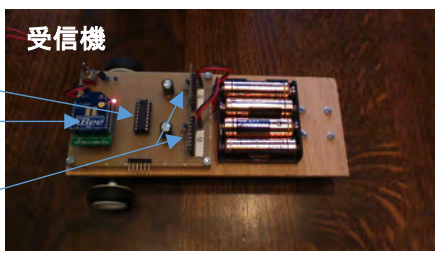


基板裏（送信機側）

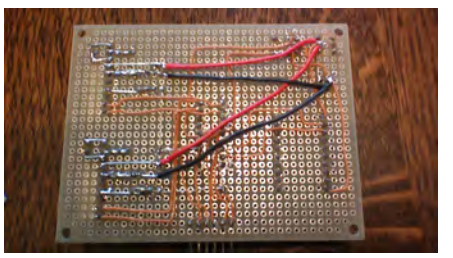




操作スイッチ
PICマイコン
XBee
モータドライバ

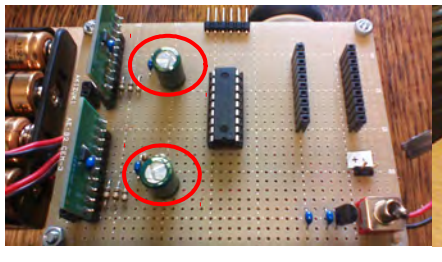


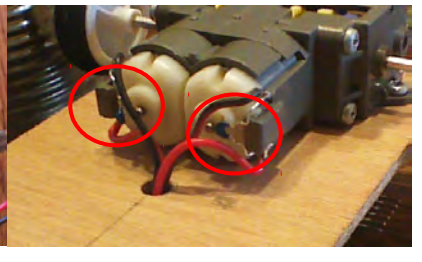
基板裏（受信機側）



実験結果

実際に起動して動かしてみたら、何故かモーターがものすごくゆっくりとしか回らなかった。モーターの電圧をテスターで測ってみたら、本来3Vあるはずの電圧が1V以下程度しかなかった。モーターの電圧を設定するモータードライバのVset端子の電圧は正常で0.75Vだった。モーターの正転・逆転を制御するIN1が3.3V・IN2が0Vとどちらも正常だった。それをすべて確認したけれども何故かうまく動作しなかった。配線ミス、電池残量低下、半田不良などもチェックしたが問題はなかった。モーターからのノイズを疑って、モーターにノイズ防止用のコンデンサー(0.1μF)をつけたところ、症状がだいぶましになった。更に原因を解明するべく色々と試した結果、モータドライバはとてもノイズに弱いと言う事がわかった。その為、電源ライン(＋6V・GND間)に1000μFを二つ入れることにした。更にモータからのノイズ対策のためにモーターの間に0.1μFを両方とも入れた。その対策をとったら、ちゃんと考えていた通りに動くようになった。





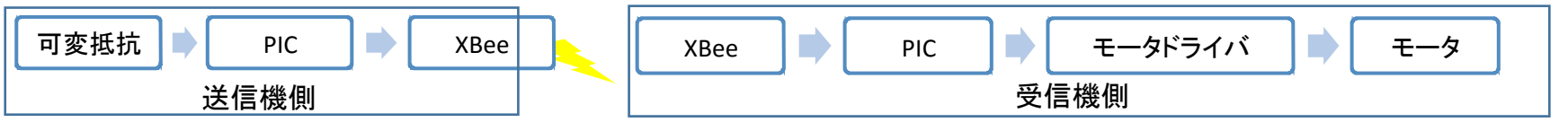
※丸印はノイズ対策用に追加したコンデンサ

操作の改良点

実際に操作してみて改めて思ったのだが、全速と停止しかないと、前後の移動はまだ良いが、方向転換の時に操作をし辛かった。モータドライバの駆動電圧制御ピンはPICマイコンのPWMの出力PIN1と2にあらかじめ繋いでおいたので、PWMで制御できるように改良する。PWM制御をするには「F」「B」「L」「R」だけでは制御できないので数値も送る必要もありそうだ。そうして送信機側からの可変抵抗の電圧数値を読み取り、PWMで駆動電圧制御ピンを制御することにした。

改良後の原理

送信側:PICが可変抵抗の電圧を確かめ、AD変換しそれに合った値をRS-232Cの方式でXBeeに送る。Xbeeは、その信号を電波にして飛ばす。
受信側:Xbeeからの値で、PICがモータドライバにそれに合ったPWM信号を送る。モータドライバは、そのPWMに合わせてモータを動かす。



(＊マイコン)

設計と制作

送信機側

3.3Vで動作するなどの基本的なところは、同じだ。速度調整の為、可変抵抗を使う。可変抵抗は左側は、左右、右側は前進・後退を行う。可変抵抗からの電圧をA/D変換して値を取り込む。片方をxの値もう片方をyの値として、16進数に変換して送信してx00y00やxFFyFFなどで文字として送る。16進数に変換するときに普通にif文を64個並べてやったらプログラムメモリに入りきらなくなったので、関数を使ってxの値の変換とyの値の変換を共通化した。

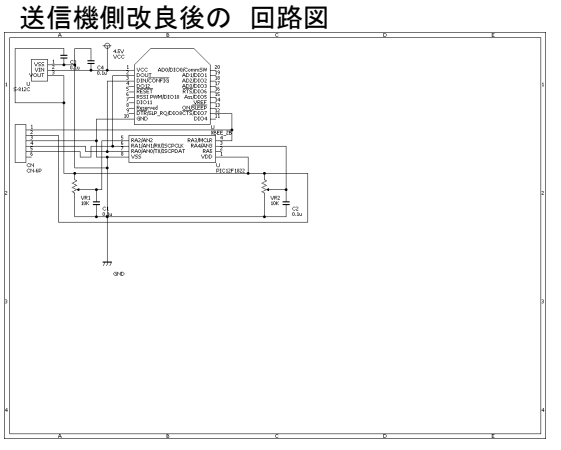
受信機側

送信機側から送られてきた16進数の文字をもとのxの数値、yの数値に変換しないといけない(これは文字として送られてきたデータを数字に戻すためだ)。変換には、送信側と同じように関数を使いxとyを共通化した。変換されたxとyの値を用いて、yは速度、xは左右のバランスを調整する。調整にはPWMを使い、下の表のように計算する。

y	0～127	128	129～255
速度(z)	255-y*2	0	y*2-255
進行方向	後	停止	前

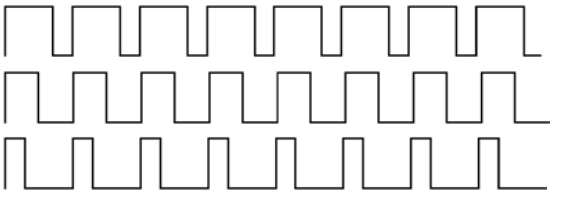
x	0～127	128	129～255
左	z*x/128	z	z
右	z	z	z*(256-x)/128

送信機側改良後の　回路図

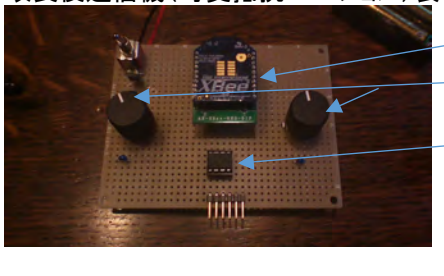


PWMとは

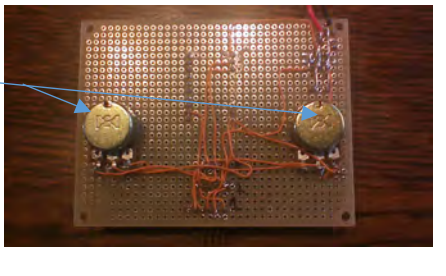
PWMは一般的にデジタル出力(ONとOFFだけの出力)だけで、あたかもアナログ出力(ONとOFFの間に半端な電圧のある出力)しているように見せかける方法である。今回使ったのは、OFFになるまでの時間をコントロールできる方式だ。ONになる周期を設定してOFFになるまでの時間(正確にはタイマーが数えている数がある一定の数AになるとOFFにしてさらにある一定の数Bになるとそのカウントがリセットされることの繰り返しの中のある一定の数A)を変更してモーターの動作する速度・バランスを変更してモータを制御している。右の図でいうと上がモーターが早いもの下がモーターの遅いものとなっている。



改良後送信機(可変抵抗バージョン)表



改良後送信機(可変抵抗バージョン)裏



実験結果

事故発生！

新しい装置での実験開始直後、電源を入れてもXbeeのLEDが付かなかった。気が付いたら電源を逆接してしまっていた。それにより高価でデリケートなXbeeが壊れてしまった。その為通信テストが出来なくなってしまった。

Xbeeの代用品

Xbee無線モジュールの代用品として、送信機と受信機間に直接1mくらいのエナメル線を2本配線し、有線で直接受信させて簡易的な動作確認を行った。その結果、前進・後退・右旋回・左旋回、思い通りに動いた。あとは、Xbee無線モジュールでの実験をするのみだ。

問題発生

Xbeeを再注文してそのXbeeを装置に繋げてみた。しかし通信が全く出来なかった。Xbeeのペアリング設定などを見直したが、すべて正常だった。注文したばかりのXbeeが壊れている訳がないのでXbeeの設定画面を見まわした。そうしてしばらくたって気が付いたのがファームウェアの設定だった。ファームウェアが送信機側と受信機側でバージョンがずれていたことにより送受信ができないという事が生じたようだ。Xbeeのファームウェアを同じバージョンにした。コンピュータのUSBシリアル通信ポートを使い、送信機側対コンピュータで通信テストをした所、正常動作した。

無線での試験、そして問題発生

コンピュータからXbeeを取り外し、受信機側に付け直し無線での動作試験を始めた。

だが、信号の送受信に問題が発生した。しばらく通信していると文字列の送信が止まり、一つ前の状態のまま次の文字列を受信せずにそのまま走って行ってしまう事が起きた。

対策

その対策として、データを送る間隔を10ms開けることにした。何故そのような対策をとったかというと、スイッチでの操作の時も実はそのような事が起きていて、信号の送り過ぎから発生したのではないかと考え、それで間隔をあけたらある程度問題が改善されたので、今回もその対策をとり、もう一度テストしてみた。

結果

そうしたら10回に1回くらいその問題が発生することもあるのだが、そのうちの9回くらいは正常動作をするという具合まで改善された。

送信機のスイッチバージョンと可変抵抗バージョンの比較

どちらも無線で実際に操作した場合である。

スイッチバージョンでの動き

Fのボタンを押すと、全速力で前進、Bのボタンを押すと、全速力で後退、Lのボタンを押すと、全速力で左旋回、Rのボタンを押すと、全速力で右旋回、と言うように、何が何でも急発進、急停車をする為、細かい操作が出来なかった。

可変抵抗バージョンでの動き

今にも止まりそうな速度から発進し走りながら徐々に速度を上げ、全速力での走行、全速力から走りながら徐々に速度を落としいまにも止まりそうな速度まで落とし停止まで自由に操作出来るようになった。更に上と同じ条件で前進しながら右旋回、左旋回、後退しながら右旋回左旋回できるようになった。それにより障害物を簡単によけることが実現可能になった。更に微妙な方向転換が出来るようになったこと、非常にゆっくり進んだり、停止状態からスムーズに発進することができるようになった事で操作性が広がった。

まとめ

初めて無線モジュールであるXBeeを使ってみて、XBeeは予想外に簡単に使えるものであって驚いた。今回の実験などを通して、XBeeの基本やPICの基本がわかりとても面白かった。けれどXBeeはたまに送受信に失敗したりするから、そこは不便だなと感じてしまう。今回、XBeeのバグ的な何かが完全には直せなかったことが特に残念だった。XBeeは思った以上にデリケートで、実験中に電源を逆接したら壊れてしまった。それでかなりのコストが飛んでしまったから、今後は起きないように注意していこうと思う。

PICのCコンパイラであるHi-Tec Cは結構使い勝手が良いがPICKIT2で直接書き込めない物があるというのは不便だと思っている。Hi-Tec Cは使い慣れていると思っていたが意外とそうでもなくて、プログラムメモリ使用状況確認用円グラフの使い方に困ったりもした。PICについては、今までにLEDを点滅させるプログラムを数十回書いているが、今回のようなまともなプログラムを書いたのは初めてで、プログラムの作成で特に関数やif文を書くのが大変だった。最初からわかる変数やif文の書き方は、少ししか知らなかったからだ。

PIC4桁のマイコンのCONFIG（プログラムの中にあるPICの設定をするための構文）を設定していると使ってみたいなと思う設定がかなりあって、興味を持てた。今の動作周波数（4MHz）でも十分に早いのだがPLLEN（CONFIGの中の構文）を使い32MHzで動作するとどのくらい今より早くなるのかに特に興味を持った。特に具体的な使い道は思いつかないが、PICがどのくらいまで速い処理ができるようになるのかに興味がある。今後は時計とか、音楽再生プレイヤ、電卓、電子オルガンや「あの楽器」（初音ミクのInnoinnocenceに出てくる楽器）などを作ってみたいと思う。

コンピュータは、情報収集機であり、研究するための文房具でもあると僕は思う。コンピュータを使いこなして更なる研究を続けていきたい。

今回、お父さんからはプログラムの関数の書き方などを習い、装置が動かないときにトラブルを解決する方法を教えてくれた。@OperaLegendさんには、研究のまとめ方や、全体の構成の方法を教えていただいた。お父さんそして@OperaLegendさんご協力に感謝します。ありがとうございました。