ハードウェアとソフトウェアの接点を探る

平成１９年２月

平成18年度　ＯＩＳＡ　技術研究会
組込みソフトウェア部会　研究成果発表

なぜ、組込みソフトウェアなのか？

～組込みソフトウェアとは？～

産業機器や家電製品などに内蔵される、特定の機能を実現するための
コンピュータシステムを組込みシステムといい、その機器の中に組み込まれている
プログラムを組み込みソフトウェアという。

自動車を例にとりますと、以下のように数多くのものがあります。
・エアバッグ、パワーウインドウ、キーレスエントリー
・AT統合制御、エンジン制御
・アンチロックブレーキ 、姿勢制御

産業機器や家電製品などに内臓される、特定の機能を実現するためのコンピュータシステムを「組込みシステム（Embedded system ）」といい、その様々な機器に組み込まれているプログラムを「組込みソフトウェア」という。

各種機器に新たな機能を追加する場合、従来は電気的・電子的な回路（ロジック）を使うため、コストが高くなる問題があったが、1980年代以降コンピュータの発達で、ソフトウェア的に機能の追加が可能になり、機能追加のコストが削減された。

このため、ほとんどの電化製品は組み込みシステムを用いて付加価値として新機能を追加するようになっている。

現在では、宇宙航空の分野から、炊飯器、携帯電話等まで幅広い分野で利用されている。

上図にも記載しているように、１台の車の中にも複数のマイコンが使用されており、組込みシステムが我々の生活の中に密接に関わっていることがわかる。

なぜ、組込みソフトウェアなのか？

～PCとは何が違うの？～

	組込みシステム	ＰＣ
ＣＰＵ性能	４～３２ビット （まれに６４ビット） ※クロックは300MHz程度	３２～６４ビット ※クロックは1GHz程度
メモリ	数Mバイト～数１０メガバイト	数百Mバイト
消費電力	極めて低いことが望まれる	性能を優先することが多い
発熱	設計で抑える	ファンや水冷で対処
電源断	「ある」が前提	「ない」が前提
起動／シャットダウン	数秒／即時	数十秒を要しても良い
堅牢性	使用状況に耐えれること	常識的な利用に耐えられること
信頼性	極めて重要	アプリケーションレベルは許容されうる
リアルタイム性	要求される場合が多い	要求されない

組込みシステムは、一般的なPCとの違いを理解するため、一般的な組込みシステムに求められる要件について考察する。

１．「CPU性能」　CPUは目的に応じて適切なものを選択する。
家電製品や携帯電話などでは、数十～数百MHｚの32ビットCPUが選択されることが多い。

２．「搭載メモリ量」
目的に対して「必要かつ十分」であればよく、32ビットのCPUに対しては、数Mバイト～数十Mバイトの範囲であることが多いが、コストの面から１Mバイト以下になる場合もある。

３．「消費電力」消費される電力はそのまま熱として放出されるため、熱対策も含めて、低消費電力を目指した設計・部品選択を行う必要がある。

４．「熱対策」：ファンレスであることが望まれるため、初期設計の段階から「熱がでない」ように設計する必要がある。

５．「電源」：シャットダウンの手続きを踏まずに電源が断たれても支障がないように、対策を講じておく必要がある。

６．「起動時間／シャットダウン」：携帯電話のように、電源スイッチをONした瞬間に使用可能であり、OFFしたとたんにシャットダウンすることが望まれる。

７．「堅牢性」：システムの利用環境により振動、電磁ノイズ、密閉性、落下などを考慮する必要がある。

８．「信頼性」：自動車の制御部品など、誤動作が発生したら人命を奪う事故にも発展する可能性が有る為、高い信頼性が求められる。

９．「リアルタイム性」：自動車の駆動系の制御・工場の生産装置の制御・高精度の計測などのコントロールを行う組込みシステムでは、処理要求への応答が迅速になされ、処理に必要な時間が「読める」ことが要求される。

なぜ、組込みソフトウェアなのか？

～現状はどうなっているの？～

１．推定組込みソフトウェア技術者

約１９万３０００人（前年：17万5,000人）

２．推定組込みソフトウェア開発規模

約２兆７，３００億円（開発費総額：約６兆7,700億円）

３．組込みシステム関連の産業比率

全産業従事者　5,200万人 製造業全体従事者数844万人 組込みシステム関連企業従業員数471万人 全産業比率9% 製造業比率56%	国内総生産（平成16年）496兆円 組込みシステム産業規模59兆円 国内総生産比率11.9%

経済産業省の「2006年度版組込みソフトウェア産業実態調査」が実施した事業責任者に対するアンケートでは、
2005年では「必要とされる技術者数24.6万人に対して約7.1万人が不足」
2006年では「必要とされる技術者数28.7万人に対して約9.4万人が不足」
としており、技術者の不足が問題となっている。

組込みソフトウェハ業界の現状について考察してみる。

IPA（独立行政法人　情報処理推進機構）の2006年の調査によると、

推定組込みソフトウェアの技術者は、約19万3,000人であり、前年の17万5,000人に対して11%増加していると考えられる。

また、組込みシステムの推定開発費総額６兆7,700億円のうち、約40.4%に当たる約２兆7,300億円が、

組込みソフトウェアの開発規模と推定される。

さらに、上表のように、組込みシステム関連の産業は、全産業比率の９%、製造業比率の56%、国内総生産の11.9%を占めており、

組込みシステムは、我が国の産業中でも主要な産業に発展しているといえる。

それに伴い、技術者の不足が問題となっており、2005年の調査時では「必要とされる技術者数24.6万人に対して約7.1万人が不足」

となっていたものが、2006年では「必要とされる技術者数28.7万人に対して約9.4万人が不足」との調査結果が報告されている。

なぜ、組込みソフトウェアなのか？

～なぜ人が足りないの？～

□ハード依存

メーカーや機種によって全く仕様が異なる。

□厳しい制約条件

メモリサイズ、リアルタイム性、省電力など厳しい制約がある。

□高い安全性と信頼性

開発ソフトにバグがあると修正が困難、人命にかかわる場合もある。

□大規模化、複雑化

従来の組込みソフトと比較して、大規模化、複雑化している

幅広い知識と熟練した職人技が求められていた。
（業務系アプリケーションの開発者からの転向が困難）

OS（Linux、T-Engine等）を搭載した機器の登場
IDE（統合開発環境）に対応した開発環境の登場
ETSS（組込みスキル標準）の公開

先にも述べたように、現在急速に発展中である組込みシステムでは、人材／スキルの不足に悩まされている。

その要因と、対策を以下に考察する。

組込みソフト開発には、企業情報システムの業務アプリケーションとの違いとして、

・メーカや機種によって全く仕様が異なるといったハード依存性

・メモリサイズやリアルタイム性、省電力等の厳しい制約

・開発ソフトにバグが発生した場合には、製品の回収、修正、返却に多大なコストが発生し、
また人命にかかわる場合もあり、より高い安全性と信頼性が求められる

・メーカの差別化や、ニーズの多様化により、組込みソフトが従来と比較しより、大規模化、複雑化している。

といった違いがあり、そのため、開発者には、メカトロニクス、エレクトロニクスなどの幅広い知識や、

熟練した職人技が求められる為、人材に育成に時間を要し、人手不足に悩まされている。

そのような中で、LinuxやT-Engine等のOSを搭載した機器の登場や、各ベンダーからEclipese等のIDEに対応した開発ツールが登場し、

業務アプリケーション開発者から組込み系の開発の現場でその手腕を生かせる環境も整いつつある。

また、IPAにて、組込みソフトウェアの技術者に必要なスキルを体系化した「組込みスキル標準」

（ETSS：Embedded　Technology Skill Standart)が公開されている。

このETSSとは、組込みソフトウェア開発に必要なスキルを明確化・体系化したものであり、組込みソフトウェア開発者の人材育成・活用に有用な「ものさし」（共通基準）として

１．スキル基準（Version1.0）
２．キャリア基準（Draft）
３．教育カリキュラム（Draft）（組込みシステム開発未経験者向けカリキュラム）
という要素を提供したものである。

なぜ、組込みソフトウェアなのか？

～今後はどうなるの？～

より便利により簡単に

□デジタル家電業界

デジカメ、デジタルビデオ、DVDレコーダ等

⇒ Linuxを用いた情報家電への展開

□自動車業界

カーナビ、ECU（電子制御装置）、車載LAN等

⇒ Bleutooth等を利用した無線への展開

□その他

医療機器、ロボット等

⇒ 小型化、低コストに向けたポストPCとしての展開

需要大幅増!!

以下に今後の組込みソフトウェアの動向に関して考察する。
現在の組込みソフトウェアを利用する主要分野としては、
デジタルカメラ、デジタルビデオ、DVDレコーダ等にて利用する「デジタル家電業界」、
カーナビ、ECU（電子制御装置）、車載LAN等で利用する「自動車業界」、
その他としては、医療、ロボットの分野でも広く利用されている。
これらの分野の今後の展開として、
「デジタル家電業界」では、Linuxを利用しネットワーク化した情報家電としての展開、
「自動車業界」では、Bleutooth等を利用した無線への展開、
その他、医療、ロボットの分野では、小型化、低コストに向けたポストPCとしての展開が期待できる。
その中で、通信や、画像・動画技術が、組込みソフト技術も今後のトレンドとして期待される。
また、製造業の好調や、RFIDの活用に伴い、益々組込みソフト技術者の需要は高まると推測される。

初めての組込みソフトウェア

～研究目標～

- 成果が目に見える形として残るようなものを作りたい ！
- みんな、自分が作ったプログラムで何か作って動かしてみたい!
- やるなら楽しい成果がほしい！

- どうやって、ハードウェアがソフトによって制御されるのか？
- ソフトウェアの知識はあるが、ハードウェアの知識は無いけど大丈夫？
- 開発環境（OS、言語等）はどうすればよいの？
- 開発設備は、どんなものが必要なの？

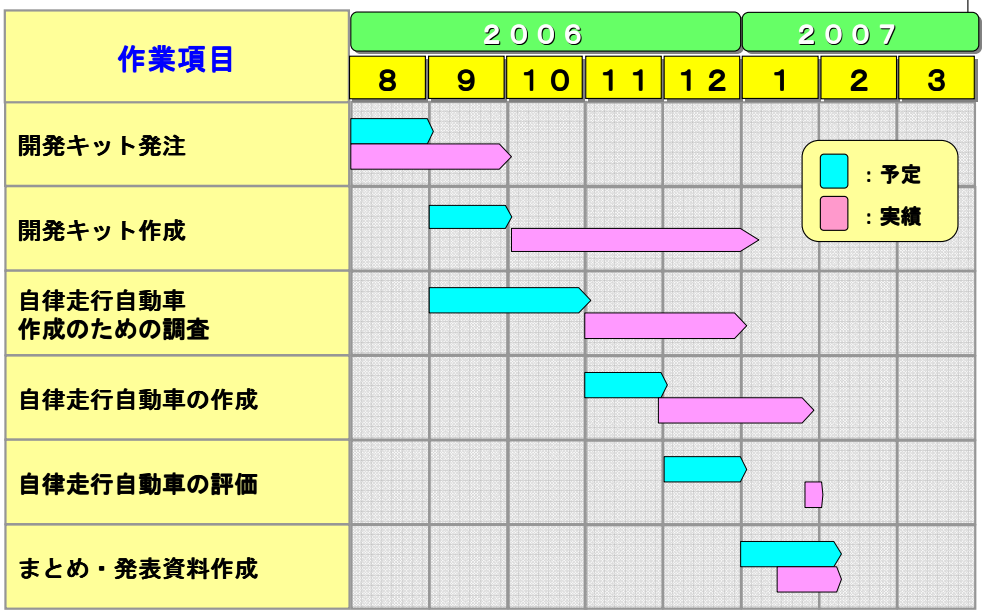
ハードウェアとソフトウェアの接点を探る

自律走行自動車の作成！

以降より本部会の研究活動内容、研究成果について記載する。
まず、研究目標の決定に際しては、部会内で以下のような意見が挙がった。
・成果が目に見える形として残るようなものを作りたい ！
・みんな、自分が作ったプログラムで何か作って動かしてみたい!
・やるなら楽しい成果がほしい！
そこから、目的達成にあたり、以下のような疑問、問題点が挙がった。
・どうやって、ハードウェアがソフトによって制御されるのか？
・ソフトウェアの知識はあるが、ハードウェアの知識は無いけど大丈夫？
・開発環境（OS、言語等）はどうすればよいの？
・開発設備は、どんなものが必要なの？
そこで、本部会の研究目標を「ハードウェアとソフトウェアの接点を探る」とし、
サブテーマを、自律型走行自動車の作成とした。

初めての組込みソフトウェア

～計画と実績～



本部会の計画と実績を上表に記載する。

LEDが光るまで

- ボードの組み立て

■使用ボード

AKI-H8/3052 LAN開発キット（秋月電子通商販売）

【仕様】

- H8/3052 CPUボード
- 10Mbpsイーサネット通信ボード
- 外部SRAM（1MB)
- ACアダプター又はUSBから電源供給
- プッシュスイッチ（ポート割付） ×4
- ディップスイッチ（プッシュスイッチとポートを共用） ×4
- ステータスLED（ポート割付） ×4
- 液晶（16文字×2行、20文字×4行に対応）
- 外部拡張コネクタ

秋月電子通商が販売しているAKI-H8/3052 LAN開発キットを使用した。
キットの内容としては、
H8／3052CPUボード、
LANボード、
液晶、
電源用ACアダプタ、
3048用Cコンパイラ、
開発・サンプルプログラム（ソース付き）一式とドキュメントが入ったCD-ROM
がキットの内容となる。

ボードの仕様は、
H8/3052 CPUボード、
LANボード、
１MByteの外部SRAMが１個、
プッシュスイッチが４個、
ディップスイッチが４個、
ステータスLEDが４個、
16文字×２行を表示できる液晶パネル、
外部拡張コネクタ、
ACアダプタ又はUSBによる電源供給が可能である。

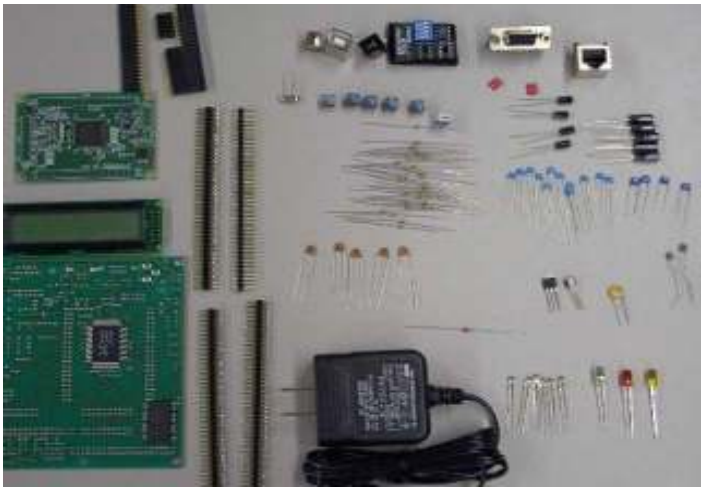
LEDが光るまで

- ボード選択理由

AKI-H8/3052 LAN開発キットを選んだ理由としては、
①インターネット等で調べた結果、初心者入門向けとして多く紹介されている。
②汎用I／Oが多く、自由度がある。
③サーボモーターの制御ができ、自律走行型自動車の制御が可能。
④イーサネット通信ボード がありLAN機器(Ethernet)の開発 も可能。
⑤Cコンパイラ、ROMライタ書き込みソフトが同梱されている。
⑥データシート等のドキュメントが同梱されている。

LEDが光るまで

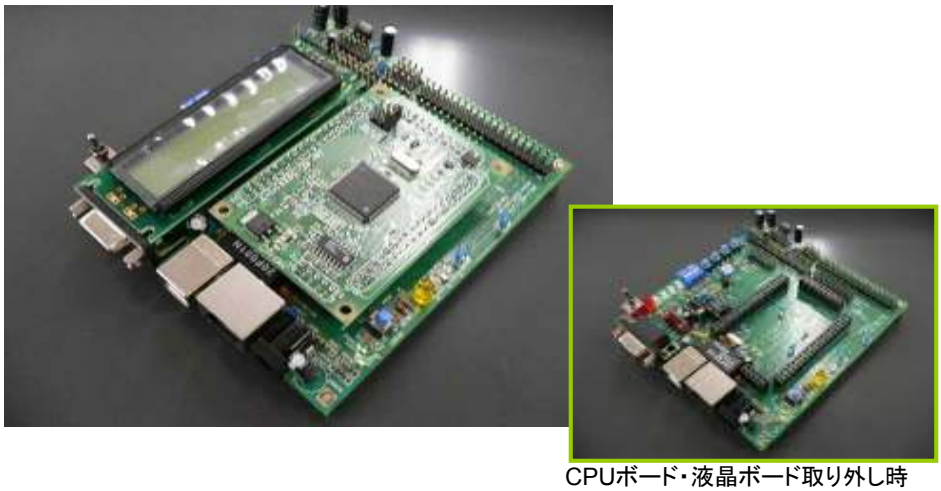
- ボード完成前



キット開封時はこのような状態だった。

LEDが光るまで

- ボード完成

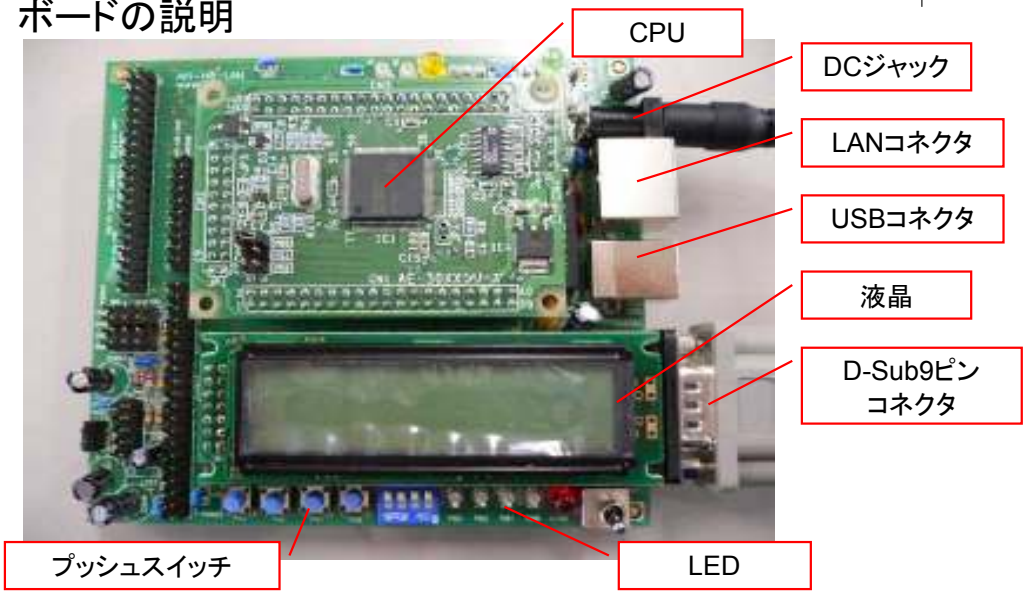


CPUボード・液晶ボード取り外し時

延べ２日間を要してボードの組み立てが完了した。
使用した道具は、半田ゴテ、ニッパー、半田。どれも量販店で購入可能。

LEDが光るまで

- ボードの説明



ボードの各部品について、

CPUは、ルネサステクノロジ社製のH8/3052F
DCジャックは、ACアダプタを接続し、電源供給。
USBコネクタは、USBから電源を供給する為のもので、通信はできない。
LANコネクタは、LANケーブルを接続することが可能。
液晶は、プログラミングすることに文字を表示することが可能。
LEDは、プログラミングすることによって点灯／消灯することが可能。
プッシュスイッチは、プログラミングすることによって入力(押下したか)を検出することが可能。
SRAMは、液晶ボードの下に配置されており、1MBの容量を有す。
D-Sub9ピンコネクタは、PCと接続し、ROMライターソフトを使いROMにプログラムを書込む為に使用する。

LEDが光るまで

- 開発環境

開発用PC	Windows XP
Cコンパイラ	CC38H.exe（キット同梱）
アセンブラ	A38H.exe（キット同梱）
リンカー	L38H.exe（キット同梱）
コンバータ	C38H.exe（キット同梱）
ライティングソフト	H8WriteTurbo（キット同梱）
転送用ケーブル	D-sub9ピンオス- D-sub9ピンメス ストレートシリアルケーブル

開発環境は、Windows端末上で行い、C言語でプログラムを作成。

Cコンパイラ、アセンブラ、リンカー、コンバータ、ライディングソフトは開発キットに同梱したものを使用。

開発用PCからプログラムを転送する為にD-Sub9ピンオス-D-Sub9ピンメスのストレートシリアルケーブルを用意した。

LEDが光るまで

- まずはサンプルプログラムを動かしてみる

【サンプルプログラムの仕様】

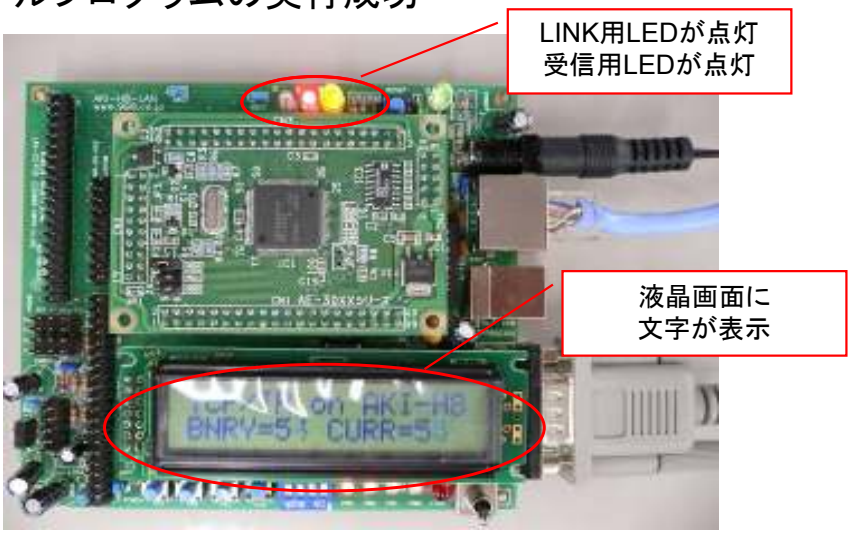
・LANケーブルを接続し、パケットを受信するとLEDが点滅する。

組み立てたボードが正常に動作できるかキット同梱のサンプルプログラムをROMに書き込み試した。

サンプルプログラムは、LANケーブルをボードに接続した場合、パケットを受信するとLEDが点滅し、液晶に文字が表示される。

LEDが光るまで

- サンプルプログラムの実行成功



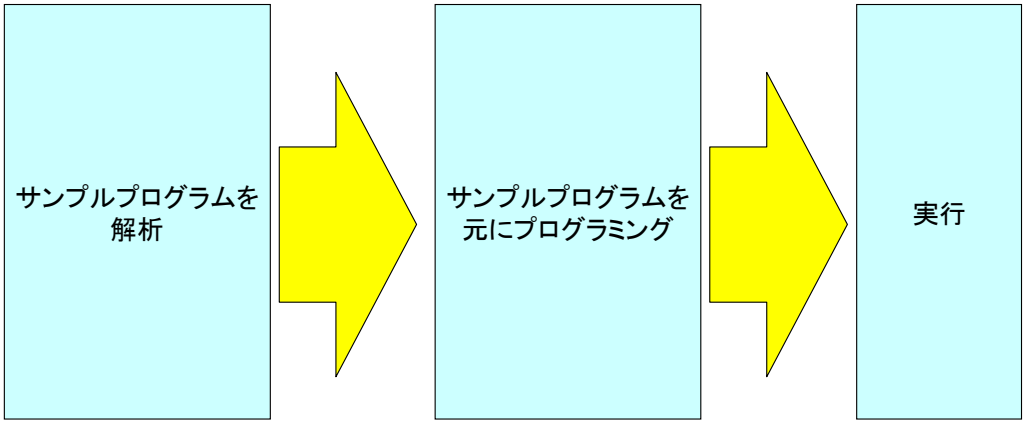
LANケーブルを接続したところ、LINK用のLEDが点灯し、受信用LEDが点滅した。

液晶画面には、何かしらのパケットを受信すると下の行に「BNRY=XX CURR=XX」と表示される。正常に表示されればネットワーク機能は正常に動作している。

サンプルプログラムの実行は成功した。

LEDが光るまで

- LEDを光らせるプログラムを作ろう



色々なプログラミングをこれから行なうのに初級編として、まずは、シンプルにLEDを光らせるだけのプログラムを作成することにした。

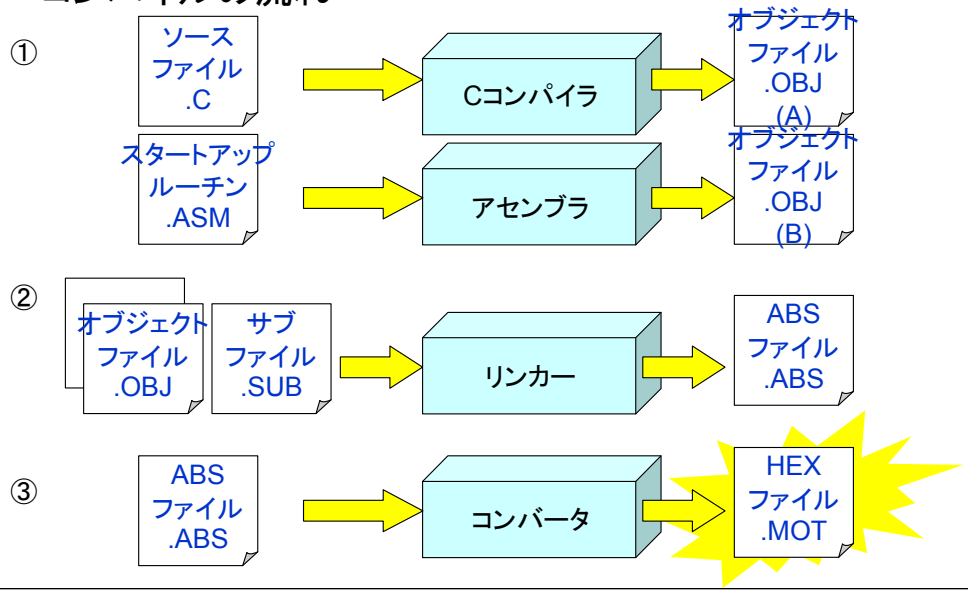
初心者としては、どこから手を付けてよいか判らなかった為、先の動作確認用で使用したキット同梱のサンプルプログラムにLEDを光らせる処理があったので、サンプルプログラムを解析し、LEDの光らせ方を解析・調査する。

次に、サンプルプログラムを元に、LEDを光らせる処理以外を省き、LEDを光らせるプログラムを作成する。

最後に、プログラムのROMに書き込みLEDを光らせてみる。

LEDが光るまで

- コンパイルの流れ



今回の開発環境では、図のような流れで、コンパイルを行なった。

まずは、コンパイル・アセンブリすることによってオブジェクトファイルを作成する。

ソースファイルは、実際に動かしたい動作をC言語でコーディングする。

スタートアップルーチンは、キット同梱のサンプルプログラムから流用した。UNIXやWindowsのプログラムはmain関数以降をコーディングすればよいが、組み込みシステムでは、 main関数が呼ばれる以前の動作を記述する必要がある。マイコンが動き始めた時、マイコンは割り込みベクトル領域のアドレスからスタートアップルーチンを読み、スタートアップルーチンはスタックの設定など初期化処理を行ないmain関数を呼ぶ役割を持つ。流用したスタートアップルーチンはアセンブラで記述されており、アセンブルする必要があった。

次に、リンカーで、生成されたオブジェクトファイルを、サブファイルにより合体しROM番地を割り当て、ABSファイルを作成する。

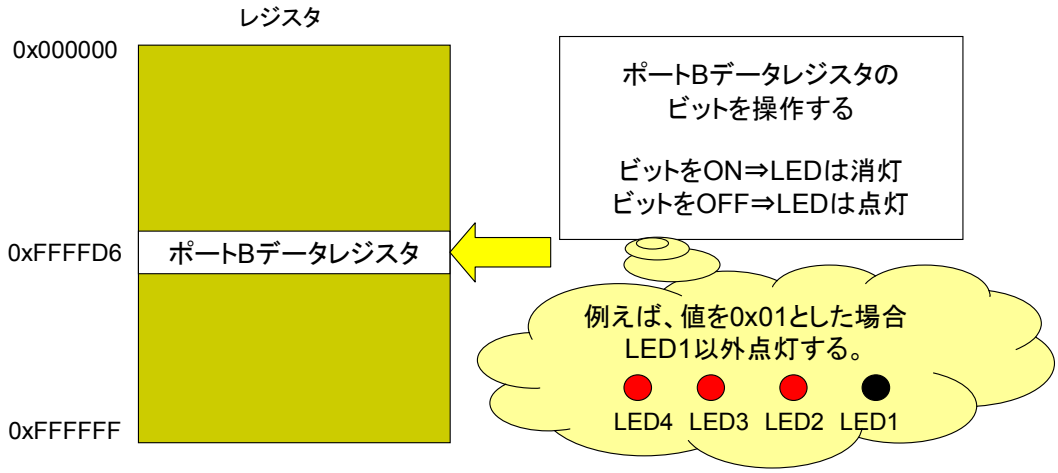
サブファイルとは、プログラム領域、データ領域などのセクションをアドレス指定で記述しており、プログラムはサブファイルの記述に従ってROMに展開される。

最後に、コンバータで、ABSファイルを書込み用HEXファイルを変換する。リンカーが生成したABSファイルはLOADERが理解できる形式（Ｓフォーマット）に変換する必要がある為である。

最終ファイルとして生成されたHEXファイルは、ライティングソフトを使ってボードに転送する。後は、マイコンの電源を入れれば実装したプログラムが実行できる。

LEDが光るまで

- LEDが光る仕組み



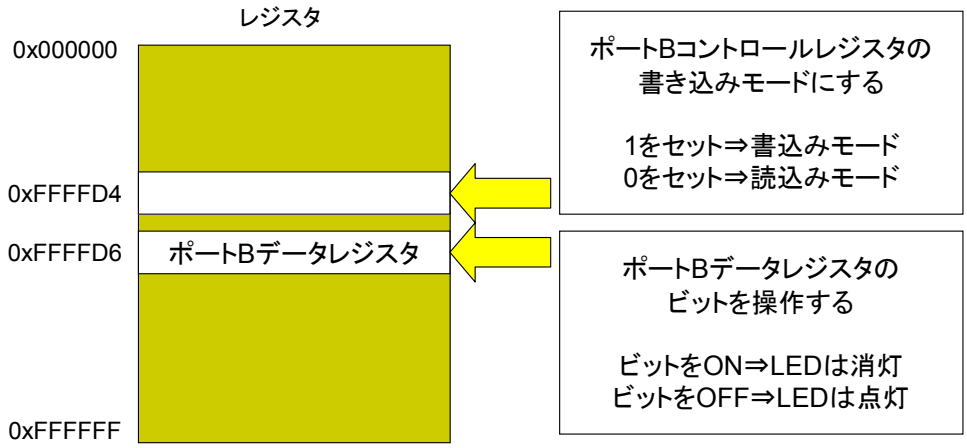
解析したところ、ポートＢデータレジスタと呼ばれる、アドレスのビットを操作することでLEDの点灯／消灯することがわかった。

例えば、0x01の場合、LED１のみが消灯し、LED2～4は点灯する。

しかし、コーディングを行い実行してみたが、LEDは点灯しなかった。

LEDが光るまで

- LEDが光る仕組み（2）



ポートBレジスタを操作したが、LEDは点灯しなかった。

さらに解析とデータシートを読んだところ、ポートBコントロールレジスタが存在し、ポートBデータレジスタを操作する為には、コントロールレジスタを書き込みモードにする必要があった。

ポートBコントロールレジスタを書き込みモードに変更し、ポートBデータレジスタを操作するプログラムを修正した。

LEDが光るまで

- モニタの導入

プログラムを修正する度にROMに書込む

↓

しかし、
ROMの書き込み回数は100回まで保証されていない

↓

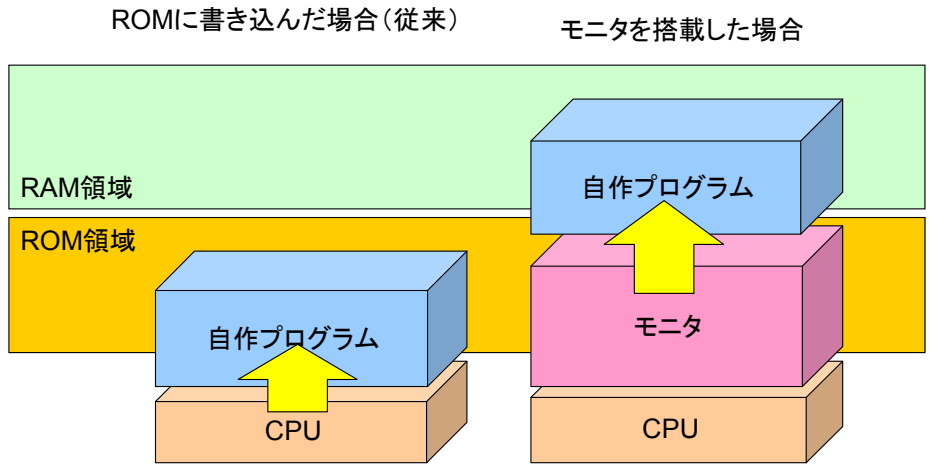
モニタを使い、プログラムをRAMへ転送し実行する

これまではプログラムを修正する度、内臓ROMに書き込み実行していた。しかし、内臓ROMの書き込み回数は、100回までしか保証されておらず、これまでの方法ではROMの寿命を縮めてしまう。

ROMに直接書き込まずにプログラムを実行する方法としてモニタというプログラム（ツール）がある。モニタを使用すると、これまで、100回までしか書き込めない制限が解消でき、開発者は書き込み回数を気にすることなく、開発を進めることができる。また、プログラムのデバックを助ける機能もあり開発効率が向上できる。

LEDが光るまで

- モニタのイメージ



ROMに書き込んだ場合、開発者が作成したプログラムは、CPUから呼ばれるのに対し、モニタを搭載した場合、開発者が作成したプログラムはモニタから呼ばれる。

但し、RAM領域に自作をプログラムを転送する為、あまりにも大きいプログラムが実行できないという欠点もある。

LEDが光るまで

- モニタの入手
  - ルネサステクノロジ社製モニタ
    http://japan.renesas.com/homepage.jsp
  - ダウンロードサイト「Vector」 AKI-H8用モニタ
    http://www.vector.co.jp/
  - 秋月電子通商製モニタ

ルネサステクノジ社、ダウンロードサイト「Vector」、秋月電子通商など有償・無償のものが多く配布されている。

また、インターネットのサイトには、モニタの作り方、改造の方法等も多く紹介されている。

LEDが光るまで

- モニタを使った場合の注意点
  - プログラミング領域の変更
  - 大きいサイズのプログラムは使用できない

・プログラム領域の変更

コンパイル時に、サブファイルを使ってリンクするが、その際、プログラム領域を指定している。サンプルプログラムでは0x000200にプログラムが展開されるように指定されているが、モニタを使用した場合モニタがその領域に置かれている為、自作プログラムはその領域を使用できない。そこで、モニタを使用した場合はRAM領域に自作プログラムを置く。今回は、プログラム領域が重複しないように「0xFFE020」を指定した。

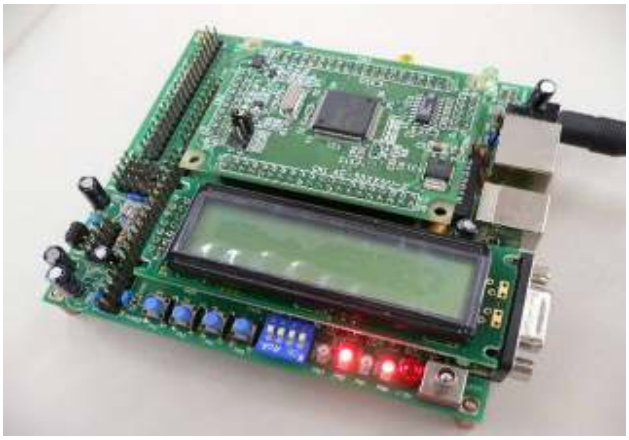
・大きいサイズのプログラムは使用できない

モニタをROMに焼いた状態で自作プログラムをRAM上に置く為、容量の関係で大きいサイズのプログラムは置くことができない。実際にボードの動作試験を行なったサンプルプログラムをモニタ上で動かそうと試みたが、プログラムサイズが大きい為RAMに置くことができなかった。

LEDが光るまで

- LED点灯！

見事、LEDが点灯しました



モニタ上でLEDを光らせるプログラムを実行した。
見事、LEDを点灯することができた。

LEDが光るまで

- LEDを光らせるまでに苦労した点
  - コンパイル時にプログラムをROMに配置する領域を考える必要があった。
  - どのポートがどのハードと繋がっているかを知る必要があった
  - データシートを読み、理解する必要があった。

LEDを点灯することができたが、そこまでに到達するまでに苦労した点は、Widows上やUnix上で動く一般的なアプリケーションのプログラミングと違い、コンパイル時にプログラムを配置する領域を考えたり、また、どのポートがどのハードと繋がっているか知る必要があった。また、データシートを読み、理解する必要もあった。

自律走行自動車の作成

～センサーを使った自律走行マイコンカー～

□製作背景

外部センサーなどからの入力を出力にフィードバックし、自律的に動作する物を
実際に作成する事で、組み込みの技術習得を目指す。

□検討仕様

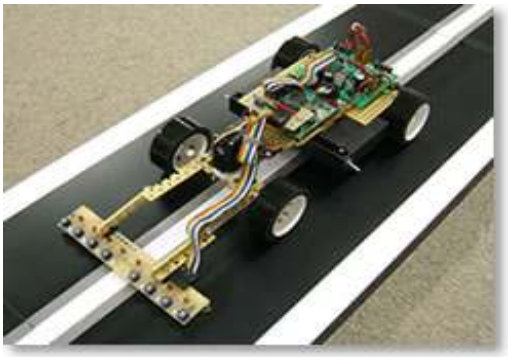
仕様方針　　：いつかマイコンカーラリー（※1）に参加できるよう検討。
ただし、自前の部材で安価に作成。

※１：マイコンカーラリー(MCR)とは、ジャパンマイコンカーラリー実行委員会の支給する指定マイコンボードを搭載し、独自に製作、プログラミングした手作りのマシンで規定のコースを完走しスピードを競う競技です。 1996年の第1回開催以来、年々参加台数が増加しており、技術レベル、走行タイムの向上が見られ、ますます注目度が高まってきています。

自律走行自動車の作成

要件　：黒線のコース上をコースを外れずに走る。
※ いかに早く走るかが重要。

基本仕様　：黒線を赤外線センサーで認識しステアリング操作を行う。
ステアリング切れ角は黒線を外れる時間から算出
スピード調整は切れ角に合わせて行う。



マイコンカーイメージ

自律走行自動車の作成

ハード仕様　：前面にステアリングと連携した赤外線センサー×最低2つ
DCモーターで駆動
上記をAKI-H8で制御
駆動は乾電池で行う。（要 必要電圧算出）

ソフト仕様　：センサーからの黒線を踏んだ入力があった場合反対方向に
ステアリングを切り始める。
ステアリングスピードはだんだんと加速する。
ステアリング切れ角にしたがって速度を操作する。
切れ角大：速度低下（大）
切れ角小：速度低下（小）
切れ角なし：速度一定（だんだんと加速）

以上の仕様を以下のステップで計画

開発ステップ　：**Step１**．基礎技術の習得
Step２．センサー入力信号取得
Step３．サーボモータ（ステアリング用）駆動制御
Step４．モーター駆動制御
Step５．ソフトウェアで連携

※車（筐体）作成、部品配置も平行で行う。

自律走行自動車の作成

STEP1　：　基礎技術の習得

見よう見まねで行ってきたLED動作で、開発環境、動作方法は理解できた。
だが、今後の作業では、現状で作成したソフトが実際にどういった意味なのか？の後回しにしていた部分の理解が必要になる。

今回必要となる（と思われる）基礎部分を下記に挙げる

１．基盤の配線図の読み方
　どこに何をつなげればいいの？
２．データシートの読み方
　どうやってコンロトールしたらいいの？
３．正確なタイマー操作
　リアルタイムな信号はどうやって出力したらいいの？

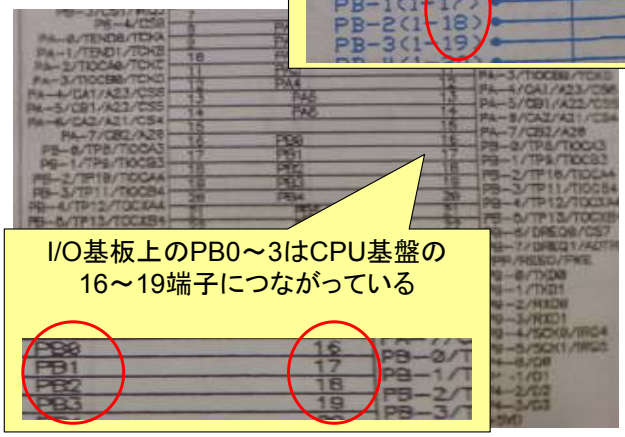
自律走行自動車の作成

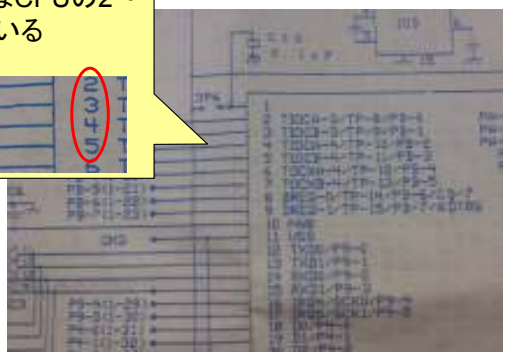
STEP1：基礎技術の習得

1. 基盤配線図の読み方

CPU基板上の16～19端子はCPUの2～5端子につながっている

I/O基盤配線図





CPU基盤配線図

I/O基板上のPB0～3はCPU基盤の16～19端子につながっている

まずは今回使用した基盤の配線図の読み方を調査する。

今回テストを行ったLEDから逆向きにどのようにつながっているか調査する。

今回使用したLEDはPB0、１、２、３と書かれた部分につながっており、配線図を元に追っていくと

CPUの2、3、4、5、の足に割り当たっていることがわかる。

今回はCPUのこの端子に対して信号を出したことがわかる。

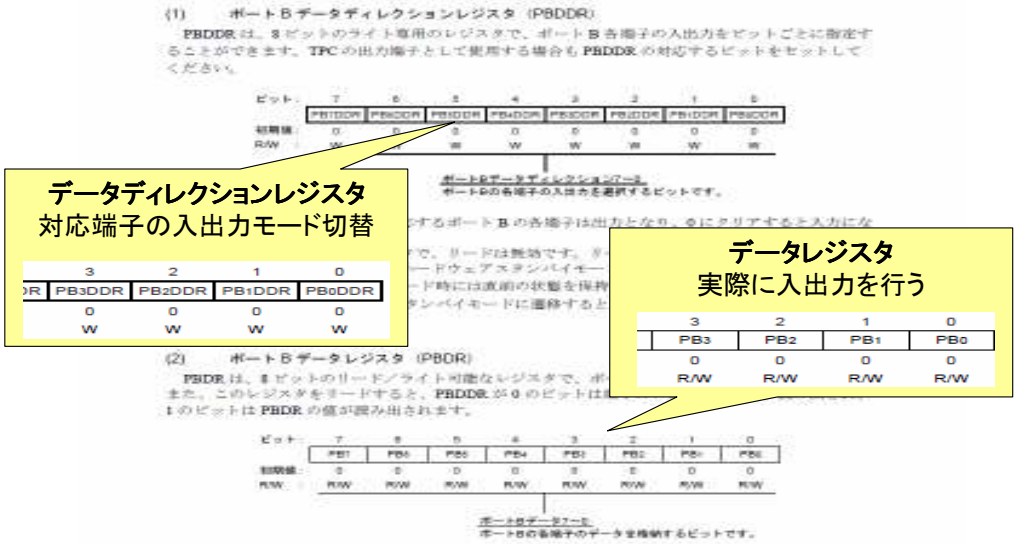
このほかにも、スイッチがどのピンに割り当たっているのか、LCDがどのピンに割り当たっているのか

などこのボードのI/OとCPUの足がどのように接続されているかが図をもって理解できる。

自律走行自動車の作成

STEP1 ： 基礎技術の習得

2. データシートの読み方



LEDの点等で使用したPBDDRと、PBDRはH8/3052Fデータシートの I/Oポートのポート B 部分に書かれている。
これを見ると、PBDDRはポートBデータディレクションレジスタの事であり、ポートBの各端子ごとの入出力モードの切り替えを行う書き込み専用のレジスタとわかる。
つまり、ここの8ビットを使用して、PB0～PB7の端子それぞれをデータ入力用
データ出力用に振り分け使用することが可能となる。
次に各PBDRだが、ポートBデータレジスタの事でありPBDDRで設定した入出力に従い
実際に各端子ごとの状態をコントロールするレジスタである。

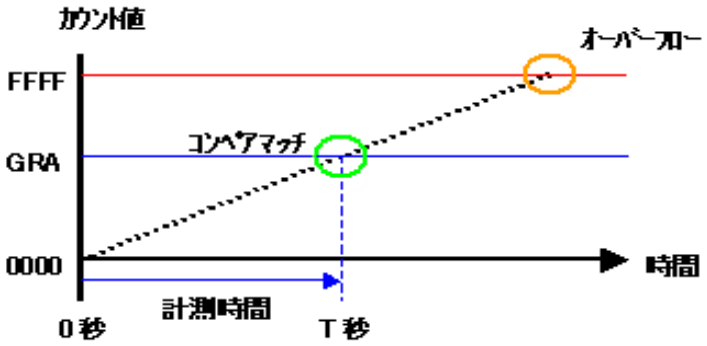
つまりLEDを光らせるのは、配線図から見て、CPUのポートBに対して出力を行えばよい事になり
データシートから見てPBDDRの該当端子の値を出力モードにし該当端子のPBDRの値を操作することで
LEDを点等させることが可能になる。

自律走行自動車の作成

STEP1 ： 基礎技術の習得

3. 正確なタイマー操作

ITU（16bitインテグレーテッドタイマーユニット）とGRA（ゼネラルレジスタA ）を使用したタイマー



次に正確なタイマーの使用方法を調査する。

これはサーボモータ動作には正確なミリセックオーダーのパルスが必要となるためである。

実際にサーボを使用する場合、20ms周期にパルス幅1から2msの幅で動作を決定しないといけない。

例えば右に駆動させたい状況でもパルス幅が0.5msずれただけで左に動作してしまうことがある。

その為、ソフトウェアでの制御では難しくCPUのタイマー機能を使用し実現しようと試みた。

今回使用したH8-3052F CPUには、ITU（16bitインテグレーテッドタイマーユニット）という機能が5チャンネル分付いている。

ITUはタイマーに関するさまざまな機能が扱えるようだが今回は単純なタイマー機能を調査する。

先ずタイマー動作をスタートさせると、16bitのTCNT（タイマカウンタ）がカウントを開始する。

このカウンタは、基準となるクロック信号が１回 H（ハイ）/L（ロー）を繰り返す度に、１ずつカウントを増やしていく。

つまり、クロック信号の周期（時間）が分かっていれば、カウント値によってカウント開始から何秒経過したかがわかる。

ここで、GRA（ゼネラルレジスタA）に、0000hからFFFFhの間の好きな値を設定しておくと、カウント値がGRAと一致（コンペアマッチ）したときに、

TSR（タイマーステータスレジスタ）のIMFAというbitが1となり、一致したことを検出することが出来る。

つまり、GRAに書き込む値によって、好きなタイマー期間を設定することが可能となる。

自律走行自動車の作成

STEP2：センサー入力信号取得

今回は 下記の汎用光センサーを使用

＜製品仕様＞

動作温度:5～40℃(推奨)
電源電圧:DC5V(±5%)
消費電流:60mA以下
出力信号:アンプ出力(0～5V)
基板サイズ:全幅約12.5mm×全長約33mm
取付穴サイズ:3.2mm

次に黒線を感知するセンサーの入力信号取得を行う。
今回はここにある汎用光センサーを使用する。これで黒色と白色の判定を行う事が可能となる。
（原理は先端の黒い部分に赤外線発光部と受光部があり、常に発光している状態で白色であれば反射され受光部感知、
黒色であれば反射せず受光部が感知しないことで判定）
これは5Vで動作出来、消費電力も少ない事から選定。

センサーは最低限の2個使用で実装を行ったが今後正確なステアリング判断、速度判断を行うために
5個程度はあったほうが良いのではと思われる。

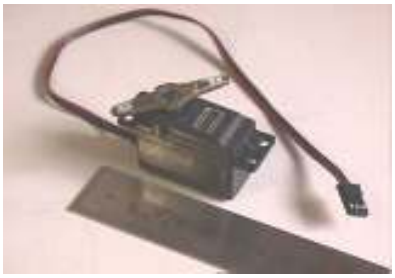
センサー単体で動作確認を行うために、直接５Vの給電を行いテスターを使用し出力信号を確認
センサー部を指で遮断し正確に出力信号が上がる事を確認。
次に基板上での動作確認を行うため光センサはまずPB5の端子につなぎモニターを使用し
手動でPBDDRをReadモードにしセンサー部を指で遮断することでPBDRの値の変化を確認。正常動作を確認する。

自律走行自動車の作成

STEP3 ： サーボモータ駆動制御

今回は下記サーボモータを使用

メーカー:三和電子機器株式会社
速度:0.18 sec/60度
寸法:39.0×20.0×36.0(mm)
トルク:3.3kg･cm
重量:45 (g)
電圧:6(V)



サーボモータとはモータの軸角度を検出する機構を内蔵し、ベースとなるスピードコントロールモータ（AC）やブラシモータ（DC）に

与える電流/電圧を電子回路で微妙に制御する事で、モータ軸1回転あたり数百から数万ステップの微小角度制御ができるモータである。

通常のDCモータでステアリング操作も考えたが、切れ角を正確に取るためには別途センサーが必要になる事を考え

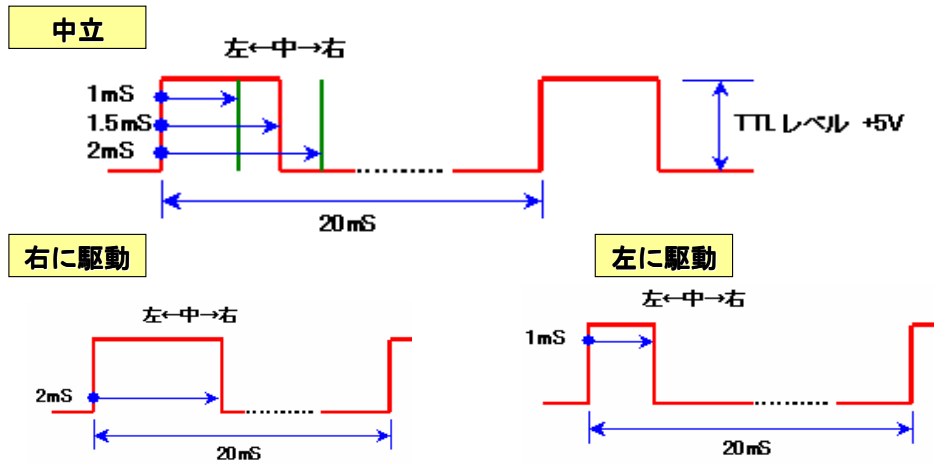
今回はサーボを使用しステアリング捜査を行うように決定した。

自律走行自動車の作成

STEP3 ： サーボモータ駆動制御

サーボモータの駆動方法

サーボモータは下記パルス信号を入力する事で動作するようになっています。



サーボモータの駆動は図のように一定の長さのパルスを出力する事でコントロールします。

サーボモータの切れ角はパルスの立ち上がっている時間で制御されます。

つまり上記図であれば右にいっぱいに切るためには2msのパルスを、左にいっぱいに切る為には1msのパルスを出力することになります。

サーボモータのトルクはパルスが続いている限り発生する事となり、切れ角を安定させておくためには

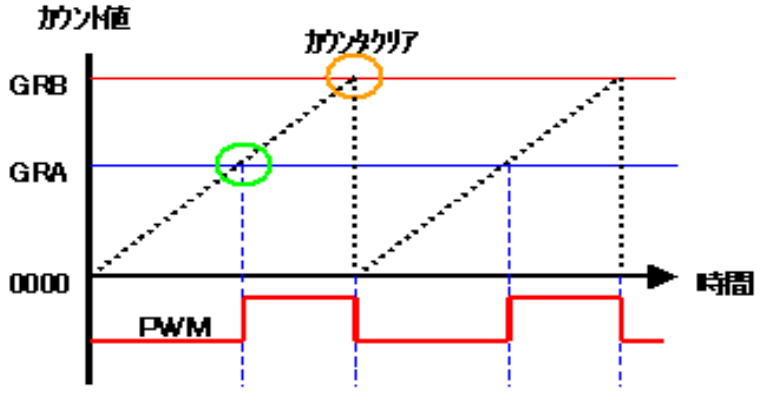
20ms単位で該当パルスを出力し続ける必要があります。

自律走行自動車の作成

STEP3 ： サーボモータ駆動制御

サーボモータの駆動方法

正確なパルス出力のためのPWM信号。



サーボモータを駆動するためには正確なパルス信号を出す必要がある。
その為、STEP１で調査したタイマー機能を使用し正確なパルスを発行する。

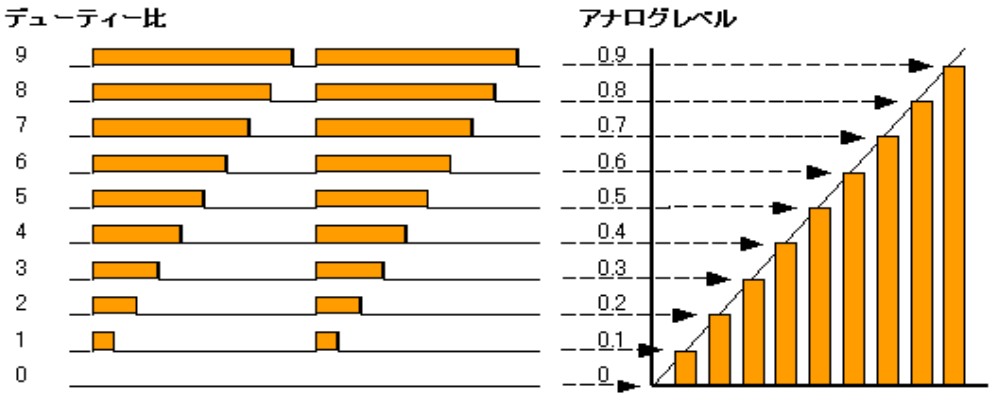
その為に、ITUタイマー機能を使用しPWM（パルス幅変調(Pulse Width Modulation ））方式を実現する。
ON、OFFの比をデューティー比 といいPWMはこの比を正確に設定が可能となる。

図の場合、GRBをカウンタクリア要因に設定した場合、カウンタ値がGRAを過ぎてからGRBに達するまでの期間、パルスが出力される。
なので、GRBでパルス周期を設定し、GRAでデューティー比を設定することが出来る。
なお、GRA > GRB の関係となるように値を設定すると、デューティー比が０（ゼロ）、つまりパルス出力されない状態とすることができる。

自律走行自動車の作成

STEP4 ： DCモータ（駆動用）制御

DCモータの駆動方法



DCモータの駆動はON、OFFを連続的に行い、ON、OFF時間を調整する事で回転数を変化させる。

つまりサーボモータの駆動と同様にPWMのデューティー比を変化させる事で速度調整を行うことが出来る。

デューティー比が高くなる事でON時間が長くなり回転速度が上がり逆で速度が下がる事となる。

ただし今回は開発時間の都合上実装は行っていない。

今回は市販のスピードコントローラを実装し手動による速度調整を行うものとした。

自律走行自動車の作成

STEP5 ： ソフトウェアでの連携

下記配置で接続を行う。

PB5：センサー1（左）

PB6：センサー2（右）

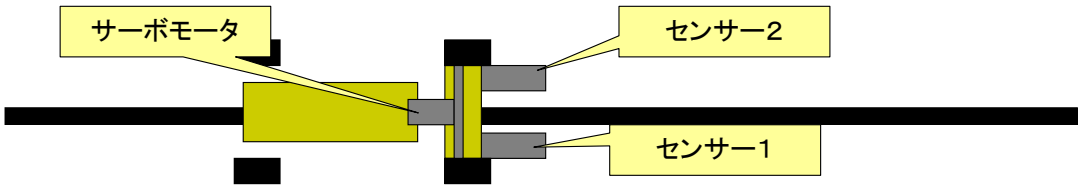
PA3：サーボモータ

下記仕様で操作を行う。

センサー1、2共に白色感知： 直線状態（サーボ中立）

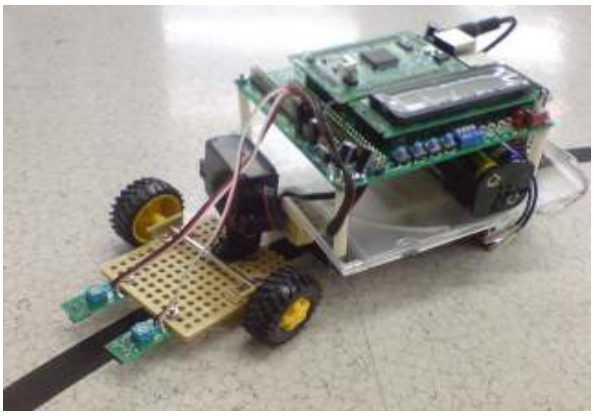
センサー1黒色感知 ： 右折状態（サーボ右に駆動）

センサー2黒色感知 ： 左折折状態（サーボ左に駆動）



自律走行自動車の作成

実際の走行
デモンストレーション



活動を通して思うこと

- 組み込みソフトに対する理解について

・敷居が高く思えていたが、組込みソフトの実装を学ぶ上でのよいきっかけになった
・組み込みソフト開発のおかれている立場、組み込みソフトの実装方法に対しての理解はできた。

- ハードウェアとソフトウェアの接点に対する理解度

・回路についてまでは到達できなかった。
・データシート等を理解し、プログラミングが可能となった。

- 成果物に対して

・１名が自律走行まで到達したが、全体的には簡単な動作しかできなかった！

- 総括

・もっと、基本を学んで計画立てればよかったかな？
・ハード面を含めて、研究するという意味では今後の糧になるものとなった。

今後の期待

～OISA活動の中で何ができる？～

- 学習、評価キットを利用して、開発手法を学ぶ
- 組込みLinux、T-EngineなどのOSを利用した開発を学ぶ
- ETSSを利用してみる

先にも述べたように、組込みソフト業界には、まだまだ多様なニーズがある。
そのような情況のなかで、組込みソフトをテーマとした本部会の意義が十分にあると思われる。
しかし、実際に組込み開発経験の無い、または少ない部会員にて、開発キットの作成から、
実際のコーディングまで行うには、時間的な制限もあり厳しいものであった。
今後、研究テーマのとしては
・学習キットや評価キットを用いてより開発に重点をおいた研究
・組込みLinux、T-Engine等のOSを用いた開発の研究
・ETSSを用いて、またはETSSがどの程度活用できるのかといった研究
に期待したい。

謝辞

本部会を実施するにあたり、多くの方にご協力をいただきました。
ここに、心より感謝の意を表します。
また、開発キットの購入に際し、大分県情報サービス産業協会殿より資金のご援助を頂きましたことに、お礼申し上げます。

組込ソフトウェア部会

【参加者】

山本　竜伸	（エスティケイテクノロジー株式会社）
森　宗美	（エスティケイテクノロジー株式会社）
工藤　香菜子	（システムエイジ株式会社）
後藤　耕平	（九州東芝エンジニアリング株式会社）(副部会長)
藤田　崇徳	（九州東芝エンジニアリング株式会社）(部会長)
諸富　雄一	（株式会社オーイーシー）
大久保　治亨	（大分交通株式会社）
安部　大介	（株式会社シーエイシー）

【アドバイザー・技術委員】

重光　貞彦	（大分シーイーシー株式会社）
平野　藤義	（株式会社オリオンネットシステム）
須藤　裕司	（株式会社システムトレンド）
三宮　由裕	（三井造船システム技術株式会社）

（名簿順）

参考文献／参考サイト

- 組込みプレス　Startup Issue（技術評論社）
- 日経ソフトウェア　2006年12月号（日経BP社）
- アットマーク・アイティ
  http://www.atmarkit.co.jp/fembedded/
- Tech総研
  http://rikunabi-next.yahoo.co.jp/tech/docs/ct_s03600.jsp?p=000062&rfr_id=kanren
- IT pro
  http://itpro.nikkeibp.co.jp/free/ITPro/OPINION/20050601/161961/