高校生ものづくりコンテスト

「電子回路組立」部門

第10回茨城大会

指　導　書

ZENJYOUKEN

平成23年2月

全国情報技術教育研究会

ご　挨　拶

埼玉県立熊谷工業高等学校長　金子　正明
（全国情報技術教育研究会長）

全国情報技術教育研究会では、（社）全国工業高等学校長協会主催の高校生ものづくりコンテスト「電子回路組立」部門を、第５回東京大会から支援させて頂いています。この大会から部門の課題が、コンピュータのソフトウェア技術とハードウェア技術、そして両者を結ぶインターフェース技術の知識と技術を必要とする「組込み技術」に関する内容となり、情報技術教育の振興に関する事項として支援させて頂いております。これまで、全国情報技術教育研究会では、ホームページ上で各県・地区大会における大会の様子や外部制御基板の頒布紹介などをしてまいりました。会員の皆様の御協力により年を追う事に大会も活発になってきており、併せて「組込み技術」に関する指導資料を求める声も出てまいりました。

これを受け全国情報技術教育研究会では、昨年度関係各位の御協力によりまして、第９回高校生ものづくりコンテスト「電子回路組立」部門の課題についての指導書を作成いたしました。今年度も同様に(株)ルネサスソリューションズ　ルネサスマイコンカーラリー事務局様の御協力を得まして、第１０回茨城大会の課題について、主にプログラムを解説した内容としてまとめました。電子回路の組立て（はんだ付け）のポイントについては、昨年度作成した指導書を御覧ください。

本書は、各地区・県大会で開催される指導資料として、そして何よりも、先生方の「組込み技術」教育の研修や実験・実習等の授業の指導資料として活用いただければ幸いです。

本書を作成するにあたり、御協力いただきました皆様に深く感謝申し上げます。

平成２３年２月吉日

第１０回高校生ものづくりコンテスト全国大会
電子回路組立部門　課題

１．課題

競技時間中に製作する『設計製作回路①』と大会当日に配布する『制御対象回路③』を、事前に製作したケーブルにより『持参コンピュータ②』と接続し、競技時間内に『制御プログラム④』を作成し、目的の動作を行うシステムを完成させる。

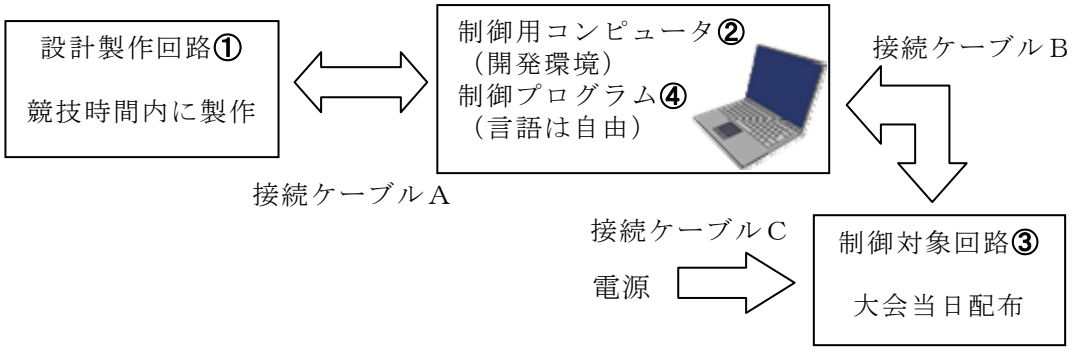


図１　課題概略図

（１）設計製作回路①

大会当日に示す設計仕様に基づく電子回路を設計し、ユニバーサル基板を用いて電子回路基板を製作する。配線はスズメッキ線を使用し、設計製作回路は以下の部品を使用する。

①ユニバーサル基板（サンハヤト ICB293 相当品）

②フォトインタラプタ、スイッチ、コネクタ、0.5φスズメッキ線等

（２）制御用コンピュータ②

開発環境を含め全て持参する。コンピュータの性能・形状等に制限はない。

（３）制御対象回路③

大会当日に競技実行委員から配布する。制御対象回路には制御対象駆動回路と制御対象が含まれている。なお、制御対象としては、次のようなものが考えられる。

①ＬＥＤ

②７セグメントＬＥＤ

③ＤＣモータ

④ステッピングモータ等

（４）制御プログラム④

大会当日に提示する仕様に基づいたプログラムを作成する。使用する言語は自由である。なお、プログラムの仕様例として、次のようなものがある。

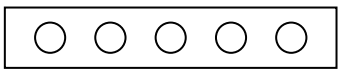
①ストップウォッチのプログラム

②回転制御のプログラム

（５）接続ケーブル

①接続ケーブルＡ（設計製作回路❶用）

ＩＣピッチ１列５ピンコネクタ・メス（ストレートピンヘッダハウジング）

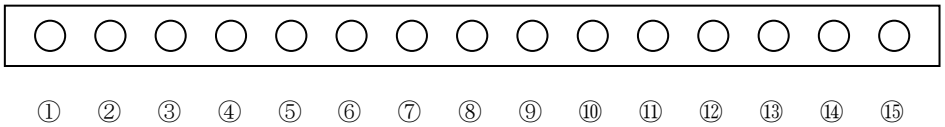


①	ＧＮＤ	②	５Ｖ	③	入力０
④	入力１	⑤	入力２		

図２　接続ケーブルＡ用コネクタのピン配置

②接続ケーブルＢ（制御対象回路❸用）

ＩＣピッチ１列１５ピンコネクタ・メス（ストレートピンヘッダハウジング）

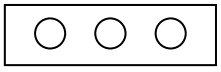


①	ＧＮＤ	②	出力０	③	出力１	④	出力２	⑤	出力３
⑥	出力４	⑦	出力５	⑧	出力６	⑨	出力７	⑩	出力８
⑪	出力９	⑫	ＮＣ	⑬	ＮＣ	⑭	ＮＣ	⑮	ＮＣ

図３　接続ケーブルＢ用コネクタのピン配置

③接続ケーブルＣ（電源供給用）

ＩＣピッチ１列３ピンコネクタ・メス（ストレートピンヘッダハウジング）



①	ＧＮＤ	②	５Ｖ	③	ＧＮＤ

図４　接続ケーブルＣ用コネクタのピン配置

注）コネクタ・メスの具体的なイメージはヒロセ電機のホームページを参照。

(http://www.hirose.co.jp/catalogj_hp/j21800074.pdf)

写真は HNC2－2.5P－8DS

（ＩＣピッチ１列８ピンコネクタ・メス）

２．作業条件

（１）競技時間　２時間 30 分（150 分）

（２）競技実行委員から配布するもの

・『設計製作回路①』で使用する部品および材料等
・『制御対象回路③』
・コンテストで使用する部品の規格表
・AC100V コンセント(2 口)
・ソースリスト印刷用パソコンおよびプリンタ
・Ａ４サイズ方眼紙

（３）競技者が準備するもの

・『制御用コンピュータ②』および開発環境
・接続ケーブルＡ、接続ケーブルＢ、接続ケーブルＣ
・＋５Ｖの電源（Max1000mA 程度）
・工具類およびテーブルタップ
・ソースリスト提出用の記録媒体（USB メモリまたは FD）
・筆記用具および定規・テンプレート類

> 工具類とは、各自の作業に必要なもので、はんだごて、こて台、ニッパ、リードペンチ、＋ドライバ、テスタ、保護メガネ、基板支持台　等

（４）競技者の服装等

・競技中は、各学校で使用している作業服を着用する。
・はんだ付けの作業時には、保護メガネを着用する。ただし、メガネをかけている場合はこの限りではない。

（５）注意事項

①作業を行うにあたっては、安全に十分注意する。
②配布された部品および材料以外のものは、使用しない。
③規格表・命令表が必要な場合は各自で準備し、大会前日に承認を受ける。
④事前に製作したヘッダーファイルは、大会前日に申請し内容の承認を受ける。
⑤接続ケーブルＡ・Ｂ・Ｃは、図２・３・４を参考に事前に製作し準備しておく。ただし、ケーブルの長さは自由である。
⑥ソースリストは、テキスト形式で記録媒体に保存・提出する。

３．審査対象

（１）『設計製作回路①』の設計図

（２）『設計製作回路①』の製作基板

（３）仕様に対応する動作

（４）プログラムのソースリスト

（５）その他（作業態度等）

４．採点基準

（１）採点項目と観点

項　　目	配点	観　　点
プログラミング技術	40	・動作 ・構造 ・書式 ・読みやすさ
組み立て技術	30	・工具類の使い方 ・部品処理 ・はんだの状態 ・配線 ・配置
設計力	20	・正確さ ・配置 ・記号 ・文字
その他	10	・作業態度 ・作業工程
合　計	100	

（２）順位の決定方法

①合計得点の高い順に、１位、２位、３位･･･とする。

②同点の場合は、「プログラミング技術」の得点の高い選手を高位とする。

③「プログラミング技術」の得点も同点の場合は、「組み立て技術」の得点の高い選手を高位とする。

④さらに同点の場合は、「設計力」の得点の高い選手を高位とする。それでもなお同点の場合は、全体の完成度から順位を決定する。

（３）その他

設計製作回路①の製作に関して、別紙『**標準的なはんだ付けについて**』を参照。

５．その他

（１）鉛フリーはんだについて

無鉛（鉛フリー）はんだ（Sn-3.0Ag-0.5Cu、0.8mm）を使用する。

（２）動作確認について

プレ審査時に競技実行委員の指示に従い､競技者が操作して課題の動作確認を行う。

（３）設計製作回路・制御対象回路・当日の課題プログラムについて

『設計製作回路①』･『制御対象回路③』の回路図・使用部品の規格等については、事前公開しない。

また、当日作成する制御プログラムに関しても、事前公開はしない。

（４）その他

大会の参考として、全国情報技術教育研究会ホームページを参照。

<〈第１０回高校生ものづくりコンテスト　全国大会　電子回路組立部門　参考資料〉>

標準的なはんだ付けについて

（１）はんだのぬれ性

イ．はんだが接合するリード線、銅箔によく流れ、長くすそを引いていること。

ロ．部品穴のはんだ付けは、ランドの表面にはんだのぬれ性があること。

（２）はんだの量

イ．はんだの量は、部品リード線の折り曲げ部分、線の切り口等をはんだで覆い、かつ、肉厚が薄く線の形がわかるものとする。（図を参照）
ただし、折り曲げず、かつ、切断せずに取り付ける部品リードのはんだ付けを行う場合は、リードの先端まで、全面はんだで覆わなくてもよい。

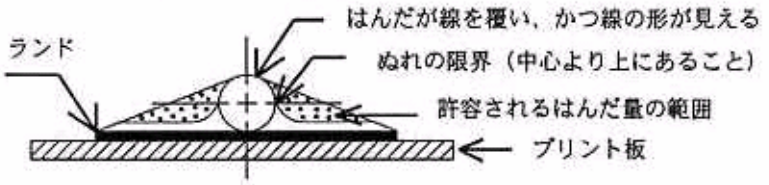
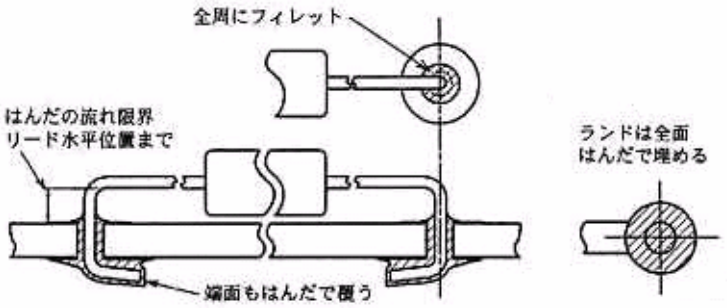
（３）その他

イ．部品端子の線材接合部は、穴あきのないようにはんだ付けすること。

ロ．ランドのないところで、線又は部品リードを接続しないこと。
（空中配線接続をしてはならない）

ハ．ランドをはく離させないこと。

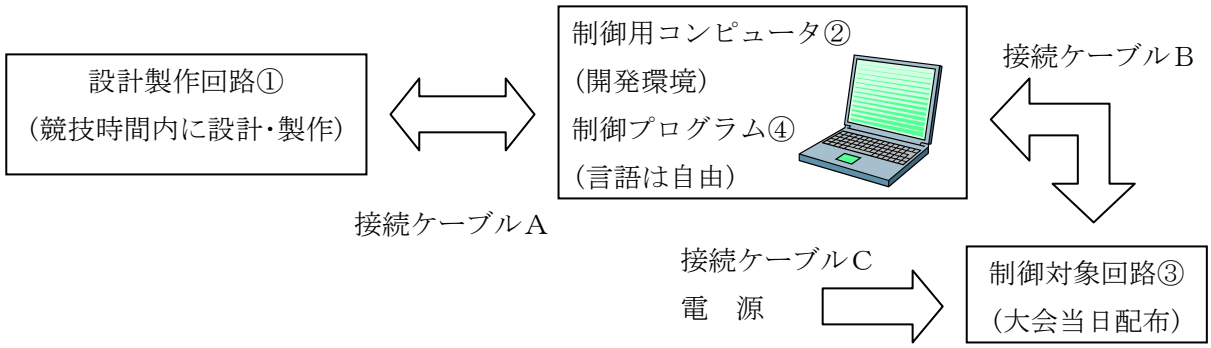
ニ．ジャンパー線を用いず、裏面のみで配線をおこなうこと。

<table>
<tr><td>はんだ</td><td> 
</td></tr>
<tr><td>ランドのはんだ付け標準</td><td> 
</td></tr>
</table>

第 10 回高校生ものづくりコンテスト全国大会（茨城）
電子回路組立部門　課題

1　システム構成

『制御用コンピュータ②』に『設計製作回路①』と『制御対象回路③』を接続し，その『制御プログラム④』を作成し，コンピュータ制御システムを完成させる。



2　設計・製作する回路

以下の条件を満たす入力回路を設計・製作しなさい。

(1) 下図に示されたピン配置による入力回路を設計する。

ＩＣピッチ１列５ピン

○	○	○	○	○
①	②	③	④	⑤

①	ＧＮＤ	②	５Ｖ	③	ＴＧＳ
④	Ｐ　Ｓ	⑤	Ｔ　Ｓ		

ＴＧＳ：トグルスイッチ
Ｐ　Ｓ：ホトインタラプタ(透過型)
Ｔ　Ｓ：タクトスイッチ

(2) 支給される方眼紙(Ａ４)に回路図を書く。

(3) 支給された部品を使用して，設計した入力回路を製作する。ホトインタラプタ(透過型)は，ユニバーサル基盤の部品面に直接取り付ける。

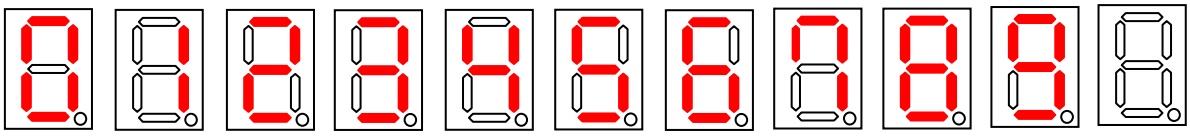
(4) トグルスイッチは，製作した入力回路の部品面の‘H’側に，配布されるシールを貼る。

3　作成するプログラム

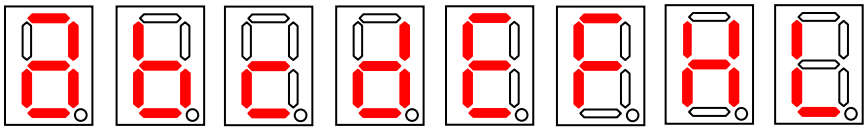
課題１から課題７のプログラムを完成させなさい。

(1) ７セグメントＬＥＤの表示は以下のようにする。(赤が点灯状態)

また，アルファベットの‘a’‘b’‘c’‘d’は小文字で点灯させる。



‘0’ ‘1’ ’2’ ’3’ ’4’ ’5’ ‘6’ ‘7’ ’8’ ’9’ 消灯



’a’ ’b’ ‘c’ ‘d’ ’E’ ’F’ ’H’ ’L’

(2) ステッピングモータおよびＤＣモータの回転は，目視および触って確認できることとする。

(3) それぞれの課題において，トグルスイッチおよびホトインタラプタ，タクトスイッチの状態を表す文中の表現は，以下のとおりとする。

ア　トグルスイッチ

(ｱ)「H」・・・トグルスイッチのレバーがH側(シール貼付あり)に倒れている状態

(ｲ)［L］・・・トグルスイッチのレバーがＬ側(シール貼付なし)に倒れている状態

イ　ホトインタラプタ

(ｱ)「透過」・・・ホトインタラプタの光が透過している状態

(ｲ)「遮断」・・・ホトインタラプタの光が遮断されている状態

(ｳ)「遮断→透過」・・・ホトインタラプタの光を１回遮断して，透過した状態

ウ　タクトスイッチ

(ｱ)「ＯＮ」・・・タクトスイッチを押している状態

(ｲ)「ＯＦＦ」・・タクトスイッチを離している状態

(ｳ)「ＯＮ→ＯＦＦ」・・・タクトスイッチを１回押して，離した状態

(4) それぞれの課題において，初期状態は以下のとおりとする。

ア　タクトスイッチＴＳ・・・「ＯＦＦ」

イ　ホトインタラプタＰＳ・・・「透過」

ウ　トグルスイッチＴＧＳ・・・・「Ｌ」

エ　ステッピングモータおよびＤＣモータ・・・「停止」

(5) 各課題において，対象となっていない制御物は動作させない。

(6) 課題の中にある動作概要図については，おおよその流れが図示してある。条件の詳細については，課題文章のとおりに動作させる。

課題1

(1) トグルスイッチが「Ｌ」の状態で，タクトスイッチが「ＯＮ」のとき，右側の７セグメントＬＥＤに‘Ｌ’を表示する。
(2) トグルスイッチが「Ｈ」の状態で，タクトスイッチが「ＯＮ」のとき，左側の７セグメントＬＥＤに‘Ｈ’を表示する。
(3) タクトスイッチが「ＯＦＦ」のとき，左右の７セグメントＬＥＤは消灯している。

課題2

(1) トグルスイッチが「Ｌ」の状態で，ホトインタラプタが「遮断」されると，ＤＣモータが時計回りに回転する。
　ホトインタラプタを「透過」すると，ＤＣモータは停止する。
(2) トグルスイッチが「Ｈ」の状態で，ホトインタラプタが「遮断」されると，ステッピングモータが時計回りに回転する。
　ホトインタラプタを「透過」すると，ステッピングモータは停止する。

課題3

(1) プログラムのスタート時，右側の７セグメントＬＥＤは‘ａ’を表示する。

(2) タクトスイッチを「ＯＮ→ＯＦＦ」するたびに，

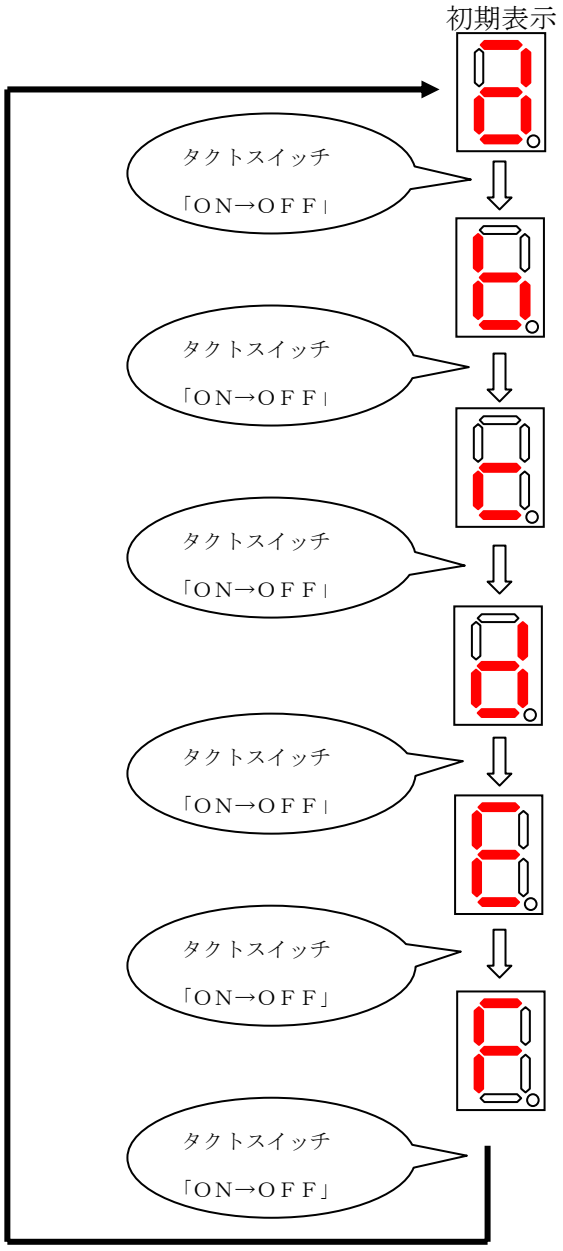
ア　右側の７セグメントＬＥＤの表示が

’a’，’b’，‘c’，‘d’，’E’，’F’の順番で変化する。

イ　７セグメントＬＥＤの表示は，タクトスイッチを押したときは変化せず，離したときに変化する。

ウ　’F’の次は’a’にもどり，繰り返し動作する。

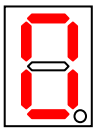
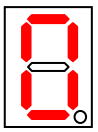
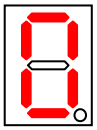
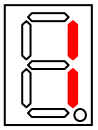
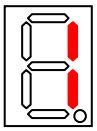
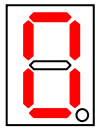
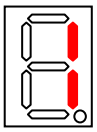
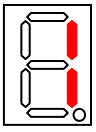
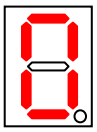
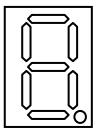
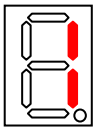
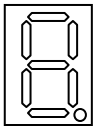
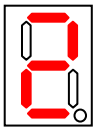
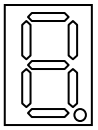
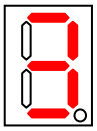
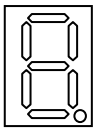
動作概要



課題4

(1) ホトインタラプタおよびトグルスイッチ，タクトスイッチの状態によって，左右の７セグメントＬＥＤを下表のように表示する。

(2) タクトスイッチがＯＦＦのとき，左右の７セグメントＬＥＤは同時に点灯し，違和感なく表示する。

トグルスイッチ ＴＧＳ	ホトインタラプタ Ｐ　Ｓ	タクトスイッチ Ｔ　Ｓ	左　側 ７セグメント ＬＥＤ	右　側 ７セグメント ＬＥＤ
Ｌ	遮断	ＯＦＦ		
Ｌ	透過	ＯＦＦ		
Ｈ	遮断	ＯＦＦ		
Ｈ	透過	ＯＦＦ		
Ｌ	遮断	ＯＮ		
Ｌ	透過	ＯＮ		
Ｈ	遮断	ＯＮ		
Ｈ	透過	ＯＮ		

課題5

(1) トグルスイッチを「H」にすると，左側の７セグメントＬＥＤに‘H’，右側の７セグメントＬＥＤに‘L’が同時に違和感なく表示され，ステッピングモータが，反時計回りに回転する。

(2) (1)の状態において

ア　タクトスイッチを「ＯＮ」すると，

(ｱ) 左側の７セグメントＬＥＤに‘０’，右側の７セグメントＬＥＤに‘L’が表示され，ステッピングモータが(1)の状態よりも，低速回転で反時計回りに回転する。

(ｲ) タクトスイッチを「ＯＦＦ」にすると，(1)の状態に戻る。

イ　ホトインタラプタを「遮断」すると，

(ｱ) 左側の７セグメントＬＥＤに‘H’，右側の７セグメントＬＥＤに‘９’が表示され，ステッピングモータが(1)の状態よりも，高速回転で反時計回りに回転する。

(ｲ) ホトインタラプタを再び「透過」すると，(1)の状態に戻る。

(3) トグルスイッチを「L」にすると，左右の７セグメントＬＥＤは消灯し，ステッピングモータは停止する。

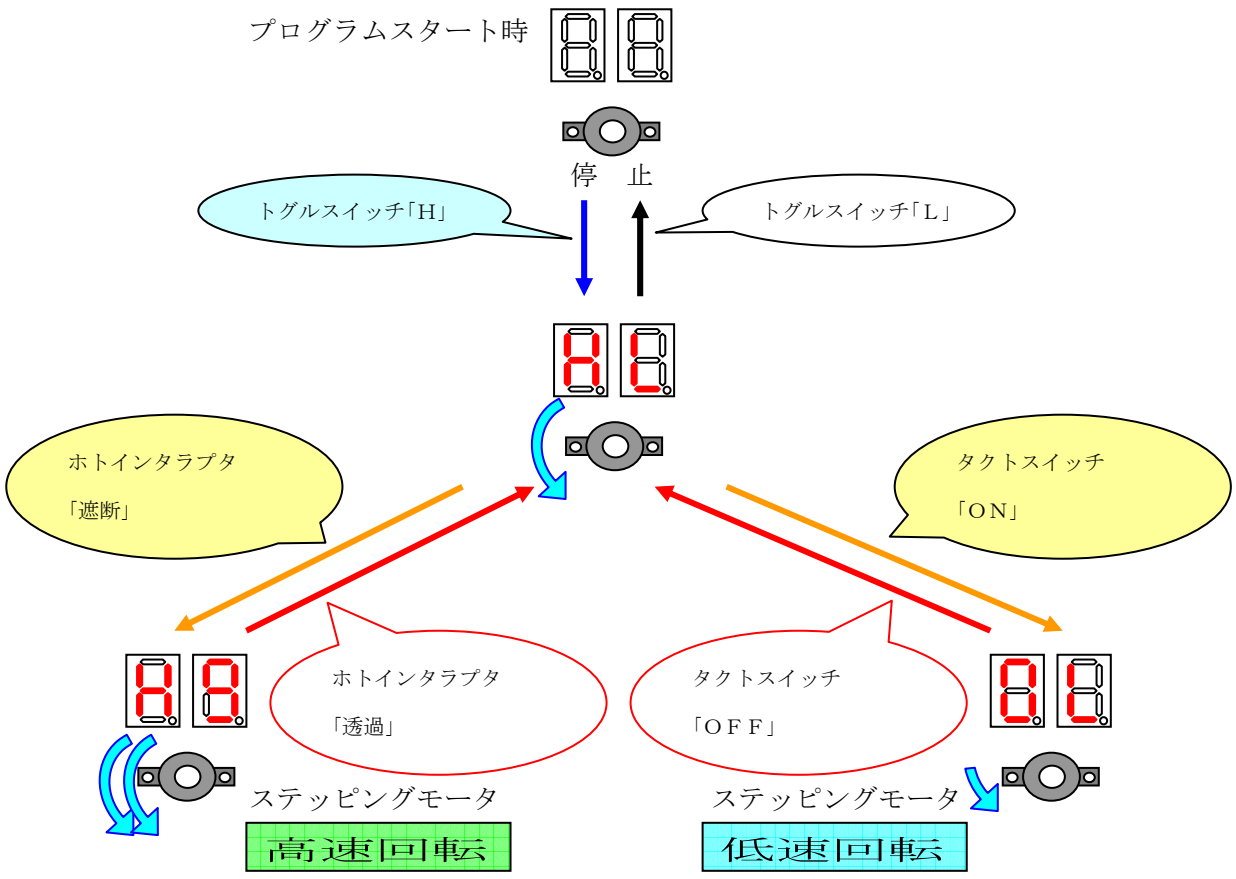
(4) トグルスイッチの「H」と「L」の切り替えは，(1)の状態のときのみ受け付ける。

(5) ステッピングモータが低速回転しているとき，ホトインタラプタの入力は受け付けない。

(6) ステッピングモータが高速回転しているとき，タクトスイッチの入力は受け付けない。

(7) 回転速度の違いは，目視で明らかに変化がわかるようにする。

動作概要



課題6

(1) プログラムスタート時，右側の７セグメントＬＥＤに‘０’が表示され，２つのモータは停止している。

(2) 右側の７セグメントＬＥＤに‘０’が表示されているとき

ア　トグルスイッチを「H」にし，タクトスイッチを１回「ＯＮ→ＯＦＦ」すると，押した直後に，ステッピングモータが時計回りに回転する。

イ　ステッピングモータが，１回転を終了するたびに，右の７セグメントＬＥＤの表示が，１ずつ増えていく。ステッピングモータは停止することなく回転する。

ウ　３回転を終了し，右側の７セグメントＬＥＤの表示が，‘３’になったとき，ステッピングモータは停止する。７セグメントＬＥＤには，そのまま‘３’が表示されている。

(3) 右側の７セグメントＬＥＤに‘３’が表示されているとき

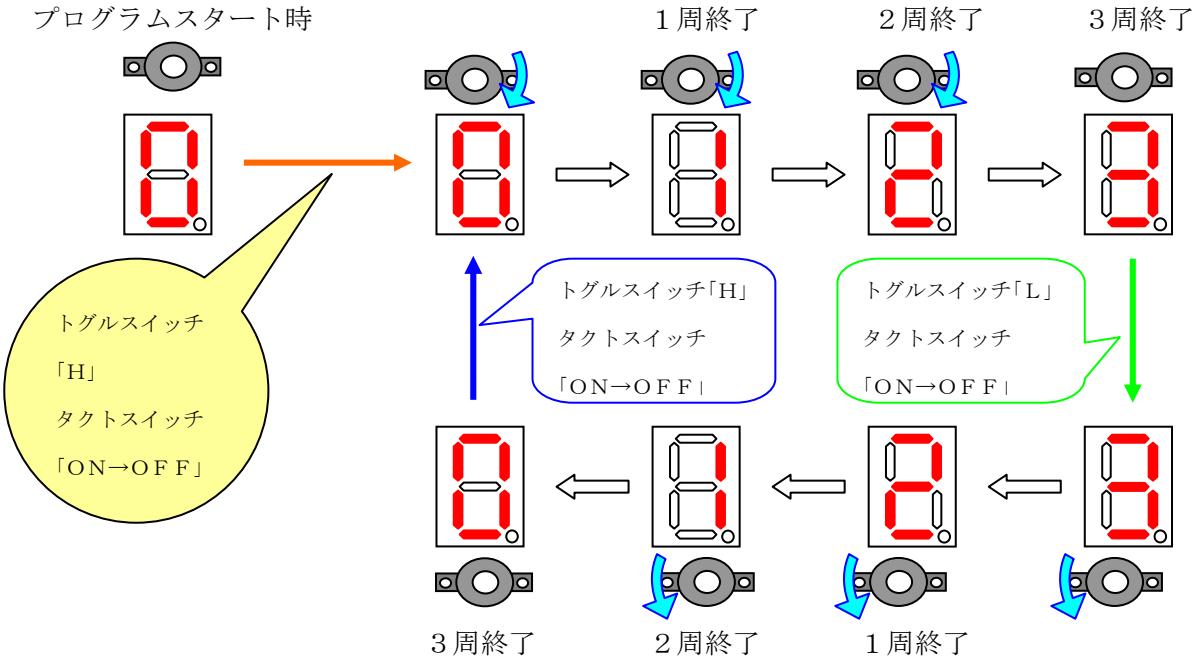
ア　トグルスイッチを「L」にし，タクトスイッチを１回「ＯＮ→ＯＦＦ」すると，押した直後に，ステッピングモータが反時計回りに回転する。

イ　ステッピングモータが，１回転を終了するたびに，右側の７セグメントＬＥＤの表示が，１ずつ減っていく。ステッピングモータは停止することなく回転する。

ウ　３回転を終了し，右側の７セグメントＬＥＤの表示が，‘０’になったとき，ステッピングモータは停止する。７セグメントＬＥＤには，そのまま‘０’が表示されている。

(4) ステッピングモータが回転している間は，いかなる入力も受け付けない。

動作概要



課題7

(1) プログラムスタート時，右側の７セグメントＬＥＤに’0’を表示する。

(2) タクトスイッチを１回「ＯＮ→ＯＦＦ」すると，

ア　約１秒間隔で右側の７セグメントＬＥＤの表示が，

’0’ →’1’ → ’2’ → ‘3’ → ‘4’ → ’5’ → ’6’ → ‘7’ → ’8’ → ’9’　のように変化する。

イ　’9’の次は’0’が表示され，繰り返し動作する。

ウ　タクトスイッチを再び「ＯＮ→ＯＦＦ」すると，押されたときの表示で停止する。

停止している数値は，７セグメントＬＥＤに表示されている。

エ　上記アからウの動作は，繰り返し実行できる。

ただし，カウントアップは，停止している表示から始まるものとする。

オ　７セグメントＬＥＤの動作および停止は，タクトスイッチが**押された瞬間**に行われる。

(3) ７セグメントＬＥＤの表示が停止しているとき，以下の動作をする。

ただし，いずれかのモータが回転しているとき，タクトスイッチの入力は受け付けない。

ア　表示されている数値が’0’から’4’

(ｱ) ホトインタラプタを１回「遮断→透過」すると，ステッピングモータが回転する。

(a) トグルスイッチが「Ｌ」ならば，時計回りに回転する。

(b) トグルスイッチが「Ｈ」ならば，反時計回りに回転する。

(c) (a)および(b)の動作は，ステッピングモータ回転中に何度でも切り替えられる。

(ｲ) トグルスイッチの状態にかかわらず，ホトインタラプタを再び「遮断→透過」すると，ステッピングモータは停止する。

イ　表示されている数値が’5’から’9’

(ｱ) ホトインタラプタを１回「遮断→透過」すると，ＤＣモータが回転する。

(a) トグルスイッチが「Ｌ」ならば，時計回りに回転する。

(b) トグルスイッチが「Ｈ」ならば，反時計回りに回転する。

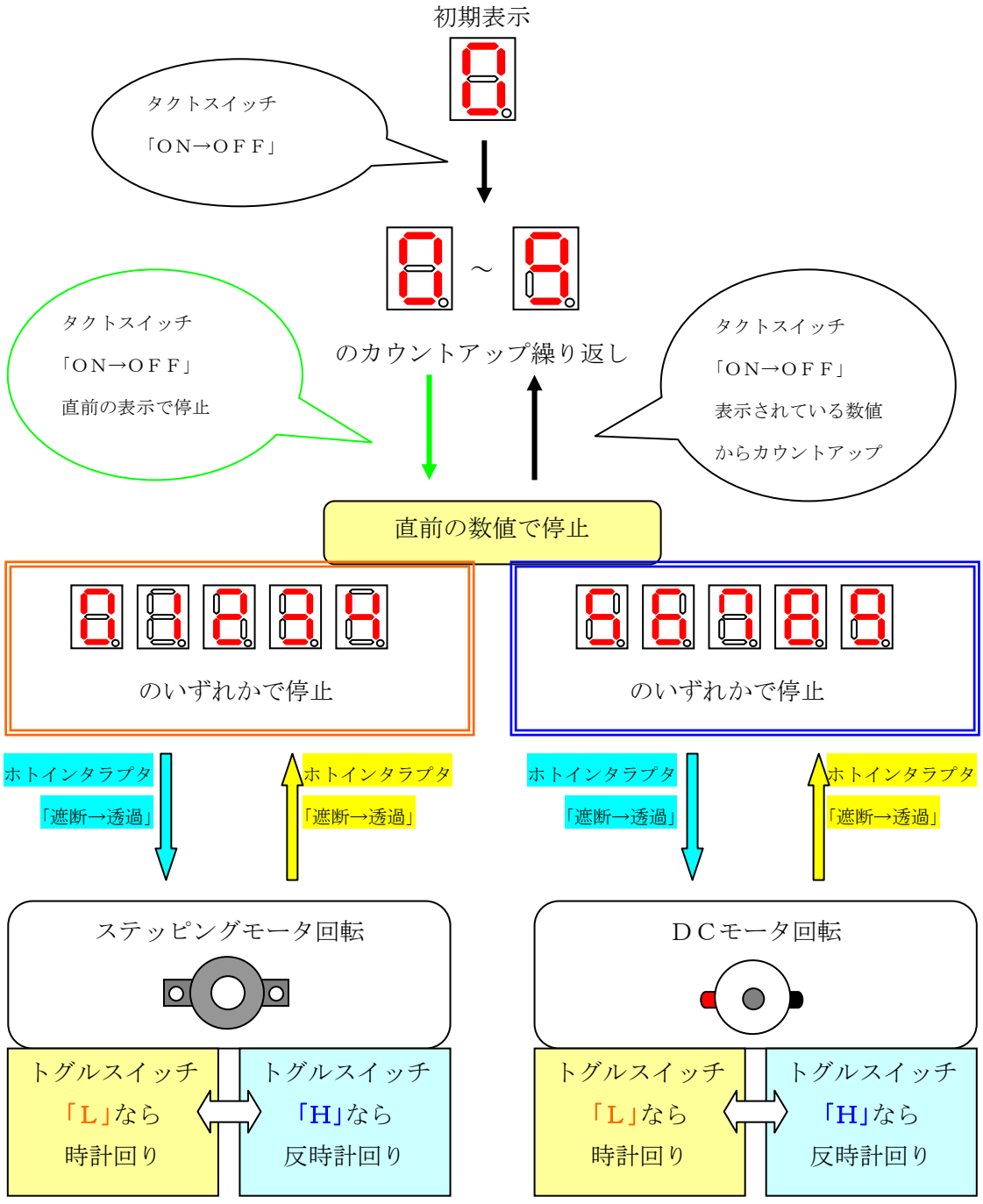
(c) (a)および(b)の動作は，ＤＣモータ回転中に何度でも切り替えられる。

(ｲ) トグルスイッチの状態にかかわらず，ホトインタラプタを再び「遮断→透過」すると，ＤＣモータは停止する。

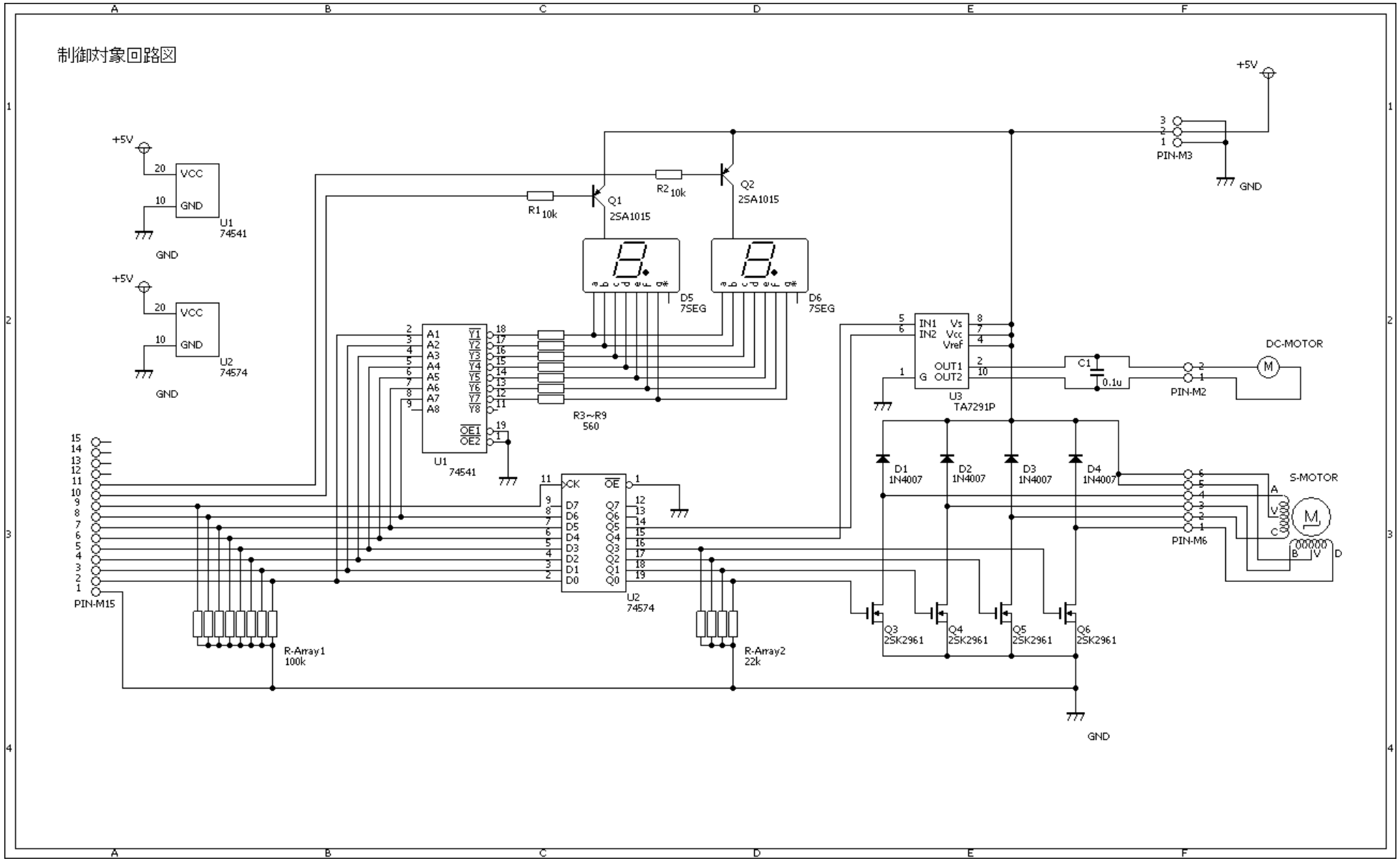
ウ　ステッピングモータおよびＤＣモータの動作および停止は，ホトインタラプタが**遮断された瞬間**に行われる。

(4) いずれかのモータが回転している間，停止している数値は，７セグメントＬＥＤに表示されている。

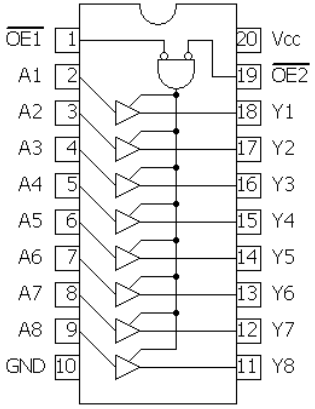
動作概要


初期表示
タクトスイッチ
「ON→OFF」
～
のカウントアップ繰り返し
タクトスイッチ
「ON→OFF」
直前の表示で停止
タクトスイッチ
「ON→OFF」
表示されている数値
からカウントアップ
直前の数値で停止
のいずれかで停止
のいずれかで停止
ホトインタラプタ
「遮断→透過」
ホトインタラプタ
「遮断→透過」
ホトインタラプタ
「遮断→透過」
ホトインタラプタ
「遮断→透過」
ステッピングモータ回転
ＤＣモータ回転
トグルスイッチ
「L」なら
時計回り
トグルスイッチ
「H」なら
反時計回り
トグルスイッチ
「L」なら
時計回り
トグルスイッチ
「H」なら
反時計回り

制御対象回路図


+5V
VCC
GND
U1
74541
U2
74574
Q1
2SA1015
Q2
2SA1015
R1 10k
R2 10k
D5
7SEG
D6
7SEG
R3~R9
560
A1
A2
A3
A4
A5
A6
A7
A8
Y1
Y2
Y3
Y4
Y5
Y6
Y7
Y8
OE1
OE2
CK
OE
D7
D6
D5
D4
D3
D2
D1
D0
Q7
Q6
Q5
Q4
Q3
Q2
Q1
Q0
PIN-M15
R-Array1
100k
R-Array2
22k
IN1
IN2
Vref
Vs
Vcc
OUT1
OUT2
G
U3
TA7291P
C1
0.1u
DC-MOTOR
PIN-M2
PIN-M3
D1
1N4007
D2
1N4007
D3
1N4007
D4
1N4007
S-MOTOR
PIN-M6
Q3
2SK2961
Q4
2SK2961
Q5
2SK2961
Q6
2SK2961

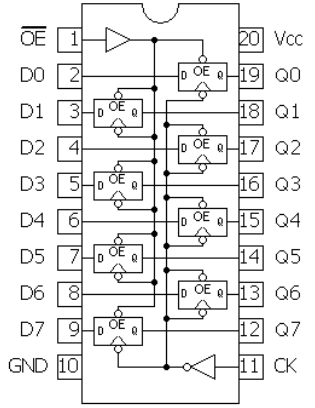
74HC541



Inputs			Output
$\overline{OE1}$	$\overline{OE2}$	An	$\overline{Yn}$
L	L	L	L
L	L	H	H
×	H	×	Z
H	×	×	Z

H: HIGH Voltege Level
L: Low Voltege Level
×: don't care
Z: high impeadance OFF-state

74HC574



Inputs			Output
$\overline{OE}$	CK	Dn	Qn
L	↑	H	H
L	↑	L	L
H	↓	×	Qn
H	×	×	Z

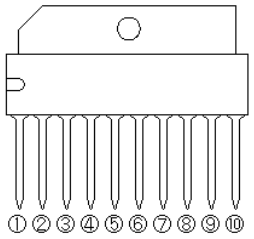
H: HIGH Voltege Level
L: Low Voltege Level
×: don't care
Z: high impeadance OFF-state
Qn: Hold

Stepping Moter

48M048C1U-N

DC Operating Voltage	5V
Step Angle	7.5°
Excitation method	4Phases Unipolar

TA7291P



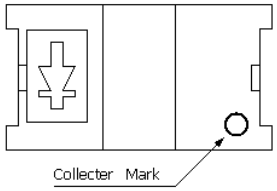
①	②	③	④	⑤	⑥	⑦	⑧	⑨	⑩
GND	OUT1	NC	Vref	IN1	IN2	Vcc	VS	NC	OUT2

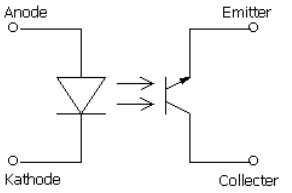
Inputs		Output		MODE
IN1	IN2	OUT1	OUT2	
L	L	Z	Z	STOP
H	L	H	L	CW / CCW
L	H	L	H	CCW / CW
H	H	L	L	BREAK

H: HIGH Voltege Level
L: Low Voltege Level
Z: high impeadance OFF-state

SG206

Top View





競技上の注意

1　入力回路は，表１に示された支給部品を使って製作すること。

表１　支給部品表

番号	部品名	型番等	個数
1	ユニバーサル基板	ＩＣＢ－293(サンハヤト)	1
2	透過型ホトインタラプタ	ＳＧ206(ＫＯＤＥＮＳＨＩ)	1
3	タクトスイッチ	青(秋月電子通商)	1
4	トグルスイッチ	秋月電子通商	1
5	ピンヘッダ	ストレート５pin	1
6	抵抗	１／４Ｗ　180Ω	1
7	抵抗	１／４Ｗ　10kΩ	2
8	抵抗	１／４Ｗ　22kΩ	1
9	ネジ	Ｍ３　10ｍｍ	4
10	ナット	Ｍ３	4
11	平ワッシャ	Ｍ３　大	4
12	平ワッシャ	Ｍ３　小	4
13	バネワッシャ	Ｍ３	4
14	ゴム足	ＢＵ－６９２－Ａ(サトーパーツ)	4
15	スズメッキ線	φ0.5	
16	鉛フリーハンダ	Sn-3.0Ag-0.5Cu　φ0.6	
17	トグルスイッチH側シール	青　丸型	1

2　制御プログラムの提出用ソースリストは，次のようなファイル名を付けること。

表２　制御プログラムのファイル名

課題	ファイル名の例
1	kadai1.c　kadai1.asm　kadai1.txt　等
2	kadai2.c　kadai2.asm　kadai2.txt　等
3	kadia3.c　kadia3.asm　kadai3.txt　等
…	…

第 10 回高校生ものづくりコンテスト全国大会

電子回路組立部門
プログラム解説マニュアル

第 1.01 版

2011.02.14

ルネサスマイコンカーラリー事務局

注 意 事 項 (rev.3.1R)

著作権

・本マニュアルに関する著作権はルネサスマイコンカーラリー事務局に帰属します。
・本マニュアルは著作権法および、国際著作権条約により保護されています。

禁止事項

ユーザーは以下の内容を行うことはできません。
・第三者に対して、本マニュアルを販売、販売を目的とした宣伝、使用、営業、複製などを行うこと
・第三者に対して、本マニュアルの使用権を譲渡または再承諾すること
・本マニュアルの一部または全部を改変、除去すること
・本マニュアルを無許可で翻訳すること
・本マニュアルの内容を使用しての、人命や人体に危害を及ぼす恐れのある用途での使用

転載、複製

本マニュアルの転載、複製については、文書によるルネサスマイコンカーラリー事務局の事前の承諾が必要です。

責任の制限

本マニュアルに記載した情報は、正確を期すため、慎重に制作したものですが万一本マニュアルの記述誤りに起因する損害が生じた場合でも、ルネサスマイコンカーラリー事務局はその責任を負いません。

その他

本マニュアルに記載の情報は本マニュアル発行時点のものであり、ルネサスマイコンカーラリー事務局は、予告なしに、本マニュアルに記載した情報または仕様を変更することがあります。製作に当たりましては、最新の内容を確認いただきますようお願いします。

連絡先

(株)ルネサスソリューションズ　ルネサスマイコンカーラリー事務局
〒162-0824　東京都新宿区揚場町 2-1　軽子坂MNビル
TEL　(03)-3266-8510
E-mail:official@mcr.gr.jp

目　次

1. 概要

本マニュアルは、第 10 回高校生ものづくりコンテスト全国大会の電子回路組立部門の課題について解説したマニュアルです。電子回路組立部門には、回路製作（半田付け）と課題のプログラム作成がありますが、本マニュアルでは主に、プログラムについて説明します。

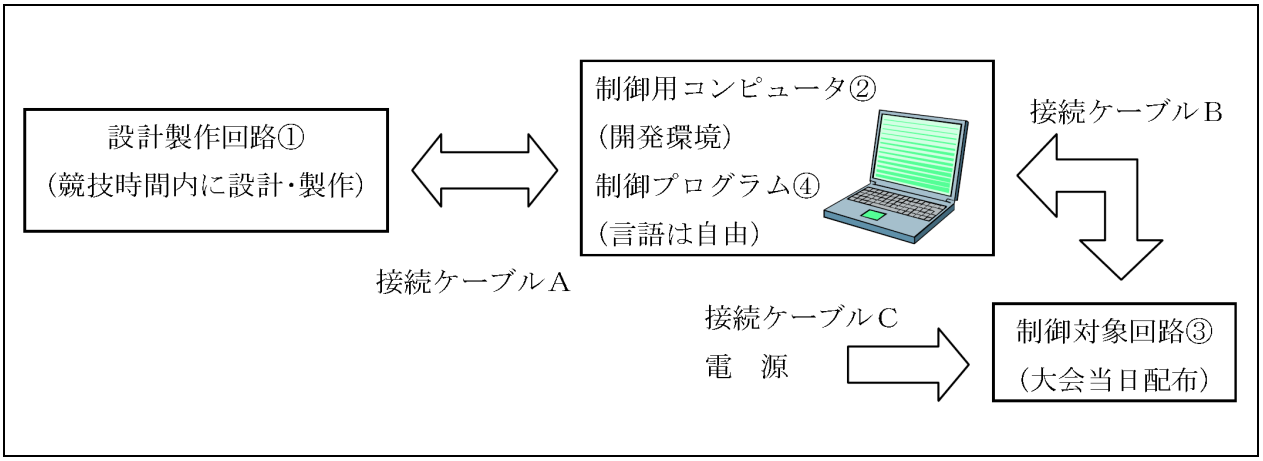
高校生ものづくりコンテストについては、全国工業高等学校長協会のホームページ（アドレス：http://www.zenkoukyo.or.jp/）を参照してください。

▲競技中の様子

2. 構成

2.1 全体構成

全体の接続構成を下図に示します。



▲事前配付資料より抜粋

設計製作回路①	部品を支給され、競技時間内に製作する回路です。事前配付資料には、下記のように記載されています。 **大会当日に示す設計仕様に基づく電子回路を設計し、ユニバーサル基板を用いて電子回路基板を製作する。配線はスズメッキ線を使用し、設計製作回路は以下の部品を使用する。** **①ユニバーサル基板（サンハヤト ICB293 相当品）** **②ホトインタラプタ、スイッチ、コネクタ、0.5φスズメッキ線等**
制御用コンピュータ②	制御用コンピュータは競技者が自由に構築することができます。事前配付資料には、下記のように記載されています。 **開発環境を含め全て持参する。コンピュータの性能・形状等に制限はない。** 実際は、ルネサス エレクトロニクス製マイコン(H8 など)、マイクロチップ・テクノロジー製マイコン(PIC など)、ポケットコンピュータなどが使われています。ただ、ポケットコンピュータは、A/D 変換機能を使った課題が 2008 年度に出題されてからほとんど使われなくなりました。 今回の制御用コンピュータ②の構成を、下記に示します。 パソコン:WindowsXP、または WindowsVista、RS232C 搭載（USB 変換ケーブル可） マイコンボード:マイコンカーラリー承認ボード　RY3048Fone ボード 開発環境:ルネサス統合開発環境 開発言語:C 言語 マイコンボードは、ジャパンマイコンカーラリーの承認ボードである「RY3048Fone」ボードを使用します。本ボードは、ルネサス エレクトロニクス製の H8/3048F-ONE マイコンを搭載したボードです。

<table>
<tr><td>制御対象回路③</td><td>大会当日に配布される基板です。事前配付資料には、下記のように記載されています。

大会当日に競技実行委員から配布する。制御対象回路には制御対象駆動回路と制御対象が含まれている。なお、制御対象としては、次のようなものが考えられる。
①LED
②7セグメントLED
③DCモータ
④ステッピングモータ等</td></tr>
<tr><td>制御プログラム④</td><td>事前配付資料には、下記のように記載されています。

大会当日に提示する仕様に基づいたプログラムを作成する。使用する言語は自由である。なお、プログラムの仕様例として、次のようなものがある。
①ストップウォッチのプログラム
②回転制御のプログラム

今回は、ルネサス統合開発環境を使い、C 言語でプログラムを行います。</td></tr>
<tr><td>接続ケーブル A</td><td>事前配付資料で、設計製作回路①側のピン配置が決められており、下記のように記載されています。

ICピッチ１列5ピンコネクタ・メス（ストレートピンヘッダ ハウジング）
○ ○ ○ ○ ○
① ② ③ ④ ⑤
<table><tr><td>①</td><td>ＧＮＤ</td><td>②</td><td>５Ｖ</td><td>③</td><td>入力０</td></tr><tr><td>④</td><td>入力１</td><td>⑤</td><td>入力２</td><td></td><td></td></tr></table>
接続ケーブル A は、あらかじめ製作しておきます。</td></tr>
<tr><td>接続ケーブル B</td><td>事前配付資料で、制御対象回路③側のピン配置が決められており、下記のように記載されています。

ICピッチ１列１５ピンコネクタ・メス（ストレートピンヘッダ ハウジング）
○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○
① ② ③ ④ ⑤ ⑥ ⑦ ⑧ ⑨ ⑩ ⑪ ⑫ ⑬ ⑭ ⑮
<table><tr><td>①</td><td>ＧＮＤ</td><td>②</td><td>出力０</td><td>③</td><td>出力１</td><td>④</td><td>出力２</td><td>⑤</td><td>出力３</td></tr><tr><td>⑥</td><td>出力４</td><td>⑦</td><td>出力５</td><td>⑧</td><td>出力６</td><td>⑨</td><td>出力７</td><td>⑩</td><td>出力８</td></tr><tr><td>⑪</td><td>出力９</td><td>⑫</td><td>ＮＣ</td><td>⑬</td><td>ＮＣ</td><td>⑭</td><td>ＮＣ</td><td>⑮</td><td>ＮＣ</td></tr></table>
接続ケーブル B は、あらかじめ製作しておきます。</td></tr>
<tr><td>接続ケーブル C</td><td>事前配付資料で、制御対象回路③側のピン配置が決められており、下記のように記載されています。

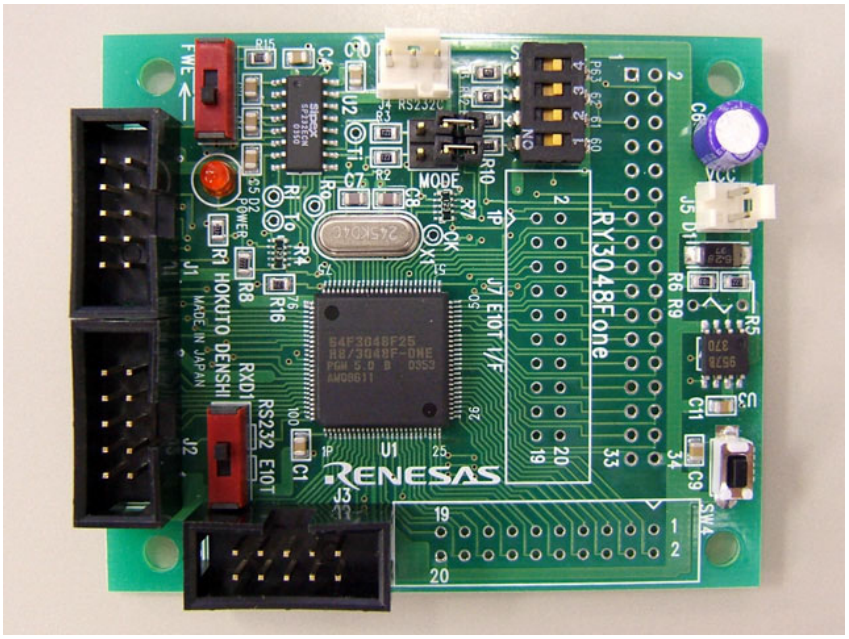
ICピッチ１列3ピンコネクタ・メス（ストレートピンヘッダ ハウジング）
○ ○ ○
① ② ③
<table><tr><td>①</td><td>ＧＮＤ</td><td>②</td><td>５Ｖ</td><td>③</td><td>ＧＮＤ</td></tr></table>
接続ケーブル C は、あらかじめ製作しておきます。</td></tr>
</table>

2.2　制御用コンピュータ②

本マニュアルでは、ジャパンマイコンカーラリーの承認ボードである「RY3048Fone」ボードを使用します。本ボードは、ルネサス エレクトロニクス製の H8/3048F-ONE マイコンを搭載したボードです。詳しい仕様やサンプルプログラムは、マイコンカーラリーサイトにある「H8/3048F-ONE 実習マニュアル(ルネサス統合開発環境版)」を参照ください。

H8/3048F-ONE 実習マニュアル(ルネサス統合開発環境版)は、
http://www.mcr.gr.jp/tech/download/main01.html
↓
マイコンに関する資料(H8 編)
より、ダウンロードできます。

マイコンボードの購入方法は、
マイコンカーラリー販売サイト URL:http://www2.himdx.net/mcr/
に記載されています。



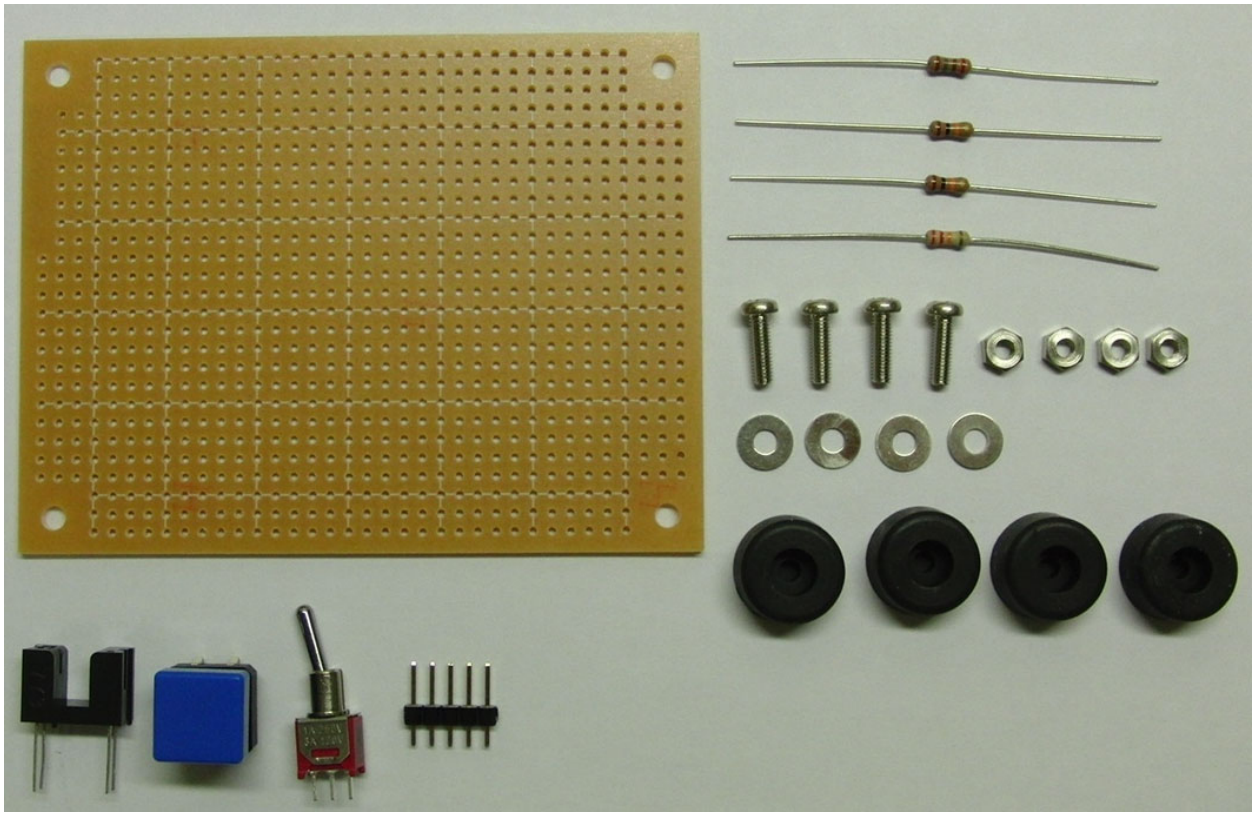
▲RY3048Fone ボード（マイコンはルネサス エレクトロニクス製の H8/3048F-ONE）

2.3　設計製作回路①

設計製作回路①は部品が支給され、競技時間内に競技者が製作します。支給部品を下記に示します。

番号	部品名	型番等	個数
1	ユニバーサル基板	ＩＣＢ－293(サンハヤト)	1
2	透過型ホトインタラプタ	ＳＧ206(ＫＯＤＥＮＳＨＩ)	1
3	タクトスイッチ	青(秋月電子通商)	1
4	トグルスイッチ	秋月電子通商	1
5	ピンヘッダ	ストレート５pin	1
6	抵抗	１／４Ｗ　180Ω	1
7	抵抗	１／４Ｗ　10kΩ	2
8	抵抗	１／４Ｗ　22kΩ	1
9	ネジ	Ｍ３　10ｍｍ	4
10	ナット	Ｍ３	4
11	平ワッシャ	Ｍ３　大	4
12	平ワッシャ	Ｍ３　小	4
13	バネワッシャ	Ｍ３	4
14	ゴム足	ＢＵ－６９２－Ａ(サトーパーツ)	4
15	スズメッキ線	φ0.5	
16	鉛フリーハンダ	Sn-3.0Ag-0.5Cu　φ0.6	
17	トグルスイッチH側シール	青　丸型	1

▲大会当日配付資料より抜粋



▲大会当日支給された部品

5 ピンのヘッダピンの接続は指定されています。接続内容を下記に示します。

ＩＣピッチ１列５ピン

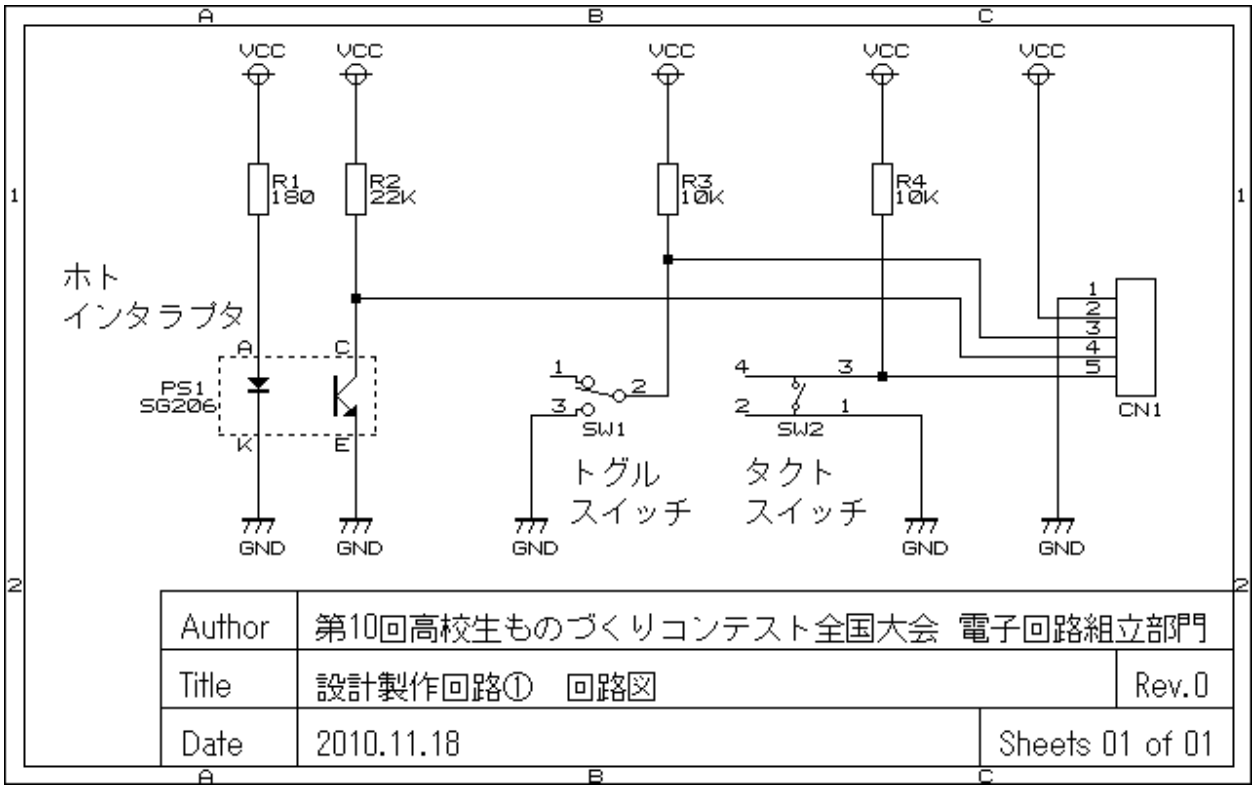
○	○	○	○	○
①	②	③	④	⑤

①	ＧＮＤ	②	５Ｖ	③	ＴＧＳ
④	Ｐ　Ｓ	⑤	Ｔ　Ｓ		

ＴＧＳ：トグルスイッチ
Ｐ　Ｓ：ホトインタラプタ(透過型)
Ｔ　Ｓ：タクトスイッチ

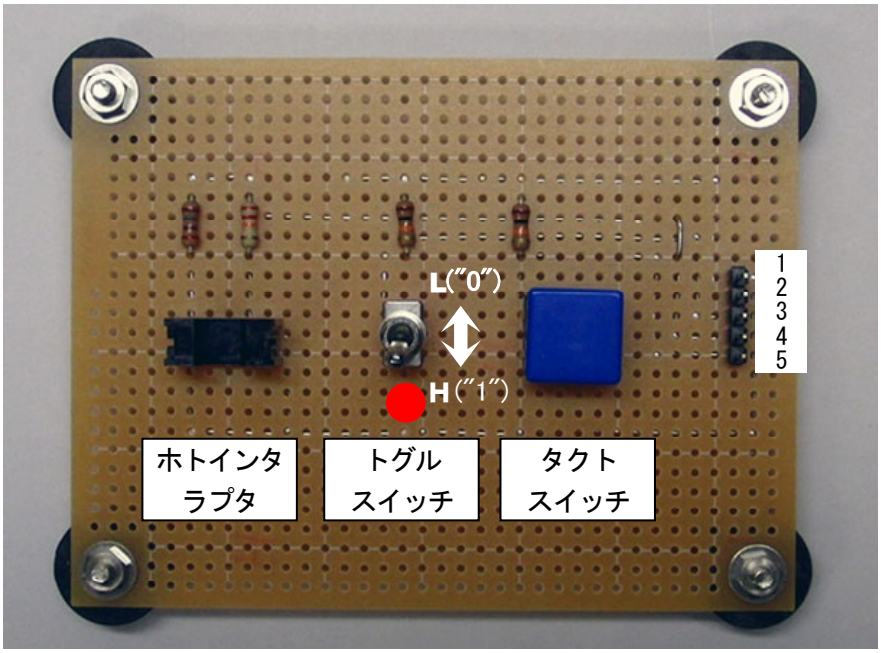
▲大会当日配付資料より抜粋

回路図はありませんので、競技者は部品から回路を予想して組み立てます。
回路図の作成例を下記に示します。



▲回路図　作成例

完成した基板の製作例を下記に示します。**基板製作後に回路図を書いている人がいましたが、先に回路図を書き、それに基づき半田付けするのが正しい順序です。**

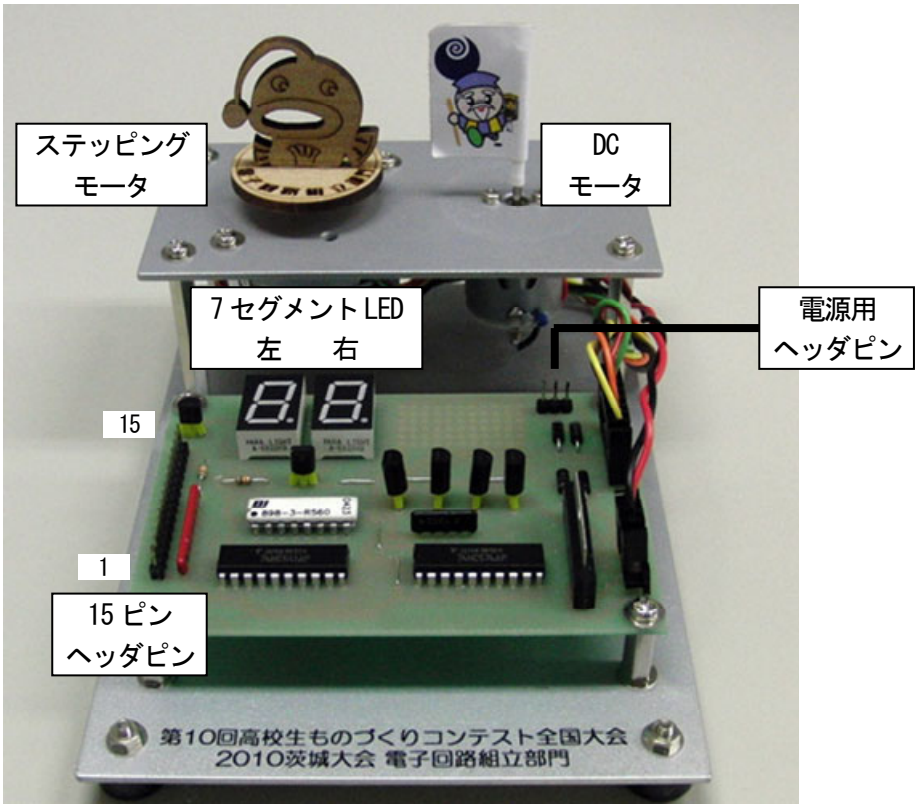


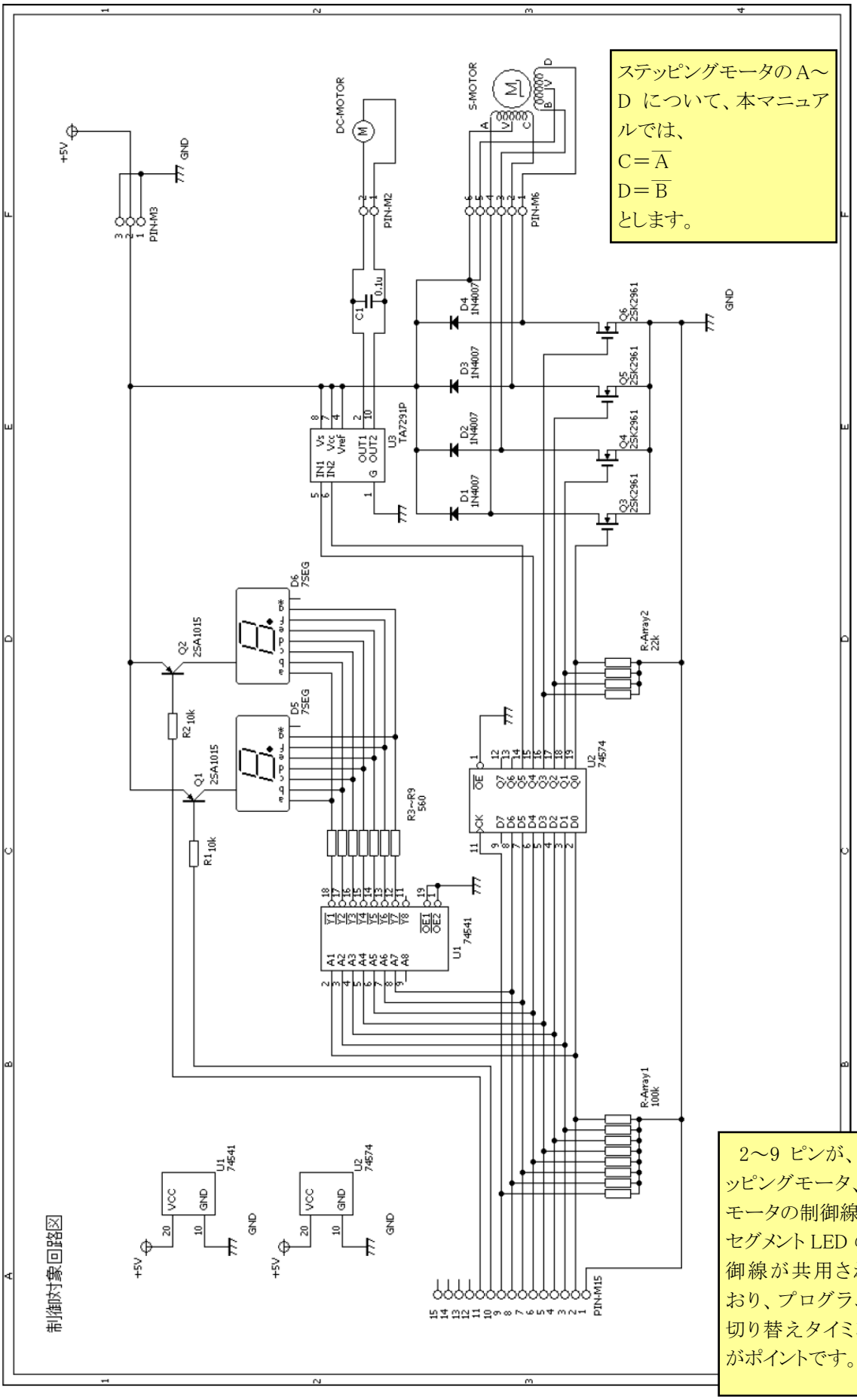
▲基板製作例

平ワッシャの位置、ナットの向きにも気をつけて組み立てましょう。細かいことですが、減点の対象となります。

2.4　制御対象回路③

制御対象回路③は、大会当日に配布される基板です。
制御対象回路③の写真を下記に、回路図を次ページに示します。





ステッピングモータの A～D について、本マニュアルでは、
C＝$\overline{A}$
D＝$\overline{B}$
とします。

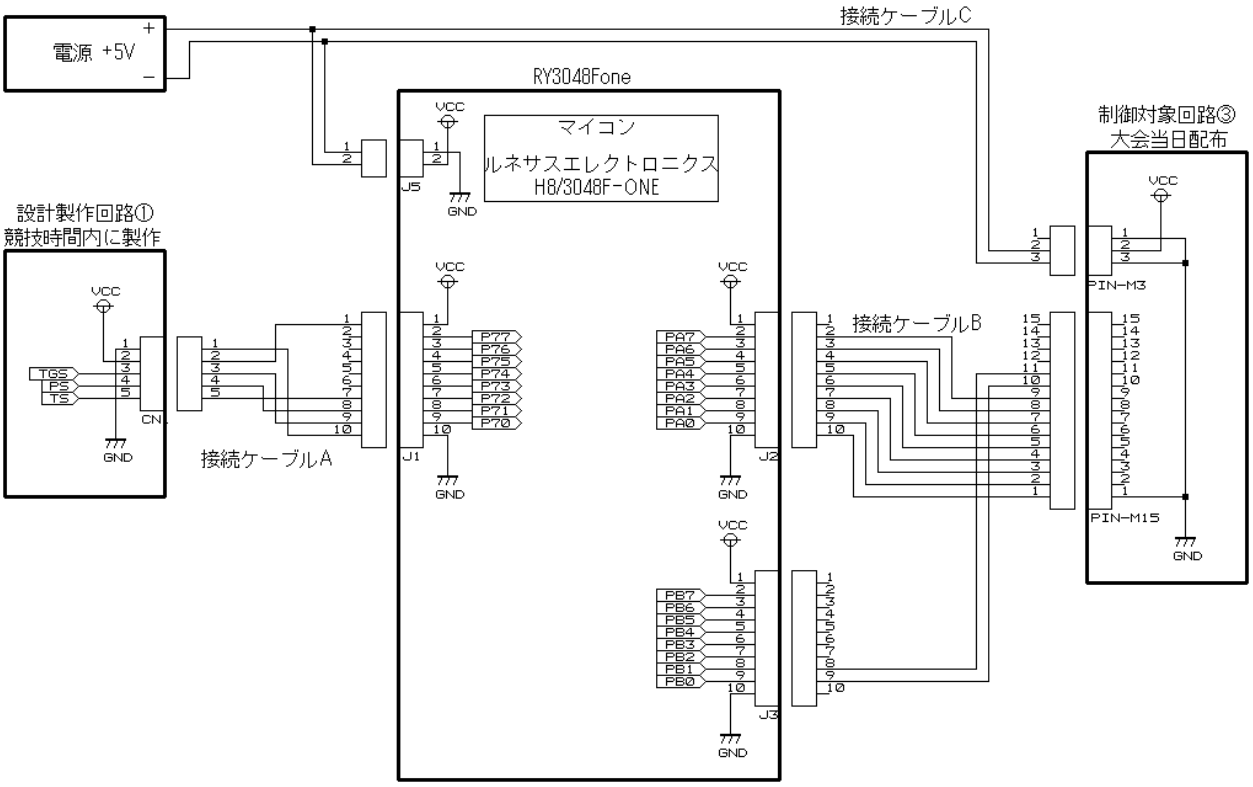
2～9 ピンが、ステッピングモータ、DC モータの制御線と 7 セグメント LED の制御線が共用されており、プログラムの切り替えタイミングがポイントです。

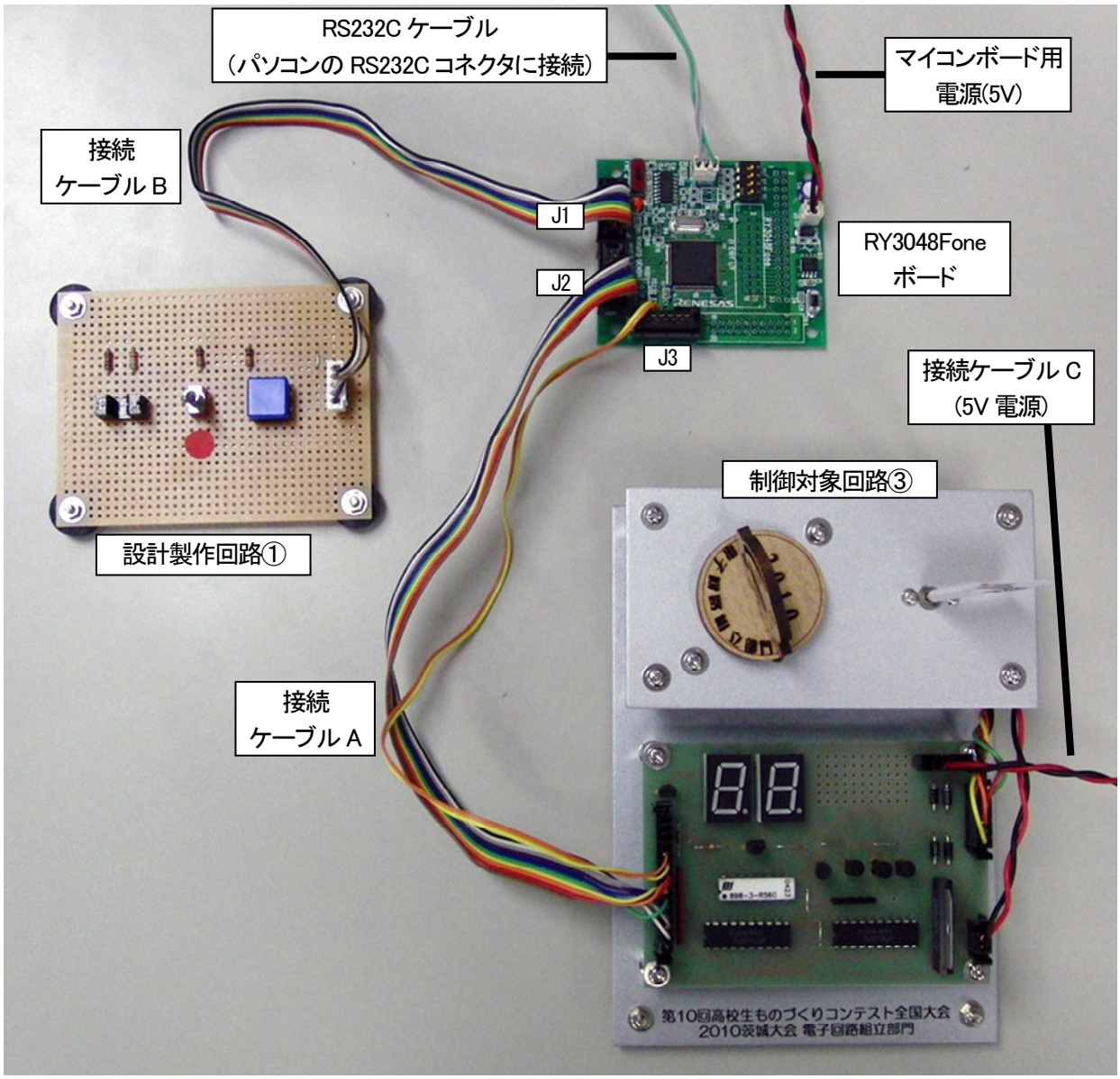
▲大会当日配付資料より

2.5 接続

今回のプログラムでは、下記のように結線されているものとします。

設計製作回路①（競技時間内に製作）、制御対象回路③（大会当日配布）、RY3048Fone ボード（持ち込み）との接続を下記に示します。接続ケーブル A、接続ケーブル B、接続ケーブル C は事前に製作しておきます。




RS232C ケーブル
（パソコンの RS232C コネクタに接続）
マイコンボード用
電源(5V)
接続
ケーブル B
J1
J2
J3
RY3048Fone
ボード
接続ケーブル C
(5V 電源)
制御対象回路③
設計製作回路①
接続
ケーブル A
第10回高校生ものづくりコンテスト全国大会
2010茨城大会 電子回路組立部門

▲接続例

3. プログラム

3.1 概要

課題のプログラムは 7 問あります。回路図作成、基板製作、プログラム作成を制限時間である 2 時間 30 分(150 分)以内で終わらせなければいけません。そのため、時間配分が重要になってきます。生徒にもよりますが、時間配分例を下記に示します。

項目		時間［分］
概要の把握(ざっと読む)		5
回路図作成		5
基板製作		20
プログラム	入力回路部分	20
	出力回路部分	50
	課題 1～7 の main 部分	50
合計		150

3.2 開発環境

本マニュアルでは、ルネサス統合開発環境(無償評価版)を使用します。

ルネサス統合開発環境やその他ファイルの入手、インストール、操作方法については、マイコンカーラリーサイトにある「ルネサス統合開発環境 操作マニュアル導入編」を参照ください。

ルネサス統合開発環境 操作マニュアル導入編は、
http://www.mcr.gr.jp/tech/download/main01.html
↓
マイコンに関する資料(H8 編)
より、ダウンロードできます。

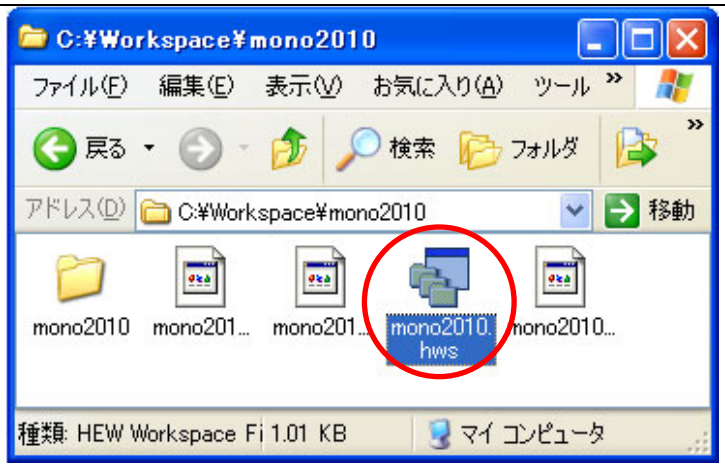
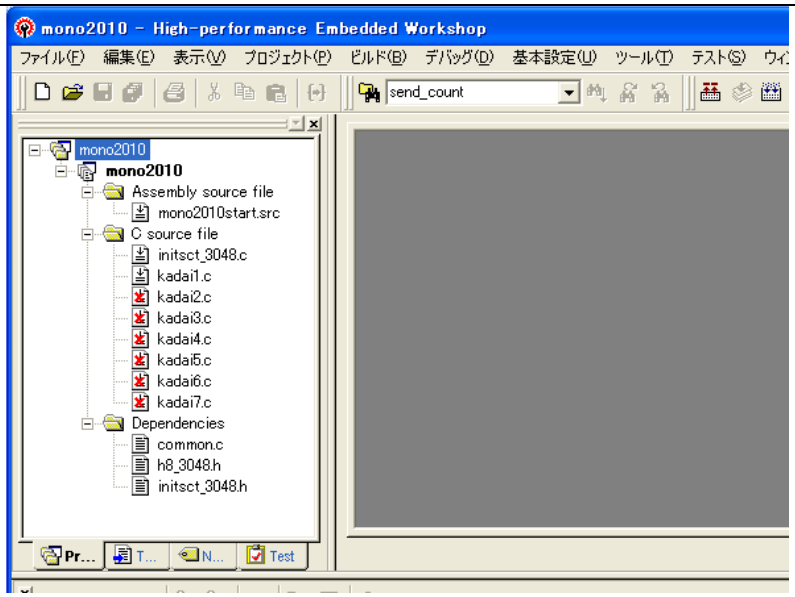
3.3 プログラムのダウンロード

「平成 22 年度　高校生ものづくりコンテスト電子回路組立部門の回答例プログラム」をダウンロードする手順を、下記に示します。

1	サンプルプログラム、書き込みソフトのダウンロード(H8編) 2010.10.07更新 サンプルプログラム、書き込みソフトのダウンロード(R8C編) 2010.10.07更新 マイコンに関する資料(H8編) 2009.05.25更新 マイコンに関する資料(R8C編) 2010.08.30更新 マイコンカー、オプションに関する資料(H8編) 2010.09.10更新 マイコンカー、オプションに関する資料(R8C編) 2010.09.01更新 実習に関する資料 2004.08.17更新 出版本に関する資料、その他資料 2010.04.01更新	http://www.mcr.gr.jp/tech/download/main01.html にアクセスします。 「出版本に関する資料、その他資料」をクリックします。
2	ダウンロード ～出版本に関する資料、その他資料～ 出版されている本に関するデータ、その他資料を掲載しています。 ●平成22年度　高校生ものづくりコンテスト電子回路組立部門の回答例プログラム 2011.02.01 全国工業高等学校長協会主催「平成22年度　高校生ものづくりコンテスト電子回路組立部門　全国大会」の回答例プログラムです。マイコンは、マイコンカーラリー承認ボードのRY3048Foneボード(H8/3048F-ONEマイコン)を使っています。マニュアルは、全国情報技術教育研究会のホームページに掲載されています。 →DOWNLOAD (EXE 約0.2MB)	「DOWNLOAD」をクリックして、ダウンロードします。
3		「workspace_monodukuri2010.exe」を実行して、サンプルプログラムをインストールします。デフォルトでは、「c:¥workspace」フォルダにインストールされます。

3.4 プログラム

ルネサス統合開発環境でのファイルの開き方、操作方法を説明します。

1		「C ドライブ→workspace→mono2010」フォルダにある、「mono2010.hws」を実行します。
2		ルネサス統合開発環境が立ち上がります。 左側にあるリストが、プログラムファイルになります。

プログラムが入っているファイルの種類と内容を、下記に示します。

ファイル名	詳細
h8_3048.h	H8/3048F-ONE マイコンのレジスタの定義をしているファイルです。全課題共通で使用します。 ファイルの場所：C:¥WorkSpace¥common¥h8_3048.h
mono2010start.src	マイコン起動時のプログラム（スタートアップルーチン）、ベクタアドレス（割り込み発生時の実行先の定義、今回はITU0を1msごとの割り込みとして使用します）が定義されているファイルです。全課題共通で使用します。 ファイルの場所：C:¥Workspace¥mono2010¥mono2010¥mono2010start.src
initsct_3048.c initsct_3048.h	H8/3048F-ONE マイコンを使う上で、メモリの初期化を行うルーチンが入っているファイルです。全課題共通で使用します。 ファイルの場所：C:¥Workspace¥common¥initsct_3048.c

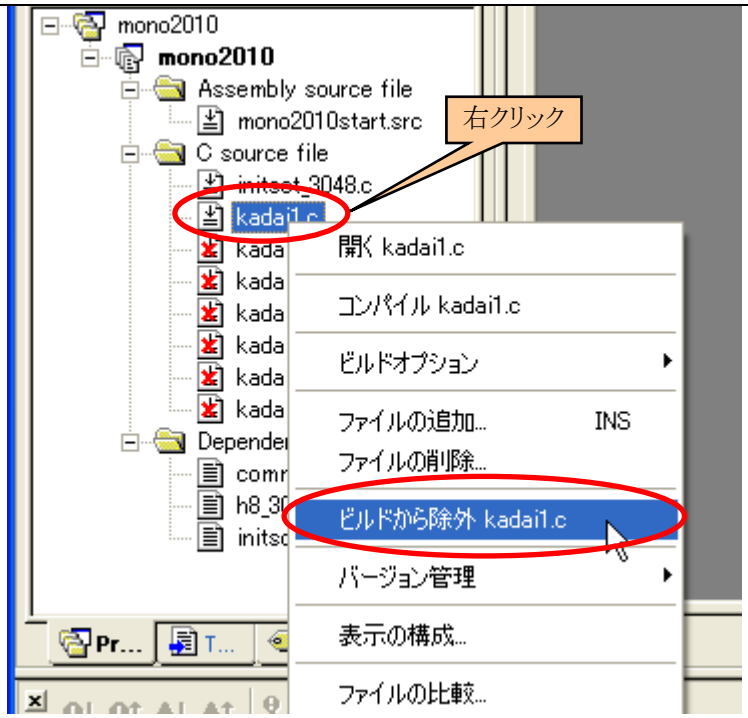
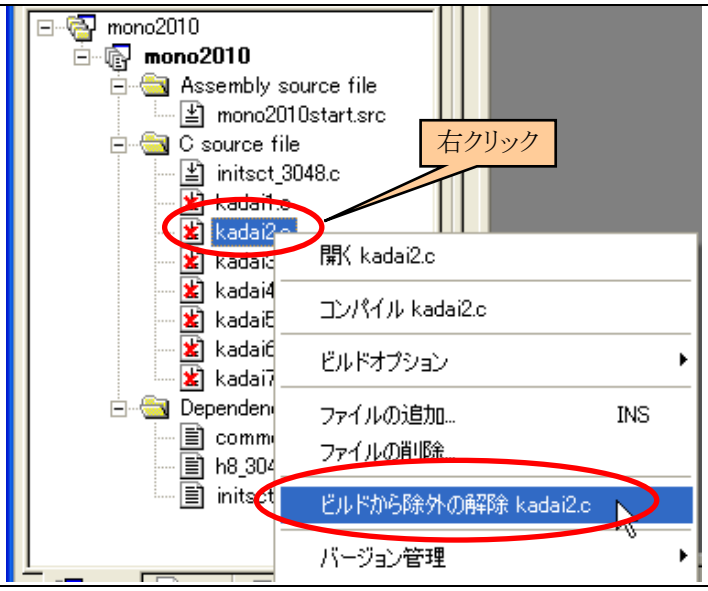
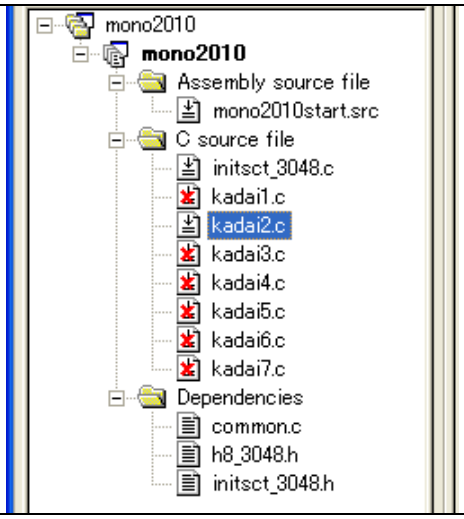
common.c	課題プログラムを作成する上で、ポートの入出力設定、入力回路部分のプログラム、出力回路部分のプログラムなど、全課題共通のプログラムをこのファイル内に入れておきます。 ファイルの場所：C:¥Workspace¥mono2010¥mono2010¥common.c
kadai1.c	課題 1 の回答例が入っているファイルです。 ファイルの場所：C:¥Workspace¥mono2010¥mono2010¥kadai1.c
kadai2.c	課題 2 の回答例が入っているファイルです。 ファイルの場所：C:¥Workspace¥mono2010¥mono2010¥kadai2.c
kadai3.c	課題 3 の回答例が入っているファイルです。 ファイルの場所：C:¥Workspace¥mono2010¥mono2010¥kadai3.c
kadai4.c	課題 4 の回答例が入っているファイルです。 ファイルの場所：C:¥Workspace¥mono2010¥mono2010¥kadai4.c
kadai5.c	課題 5 の回答例が入っているファイルです。 ファイルの場所：C:¥Workspace¥mono2010¥mono2010¥kadai5.c
kadai6.c	課題 6 の回答例が入っているファイルです。 ファイルの場所：C:¥Workspace¥mono2010¥mono2010¥kadai6.c
kadai7.c	課題 7 の回答例が入っているファイルです。 ファイルの場所：C:¥Workspace¥mono2010¥mono2010¥kadai7.c

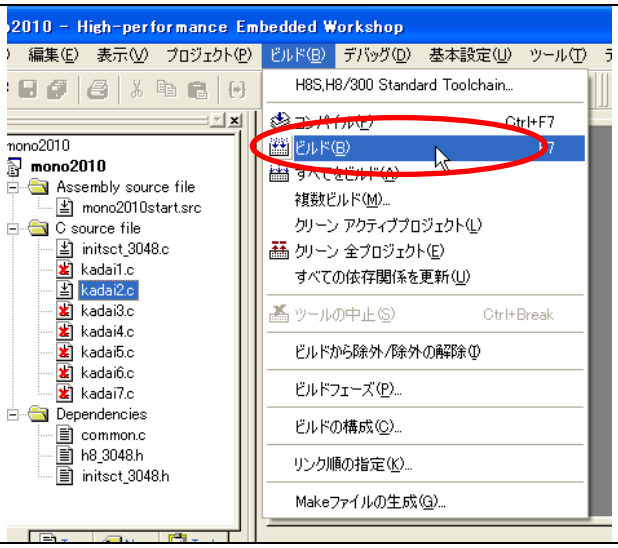
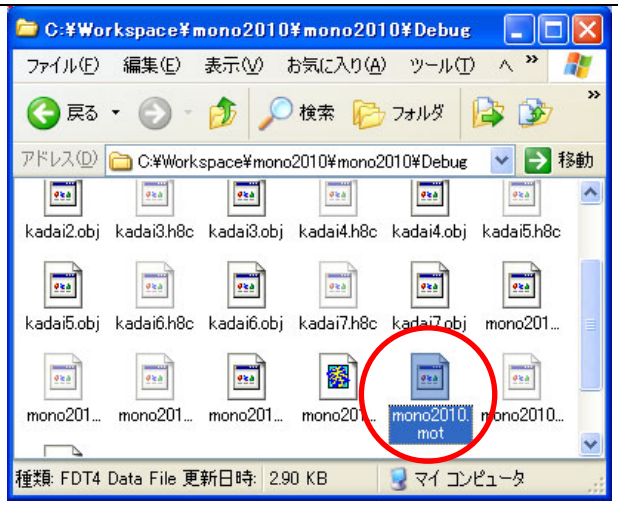
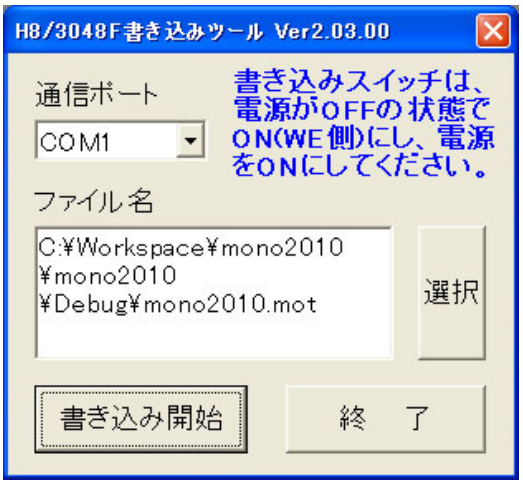
本プロジェクトには、kadai1.c～kadai7.c の課題ファイルが登録されていますが、この中で有効にできるのは 1 つだけです。例えば、課題 1 のときは、「kadai1.c」のみ有効にして、「kadai2.c～kadai7.c」はビルドから除外（ファイル左の赤い×マーク）にしておきます。

各課題のプログラムをビルドするときに、ビルドから除外するファイルを下記に示します。

ファイル名	課題 1	課題 2	課題 3	課題 4	課題 5	課題 6	課題 7
mono2010start.src	○	○	○	○	○	○	○
initsct_3048.c	○	○	○	○	○	○	○
common.c	○	○	○	○	○	○	○
kadai1.c	○	×	×	×	×	×	×
kadai2.c	×	○	×	×	×	×	×
kadai3.c	×	×	○	×	×	×	×
kadai4.c	×	×	×	○	×	×	×
kadai5.c	×	×	×	×	○	×	×
kadai6.c	×	×	×	×	×	○	×
kadai7.c	×	×	×	×	×	×	○

○：有効　×：ビルドから除外するファイル

3		例えば、課題 2 のファイルである、「kadai2.c」をビルド（MOT ファイルの作成）したい場合、次の操作を行います。 「kadai1.c」の上で右クリックし、「ビルドから除外」をクリックします。
4		「kadai2.c」の上で右クリックし、「ビルドから除外の解除」をクリックします。
5		リストが、左画面のようになれば完了です。

6		「ビルド→ビルド」で、kadai2.c などの登録されているファイルがビルド(アセンブル、コンパイル、リンク)され最終ファイル(MOT ファイル)ができあがります。
7		MOT ファイルは、 「C:¥Workspace¥mono2010¥mono2010¥Debug」フォルダ内にできます。
8		CPU Write などの書き込みソフトを使って、プログラムを書き込んでください。

3.5 H8/3048F-ONE 内蔵周辺機能の初期化

H8/3048F-ONE マイコンの内蔵周辺機能に関わる設定は、init 関数内で行います。init は、「initialize（イニシャライズ）」の略です。

3.5.1 ポートの入出力設定

(1) ポートの接続

ポートの接続を下記に示します。記入の無いビットは未接続、斜線の欄は端子が無いビットです。

ポート	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
1								
2								
3								
4								
5								
6					マイコンボード上のディップスイッチ 3（入力）	マイコンボード上のディップスイッチ 2（入力）	マイコンボード上のディップスイッチ 1（入力）	マイコンボード上のディップスイッチ 0（入力）
7						**設計製作回路① タクトスイッチ（入力）**	**設計製作回路① ホトインタラプタ（入力）**	**設計製作回路① トグルスイッチ（入力）**
8								
9					通信線（入力）		通信線（出力）	
A	**制御対象回路③ 9 ピン（出力）**	**制御対象回路③ 8 ピン（出力）**	**制御対象回路③ 7 ピン（出力）**	**制御対象回路③ 6 ピン（出力）**	**制御対象回路③ 5 ピン（出力）**	**制御対象回路③ 4 ピン（出力）**	**制御対象回路③ 3 ピン（出力）**	**制御対象回路③ 2 ピン（出力）**
B							**制御対象回路③ 11 ピン（出力）**	**制御対象回路③ 10 ピン（出力）**

(2) プログラム

ポートの入出力設定は、「P○DDR」レジスタで行い、○には 1～B まで入ります。ただし、ポート 7 は入力専用のため、「P7DDR」はありません。

入力にしたいビットは"0"、出力にしたいビットは"1"を設定します。未接続のビットの端子は出力にしておきます。今回の入出力設定プログラムを、下記に示します。

```
87 :      // ポートの入出力設定
88 :      P1DDR = 0xff;
89 :      P2DDR = 0xff;
90 :      P3DDR = 0xff;
91 :      P4DDR = 0xff;
92 :      P5DDR = 0xff;
93 :      P6DDR = 0xf0;                    // マイコンボード上の DIP SW
94 :      // P7 は入力専用                  // 設計製作回路①と接続
95 :      P8DDR = 0xff;
96 :      P9DDR = 0xf7;
97 :      PADDR = 0xff;                    // 制御対象回路③と接続
98 :      PBDDR = 0xff;                    // 制御対象回路③と接続
```

3.5.2 ITU0 の設定

今回は、H8/3048F-ONE の ITU というタイマのチャネル 0 を使って、1ms ごとに割り込みを発生させます。

詳しくは、マイコンカーラリーホームページのダウンロードサイト「http://www.mcr.gr.jp/tech/download/main01.html」→「マイコンに関する資料(H8 編)」→「H8/3048F-ONE 実習マニュアル(ルネサス統合開発環境版)」など、ITU について書かれている資料を参照してください。

プログラムを下記に示します。

```
100 :     // ITU0 1ms 毎の割り込み
101 :     ITU0_TCR = 0x20;                 // GRA=CNT でカウンタクリア
102 :     ITU0_GRA = 24575;                // 割り込み周期 0.001/(1/24.576M)-1
103 :     ITU0_IER = 0x01;                 // 割り込み許可
104 :     ITU_STR  = 0x01;                 // ITU0 カウント開始
```

3.6 H8/3048F-ONE 内蔵周辺機能の初期化

3.6.1 割り込み許可

H8 マイコンは、個別の割り込み設定の他に、全体の割り込みを許可する設定が必要です。プログラムを下記に示します。

```
set_ccr( 0x00 );                    // 全体割り込み許可
```

ITU0 の割り込み設定を行った後で、全体の割り込みを許可してください。

3.6.2 割り込みプログラム

今回は interrupt_timer0 関数が 1ms ごとに実行されます。プログラムを下記に示します。

```
#pragma interrupt interrupt_timer0(vect=24)
void interrupt_timer0( void )
{
    ITU0_TSR &= 0xfe;                    // フラグクリア

    割り込みプログラム
}
```

「#pragma interrupt interrupt_timer0(vect=24)」は、『ベクタテーブル 24 番の割り込み関数は interrupt_timer0 関数です』、という意味です。24 番は、ITU0 の IMIA0 割り込みが発生したときに実行される番号です。よって、ITU0 の IMIA0 割り込み（今回設定した割り込み）が発生したときは、interrupt_timer0 関数が実行されます。

割り込みが発生したときに、ITU0_TSR の bit0 が″1″になります。割り込みプログラムの最初でこの bit を″0″にクリアしておきます。

3.7　入力信号を取り込むプログラム

入力信号は、設計製作回路①基板からの信号で、トグルスイッチ、ホトインタラプタ、タクトスイッチの 3 種類です。

3.7.1　入力信号のポート

各入力回路と RY3048Fone ボードの接続を、下記に示します。

入力回路	マイコンのポート	″1″が入力される条件	″0″が入力される条件	備考
トグルスイッチ（TGS）	P70	下側にしたとき	上側にしたとき	基板の製作例の赤●のシールが貼っている側が″1″(High)側、逆が″0″(Low)側です。今回の基板製作例では、下側が″1″になってしまいましたが、逆のほうが分かりやすいと思います。 プログラムでは、論理はそのままで使います。
ホトインタラプタ（PS）	P71	光りが遮断している状態（PS の間に障害物がある状態）	光りが透過している状態（PS の間に障害物が無い状態）	プログラムでは、論理はそのままで使います。 ″1″…遮断 ″0″…透過
タクトスイッチ（TS）	P72	離している状態 ※備考参照	押している状態 ※備考参照	「押している=″1″」の方が分かりやすいので、プログラムでは論理を反転して使用します。よって次のようになります。 ″1″…押している状態 ″0″…離している状態

※P70…ポート 7 の bit0 に接続されます。P71、P72 も同じ意味です。

3.7.2 課題での使われ方

入力信号を取り込むためのプログラムを作り始める前に、課題 1～7 が何を要求しているのか把握します。限られた時間しかありませんので早く main 関数内のプログラムを作りたいとは思いますが、入力信号を取り込むプログラム部分を作り誤ると main 関数のプログラムが大変になりますので、この部分はある程度時間をかけましょう。

課題 1～7 で操作する内容を、下表に示します。

	トグルスイッチ（TGS）	ホトインタラプタ(PS)	タクトスイッチ(TS)
課題 1	L、H		ON、OFF
課題 2	L、H	遮断、透過	
課題 3			ON→OFF するたびに （離したときの変化を検出）
課題 4	L、H	遮断、透過	ON、OFF
課題 5	L、H	遮断、透過	ON、OFF
課題 6	L、H		ON→OFF すると （押された瞬間に） ※課題では「押した直後に」という表記です。
課題 7	L、H	遮断→透過すると （遮断された瞬間に）	ON→OFF すると （押された瞬間に）
課題 1～7 の状態をまとめると	・L を検出する ・H を検出する	・遮断を検出する ・透過を検出する ・遮断された瞬間を検出する	・ON を検出する ・OFF を検出する ・離したときの変化を検出 ・押された瞬間を検出
プログラムの処理	・L を検出する ・H を検出する	・遮断を検出する ・透過を検出する ・遮断された瞬間を検出する	・ON を検出する ・OFF を検出する ・押された瞬間を検出 ※課題 3 は、離したときの変化を検出しなければいけませんが、これは main 関数内で検出します。

トグルスイッチ、ホトインタラプタ、タクトスイッチの状態を検出するプログラムは、1ms ごとに実行される割り込み処理に入れます。ただし、検出は 10ms ごとに行います。

ホトインタラプタやタクトスイッチの状態が変化した瞬間の検出は、瞬間を検出したときに 1 になる専用の変数を用意して、割り込みプログラム内で検出します。筆者としては、割り込みプログラムが少し複雑になりますが、それ以上に main 関数が簡単になるので、プログラム作成時間は短くなると考えています。

3.7.3　10msごとの処理

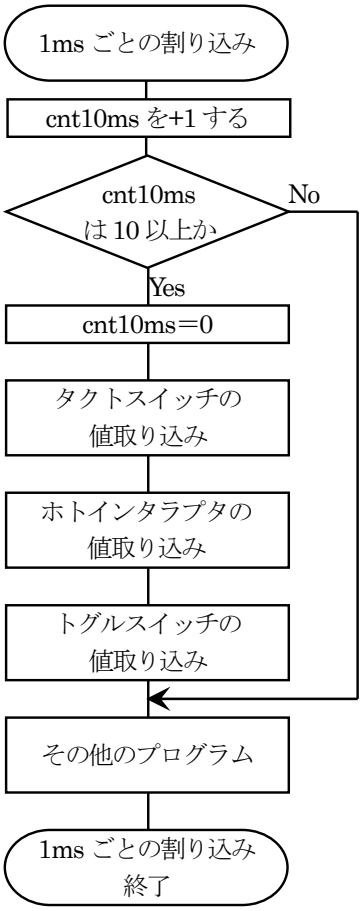
トグルスイッチ、ホトインタラプタ、タクトスイッチの状態は、10ms ごとにプログラムで読み込んでいます。
割り込みは 1ms ごとに発生するので、10 回に 1 回、処理すれば 10ms ごとに処理することになります。

(1) 変数

10ms ごとの処理で使用する変数を、下記に示します。

変数	内容
cnt10ms	1ms ごとに＋1して、10 になったら 10ms たったと判断します。

(2) フローチャート



(3) プログラム

```
// グローバル変数の宣言

  31 :  int     cnt10ms;                        // 10ms ごとの処理用

// プログラム

 107 :  ////////////////////////////////////////////////////////////////////////
 108 :  // ITU0 割り込み処理
 109 :  ////////////////////////////////////////////////////////////////////////
 110 :  #pragma interrupt interrupt_timer0(vect=24)
 111 :  void interrupt_timer0( void )
 112 :  {
 113 :      ITU0_TSR &= 0xfe;                   // フラグクリア
中略
 120 :      // スイッチ、ホトインタラプタは 10ms ごとにチェックする
 121 :      cnt10ms++;
 122 :      if( cnt10ms >= 10 ) {
 123 :          cnt10ms = 0;

              タクトスイッチの値取り込み処理

              ホトインタラプタの値取り込み

              トグルスイッチの値取り込み
          }

          その他のプログラム

 228 :  }
```

3.7.4 タクトスイッチの値取り込み処理

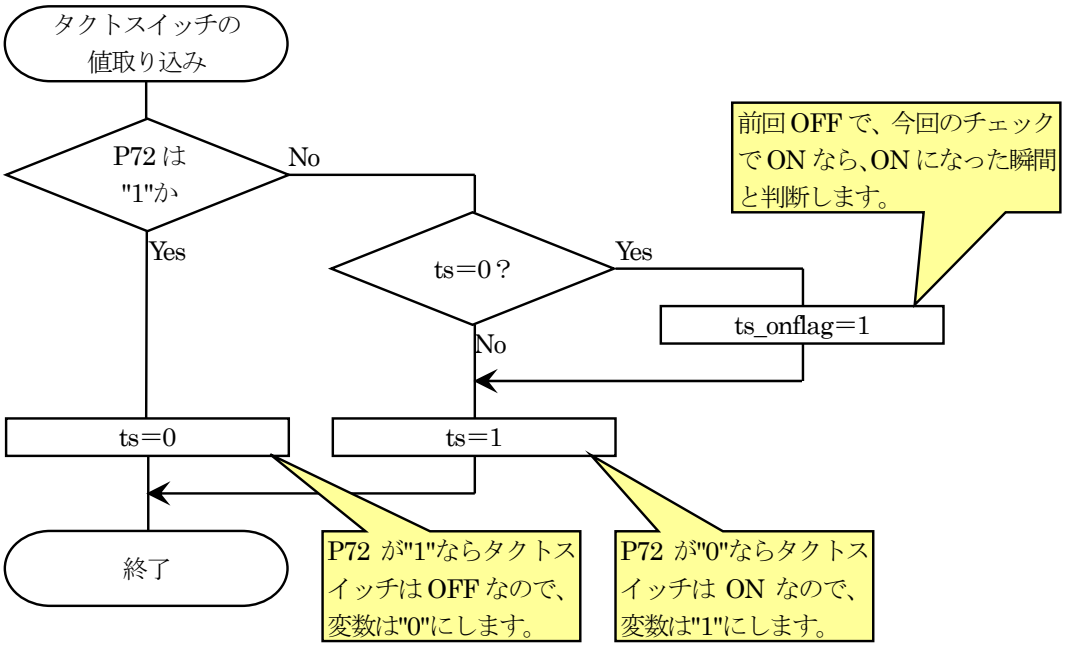
(1) 変数

タクトスイッチ関連で使用する変数を、下記に示します。

変数	内容
ts	タクトスイッチの状態が入っている変数です。 0:タクトスイッチ OFF 1:タクトスイッチ ON

ts_onflag	タクトスイッチが OFF から ON になった瞬間に 1 になる変数です。この変数が 1 になったら、タクトスイッチが押されたと判断します。
	if 文などで ts_onflag 変数が 1 になったことを検出したら、プログラムで ts_onflag 変数を 0 にしておきます。0 にしないと 1 のままなので、ずっと「ON になった瞬間」と判断してしまいます。
	
	例)
	`if( ts_onflag == 1 ) {`
	`  ts_onflag = 0;  // 1 を検出したので次のチェックに備え 0 にしておく`
	`  その他プログラム`
	`}`

(2) フローチャート



(3) プログラム例

プログラムを、下記に示します。

```
// グローバル変数の宣言

  33 :  ////////////////////////////////////////////////////////////
  34 :  // タクトスイッチの状態
  35 :  int     ts;                             // 0:OFF 1:ON
  36 :  int     ts_onflag;                      // OFF→ON になった瞬間に 1

// プログラム

 125 :          // タクトスイッチの値取り込み
 126 :          if( (P7DR & 0x04) == 0x04 ) {
 127 :              ts = 0;                     // OFF なら
 128 :          } else {
 129 :              if( ts == 0 ) {
 130 :                  ts_onflag = 1;          // ON になった瞬間なら 1
 131 :              }
 132 :              ts = 1;                     // ON なら
 133 :          }
```

3.7.5 ホトインタラプタの値取り込み処理

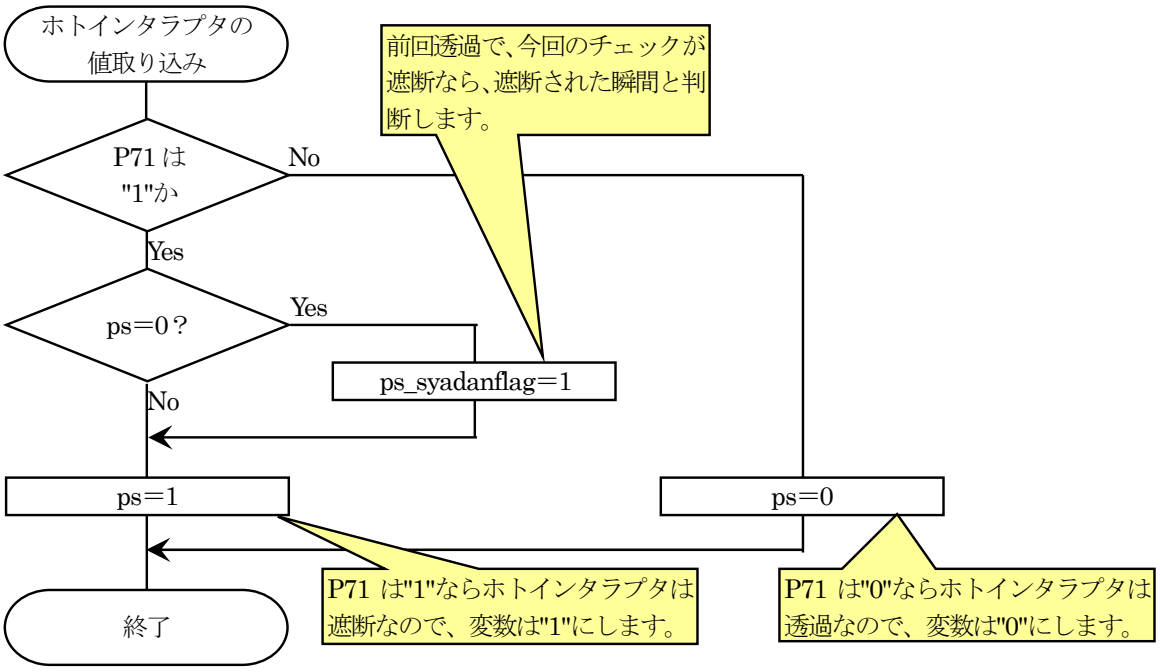
(1) 変数

ホトインタラプタ関連で使用する変数を、下記に示します。

変数	内容
ps	ホトインタラプタの状態が入っている変数です。 0:ホトインタラプタ透過 1:ホトインタラプタ遮断

ps_syadanflag	ホトインタラプタが透過から遮断された瞬間に 1 になる変数です。この変数が 1 になったら、ホトインタラプタが遮断されたと判断します。
	if 文などで ps_syadanflag 変数が 1 になったことを検出したら、プログラムで ps_syadanflag 変数を 0 にしておきます。0 にしないと 1 のままなので、ずっと「遮断された瞬間」と判断してしまいます。
	
	例)
	`if( ps_syadanflag == 1 ) {`
	`  ps_syadanflag = 0;  // 1 を検出したので次のチェックに備え 0 にしておく`
	`  その他プログラム`
	`}`

(2) フローチャート



(3) プログラム

```
// グローバル変数の宣言

  38 :  ////////////////////////////////////////////////////////////
  39 :  // ホトインタラプタの状態
  40 :  int     ps;                             // 0:透過 1:遮断
  41 :  int     ps_syadanflag;                  // 透過→遮断された瞬間に 1

// プログラム

 135 :          // ホトインタラプタの値取り込み
 136 :          if( (P7DR & 0x02) == 0x02 ) {
 137 :              if( ps == 0 ) {
 138 :                  ps_syadanflag = 1;      // 遮断した瞬間なら 1
 139 :              }
 140 :              ps = 1;                     // 遮断なら
 141 :          } else {
 142 :              ps = 0;                     // 透過なら
 143 :          }
```

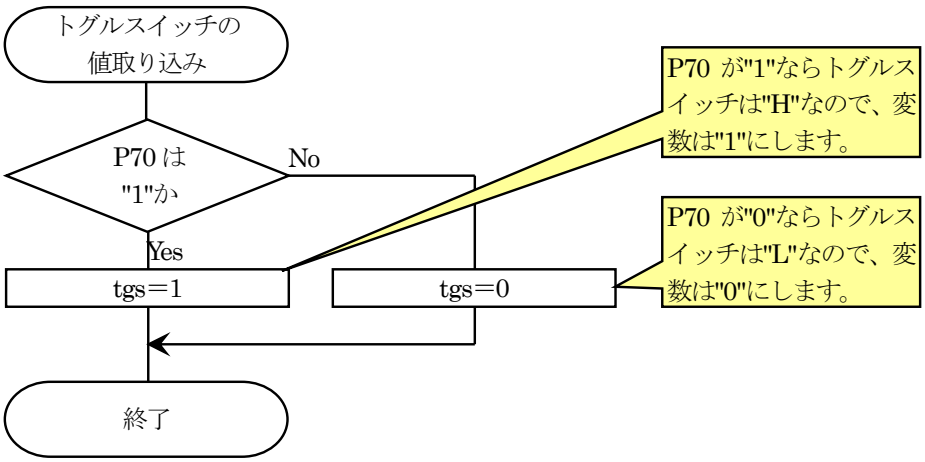
3.7.6 トグルスイッチの値取り込み処理

(1) 変数

トグルスイッチ関連で使用する変数を、下記に示します。

変数	内容
tgs	トグルスイッチの状態が入っている変数です。 0:トグルスイッチ"L" 1:トグルスイッチ"H"

(2) フローチャート



(3) プログラム

```
// グローバル変数の宣言
 43 :  ////////////////////////////////////////////////////////////
 44 :  // トグルスイッチの状態
 45 :  int     tgs;                            // 0:Low 1:High

// プログラム
145 :          // トグルスイッチの値取り込み
146 :          if( (P7DR & 0x01) == 0x01 ) {
147 :              tgs = 1;                    // "H"なら
148 :          } else {
149 :              tgs = 0;                    // "L"なら
150 :          }
```

3.8 出力回路へ信号を出力するプログラム

出力信号は、制御対象回路③基板へ出力する信号で、制御する対象は左側 7 セグメント LED、右側 7 セグメント LED、ステッピングモータ、DC モータの 4 種類です。

3.8.1 出力信号のポート

各出力回路と RY3048Fone ボードの接続を、下記に示します。

出力先		マイコンのポート	"1"を出力したときの状態	"0"を出力したときの状態	備考
左側7セグメントLED	アノードコモン端子の選択	PB0	全消灯	PA6～PA0 の値に従って点灯	7 セグメント LED はアノードコモンで、このコモン端子を ON/OFF するポートです。 PB0="1"なら各セグメント LED は PA6～PA0 の値に関係なく消灯します。
	各セグメントの LED の制御	PA6(g セグメント)	消灯	点灯	PB0="0"（点灯）のとき、PA6～PA0 でどの LED を点灯させるか制御します。 右側 7 セグメント LED と共通なので、PB0="0"のときだけ、左側 7 セグメント LED に出力したい値を設定します。
		PA5(f セグメント)	消灯	点灯	
		PA4(e セグメント)	消灯	点灯	
		PA3(d セグメント)	消灯	点灯	
		PA2(c セグメント)	消灯	点灯	
		PA1(b セグメント)	消灯	点灯	
		PA0(a セグメント)	消灯	点灯	
右側7セグメントLED	アノードコモン端子の選択	PB1	全消灯	PA6～PA0 の値に従って点灯	7 セグメント LED はアノードコモンで、このコモン端子を ON/OFF するポートです。 PB1="1"なら各セグメント LED は PA6～PA0 の値に関係なく消灯します。
	各セグメントの LED の制御	PA6(g セグメント)	消灯	点灯	PB1="0"（点灯）のとき、PA6～PA0 でどの LED を点灯させるか制御します。 左側 7 セグメント LED と共通なので、PB1="0"のときだけ、右側 7 セグメント LED に出力したい値を設定します。
		PA5(f セグメント)	消灯	点灯	
		PA4(e セグメント)	消灯	点灯	
		PA3(d セグメント)	消灯	点灯	
		PA2(c セグメント)	消灯	点灯	
		PA1(b セグメント)	消灯	点灯	
		PA0(a セグメント)	消灯	点灯	

（次のページへ続く）

出力先		マイコンのポート	″1″を出力したときの状態	″0″を出力したときの状態	備考
ステッピングモータ	74HC574 出力端子	PA7	″0″→″1″になった瞬間に PA6～PA0 の信号が、ステッピングモータ、DC モータに出力される	変化なし	PA5,PA4 は DC モータが接続されています。よって、PA5～PA0 に DC モータ、ステッピングモータへの出力値をセットした後で PA7 を″1″にします。″1″にした瞬間に 74HC574 の出力端子から PA5～PA0 の値が出力され、各モータが動作します。″1″にした後は、74HC574 によって出力値が保持されるので PA5～PA0 の値を変化させてもモータは動作し続けます。
	励磁コイルの選択	PA3($\overline{B}$)	$\overline{B}$コイルを励磁する	$\overline{B}$コイルを励磁しない	A→B→$\overline{A}$→$\overline{B}$の順に励磁するとステッピングモータは正回転し、その逆の順番に励磁すると逆回転します。1 パルスで 7.5 度動きます。1 周は 360÷7.5＝48 パルスです。 ※制御対象回路③の回路図にあるステッピングモータについて、コイルの記載は本マニュアルでは C を$\overline{A}$、D を$\overline{B}$としています。
		PA2($\overline{A}$)	$\overline{A}$コイルを励磁する	$\overline{A}$コイルを励磁しない	
		PA1(B)	B コイルを励磁する	B コイルを励磁しない	
		PA0(A)	A コイルを励磁する	A コイルを励磁しない	
DC モータ	74HC574 出力端子	PA7	″0″→″1″になった瞬間に PA6～PA0 の信号が、ステッピングモータ、DC モータに出力される	変化なし	ステッピングモータ部分を参照してください。
	DC モータの動作選択	PA5、PA4	″00″:ストップ ″01″:時計回りに回転 ″10″:反時計回りに回転 ″11″:ブレーキ		ストップは DC モータの端子間を解放、ブレーキは DC モータの端子間をショートさせます。

3.8.2 課題での使われ方

入力信号を取り込むためのプログラムを作り始める前に、課題は何を要求しているのか把握します。限られた時間しかありませんので早く main 関数内のプログラムを作りたいとは思いますが、入力信号を取り込むプログラム部分を作り誤ると main 関数のプログラムが大変になりますので、この部分はある程度時間をかけましょう。

課題 1～7 で制御する内容を、下表に示します。

※7 セグメント LED の消灯、ステッピングモータの停止、DC モータの停止は省略

	左側 7 セグメント LED	右側 7 セグメント LED	ステッピングモータ	DC モータ
課題 1	・H 表示	・L 表示		
課題 2			・時計回りに回転	・時計回りに回転
課題 3		・a,b,c,d,E,F 表示		
課題 4	・0～3 表示	・0、1 表示		
課題 5	・H 表示 ・0 表示	・L 表示 ・9 表示	・反時計回りに回転 ・**低速回転で**反時計回りに回転 ・**高速回転で**反時計回りに回転	
課題 6		・0～3 表示	・時計回りに回転 ・反時計回りに回転 **※7 セグメント LED の 0～3 の表示とモータの回転数を合わせる制御が必要**	
課題 7		・0～9 表示	・時計回りに回転 ・反時計回りに回転	・時計回りに回転 ・反時計回りに回転
課題 1～7 の状態をまとめると	・0～3 表示 ・H 表示	・0～9 表示 ・a～F 表示 ・L 表示	・反時計回りに回転 ・低速回転で反時計回りに回転 ・高速回転で反時計回りに回転 ・時計回りに回転	・時計回りに回転 ・反時計回りに回転
プログラムの処理	・0～9 表示 ・a～F 表示 ・H,L 表示	・0～9 表示 ・a～F 表示 ・H,L 表示	・時計回りに回転 このとき、低速、標準、高速回転を選択できるようにする ・反時計回りに回転 このとき、低速、標準、高速回転を選択できるようにする ・回転数をカウントする変数を用意する（正転時はプラス、逆転時はマイナスするようにする）	・時計回りに回転 ・反時計回りに回転

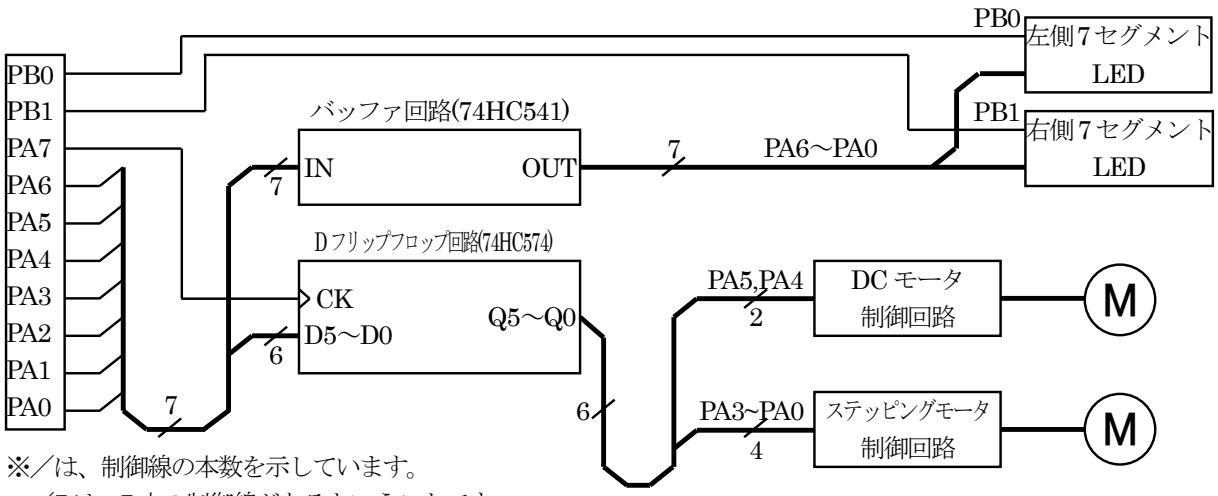
3.8.3　制御手順

制御対象回路は特に難しい回路ではありませんが、PA6～PA0 が次の回路を兼用しています。

・ステッピングモータ、DC モータ
・左側 7 セグメント LED の各セグメントの LED
・右側 7 セグメント LED の各セグメントの LED

よって、これらをどう混同しないよう制御するかがポイントです。
プログラムでは、1ms ごとの割り込み内でこれらの制御を行います。

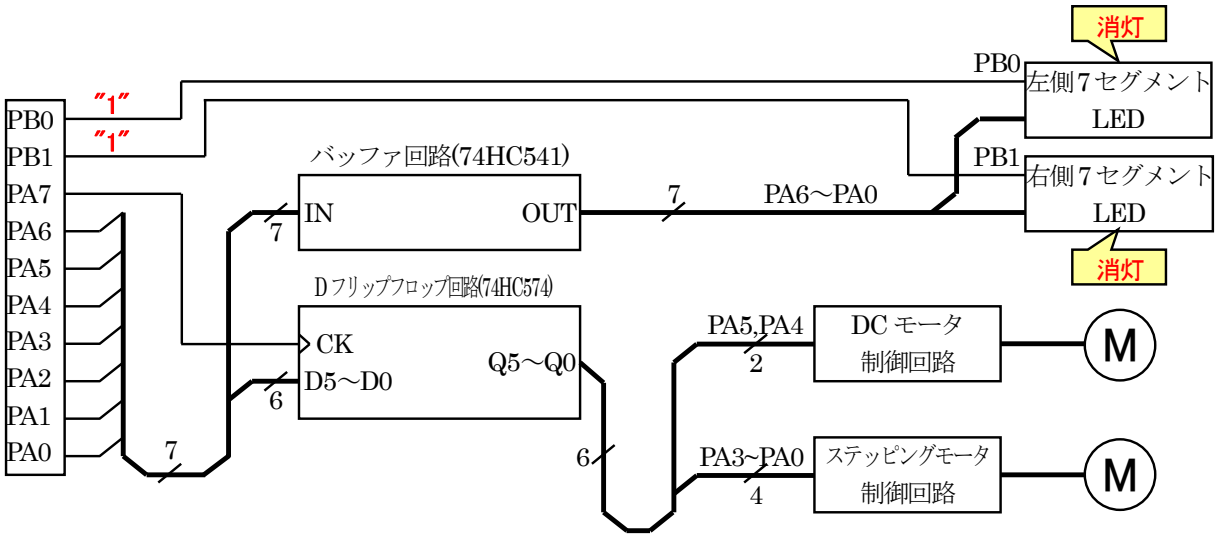
回路の簡略図を、下図に示します。



※／は、制御線の本数を示しています。
　／7 は、7 本の制御線があるということです。

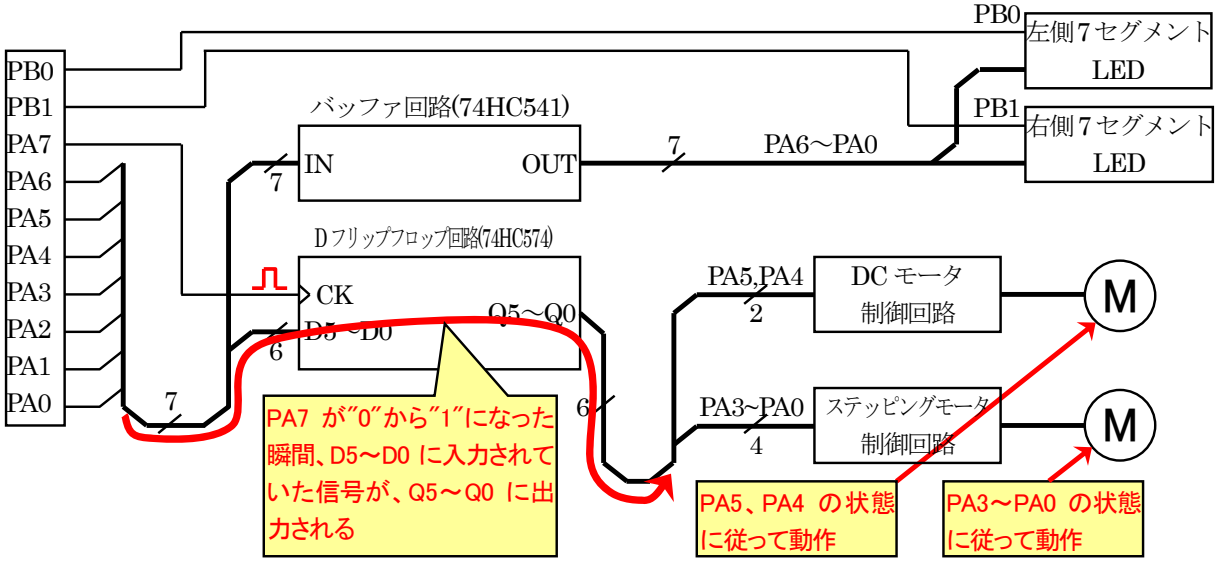
①7 セグメント LED の消灯

まず、PB1 と PB0 を″1″にして、7 セグメント LED を消灯させます。



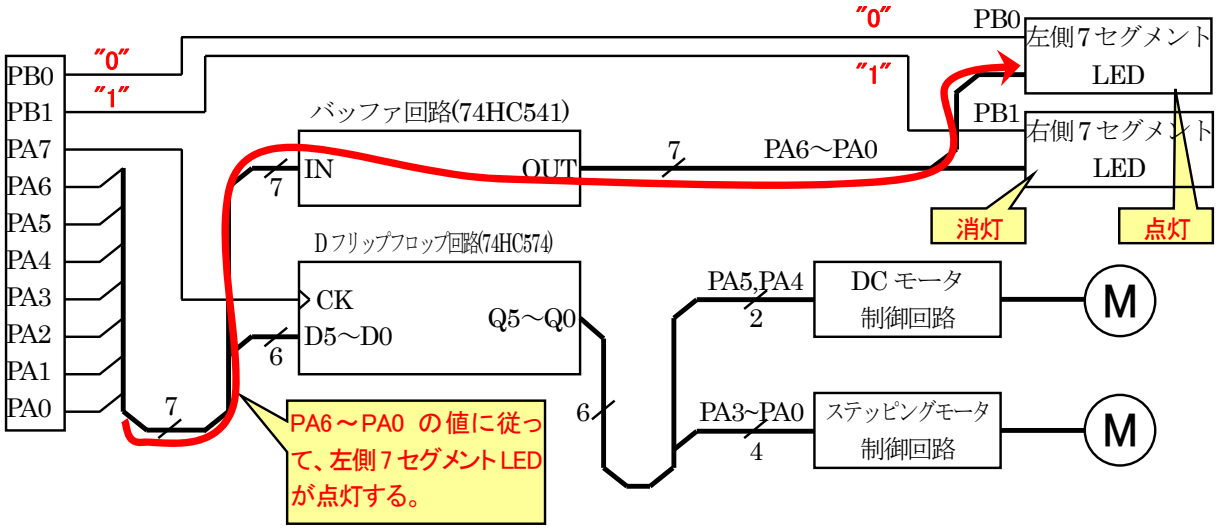
②ステッピングモータ、DC モータの制御

ステッピングモータを制御するために PA3～PA0、DC モータを制御するために PA5、PA4 の各端子からデータを出力します。まだ、各モータは前の動作のまま変化しません。PA7 の端子が"0"から"1"になった瞬間に 74HC574 の入力端子 D5～D0 の値が Q5～Q0 から出力されます。よって、PA7 を"0"→"1"にした瞬間に各モータの動作が変化します。次に PA7 の端子を"0"から"1"にするまで変化しません。



③7 セグメント LED の制御

左側 7 セグメント LED、右側 7 セグメント LED を制御します。PA6～PA0 が兼用のため、同時に点灯させることはできません。そのため、1ms ごと左側 7 セグメント LED と右側 7 セグメント LED の点灯を交互に繰り返します。左と右の 7 セグメント LED は同時に点灯しませんが、1ms ごとに点灯しているので、人間の目で見ると同時に点灯しているように見えます。左側 7 セグメント LED を点灯させるときの信号を、下図に示します。



このように、「ステッピングモータと DC モータ」→「左側 7 セグメント LED 表示」→「右側 7 セグメント LED 表示」の順番に制御していきます。

3.8.4　ステッピングモータの制御

(1) 励磁する手順

※励磁(れいじ)とは、ステッピングモータのコイルに電流を流すことです。

ステッピングモータには、A、$\overline{A}$、B、$\overline{B}$の 4 つのコイルがあります。時計回りに回転させるには「A→B→$\overline{A}$→$\overline{B}$」の順番に励磁します。反時計回りに回転させるには、「$\overline{B}$→$\overline{A}$→B→A」の順番に励磁します。
コイルとポートの関係を、下表に示します。

PA3	PA2	PA1	PA0	A のコイル	B のコイル	$\overline{A}$のコイル	$\overline{B}$のコイル
″0″	″0″	″0″	**″1″**	**ON**	OFF	OFF	OFF
″0″	″0″	**″1″**	″0″	OFF	**ON**	OFF	OFF
″0″	**″1″**	″0″	″0″	OFF	OFF	**ON**	OFF
″1″	″0″	″0″	″0″	OFF	OFF	OFF	**ON**

よって、ステッピングモータを時計回りに回転させるには、ポート A(PA3～PA0)を次のように設定します。

```
0x01→0x02→0x04→0x08→繰り返し
```

ステッピングモータを反時計回りに回転させるには、ポート A(PA3～PA0)を次のように設定します。

```
0x08→0x04→0x02→0x01→繰り返し
```

(2) 出力データに配列を使う

プログラムでは stepper_data という配列を作り、下記のようにプログラムしています。

```
////////////////////////////////////////////////
// ステッピングモータの励磁出力信号
//                                A     B     C(/A) D(/B)
unsigned char stepper_data[] = { 0x01, 0x02, 0x04, 0x08 };
```

添字([ ]の中に入れる数字のこと)に 0～3 の値をセットすると、0x01、0x02、0x04、0x08 の値が返ってきます。例えば、添字に 1 をセットしたところを、下記に示します。

```
    PADR = stepper_data[ 1 ];     // ポート A(PA)には、0x02 が設定されます。
```

したがって、添字を 0,1,2,3 と増やしていくとステッピングモータが時計回りに回転して、逆に 3,2,1,0 と減らしていくとステッピングモータが反時計回りに回転します。

(3) 回転数

ステッピングモータを回転させるスピードを変えるには、励磁コイルを切り替える間隔を変えます。セットする間隔が短ければステッピングモータは速く回転し、間隔が長ければステッピングモータはゆっくりと回転します。課題 5 は、「低速回転」、「高速回転」、「低速と高速の間の回転(標準回転)」の 3 種類の回転速度で回します。

今回のステッピングモータは励磁コイルを 1 つ切り替えるたびに、7.5 度回転します。1 回転するために切り替える回数は次のようになります。

360÷1 回当たりの角度 7.5＝48

よって、48 回切り替えると、ステッピングモータは 1 回転します。

今回のプログラムでは、stepper_kaiten という変数を用意して、時計回りに 48 回励磁コイルを切り替えたなら stepper_kaiten 変数を＋1, 反時計回りに 48 回励磁コイルを切り替えたなら stepper_kaiten 変数を－1して、ステッピングモータの回転数を検出します。

(4) 使用する変数

ステッピングモータ関連で使用する変数を、下記に示します。

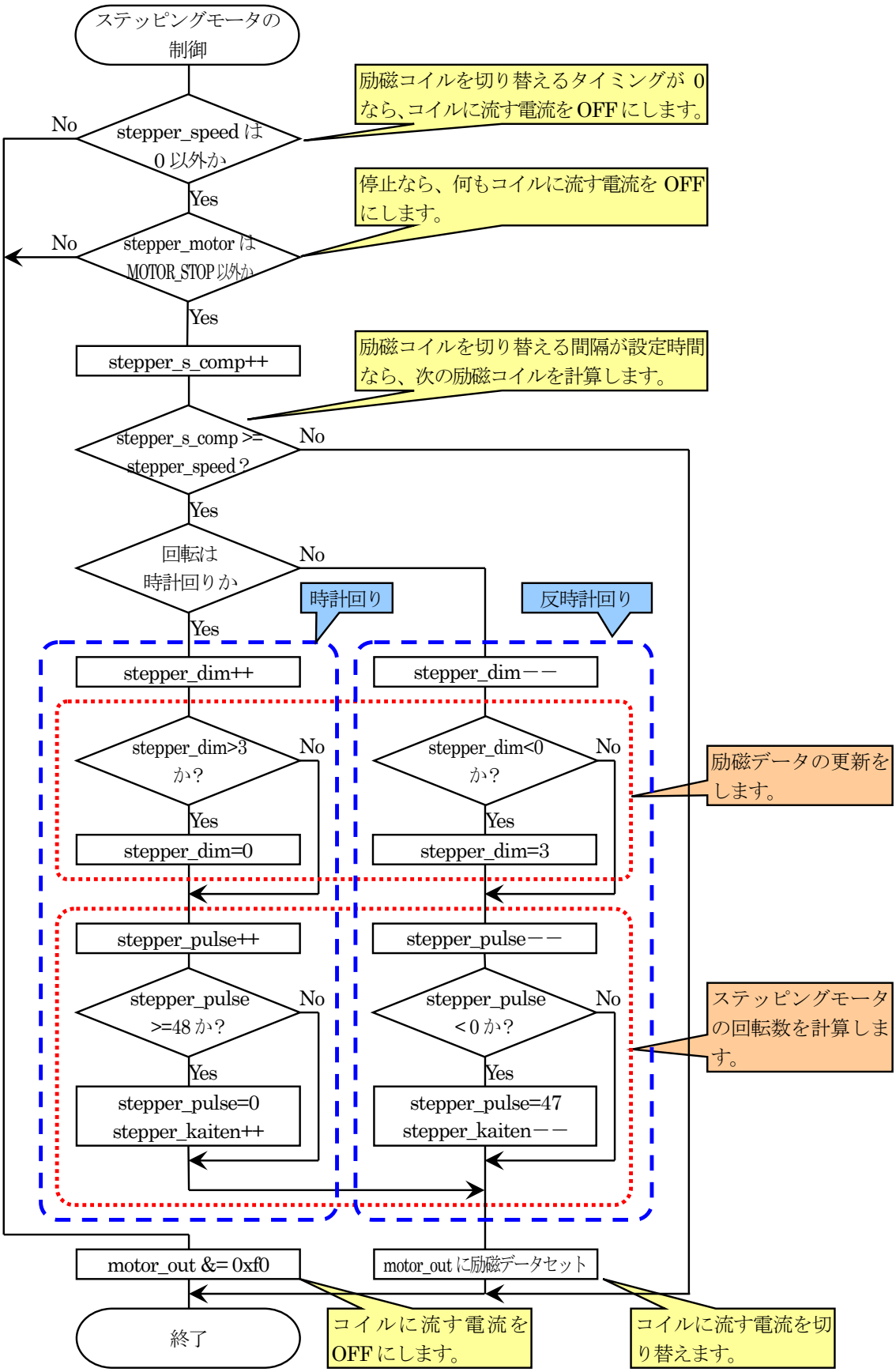
変数	内容
stepper_speed	励磁コイルを切り替えるスピードを[ms]単位で設定します。 例えば、励磁コイルを 10ms ごとに切り替えたい場合は、main 関数内でこの変数に 10 を設定します。

stepper_s_comp	stepper_speed 変数の値以上の時間になったかチェックする変数です。main 関数では、この変数は操作しません。
	stepper_s_comp 変数は割り込み関数内で 1ms ごとに＋1しています。次のプログラムで stepper_speed 変数の時間ごとに励磁コイルを切り替えます。
	
	`stepper_s_comp++;` 1ms ごとに＋１される
	`if( stepper_s_comp >= stepper_speed ) {`
	`  stepper_s_comp = 0;`
	`  励磁コイルを切り替えるプログラム`
	`}`

stepper_motor	ステッピングモータの回転方向を設定します。
	1：時計周りに回転
	0：停止
	-1：反時計回りに回転
	
	これらは、define 文で
	
	`#define MOTOR_TOKEI  1  // 時計回りに回転`
	`#define MOTOR_STOP  0  // 停止`
	`#define MOTOR_HANTOKEI  -1  // 反時計回りに回転`
	
	と定義しています。プログラムでは下記のように使います。
	
	`stepper_motor = MOTOR_TOKEI  // 時計回りに回転`
	`stepper_motor = MOTOR_STOP  // 停止`
	`stepper_motor = MOTOR_HANTOKEI  // 反時計回りに回転`

stepper_pulse	ステッピングモータの励磁コイルを切り替えた回数です。main 関数では、この変数は操作しません。 時計回りに励磁コイルを切り替えた場合は＋1、反時計回りに励磁コイルを切り替えた場合は－1します。
stepper_kaiten	励磁コイルを 48 回切り替えると 1 回転です。そのため、stepper_pulse 変数が 48 になると stepper_pulse 変数を 0 にして、stepper_kaiten 変数を＋1します。stepper_pulse 変数が -1 になると stepper_pulse 変数を 47 にして、stepper_kaiten 変数を－1します。 main 関数では、この変数の値を読み込むことにより、ステッピングモータの回転数が分かります。
stepper_dim	stepper_data 配列の添字です。main 関数では、この変数は操作しません。 時計回りの時は励磁コイルを切り替えるごとに＋1、反時計回りの時は励磁コイルを切り替えるごとに－1します。stepper_data 配列の添字は、0～3 の範囲なので、4 以上になったら 0 に、-1 になったら 3 にします。
motor_out	ステッピングモータは PA3～PA0、DC モータは PA5～PA4 なので、ポート A にステッピングモータの励磁データを出力すると、DC モータのビットにも影響を与えてしまいます。そこで、motor_out 変数を用意して、そこにステッピングモータの出力データと DC モータの出力データをセットして、その後、ポート A に二つのモータのデータを出力します。

(5) フローチャート


ステッピングモータの制御
stepper_speed は 0 以外か
No
Yes
励磁コイルを切り替えるタイミングが 0 なら、コイルに流す電流を OFF にします。
stepper_motor は MOTOR_STOP 以外か
No
Yes
停止なら、何もコイルに流す電流を OFF にします。
stepper_s_comp++
stepper_s_comp >= stepper_speed ?
No
Yes
励磁コイルを切り替える間隔が設定時間なら、次の励磁コイルを計算します。
回転は 時計回りか
No
Yes
時計回り
反時計回り
stepper_dim++
stepper_dim−−
stepper_dim>3 か?
stepper_dim<0 か?
No
Yes
stepper_dim=0
stepper_dim=3
励磁データの更新をします。
stepper_pulse++
stepper_pulse−−
stepper_pulse >=48 か?
stepper_pulse < 0 か?
No
Yes
stepper_pulse=0 stepper_kaiten++
stepper_pulse=47 stepper_kaiten−−
ステッピングモータの回転数を計算します。
motor_out &= 0xf0
motor_out に励磁データセット
終了
コイルに流す電流を OFF にします。
コイルに流す電流を切り替えます。

(6) プログラム

```
// グローバル変数の宣言

 64 : ////////////////////////////////////////////////////////////
 65 : // ステッピングモータの励磁出力信号
 66 : //                              A     B     C(/A) D(/B)
 67 : unsigned char stepper_data[] = { 0x01, 0x02, 0x04, 0x08 };
 68 :
 69 : // ステッピングモータ
 70 : int     stepper_speed;                  // スピード
 71 : int     stepper_motor;                  // 回転方向
 72 :
 73 : int     stepper_pulse;                  // 加えたパルス数 48 で 1 回転
 74 : int     stepper_kaiten;                 // 回転数 48 でこの変数+1
 75 : int     stepper_s_comp;                 // スピード比較値
 76 : int     stepper_dim;                    // 励磁出力信号の添字

// プログラム

153 :      // ステッピングモータの制御
154 :      if( stepper_speed != 0 && stepper_motor != MOTOR_STOP) {
155 :          // 回転速度が 0 以上、かつ停止以外なら処理
156 :
157 :          // 励磁コイルを替える間隔のチェック
158 :          stepper_s_comp++;
159 :          if( stepper_s_comp >= stepper_speed ) {
160 :              stepper_s_comp = 0;
161 :              if( stepper_motor == MOTOR_TOKEI ) {            // 時計回りか？
162 :                  // 励磁コイルの切り替え
163 :                  stepper_dim++;
164 :                  if( stepper_dim > 3 ) stepper_dim = 0;
165 :
166 :                  // 回転数の計算
167 :                  stepper_pulse++;
168 :                  if( stepper_pulse >= 48 ) {
169 :                      stepper_pulse = 0;
170 :                      stepper_kaiten++;
171 :                  }
172 :              } else if( stepper_motor == MOTOR_HANTOKEI ) {  // 反時計回りか？
173 :                  // 励磁コイルの切り替え
174 :                  stepper_dim--;
175 :                  if( stepper_dim < 0 ) stepper_dim = 3;
176 :
177 :                  // 回転数の計算
178 :                  stepper_pulse--;
179 :                  if( stepper_pulse < 0 ) {
180 :                      stepper_pulse = 47;
181 :                      stepper_kaiten--;
182 :                  }
183 :              }
184 :
185 :              motor_out &= 0xf0;
186 :              motor_out |= stepper_data[ stepper_dim ];       // 励磁データセット
187 :          }
188 :      } else {
189 :          // 停止
190 :          motor_out &= 0xf0;
191 :      }
```

3.8.5 DCモータの制御

(1) 回転させる方法

制御対象回路③には、DC モータを制御する専用の IC が取り付けてあり、2bit の信号でモータを制御することができます。また、ステッピングモータのように回転速度を変える必要がないため、PWM 出力(速度制御する方法の一つ)などは必要なく、単純に"0"または"1"を出力するだけで制御可能です。

ポートと DC モータの動作の関係を、下表に示します。

PA5	PA4	DC モータの動作
"0"	"0"	ストップ
"0"	"1"	時計回りに回転
"1"	"0"	反時計回りに回転
"1"	"1"	ブレーキ

ストップは DC モータの端子間を解放、ブレーキは DC モータの端子間をショートさせます。今回モータを止めるのはストップを使います。

(2) 使用する変数

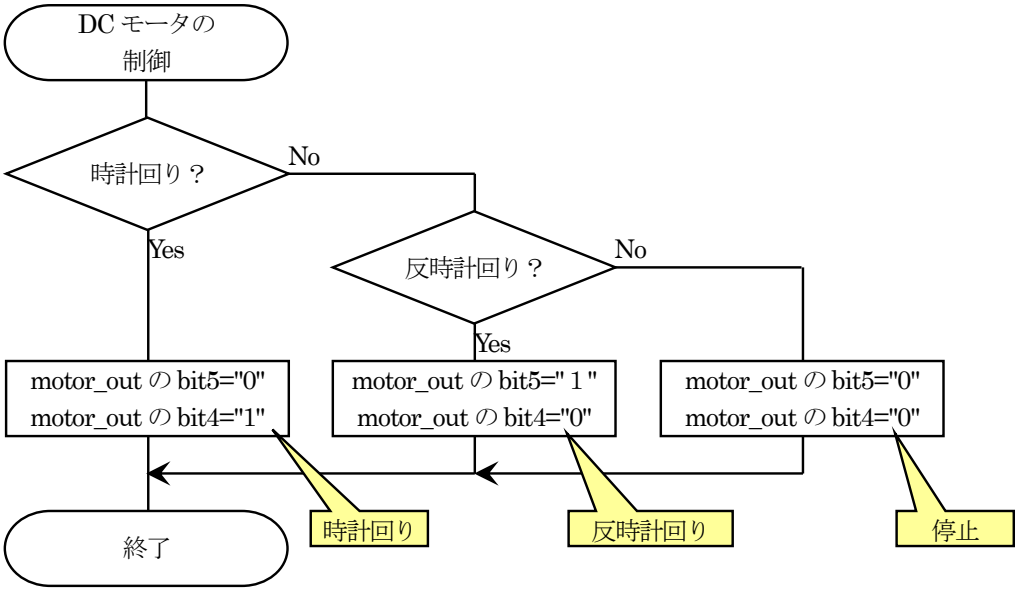
DC モータ関連で使用する変数を、下記に示します。

変数	内容

dc_motor	DC モータの回転方向を設定します。
	1：時計周りに回転
	0：停止
	-1：反時計回りに回転
	
	これらは、define 文で
	
	#define MOTOR_TOKEI 1 // 時計回りに回転
	#define MOTOR_STOP 0 // 停止
	#define MOTOR_HANTOKEI -1 // 反時計回りに回転
	
	と定義しています。プログラムでは下記のように使います。
	
	dc_motor = MOTOR_TOKEI // 時計回りに回転
	dc_motor = MOTOR_STOP // 停止
	dc_motor = MOTOR_HANTOKEI // 反時計回りに回転

motor_out	ステッピングモータは PA3～PA0、DC モータは PA5～PA4 なので、ポート A に DC モータの制御データを出力すると、ステッピングモータのビットにも影響を与えてしまいます。そこで、motor_out 変数を用意して、そこにステッピングモータの出力データと DC モータの出力データをセットして、その後、ポート A に二つのモータのデータを出力します。

(3) フローチャート



(4) プログラム

```
// グローバル変数の宣言

  78 :  ////////////////////////////////////////////////////////////
  79 :  // DC モータ
  80 :  int     dc_motor;                       // 回転方向

// プログラム

 193 :      // DC モータの制御
 194 :      if( dc_motor == MOTOR_TOKEI ) {             // 時計回りか？
 195 :          motor_out &= 0xdf;
 196 :          motor_out |= 0x10;
 197 :      } else if( dc_motor == MOTOR_HANTOKEI ) {   // 反時計回りか？
 198 :          motor_out &= 0xef;
 199 :          motor_out |= 0x20;
 200 :      } else {                                    // それ以外は停止
 201 :          motor_out &= 0xcf;
 202 :      }
```

3.8.6 モータ出力処理

(1) モータへ信号を出力する方法

ステッピングモータとDC モータへ出力するデータを、motor_out 変数に格納しました。motor_out 変数の値をポート A へ出力して実際にモータを制御します。ただし、ポート A は左側 7 セグメント LED、右側 7 セグメント LED も接続されているので、混同しないようにしなければいけません。

(2) プログラム

```
204 :       // DC モータ、ステッピングモータへ出力
205 :       PBDR = 0x03;                        // 7 セグメント LED 消灯(兼用のため)
206 :
207 :       PADR = motor_out;                   // DC ﾓｰﾀ、ｽﾃｯﾋﾟﾝｸﾞﾓｰﾀへ信号出力
208 :
209 :       PADR |= 0x80;                       // CK HIGH
210 :       PADR &= 0x7f;                       // CK LOW
```

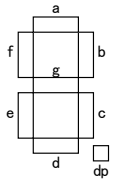
行	詳細
205	PB1 と PB0 に″1″を出力して、左側 7 セグメント LED、右側 7 セグメント LED を消灯させます。
207	ポート A に motor_out 変数の値を出力します。74HC574 の入力端子(D5～D0)に出力されます。まだモータには出力されていません。
209	PA7 を″1″にします。″1″にした瞬間、74HC574 の D5～D0 に入力されているポート A の出力データが、Q5～Q0 端子から出力され、モータの回路へ信号が出力されます。
210	PA7 を″0″にします。次に PA7 を″1″にするまで 74HC574 の出力データは変化しません。よって、ポート A の値を変えてもモータの動作は変わりません。この間に 7 セグメント LED へデータを出力し表示させます。

3.8.7　7 セグメントLEDの制御

(1) 表示する方法

7 セグメント LED には、名前のとおり 7 個の LED があります。点灯する LED を組み合わせることにより数値や一部のアルファベットを表示することが出来ます。各 LED には名前が付いており、いちばん上を"a"、時計回りに"b"→"c"…"f"、真ん中を"g"とします。小数点を表示する点"dp"もありますが、今回は表示しません（回路的に繋がっていません）。表示する値と、7 セグメント LED に送るデータを下図に示します。

この表示パターンは課題のプリントで指定されています。指定パターン以外で表示すると、減点の対象となりますので気をつけてください。

表示 イメージ 	dp 今回は 未接続	g PA6	f PA5	e PA4	d PA3	c PA2	b PA1	a PA0	16 進数
	0	0	1	1	1	1	1	1	0x3f
	0	0	0	0	0	1	1	0	0x06
	0	1	0	1	1	0	1	1	0x5b
	0	1	0	0	1	1	1	1	0x4f
	0	1	1	0	0	1	1	0	0x66
	0	1	1	0	1	1	0	1	0x6d
	0	1	1	1	1	1	0	1	0x7d
	0	0	1	0	0	1	1	1	0x27
	0	1	1	1	1	1	1	1	0x7f
	0	1	1	0	1	1	1	1	0x6f
	0	1	0	1	1	1	1	1	0x5f
	0	1	1	1	1	1	0	0	0x7c
	0	1	0	1	1	0	0	0	0x58
	0	1	0	1	1	1	1	0	0x5e
	0	1	1	1	1	0	0	1	0x79

	0	1	1	1	0	0	0	1	0x71
	0	0	1	1	1	0	0	0	0x38
	0	1	1	1	0	1	1	0	0x76

上表の 16 進数の値をポート A から出力し、PB0 を"0"にすると左側 7 セグメント LED が点灯、PB1 を"0"にすると右側 7 セグメント LED が点灯します。

ポート A は、左側 7 セグメント LED、右側 7 セグメント LED を共用しているので、交互に点灯させます。具体的には 1ms 間は左側 7 セグメント LED を点灯、次の 1ms 間は右側 7 セグメント LED を点灯、これを交互に繰り返します。1ms ごとなので、人間の目には早すぎて同時に点灯しているように見えます。もし 1 秒ごとに交互に点灯させたなら、遅すぎて交互に点灯しているのが分かってしまいます。どれくらいのスピードまで同時に点灯して見えるのか実験するのも良いでしょう。

(2) 表示データに配列を使う

プログラムでは seg_data という配列を作り、下記のようにプログラムしています。

```
48 :  // 7 セグメント LED の表示データ 0～F,L,H
49 :  const unsigned char seg_data[] =
50 :      { 0x3f, 0x06, 0x5b, 0x4f, 0x66, 0x6d, 0x7d, 0x27,     // 01234567
51 :        0x7f, 0x6f, 0x5f, 0x7c, 0x58, 0x5e, 0x79, 0x71,     // 89abcdEF
52 :        0x38, 0x76                                          // HL
53 :  };
```

添字([ ] の中に入れる数字のこと)に 0～17 の値をセットすると、7 セグメント LED に出力する値が返ってきます。値は、0～9 が数字、10～15 が"A"～"f"のアルファベット、16 が"H"、17 が"L"となります。例えば、添字に 10 をセットしたところを、下記に示します。

```
PADR = seg_data[ 10 ];     // ポート A(PA)には、0x5f が設定されます。
```

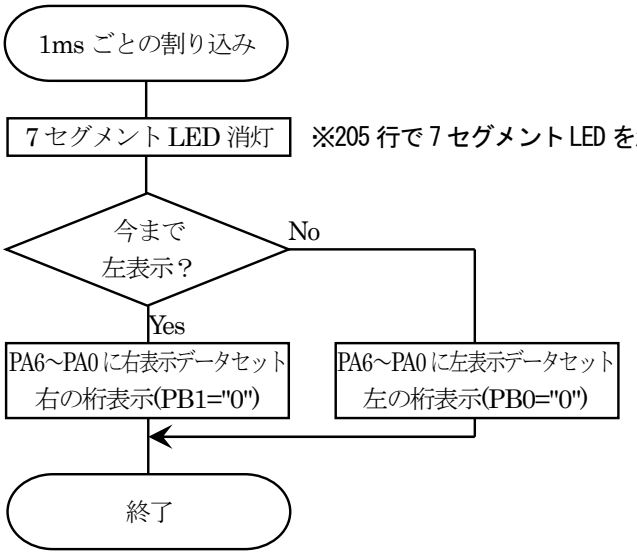
ポート A には、'a'を表示する値である、0x5f がセットされます。このように、添字にそのまま 0～17 の値をセットすればポート A に出力する値が取り出せるので、いちいち 0 だから 0x3f をセットして･･･ 1 だから 0x06 をセットして･･･ というようにプログラムする必要が無くなります。

(3) 使用する変数

7 セグメント LED 関連で使用する変数を、下記に示します。

変数	内容
seg_left	左側 7 セグメント LED に表示する値を設定します。 0～9 までが数字、10～15 までが"a"～"F"のアルファベットを表示、16 が"H"、17 が"L"、負の数なら消灯となります。 "H"、"L"、消灯は、define 文で下記のように定義しています。 #define SEG_L　　16　　// 7 セグ L 表示 #define SEG_H　　17　　// 7 セグ H 表示 #define SEG_NULL　-1　　// 7 セグ 消灯 (例) seg_left = 0;　　// '0'を表示 seg_left = 0xb;　　// 'b'を表示 seg_left = SEG_L　　// 'L'を表示 seg_left = SEG_NULL;　// 消灯
seg_right	右側 7 セグメント LED に表示する値を設定します。表示方法は、seg_left 変数と同様です。
seg_digit	左側、右側のどちらの 7 セグメント LED を表示するか設定します。main 関数内では使いません。 0:左表示 1:右表示

(4) フローチャート



(5) プログラム

```
// グローバル変数の宣言

 47 : ////////////////////////////////////////////////////////////
 48 : // 7 セグメント LED の表示データ 0～F,L,H
 49 : const unsigned char seg_data[] =
 50 :     { 0x3f, 0x06, 0x5b, 0x4f, 0x66, 0x6d, 0x7d, 0x27,     // 01234567
 51 :       0x7f, 0x6f, 0x5f, 0x7c, 0x58, 0x5e, 0x79, 0x71,     // 89abcdEF
 52 :       0x38, 0x76                                          // HL
 53 : };
 54 :
 55 : // 7 セグメント LED の表示値
 56 : int     seg_left  = SEG_NULL;          // 左の桁　表示値
 57 : int     seg_right = SEG_NULL;          // 右の桁　表示値
 58 : int     seg_digit;                     // 0:左表示 1:右表示

// プログラム

212 :      // 7 セグメント LED の表示　左と右の桁を交互に表示
213 :      if( seg_digit == 0 ) {             // 左表示中なら
214 :          // 右を表示
215 :          seg_digit = 1;
216 :          if( seg_right >= 0 ) {
217 :              PADR = ~seg_data[seg_right];// 値のセット
218 :              PBDR = 0x01;               // 右 PB1="0"(ON) 左 PB0="1"(OFF)
219 :          }
220 :      } else {
221 :          // 左を表示
222 :          seg_digit = 0;
223 :          if( seg_left >= 0 ) {
224 :              PADR = ~seg_data[seg_left]; // 値のセット
225 :              PBDR = 0x02;               // 右 PB1="1"(OFF) 左 PB0="0"(ON)
226 :          }
227 :      }
```

行	詳細
213	seg_digit 変数が 0 なら左側 7 セグメント LED 表示中です。 seg_digit 変数が 0 かチェックして、0 なら 215～219 行を実行して右側 7 セグメント LED を表示、それ以外(右側表示中)なら、else 文の 222～226 行を実行します。
215	seg_digit 変数を 1 にして、右側表示にします。
216	表示値がマイナスなら、何もセットせずに終了します。右側 7 セグメント LED は消灯です。
217	seg_data 配列から表示データを読み込んで、ポート A に設定します。
218	PB1 を"0"にして、右側 7 セグメント LED を点灯します。
222～226	左側 7 セグメント LED を表示させる処理です。意味は、右側と同様です。

3.9 その他

3.9.1 時間の測定

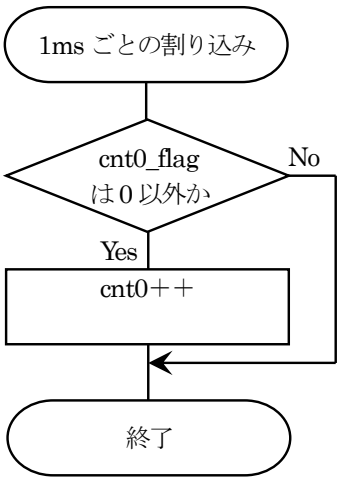
課題 7 は、7 セグメント LED を 1 秒ごとに＋1していきます。そのため、専用の変数を作り、1ms ごとに＋1するようにして、この変数を確認することで時間を計るようにします。

(1) 使用する変数

時間の測定で使用する変数を、下記に示します。

変数	内容
cnt0_flag	cnt0 変数を 1ms ごとに＋1するかしないかを設定します。 0:cnt0 の値を変化させない 1:cnt0 を 1ms ごとに＋1する
cnt0	cnt0_flag が 1 なら、1ms ごとに＋1します。cnt0_flag が 0 なら値は変化しません。

(2) フローチャート



(3) プログラム

```
// グローバル変数の宣言

  26 :  ////////////////////////////////////////////////////////////
  27 :  // カウンタ
  28 :  long    cnt0;                           // 1ms ごとに+1
  29 :  int     cnt0_flag;                      // 0:cnt0 のカウントを停止

// プログラム

 115 :      // カウンタカウントアップ
 116 :      if( cnt0_flag != 0 ) {
 117 :          cnt0++;
 118 :      }
```

3.10　共通Cファイル「common.c」のまとめ

今まで説明した H8/3048F-ONE 内蔵周辺機能の初期化、入力信号を取り込むプログラム、出力回路へ信号を出力するプログラムなどを「common.c」としてまとめます。「kadai1.c」～「kadai7.c」プログラムの共通ファイルとしました。下記に、プログラムを示します。

コンテスト時は限られた時間しかありませんが、できるだけ日本語のコメントをつけてください。自分でも分かりやすいですし、間違いも少なくなります（審査もしやすくなります）。

```
 1 : ////////////////////////////////////////////////////////////////////////
 2 : // 第 10 回高校生ものづくりコンテスト全国大会　電子回路部門　共通ファイル
 3 : // 座席番号：●
 4 : // 氏　　名：○○　○○
 5 : //
 6 : // Copyright (C) 2010 ルネサスマイコンカーラリー事務局
 7 : ////////////////////////////////////////////////////////////////////////
 8 :
 9 : ////////////////////////////////////////////////////////////////////////
10 : // シンボル定義
11 : ////////////////////////////////////////////////////////////////////////
12 : // seg_left と seg_right 変数に入れる値
13 : #define SEG_L       16                  // 7 セグ L 表示
14 : #define SEG_H       17                  // 7 セグ H 表示
15 : #define SEG_NULL    -1                  // 7 セグ 消灯
16 :
17 : // モータ(ステッピングモータ、DC モータ共通)
18 : #define MOTOR_TOKEI     1               // 時計回りに回転
19 : #define MOTOR_STOP      0               // 停止
20 : #define MOTOR_HANTOKEI  -1              // 反時計回りに回転
21 :
22 : ////////////////////////////////////////////////////////////////////////
23 : // グローバル変数の宣言
24 : ////////////////////////////////////////////////////////////////////////
25 :
26 : //////////////////////////////////////////////////////////
27 : // カウンタ
28 : long    cnt0;                           // 1ms ごとに+1
29 : int     cnt0_flag;                      // 0:cnt0 のカウントを停止
30 :
31 : int     cnt10ms;                        // 10ms ごとの処理用
32 :
33 : //////////////////////////////////////////////////////////
34 : // タクトスイッチの状態
35 : int     ts;                             // 0:OFF 1:ON
36 : int     ts_onflag;                      // OFF→ON になった瞬間に 1
37 :
38 : //////////////////////////////////////////////////////////
39 : // ホトインタラプタの状態
40 : int     ps;                             // 0:透過 1:遮断
41 : int     ps_syadanflag;                  // 透過→遮断された瞬間に 1
42 :
43 : //////////////////////////////////////////////////////////
44 : // トグルスイッチの状態
45 : int     tgs;                            // 0:Low 1:High
46 :
47 : //////////////////////////////////////////////////////////
48 : // 7 セグメント LED の表示データ 0～F,L,H
49 : const unsigned char seg_data[] =
50 :     { 0x3f, 0x06, 0x5b, 0x4f, 0x66, 0x6d, 0x7d, 0x27,     // 01234567
51 :       0x7f, 0x6f, 0x5f, 0x7c, 0x58, 0x5e, 0x79, 0x71,     // 89abcdEF
52 :       0x38, 0x76                                          // HL
53 : };
54 :
55 : // 7 セグメント LED の表示値
56 : int     seg_left  = SEG_NULL;           // 左の桁　表示値
57 : int     seg_right = SEG_NULL;           // 右の桁　表示値
58 : int     seg_digit;                      // 0:左表示 1:右表示
59 :
```

```
60 : ////////////////////////////////////////////////////////////
61 : // DC モータ、ステッピングモータのポート出力データ　バッファ
62 : unsigned char motor_out;
63 :
64 : ////////////////////////////////////////////////////////////
65 : // ステッピングモータの励磁出力信号
66 : //                              A     B     C(/A) D(/B)
67 : unsigned char stepper_data[] = { 0x01, 0x02, 0x04, 0x08 };
68 :
69 : // ステッピングモータ
70 : int     stepper_speed;                  // スピード
71 : int     stepper_motor;                  // 回転方向
72 :
73 : int     stepper_pulse;                  // 加えたパルス数 48 で 1 回転
74 : int     stepper_kaiten;                 // 回転数 48 でこの変数+1
75 : int     stepper_s_comp;                 // スピード比較値
76 : int     stepper_dim;                    // 励磁出力信号の添字
77 :
78 : ////////////////////////////////////////////////////////////
79 : // DC モータ
80 : int     dc_motor;                       // 回転方向
81 :
82 : ////////////////////////////////////////////////////////////////////////
83 : // H8/3048F-ONE 内蔵周辺機能の初期化
84 : ////////////////////////////////////////////////////////////////////////
85 : void init( void )
86 : {
87 :     // ポートの入出力設定
88 :     P1DDR = 0xff;
89 :     P2DDR = 0xff;
90 :     P3DDR = 0xff;
91 :     P4DDR = 0xff;
92 :     P5DDR = 0xff;
93 :     P6DDR = 0xf0;                       // マイコンボード上の DIP SW
94 :     // P7 は入力専用                     // 設計製作回路①と接続
95 :     P8DDR = 0xff;
96 :     P9DDR = 0xf7;
97 :     PADDR = 0xff;                       // 制御対象回路③と接続
98 :     PBDDR = 0xff;                       // 制御対象回路③と接続
99 :
100 :    // ITU0 1ms 毎の割り込み
101 :    ITU0_TCR = 0x20;                    // GRA=CNT でカウンタクリア
102 :    ITU0_GRA = 24575;                   // 割り込み周期 0.001/(1/24.576M)-1
103 :    ITU0_IER = 0x01;                    // 割り込み許可
104 :    ITU_STR  = 0x01;                    // ITU0 カウント開始
105 : }
106 :
107 : ////////////////////////////////////////////////////////////////////////
108 : // ITU0 割り込み処理
109 : ////////////////////////////////////////////////////////////////////////
110 : #pragma interrupt interrupt_timer0(vect=24)
111 : void interrupt_timer0( void )
112 : {
113 :     ITU0_TSR &= 0xfe;                   // フラグクリア
114 :
115 :     // カウンタカウントアップ
116 :     if( cnt0_flag != 0 ) {
117 :         cnt0++;
118 :     }
119 :
120 :     // スイッチ、ホトインタラプタは 10ms ごとにチェックする
121 :     cnt10ms++;
122 :     if( cnt10ms >= 10 ) {
123 :         cnt10ms = 0;
124 :
```

```
125 :          // タクトスイッチの値取り込み
126 :          if( (P7DR & 0x04) == 0x04 ) {
127 :              ts = 0;                      // OFF なら
128 :          } else {
129 :              if( ts == 0 ) {
130 :                  ts_onflag = 1;           // ON になった瞬間なら 1
131 :              }
132 :              ts = 1;                      // ON なら
133 :          }
134 :
135 :          // ホトインタラプタの値取り込み
136 :          if( (P7DR & 0x02) == 0x02 ) {
137 :              if( ps == 0 ) {
138 :                  ps_syadanflag = 1;       // 遮断した瞬間なら 1
139 :              }
140 :              ps = 1;                      // 遮断なら
141 :          } else {
142 :              ps = 0;                      // 透過なら
143 :          }
144 :
145 :          // トグルスイッチの値取り込み
146 :          if( (P7DR & 0x01) == 0x01 ) {
147 :              tgs = 1;                     // "H"なら
148 :          } else {
149 :              tgs = 0;                     // "L"なら
150 :          }
151 :      }
152 :
153 :      // ステッピングモータの制御
154 :      if( stepper_speed != 0 && stepper_motor != MOTOR_STOP) {
155 :          // 回転速度が 0 以上、かつ停止以外なら処理
156 :
157 :          // 励磁を替える間隔のチェック
158 :          stepper_s_comp++;
159 :          if( stepper_s_comp >= stepper_speed ) {
160 :              stepper_s_comp = 0;
161 :              if( stepper_motor == MOTOR_TOKEI ) {            // 時計回りか？
162 :                  // 励磁の切り替え
163 :                  stepper_dim++;
164 :                  if( stepper_dim > 3 ) stepper_dim = 0;
165 :
166 :                  // 回転数の計算
167 :                  stepper_pulse++;
168 :                  if( stepper_pulse >= 48 ) {
169 :                      stepper_pulse = 0;
170 :                      stepper_kaiten++;
171 :                  }
172 :              } else if( stepper_motor == MOTOR_HANTOKEI ) {  // 反時計回りか？
173 :                  // 励磁の切り替え
174 :                  stepper_dim--;
175 :                  if( stepper_dim < 0 ) stepper_dim = 3;
176 :
177 :                  // 回転数の計算
178 :                  stepper_pulse--;
179 :                  if( stepper_pulse < 0 ) {
180 :                      stepper_pulse = 47;
181 :                      stepper_kaiten--;
182 :                  }
183 :              }
184 :
185 :              motor_out &= 0xf0;
186 :              motor_out |= stepper_data[ stepper_dim ];       // 励磁データセット
187 :          }
188 :      } else {
189 :          // 停止
190 :          motor_out &= 0xf0;
191 :      }
192 :
```

```
193 :      // DC モータの制御
194 :      if( dc_motor == MOTOR_TOKEI ) {             // 時計回りか？
195 :          motor_out &= 0xdf;
196 :          motor_out |= 0x10;
197 :      } else if( dc_motor == MOTOR_HANTOKEI ) {   // 反時計回りか？
198 :          motor_out &= 0xef;
199 :          motor_out |= 0x20;
200 :      } else {                                    // それ以外は停止
201 :          motor_out &= 0xcf;
202 :      }
203 :
204 :      // DC モータ、ステッピングモータへ出力
205 :      PBDR = 0x03;                            // 7 セグメント LED 消灯(兼用のため)
206 :
207 :      PADR = motor_out;                       // DC ﾓｰﾀ、ｽﾃｯﾋﾟﾝｸﾞﾓｰﾀへ信号出力
208 :
209 :      PADR |= 0x80;                           // CK HIGH
210 :      PADR &= 0x7f;                           // CK LOW
211 :
212 :      // 7 セグメント LED の表示　左と右の桁を交互に表示
213 :      if( seg_digit == 0 ) {                  // 左表示中なら
214 :          // 右を表示
215 :          seg_digit = 1;
216 :          if( seg_right >= 0 ) {
217 :              PADR = ~seg_data[seg_right];// 値のセット
218 :              PBDR = 0x01;                // 右 PB1="0"(ON) 左 PB0="1"(OFF)
219 :          }
220 :      } else {
221 :          // 左を表示
222 :          seg_digit = 0;
223 :          if( seg_left >= 0 ) {
224 :              PADR = ~seg_data[seg_left]; // 値のセット
225 :              PBDR = 0x02;                // 右 PB1="1"(OFF) 左 PB0="0"(ON)
226 :          }
227 :      }
228 :  }
229 :
230 :  ////////////////////////////////////////////////////////////////////////
231 :  // End of File
232 :  ////////////////////////////////////////////////////////////////////////
```

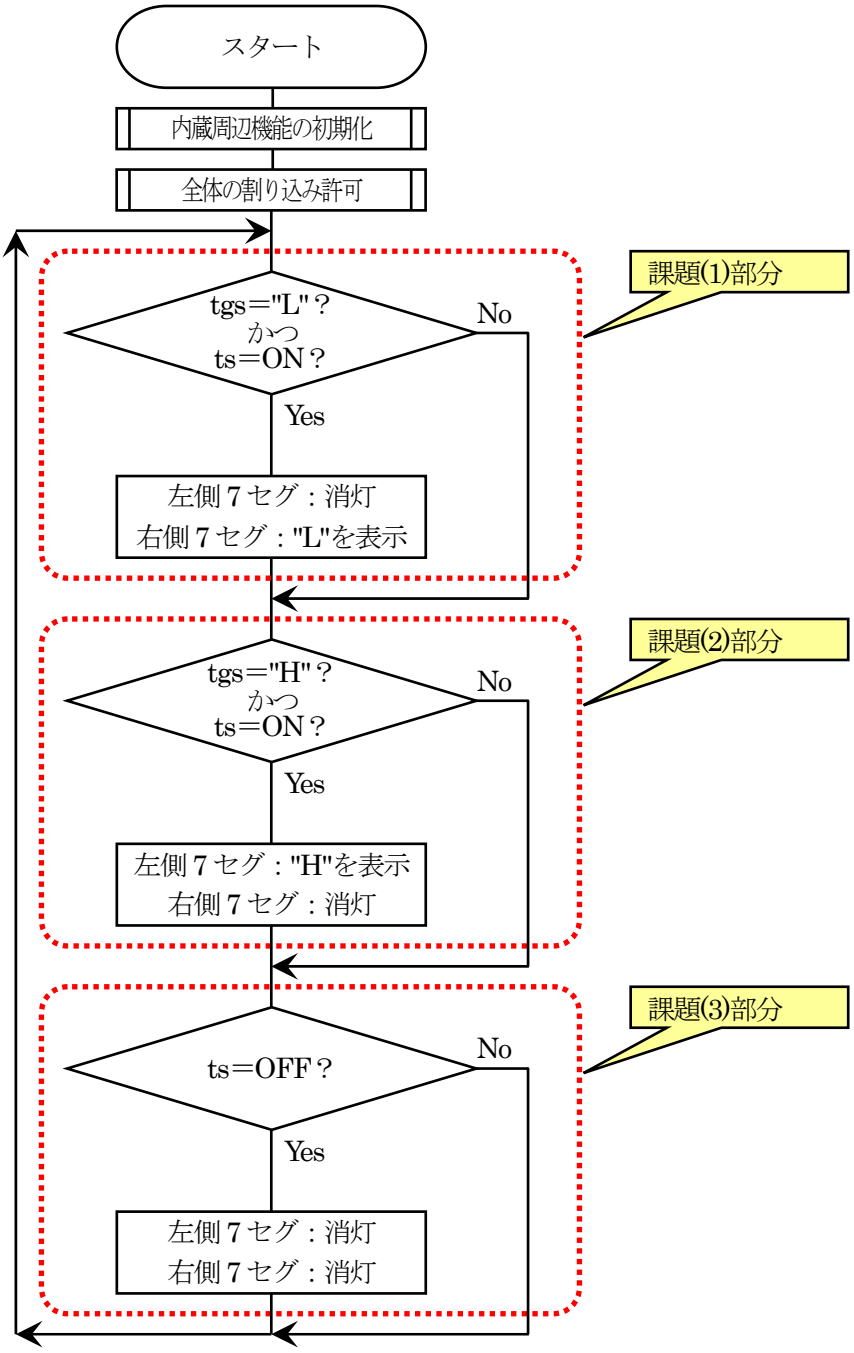
3.11　課題1

3.11.1　課題

(1) トグルスイッチが「Ｌ」の状態で，タクトスイッチが「ＯＮ」のとき，右側の７セグメントＬＥＤに‘Ｌ’を表示する。
(2) トグルスイッチが「Ｈ」の状態で，タクトスイッチが「ＯＮ」のとき，左側の７セグメントＬＥＤに‘Ｈ’を表示する。
(3) タクトスイッチが「ＯＦＦ」のとき，左右の７セグメントＬＥＤは消灯している。

▲大会当日配付資料より抜粋

3.11.2　フローチャート



3.11.3 プログラム例

```
 1 : ////////////////////////////////////////////////////////////////////////////
 2 : // 第 10 回高校生ものづくりコンテスト全国大会　電子回路部門　課題 1
 3 : // 座席番号：●
 4 : // 氏　　名：○○　○○
 5 : //
 6 : // Copyright (C) 2010 ルネサスマイコンカーラリー事務局
 7 : ////////////////////////////////////////////////////////////////////////////
 8 :
 9 : ////////////////////////////////////////////////////////////////////////////
10 : // インクルードファイル設定
11 : ////////////////////////////////////////////////////////////////////////////
12 : #include <machine.h>                    // H8 マイコン特有の命令取り込み
13 : #include "h8_3048.h"                   // H8/3048F-ONE 用 I/O レジスタ定義
14 : #include "common.c"                    // 共通ファイルの取り込み
15 :
16 : ////////////////////////////////////////////////////////////////////////////
17 : // メイン関数
18 : ////////////////////////////////////////////////////////////////////////////
19 : void main( void )
20 : {
21 :     init();                            // 内蔵周辺機能の初期化
22 :     set_ccr( 0x00 );                   // 全体割り込み許可
23 :
24 :     while( 1 ) {
25 :         // トグルスイッチ="L" タクトスイッチ ON なら
26 :         if( tgs == 0 && ts == 1 ) {
27 :             seg_left  = SEG_NULL;
28 :             seg_right = SEG_L;
29 :         }
30 :
31 :         // トグルスイッチ="H" タクトスイッチ ON なら
32 :         if( tgs == 1 && ts == 1 ) {
33 :             seg_left  = SEG_H;
34 :             seg_right = SEG_NULL;
35 :         }
36 :
37 :         // タクトスイッチ OFF なら
38 :         if( ts == 0 ) {
39 :             seg_left  = SEG_NULL;
40 :             seg_right = SEG_NULL;
41 :         }
42 :     }
43 : }
44 :
45 : ////////////////////////////////////////////////////////////////////////////
46 : // End of File
47 : ////////////////////////////////////////////////////////////////////////////
```

3.11.4 プログラムの解説

行	詳細
14	通常は、common.c と kadai1.c は別々にコンパイルします。それぞれのファイルの関数を呼び出すために、ヘッダファイル（通常は拡張子 h ファイル)を作り関数のプロトタイプ宣言をしたり、ルネサス統合開発環境に登録するなど、決められた手続きが必要です。また、変数を共用するには、extern 宣言しなければいけないなど、手続きに時間がかかってしまいます。 今回は 150 分という限られた時間しかありませんので、直接 C 言語ソースファイルを取り込んで、kadai1.c のプログラムの一部とします。
26	トグルスイッチが"L"、かつタクトスイッチが ON なら、27 行、28 行を実行します。 27 行は左側 7 セグメント LED を消灯、28 行は右側 7 セグメント LED に"L"を表示させます。
32	トグルスイッチが"H"、かつタクトスイッチが ON なら、33 行、34 行を実行します。 33 行は左側 7 セグメント LED に"H"を表示、34 行は右側 7 セグメント LED を消灯させます。
38	タクトスイッチが OFF なら、39 行、40 行を実行します。 39 行は左側 7 セグメント LED を消灯、40 行は右側 7 セグメント LED を消灯させます。

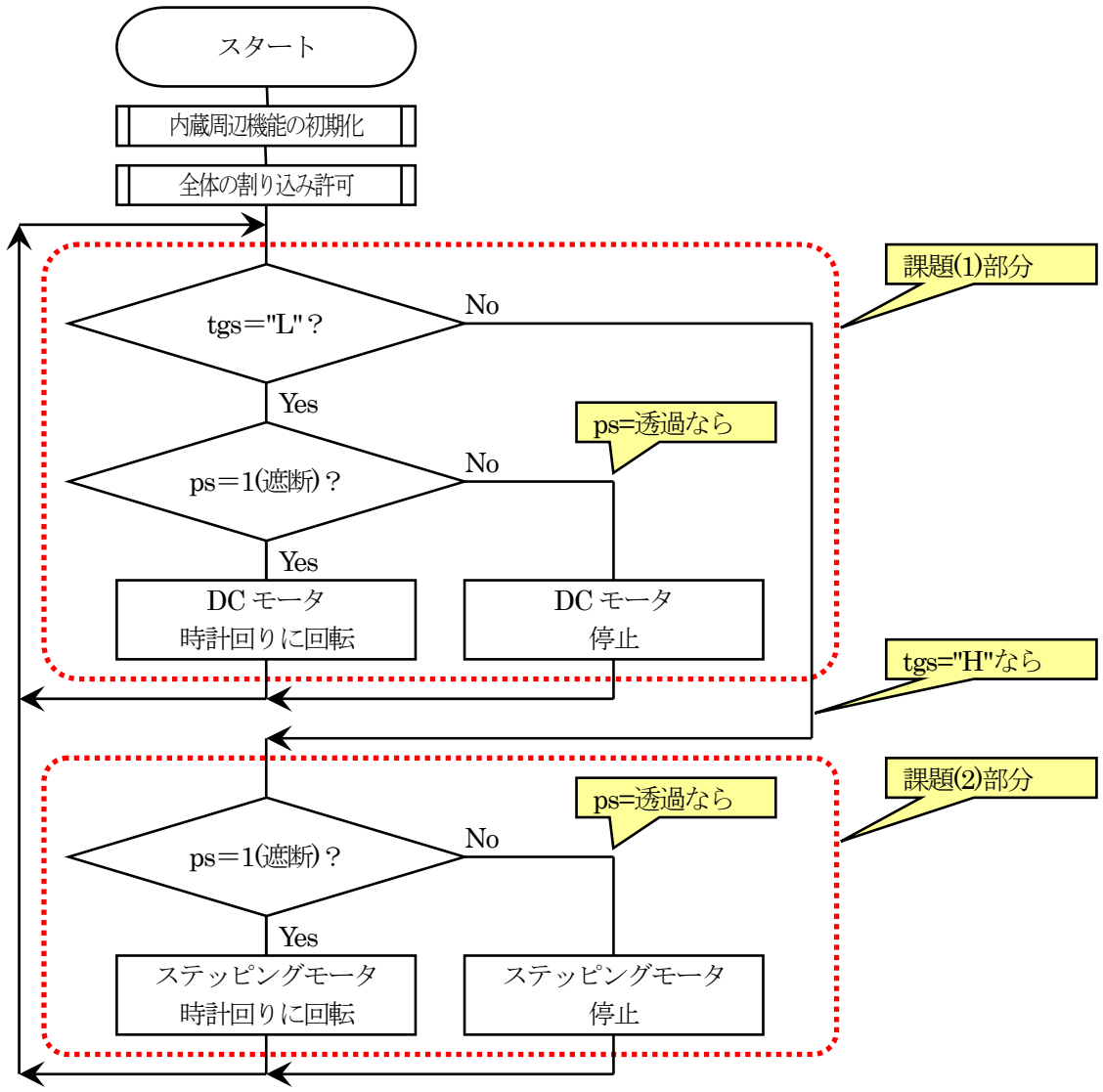
3.12　課題 2

3.12.1　課題

(1) トグルスイッチが「L」の状態で，ホトインタラプタが「遮断」されると，ＤＣモータが時計回りに回転する。
　ホトインタラプタを「透過」すると，ＤＣモータは停止する。
(2) トグルスイッチが「H」の状態で，ホトインタラプタが「遮断」されると，ステッピングモータが時計回りに回転する。
　ホトインタラプタを「透過」すると，ステッピングモータは停止する。

▲大会当日配付資料より抜粋

3.12.2　フローチャート



3.12.3 プログラム

```
 1 : ////////////////////////////////////////////////////////////////////////
 2 : // 第 10 回高校生ものづくりコンテスト全国大会　電子回路部門　課題 2
 3 : // 座席番号：●
 4 : // 氏　　名：○○　○○
 5 : //
 6 : // Copyright (C) 2010 ルネサスマイコンカーラリー事務局
 7 : ////////////////////////////////////////////////////////////////////////
 8 :
 9 : ////////////////////////////////////////////////////////////////////////
10 : // インクルードファイル設定
11 : ////////////////////////////////////////////////////////////////////////
12 : #include <machine.h>                    // H8 マイコン特有の命令取り込み
13 : #include "h8_3048.h"                   // H8/3048F-ONE 用 I/O レジスタ定義
14 : #include "common.c"                    // 共通ファイルの取り込み
15 :
16 : ////////////////////////////////////////////////////////////////////////
17 : // メイン関数
18 : ////////////////////////////////////////////////////////////////////////
19 : void main( void )
20 : {
21 :     init();                            // 内蔵周辺機能の初期化
22 :     set_ccr( 0x00 );                   // 全体割り込み許可
23 :
24 :     stepper_speed = 10;                // ｽﾃｯﾋﾟﾝｸﾞﾓｰﾀ 励磁を替える間隔[ms]
25 :
26 :     while( 1 ) {
27 :         if( tgs == 0 ) {
28 :             // トグルスイッチが"L"なら
29 :             if( ps == 1 ) {            // ホトインタラプタが遮断なら
30 :                 dc_motor = MOTOR_TOKEI;
31 :             } else {                   // ホトインタラプタが透過なら
32 :                 dc_motor = MOTOR_STOP;
33 :             }
34 :         } else {
35 :             // トグルスイッチが"H"なら
36 :             if( ps == 1 ) {            // ホトインタラプタが遮断なら
37 :                 stepper_motor = MOTOR_TOKEI;
38 :             } else {                   // ホトインタラプタが透過なら
39 :                 stepper_motor = MOTOR_STOP;
40 :             }
41 :         }
42 :     }
43 : }
44 :
45 : ////////////////////////////////////////////////////////////////////////
46 : // End of File
47 : ////////////////////////////////////////////////////////////////////////
```

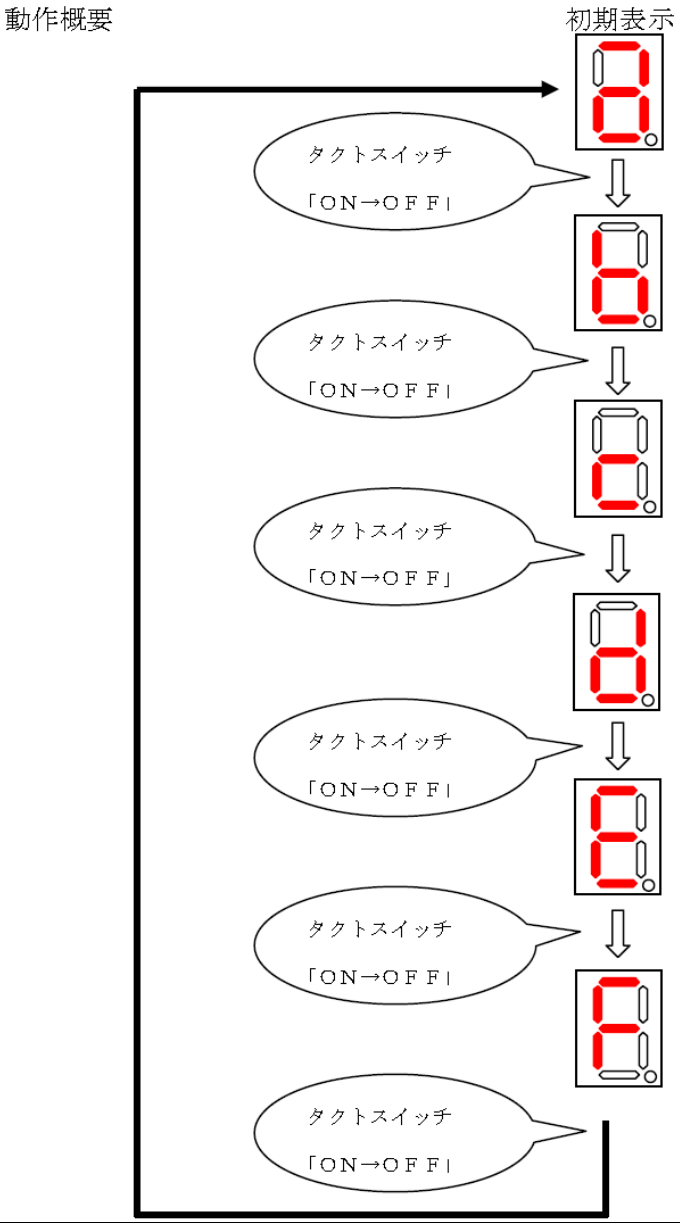
3.12.4 プログラムの解説

行	詳細
24	ステッピングモータの励磁コイルを替えるタイミングを設定します。stepper_speed 変数に「ms」単位で設定します。今回は、10msごとです。48パルスで1回転なので、48×10ms＝480msで1回転します。
27	トグルスイッチが"L"かどうかチェックします。"L"なら 29～33 行を、"H"なら else 文部分の 36～40 行を実行します。
29～33	トグルスイッチが"L"のときに実行される部分です。 ホトインタラプタが遮断されたかチェックします。遮断されたなら30行を実行して、DCモータを時計回りに回転させます。透過されたなら 32 行を実行して、DC モータを停止させます。
36～40	トグルスイッチが"H"のときに実行される部分です。 ホトインタラプタが遮断されたかチェックします。遮断されたなら 37 行を実行して、ステッピングモータを時計回りに回転させます。透過されたなら 39 行を実行して、ステッピングモータを停止させます。

3.13　課題 3

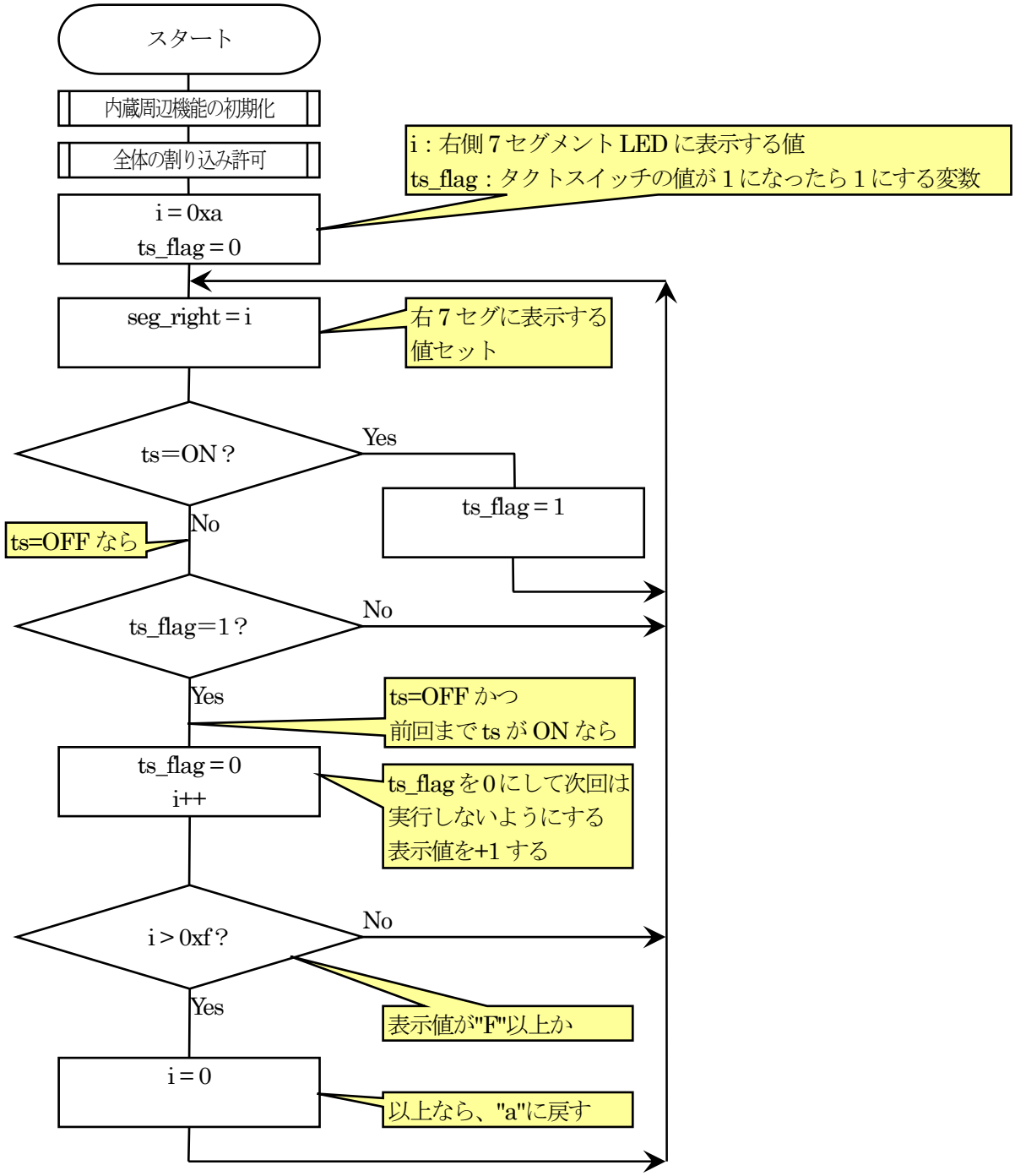
3.13.1　課題

(1) プログラムのスタート時，右側の７セグメントＬＥＤは‘ａ’を表示する。
(2) タクトスイッチを「ＯＮ→ＯＦＦ」するたびに，
ア　右側の７セグメントＬＥＤの表示が
’a’，’b’，‘c’，‘d’，’E’，’F’の順番で変化する。
イ　７セグメントＬＥＤの表示は，タクトスイッチを押したときは変化せず，離したときに変化する。
ウ　’F’の次は’a’にもどり，繰り返し動作する。



▲大会当日配付資料より抜粋

3.13.2 フローチャート



3.13.3 プログラム例

```
 1 : ////////////////////////////////////////////////////////////////////////////
 2 : // 第 10 回高校生ものづくりコンテスト全国大会　電子回路部門　課題 3
 3 : // 座席番号：●
 4 : // 氏　　名：○○　○○
 5 : //
 6 : // Copyright (C) 2010 ルネサスマイコンカーラリー事務局
 7 : ////////////////////////////////////////////////////////////////////////////
 8 :
 9 : ////////////////////////////////////////////////////////////////////////////
10 : // インクルードファイル設定
11 : ////////////////////////////////////////////////////////////////////////////
```

```
12 :  #include <machine.h>                    // H8 マイコン特有の命令取り込み
13 :  #include "h8_3048.h"                    // H8/3048F-ONE 用 I/O レジスタ定義
14 :  #include "common.c"                     // 共通ファイルの取り込み
15 :
16 :  ////////////////////////////////////////////////////////////////////////
17 :  // メイン関数
18 :  ////////////////////////////////////////////////////////////////////////
19 :  void main( void )
20 :  {
21 :      int i       = 0xa;                 // 右側 7 セグメント LED に表示する値
22 :      int ts_flag = 0;                   // ﾀｸﾄｽｲｯﾁ ON になったとき 1 にする
23 :
24 :      init();                            // 内蔵周辺機能の初期化
25 :      set_ccr( 0x00 );                   // 全体割り込み許可
26 :
27 :      while( 1 ) {
28 :          seg_right = i;                 // 右 7 セグメント LED 表示値セット
29 :
30 :          if( ts == 1 ) {
31 :              // タクトスイッチ ON なら
32 :              ts_flag = 1;               // フラグを 1 にする
33 :          } else {
34 :              // タクトスイッチ OFF なら
35 :              if( ts_flag == 1 ) {       // フラグが 1 なら
36 :                  ts_flag = 0;
37 :                  i++;                   // 表示値＋1
38 :                  if( i > 0xf ) i = 0xa; // 上限のチェック
39 :              }
40 :          }
41 :      }
42 :  }
43 :
44 :  ////////////////////////////////////////////////////////////////////////
45 :  // End of File
46 :  ////////////////////////////////////////////////////////////////////////
```

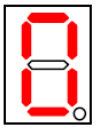
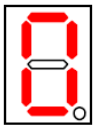
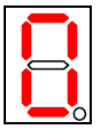
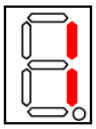
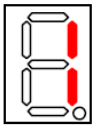
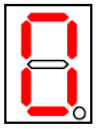
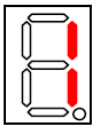
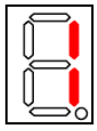
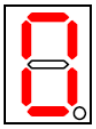
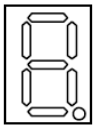
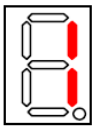
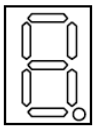
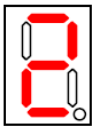
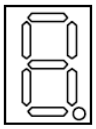
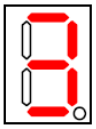
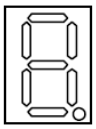
3.13.4 プログラムの解説

行	詳細
21	i 変数は、右側 7 セグメント LED に表示する値です。0xa～0xf の範囲です。
22	ts_flag 変数は、タクトスイッチの値が 1 になったら 1 にする変数です。
28	seg_right 変数は、右側 7 セグメント LED 表示する値をセットする変数です。表示値である、i 変数の値を代入しています。
32	タクトスイッチが ON なら、ts_flag 変数に 1 を代入します。
35	タクトスイッチが OFF ならこの行を実行します。ts_flag 変数が 1 かチェックします。すなわち、今のタクトスイッチが OFF で、前回のタクトスイッチが ON なら(ts_flag=1 なら)36 行以降を実行します。**この部分で、タクトスイッチが離された瞬間を検出しています。**
36	次に 35 行目を実行するとき、また実行されないように ts_flag を 0 にします。もし、この行がなければどうなるか実験するのも良いでしょう。
37	右側 7 セグメント LED に表示する値を+1 します。
38	"F"以上の場合、"a"に戻します。

3.14　課題 4

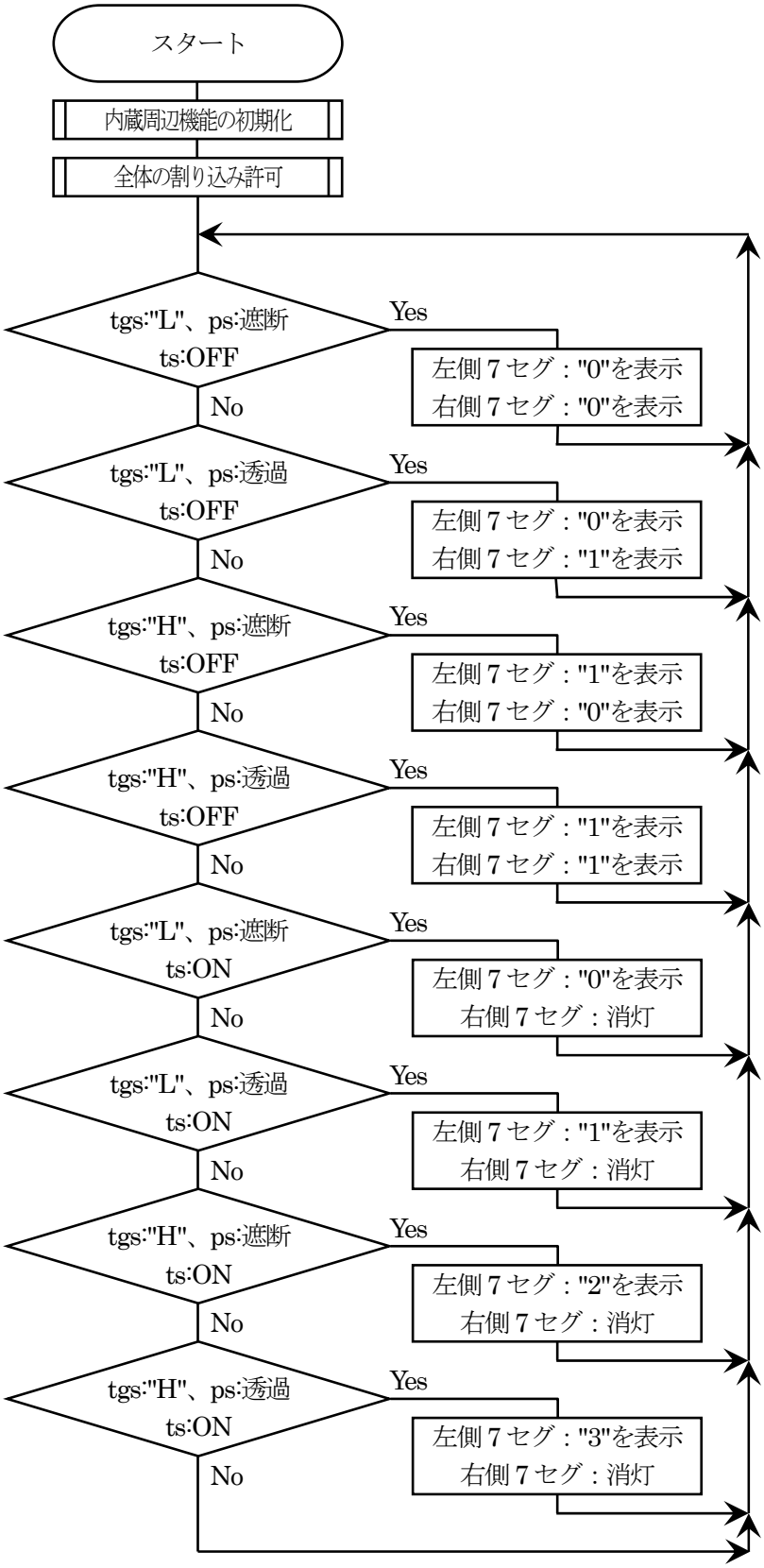
3.14.1　課題

(1) ホトインタラプタおよびトグルスイッチ，タクトスイッチの状態によって，左右の７セグメントＬＥＤを下表のように表示する。
(2) タクトスイッチがＯＦＦのとき，左右の７セグメントＬＥＤは同時に点灯し，違和感なく表示する。

トグルスイッチ ＴＧＳ	ホトインタラプタ Ｐ　Ｓ	タクトスイッチ Ｔ　Ｓ	左　側 ７セグメント ＬＥＤ	右　側 ７セグメント ＬＥＤ
Ｌ	遮断	ＯＦＦ		
Ｌ	透過	ＯＦＦ		
Ｈ	遮断	ＯＦＦ		
Ｈ	透過	ＯＦＦ		
Ｌ	遮断	ＯＮ		
Ｌ	透過	ＯＮ		
Ｈ	遮断	ＯＮ		
Ｈ	透過	ＯＮ		

▲大会当日配付資料より抜粋

3.14.2 フローチャート



※判断式は、すべて AND 条件（「tgs:○○」かつ「ps:○○」かつ「ts:○○」）

3.14.3 プログラム例

```
 1 : ////////////////////////////////////////////////////////////////////////
 2 : // 第 10 回高校生ものづくりコンテスト全国大会　電子回路部門　課題 4
 3 : // 座席番号：●
 4 : // 氏　　名：○○　○○
 5 : //
 6 : // Copyright (C) 2010 ルネサスマイコンカーラリー事務局
 7 : ////////////////////////////////////////////////////////////////////////
 8 :
 9 : ////////////////////////////////////////////////////////////////////////
10 : // インクルードファイル設定
11 : ////////////////////////////////////////////////////////////////////////
12 : #include <machine.h>                    // H8 マイコン特有の命令取り込み
13 : #include "h8_3048.h"                   // H8/3048F-ONE 用 I/O レジスタ定義
14 : #include "common.c"                    // 共通ファイルの取り込み
15 :
16 : ////////////////////////////////////////////////////////////////////////
17 : // メイン関数
18 : ////////////////////////////////////////////////////////////////////////
19 : void main( void )
20 : {
21 :     init();                            // 内蔵周辺機能の初期化
22 :     set_ccr( 0x00 );                   // 全体割り込み許可
23 :
24 :     while( 1 ) {
25 :         if( tgs == 0 && ps == 1 && ts == 0 ) {
26 :             // tgs="L" ps=遮断 ts=off なら
27 :             seg_left  = 0;
28 :             seg_right = 0;
29 :         } else if( tgs == 0 && ps == 0 && ts == 0 ) {
30 :             // tgs="L" ps=透過 ts=off なら
31 :             seg_left  = 0;
32 :             seg_right = 1;
33 :         } else if( tgs == 1 && ps == 1 && ts == 0 ) {
34 :             // tgs="H" ps=遮断 ts=off なら
35 :             seg_left  = 1;
36 :             seg_right = 0;
37 :         } else if( tgs == 1 && ps == 0 && ts == 0 ) {
38 :             // tgs="H" ps=透過 ts=off なら
39 :             seg_left  = 1;
40 :             seg_right = 1;
41 :         } else if( tgs == 0 && ps == 1 && ts == 1 ) {
42 :             // tgs="L" ps=遮断 ts=on なら
43 :             seg_left  = 0;
44 :             seg_right = SEG_NULL;
45 :         } else if( tgs == 0 && ps == 0 && ts == 1 ) {
46 :             // tgs="L" ps=透過 ts=on なら
47 :             seg_left  = 1;
48 :             seg_right = SEG_NULL;
49 :         } else if( tgs == 1 && ps == 1 && ts == 1 ) {
50 :             // tgs="H" ps=遮断 ts=on なら
51 :             seg_left  = 2;
52 :             seg_right = SEG_NULL;
53 :         } else if( tgs == 1 && ps == 0 && ts == 1 ) {
54 :             // tgs="H" ps=透過 ts=on なら
55 :             seg_left  = 3;
56 :             seg_right = SEG_NULL;
57 :         }
58 :     }
59 : }
60 :
61 : ////////////////////////////////////////////////////////////////////////
62 : // End of File
63 : ////////////////////////////////////////////////////////////////////////
```

3.14.4 プログラムの解説

if 文を使って、トグルスイッチ、ホトインタラプタ、タクトスイッチの状態を判断して、条件が成り立ったならば左側 7 セグメント LED、右側 7 セグメント LED へ指定された値を出力します。

switch～case 文を使う方法もあります。switch～case 文を使ったプログラムにもチャレンジしてみてください。

3.15 課題 5

3.15.1 課題

(1) トグルスイッチを「H」にすると，左側の７セグメントＬＥＤに‘H’，右側の７セグメントＬＥＤに‘L’が同時に違和感なく表示され，ステッピングモータが，反時計回りに回転する。
(2) (1)の状態において
ア　タクトスイッチを「ＯＮ」すると，
(ｱ) 左側の７セグメントＬＥＤに‘０’，右側の７セグメントＬＥＤに‘L’が表示され，ステッピングモータが(1)の状態よりも，低速回転で反時計回りに回転する。
(ｲ) タクトスイッチを「ＯＦＦ」にすると，(1)の状態に戻る。
イ　ホトインタラプタを「遮断」すると，
(ｱ) 左側の７セグメントＬＥＤに‘H’，右側の７セグメントＬＥＤに‘９’が表示され，ステッピングモータが(1)の状態よりも，高速回転で反時計回りに回転する。
(ｲ) ホトインタラプタを再び「透過」すると，(1)の状態に戻る。
(3) トグルスイッチを「L」にすると，左右の７セグメントＬＥＤは消灯し，ステッピングモータは停止する。
(4) トグルスイッチの「H」と「L」の切り替えは，(1)の状態のときのみ受け付ける。
(5) ステッピングモータが低速回転しているとき，ホトインタラプタの入力は受け付けない。
(6) ステッピングモータが高速回転しているとき，タクトスイッチの入力は受け付けない。
(7) 回転速度の違いは，目視で明らかに変化がわかるようにする。

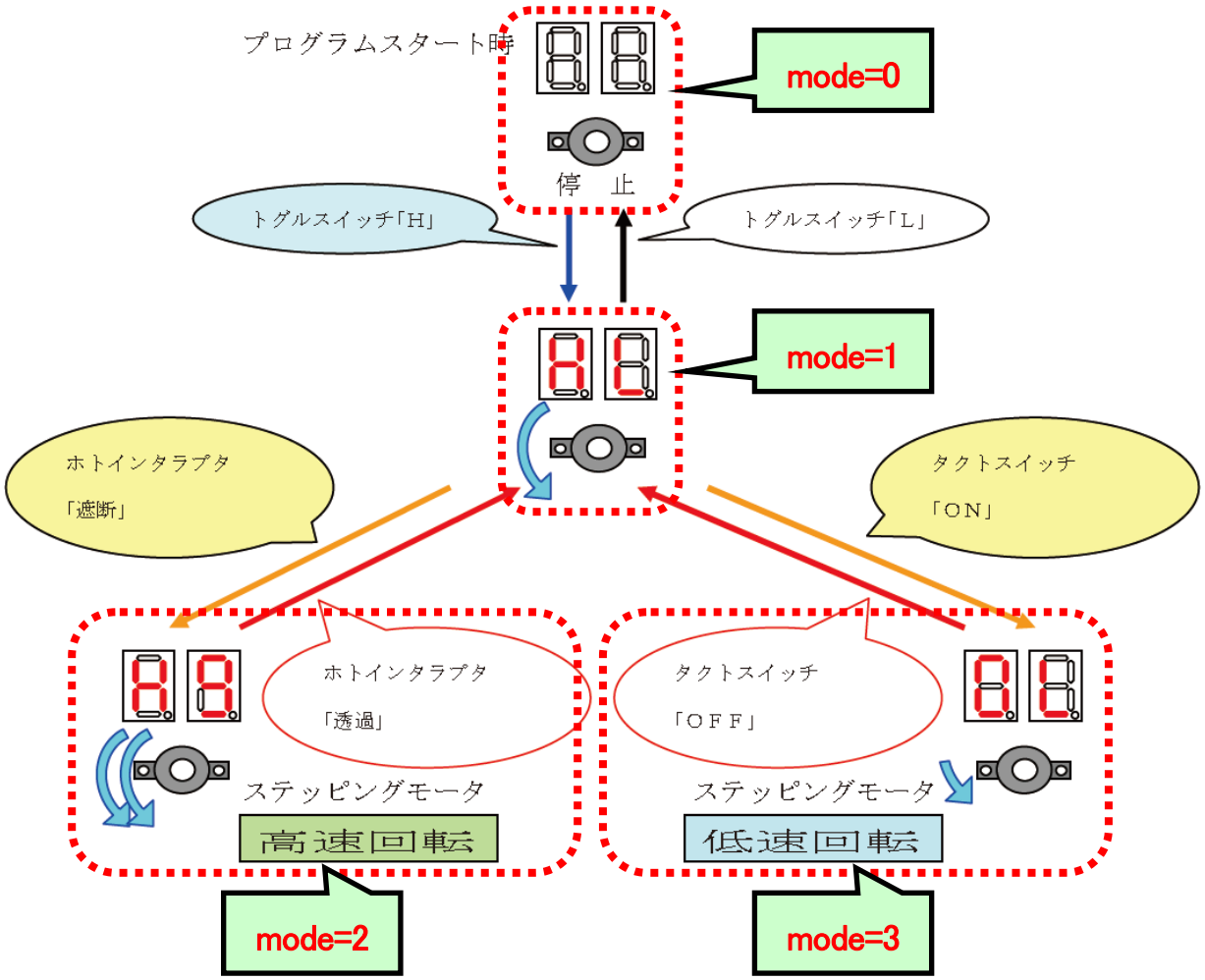
▲大会当日配付資料より抜粋

3.15.2　フローチャート

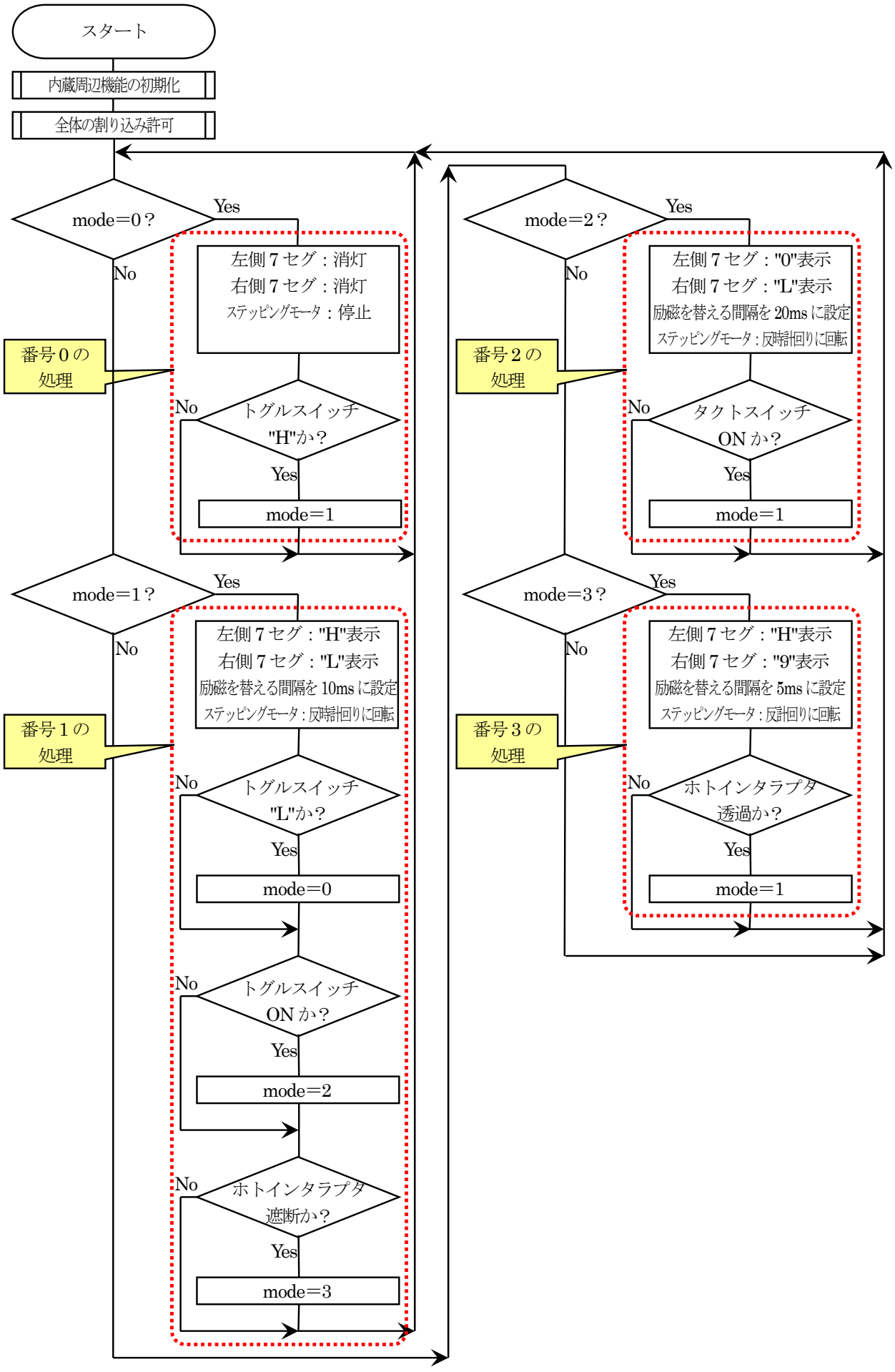
今回は、それぞれの処理に番号を付けて、番号ごとに区切ってプログラムを作ります。実際のプログラムでは、mode 変数を宣言して、この変数の値を処理番号とし、switch-case 文で処理を分けます。

処理番号とプログラムを、下表に示します。

番号	状態	左 7 セグ	右 7 セグ	ステッピングモータ	処理番号が変わる条件
0	下図の「mode=0」の部分	消灯	消灯	停止	・トグルスイッチが"H"になったら 1 番へ移る
1	下図の「mode=1」の部分	"H"表示	"H"表示	標準回転	・トグルスイッチが"L"になったら 0 番へ移る ・ホトインタラプタが遮断されたら 2 番に移る ・タクトスイッチが ON になったら 3 番に移る
2	下図の「mode=2」の部分	"H"表示	"9"表示	高速回転	・ホトインタラプタが透過したら 1 番に移る
3	下図の「mode=3」の部分	"0"表示	"L"表示	低速回転	・タクトスイッチが OFF になったら 1 番に移る



表に従って、フローチャートを作ります。


スタート
内蔵周辺機能の初期化
全体の割り込み許可
mode＝0？
Yes
No
左側 7 セグ：消灯
右側 7 セグ：消灯
ステッピングモータ：停止
番号 0 の処理
トグルスイッチ "H"か？
No
Yes
mode＝1
mode＝1？
Yes
No
左側 7 セグ："H"表示
右側 7 セグ："L"表示
励磁を替える間隔を 10ms に設定
ステッピングモータ：反時計回りに回転
番号 1 の処理
トグルスイッチ "L"か？
No
Yes
mode＝0
トグルスイッチ ON か？
No
Yes
mode＝2
ホトインタラプタ 遮断か？
No
Yes
mode＝3
mode＝2？
Yes
No
左側 7 セグ："0"表示
右側 7 セグ："L"表示
励磁を替える間隔を 20ms に設定
ステッピングモータ：反時計回りに回転
番号 2 の処理
タクトスイッチ ON か？
No
Yes
mode＝1
mode＝3？
Yes
No
左側 7 セグ："H"表示
右側 7 セグ："9"表示
励磁を替える間隔を 5ms に設定
ステッピングモータ：反計回りに回転
番号 3 の処理
ホトインタラプタ 透過か？
No
Yes
mode＝1

3.15.3 プログラム例

```
 1 : ////////////////////////////////////////////////////////////////////////
 2 : // 第 10 回高校生ものづくりコンテスト全国大会　電子回路部門　課題 5
 3 : // 座席番号：●
 4 : // 氏　　名：○○　○○
 5 : //
 6 : // Copyright (C) 2010 ルネサスマイコンカーラリー事務局
 7 : ////////////////////////////////////////////////////////////////////////
 8 :
 9 : ////////////////////////////////////////////////////////////////////////
10 : // インクルードファイル設定
11 : ////////////////////////////////////////////////////////////////////////
12 : #include <machine.h>                    // H8 マイコン特有の命令取り込み
13 : #include "h8_3048.h"                   // H8/3048F-ONE 用 I/O レジスタ定義
14 : #include "common.c"                    // 共通ファイルの取り込み
15 :
16 : ////////////////////////////////////////////////////////////////////////
17 : // メイン関数
18 : ////////////////////////////////////////////////////////////////////////
19 : void main( void )
20 : {
21 :     int mode = 0;                      // 処理番号
22 :
23 :     init();                            // 内蔵周辺機能の初期化
24 :     set_ccr( 0x00 );                   // 全体割り込み許可
25 :
26 :     while( 1 ) {
27 :         switch( mode ) {
28 :             case 0:
29 :                 // トグルスイッチ"L"なら
30 :                 seg_left  = SEG_NULL;
31 :                 seg_right = SEG_NULL;
32 :                 stepper_motor = MOTOR_STOP;
33 :
34 :                 if( tgs == 1 ) mode = 1; // トグルスイッチ"H"なら
35 :                 break;
36 :
37 :             case 1:
38 :                 // トグルスイッチ"H"なら
39 :                 seg_left  = SEG_H;
40 :                 seg_right = SEG_L;
41 :                 stepper_speed = 10;     // ｽﾃｯﾋﾟﾝｸﾞﾓｰﾀ 励磁を替える間隔[ms]
42 :                 stepper_motor = MOTOR_HANTOKEI;
43 :
44 :                 if( tgs == 0 ) mode = 0; // トグルスイッチ"L"なら
45 :                 if( ts  == 1 ) mode = 2; // タクトスイッチ ON なら
46 :                 if( ps  == 1 ) mode = 3; // ホトインタラプタ遮断なら
47 :                 break;
48 :
49 :             case 2:
50 :                 // タクトスイッチ ON なら
51 :                 seg_left  = 0;
52 :                 seg_right = SEG_L;
53 :                 stepper_speed = 20;     // ｽﾃｯﾋﾟﾝｸﾞﾓｰﾀ 励磁を替える間隔[ms]
54 :                 stepper_motor = MOTOR_HANTOKEI;
55 :
56 :                 if( ts == 0 ) mode = 1; // タクトスイッチ OFF なら
57 :                 break;
58 :
59 :             case 3:
60 :                 // ホトインタラプト遮断なら
61 :                 seg_left  = SEG_H;
62 :                 seg_right = 9;
63 :                 stepper_speed = 5;      // ｽﾃｯﾋﾟﾝｸﾞﾓｰﾀ 励磁を替える間隔[ms]
64 :                 stepper_motor = MOTOR_HANTOKEI;
65 :
66 :                 if( ps == 0 ) mode = 1; // ホトインタラプタ透過なら
67 :                 break;
68 :         }
```

```
69 :       }
70 :  }
71 :
72 :  ////////////////////////////////////////////////////////////////////////////
73 :  // End of File
74 :  ////////////////////////////////////////////////////////////////////////////
```

3.15.4 プログラムの解説

行	詳細
27	mode 変数の値の確認は、switch-case 文を使いました。if 文でも構いません。
30～35	番号 0 の処理です。 タクトスイッチが"H"になったら番号を 1 にして、次に switch-csae 文を実行するときは case 1 の部分を実行します。
39～47	番号 1 の処理です。 トグルスイッチが"L"になったら番号を 0 にして、次に switch-csae 文を実行するときは case 0 の部分を実行します。 タクトスイッチが ON になったら番号を 2 にして、次に switch-csae 文を実行するときは case 2 の部分を実行します。 ホトインタラプタが遮断されたなら番号を 3 にして、次に switch-csae 文を実行するときは case 3 の部分を実行します。
51～57	番号 2 の処理です。 タクトスイッチが OFF になったら番号を 1 にして、次に switch-csae 文を実行するときは case 1 の部分を実行します。
61～67	番号 3 の処理です。 ホトインタラプタが透過されたなら番号を 1 にして、次に switch-csae 文を実行するときは case 1 の部分を実行します。

3.16　課題 6

3.16.1　課題

(1) プログラムスタート時，右側の７セグメントＬＥＤに‘０’が表示され，２つのモータは停止している。
(2) 右側の７セグメントＬＥＤに‘０’が表示されているとき
ア　トグルスイッチを「H」にし，タクトスイッチを１回「ＯＮ→ＯＦＦ」すると，押した直後に，ステッピングモータが時計回りに回転する。
イ　ステッピングモータが，１回転を終了するたびに，右の７セグメントＬＥＤの表示が，１ずつ増えていく。ステッピングモータは停止することなく回転する。
ウ　３回転を終了し，右側の７セグメントＬＥＤの表示が，‘３’になったとき，ステッピングモータは停止する。７セグメントＬＥＤには，そのまま‘３’が表示されている。
(3) 右側の７セグメントＬＥＤに‘３’が表示されているとき
ア　トグルスイッチを「L」にし，タクトスイッチを１回「ＯＮ→ＯＦＦ」すると，押した直後に，ステッピングモータが反時計回りに回転する。
イ　ステッピングモータが，１回転を終了するたびに，右側の７セグメントＬＥＤの表示が，１ずつ減っていく。ステッピングモータは停止することなく回転する。
ウ　３回転を終了し，右側の７セグメントＬＥＤの表示が，‘０’になったとき，ステッピングモータは停止する。７セグメントＬＥＤには，そのまま‘０’が表示されている。
(4) ステッピングモータが回転している間は，いかなる入力も受け付けない。

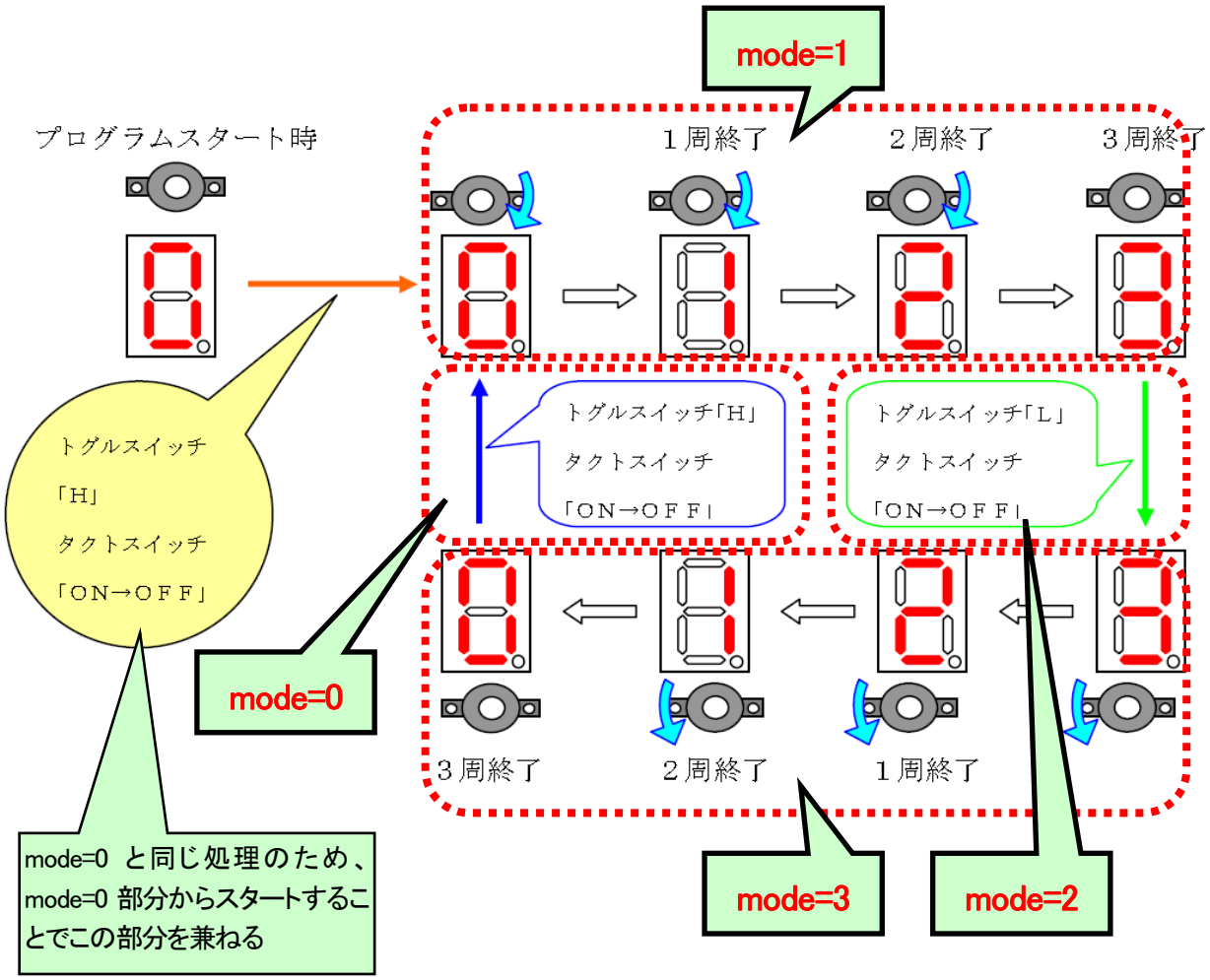
▲大会当日配付資料より抜粋

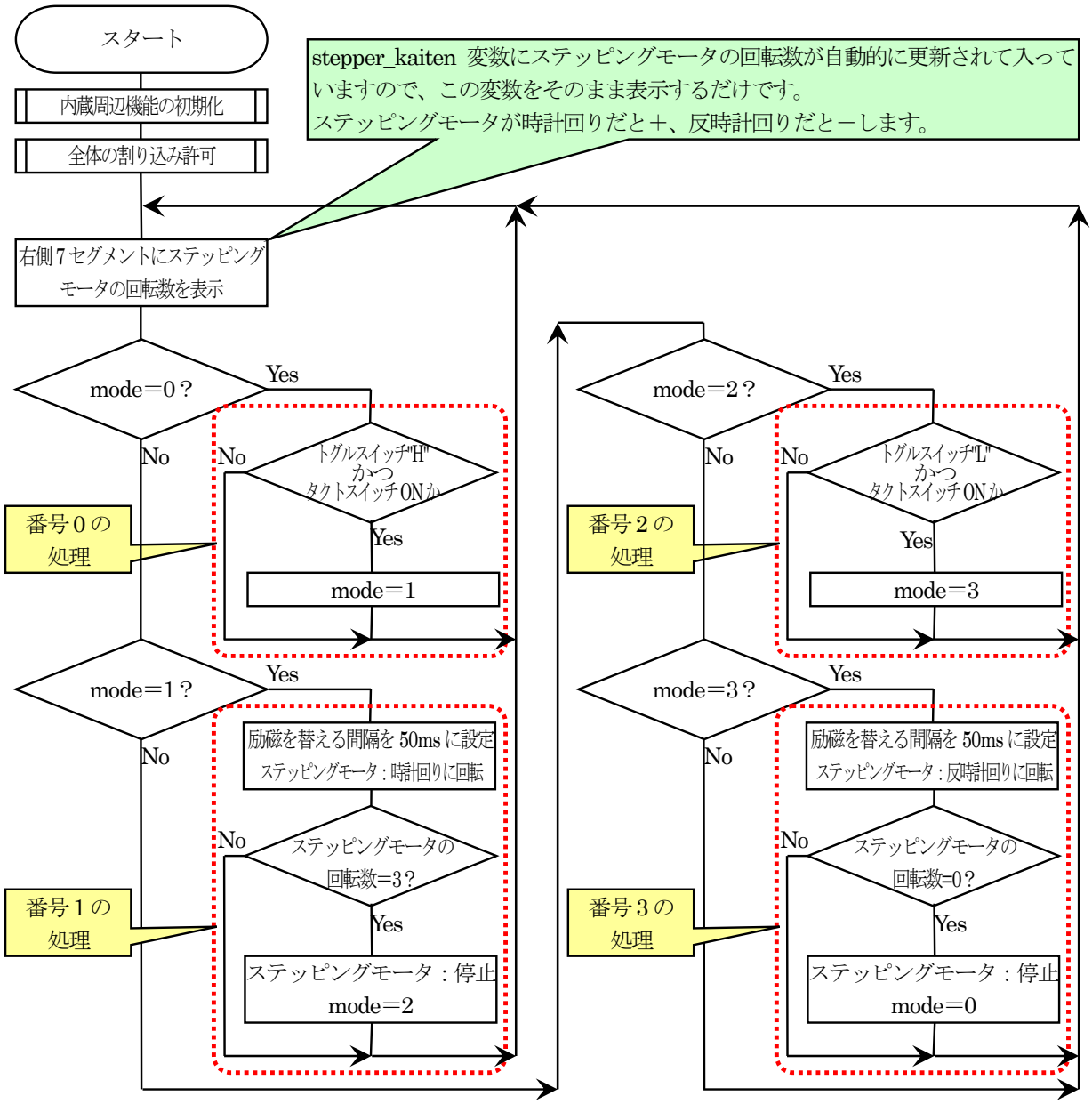
3.16.2　フローチャート

今回は、それぞれの処理に番号を付けて、番号ごとに区切ってプログラムを作ります。実際のプログラムでは、mode 変数を宣言して、この変数の値を処理番号とし、switch-case 文で処理を分けます。

処理番号とプログラムを、下表に示します。

番号	状態	右 7 セグ	ステッピングモータ	処理番号が変わる条件
0	下図の「mode=0」の部分	″0″表示	停止	・トグルスイッチが″H″、かつタクトスイッチが ON になった瞬間に 1 番へ移る
1	下図の「mode=1」の部分	ステッピングモータの回転に合わせて″0″→″1″→″2″→″3″表示	時計回りに 3 回転	・時計回りに 3 回転したなら、2 番へ移る
2	下図の「mode=2」の部分	″3″表示	停止	・トグルスイッチが″L″、かつタクトスイッチが ON になった瞬間に 1 番へ移る
3	下図の「mode=3」の部分	ステッピングモータの回転に合わせて″3″→″2″→″1″→″0″表示	反時計回りに 3 回転	・反時計回りに 3 回転したなら、0 番へ移る





3.16.3 プログラム例

```
 1 : ////////////////////////////////////////////////////////////////////////////
 2 : // 第 10 回高校生ものづくりコンテスト全国大会　電子回路部門　課題 6
 3 : // 座席番号：●
 4 : // 氏　　名：○○　○○
 5 : //
 6 : // Copyright (C) 2010 ルネサスマイコンカーラリー事務局
 7 : ////////////////////////////////////////////////////////////////////////////
 8 :
 9 : ////////////////////////////////////////////////////////////////////////////
10 : // インクルードファイル設定
11 : ////////////////////////////////////////////////////////////////////////////
12 : #include <machine.h>                    // H8 マイコン特有の命令取り込み
13 : #include "h8_3048.h"                   // H8/3048F-ONE 用 I/O レジスタ定義
14 : #include "common.c"                    // 共通ファイルの取り込み
15 :
```

```
16 : ////////////////////////////////////////////////////////////////////////
17 : // メイン関数
18 : ////////////////////////////////////////////////////////////////////////
19 : void main( void )
20 : {
21 :     int mode = 0;                      // 処理番号
22 :
23 :     init();                            // 内蔵周辺機能の初期化
24 :     set_ccr( 0x00 );                   // 全体割り込み許可
25 :
26 :     while( 1 ) {
27 :         seg_right = stepper_kaiten;    // ｽﾃｯﾋﾟﾝｸﾞﾓｰﾀの回転数を表示
28 :
29 :         switch( mode ) {
30 :             case 0:
31 :                 // トグルスイッチ"H"、タクトスイッチ ON 待ち
32 :                 if( tgs == 1 && ts == 1 ) {
33 :                     mode = 1;
34 :                 }
35 :                 break;
36 :
37 :             case 1:
38 :                 // 7 セグ："0"→"1"→"2"→"3"表示、ｽﾃｯﾋﾟﾝｸﾞﾓｰﾀ時計回り処理
39 :                 stepper_speed = 50;     // ｽﾃｯﾋﾟﾝｸﾞﾓｰﾀ 励磁を替える間隔[ms]
40 :                 stepper_motor = MOTOR_TOKEI;
41 :
42 :                 if( stepper_kaiten == 3 ) { // 3 回転した？
43 :                     stepper_motor = MOTOR_STOP;
44 :                     mode = 2;
45 :                 }
46 :                 break;
47 :
48 :             case 2:
49 :                 // トグルスイッチ"L"、タクトスイッチ ON 待ち
50 :                 if( tgs == 0 && ts == 1 ) {
51 :                     mode = 3;
52 :                 }
53 :                 break;
54 :
55 :             case 3:
56 :                 // 7 セグ："3"→"2"→"1"→"0"表示、ｽﾃｯﾋﾟﾝｸﾞﾓｰﾀ反時計回り処理
57 :                 stepper_speed = 50;     // ｽﾃｯﾋﾟﾝｸﾞﾓｰﾀ 励磁を替える間隔[ms]
58 :                 stepper_motor = MOTOR_HANTOKEI;
59 :
60 :                 if( stepper_kaiten == 0 ) { // 0 に戻った？
61 :                     stepper_motor = MOTOR_STOP;
62 :                     mode = 0;
63 :                 }
64 :                 break;
65 :         }
66 :     }
67 : }
68 :
69 : ////////////////////////////////////////////////////////////////////////
70 : // End of File
71 : ////////////////////////////////////////////////////////////////////////
```

3.16.4 プログラムの解説

行	詳細
27	stepper_kaiten 変数の値は「common.c」で随時更新されている変数です。この変数は、ステッピングモータが時計回りに1回転したなら＋1，反時計回りに1回転したなら－1します。 今回の問題は、 (2)イ　ステッピングモータが、(時計回りに)1回転を終了するたびに、右の7セグメント LED の表示が、1ずつ増えていく。 (3)イ　ステッピングモータが、(反時計回りに)1回転を終了するたびに、右の7セグメント LED の表示が、1ずつ減っていく。 となっていますので、stepper_kaiten 変数の値をそのまま右側 7 セグメント LED に表示すれば良いことになります。 ステッピングモータを制御する割り込みプログラムが大変だった分、main 関数のプログラムが非常に簡潔にできます。
32～35	番号 0 の処理です。 トグルスイッチが"H"、かつタクトスイッチが ON になったなら番号を 1 にします。次に switch-csae 文を実行するときは、case 1 の部分を実行します。
39～46	番号 1 の処理です。 ステッピングモータの励磁コイルを替える時間を 50ms ごとにして、時計回りに回転させます。 3 回転したならば(stepper_kaiten 変数が 3 になったなら)、ステッピングモータを停止させ番号を 2 にします。次に switch-csae 文を実行するときは、case 2 の部分を実行します。
50～53	番号 2 の処理です。 トグルスイッチが"L"、かつタクトスイッチが ON になったなら番号を 3 にします。次に switch-csae 文を実行するときは、case 3 の部分を実行します。
57～64	番号 3 の処理です。 ステッピングモータの励磁コイルを替える時間を 50ms ごとにして、反時計回りに回転させます。 ステッピングモータは反時計回りに回っているので、1 回転するごとに stepper_kaiten 変数は－1します。 反時計回りに 3 回転したならば(stepper_kaiten 変数が 0 になったなら)、ステッピングモータを停止させ番号を 0 にします。次に switch-csae 文を実行するときは、case 0 の部分を実行します。

3.17　課題 7

3.17.1　課題

課題 7 の問題を、下記に示します。

(1) プログラムスタート時，右側の７セグメントＬＥＤに’0’を表示する。
(2) タクトスイッチを１回「ＯＮ→ＯＦＦ」すると，
　ア　約１秒間隔で右側の７セグメントＬＥＤの表示が，
　　’0’ →’1’ → ’2’ → ‘3’ → ‘4’ → ’5’ → ’6’ → ‘7’ → ’8’ → ’9’　のように変化する。
　イ　’9’の次は’0’が表示され，繰り返し動作する。
　ウ　タクトスイッチを再び「ＯＮ→ＯＦＦ」すると，押されたときの表示で停止する。
　　停止している数値は，７セグメントＬＥＤに表示されている。
　エ　上記アからウの動作は，繰り返し実行できる。
　　ただし，カウントアップは，停止している表示から始まるものとする。
　オ　７セグメントＬＥＤの動作および停止は，タクトスイッチが**押された瞬間**に行われる。
(3) ７セグメントＬＥＤの表示が停止しているとき，以下の動作をする。
　　ただし，いずれかのモータが回転しているとき，タクトスイッチの入力は受け付けない。
　ア　表示されている数値が’0’から’4’
　　(ｱ) ホトインタラプタを１回「遮断→透過」すると，ステッピングモータが回転する。
　　　(a) トグルスイッチが「Ｌ」ならば，時計回りに回転する。
　　　(b) トグルスイッチが「Ｈ」ならば，反時計回りに回転する。
　　　(c) (a)および(b)の動作は，ステッピングモータ回転中に何度でも切り替えられる。
　　(ｲ) トグルスイッチの状態にかかわらず，ホトインタラプタを再び「遮断→透過」すると，ステッピングモータは停止する。
　イ　表示されている数値が’5’から’9’
　　(ｱ) ホトインタラプタを１回「遮断→透過」すると，ＤＣモータが回転する。
　　　(a) トグルスイッチが「Ｌ」ならば，時計回りに回転する。
　　　(b) トグルスイッチが「Ｈ」ならば，反時計回りに回転する。
　　　(c) (a)および(b)の動作は，ＤＣモータ回転中に何度でも切り替えられる。
　　(ｲ) トグルスイッチの状態にかかわらず，ホトインタラプタを再び「遮断→透過」すると，ＤＣモータは停止する。
　ウ　ステッピングモータおよびＤＣモータの動作および停止は，ホトインタラプタが**遮断された瞬間**に行われる。
(4) いずれかのモータが回転している間，停止している数値は，７セグメントＬＥＤに表示されている。

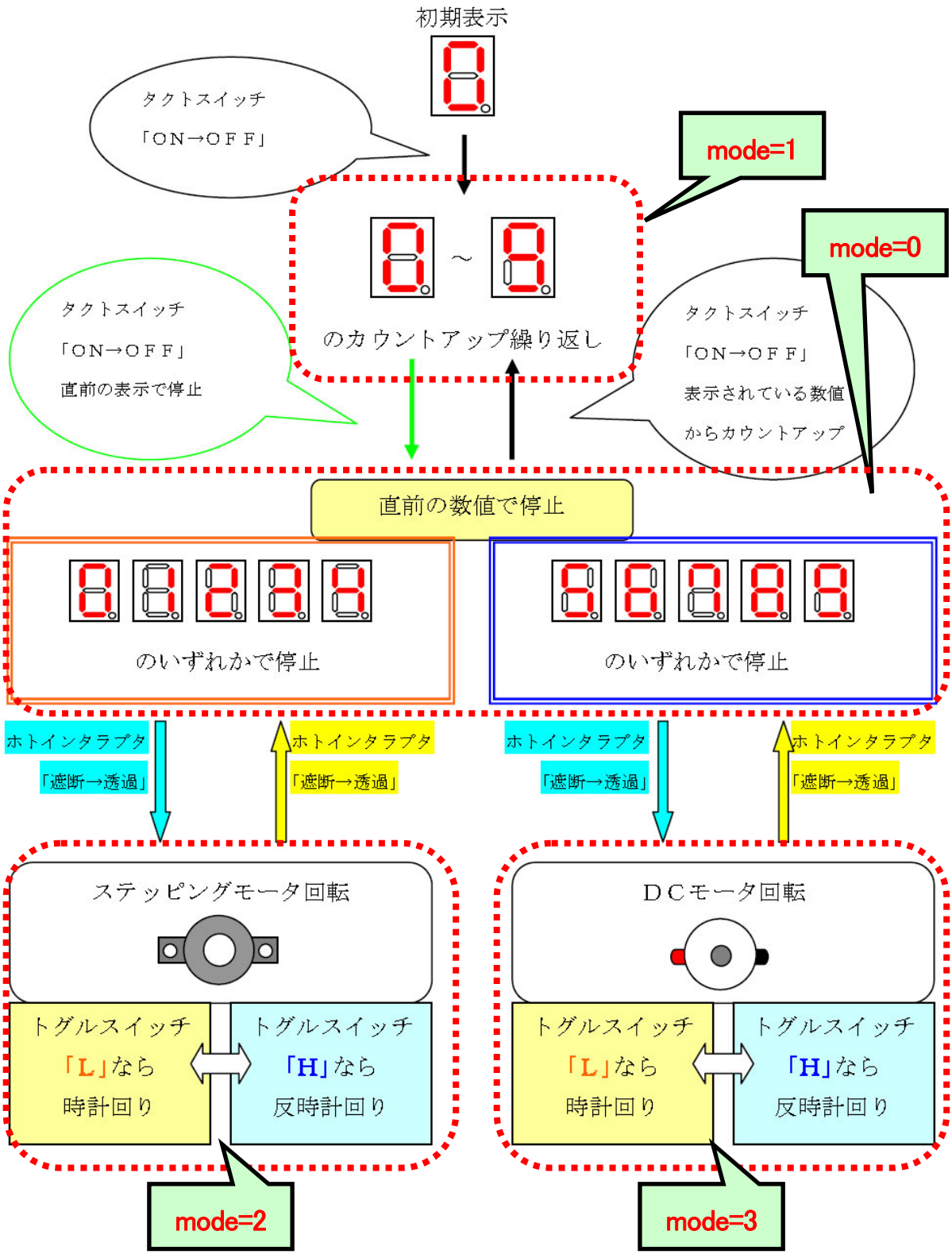
▲大会当日配付資料より抜粋

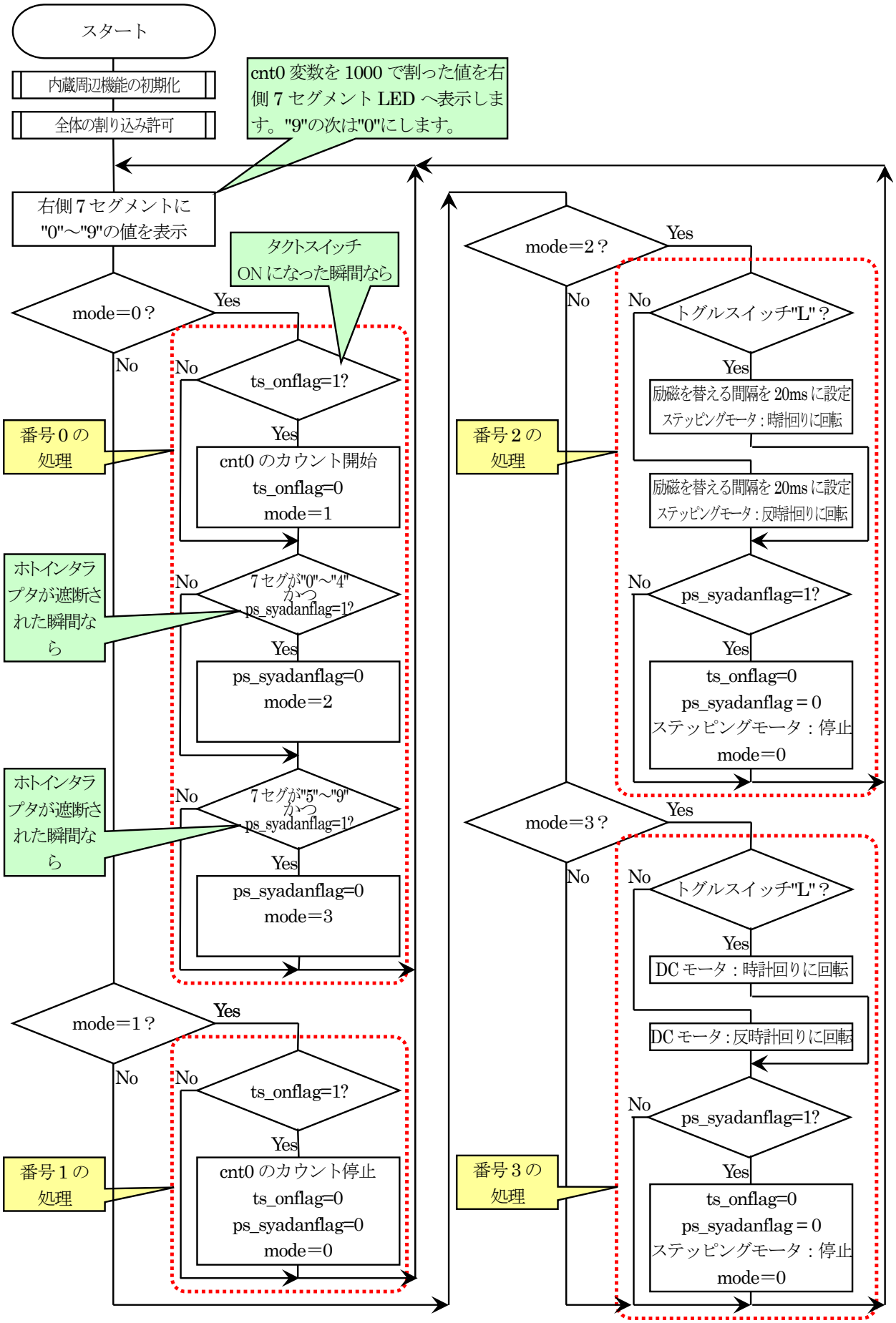
3.17.2 フローチャート

今回は、それぞれの処理に番号を付けて、番号ごとに区切ってプログラムを作ります。実際のプログラムでは、mode 変数を宣言して、この変数の値を処理番号とし、switch-case 文で処理を分けます。

処理番号とプログラムを、下表に示します。

番号	状態	右側 7 セグメント LED	ステッピングモータ	DC モータ	処理番号が変わる条件
0	図の「mode=0」の部分	″0″～″9″の間で停止	停止	停止	・タクトスイッチが押された瞬間に 1 番へ移る ・右側 7 セグメント LED の値が″0″～″4″でホトインタラプタが遮断された瞬間に 2 番へ移る ・右側 7 セグメント LED の値が″5″～″9″でホトインタラプタが遮断された瞬間に 2 番へ移る
1	図の「mode=1」の部分	1 秒ごとに＋1 ″9″の次は、″0″に戻る	停止	停止	・タクトスイッチが押された瞬間に 0 番へ移る
2	図の「mode=2」の部分	前のままの表示	トグルスイッチが″L″なら時計回り、″H″なら反時計回り	停止	・ホトインタラプタが遮断された瞬間に 0 番へ移る
3	図の「mode=3」の部分	前のままの表示	停止	トグルスイッチが″L″なら時計回り、″H″なら反時計回り	・ホトインタラプタが遮断された瞬間に 0 番へ移る


初期表示
タクトスイッチ
「ON→OFF」
mode=1
mode=0
～
のカウントアップ繰り返し
タクトスイッチ
「ON→OFF」
直前の表示で停止
タクトスイッチ
「ON→OFF」
表示されている数値
からカウントアップ
直前の数値で停止
のいずれかで停止
のいずれかで停止
ホトインタラプタ
「遮断→透過」
ホトインタラプタ
「遮断→透過」
ホトインタラプタ
「遮断→透過」
ホトインタラプタ
「遮断→透過」
ステッピングモータ回転
トグルスイッチ
「L」なら
時計回り
トグルスイッチ
「H」なら
反時計回り
DCモータ回転
トグルスイッチ
「L」なら
時計回り
トグルスイッチ
「H」なら
反時計回り
mode=2
mode=3


スタート
内蔵周辺機能の初期化
全体の割り込み許可
cnt0 変数を 1000 で割った値を右側 7 セグメント LED へ表示します。"9"の次は"0"にします。
右側 7 セグメントに "0"～"9"の値を表示
タクトスイッチ ON になった瞬間なら
mode＝0？
Yes
No
No
ts_onflag=1?
Yes
番号 0 の処理
cnt0 のカウント開始
ts_onflag=0
mode＝1
ホトインタラプタが遮断された瞬間なら
No
7 セグが"0"～"4" かつ ps_syadanflag=1?
Yes
ps_syadanflag=0
mode＝2
ホトインタラプタが遮断された瞬間なら
No
7 セグが"5"～"9" かつ ps_syadanflag=1?
Yes
ps_syadanflag=0
mode＝3
mode＝1？
Yes
No
No
ts_onflag=1?
Yes
番号 1 の処理
cnt0 のカウント停止
ts_onflag=0
ps_syadanflag=0
mode＝0
mode＝2？
Yes
No
No
トグルスイッチ"L"？
Yes
番号 2 の処理
励磁を替える間隔を 20ms に設定
ステッピングモータ：時計回りに回転
励磁を替える間隔を 20ms に設定
ステッピングモータ：反時計回りに回転
No
ps_syadanflag=1?
Yes
ts_onflag=0
ps_syadanflag＝0
ステッピングモータ：停止
mode＝0
mode＝3？
Yes
No
No
トグルスイッチ"L"？
Yes
番号 3 の処理
DC モータ：時計回りに回転
DC モータ：反時計回りに回転
No
ps_syadanflag=1?
Yes
ts_onflag=0
ps_syadanflag＝0
ステッピングモータ：停止
mode＝0

3.17.3 プログラム例

```
 1 : ////////////////////////////////////////////////////////////////////////
 2 : // 第 10 回高校生ものづくりコンテスト全国大会　電子回路部門　課題 7
 3 : // 座席番号：●
 4 : // 氏　　名：○○　○○
 5 : //
 6 : // Copyright (C) 2010 ルネサスマイコンカーラリー事務局
 7 : ////////////////////////////////////////////////////////////////////////
 8 :
 9 : ////////////////////////////////////////////////////////////////////////
10 : // インクルードファイル設定
11 : ////////////////////////////////////////////////////////////////////////
12 : #include <machine.h>                    // H8 マイコン特有の命令取り込み
13 : #include "h8_3048.h"                   // H8/3048F-ONE 用 I/O レジスタ定義
14 : #include "common.c"                    // 共通ファイルの取り込み
15 :
16 : ////////////////////////////////////////////////////////////////////////
17 : // メイン関数
18 : ////////////////////////////////////////////////////////////////////////
19 : void main( void )
20 : {
21 :     int     mode     = 0;              // 処理番号
22 :     int     i        = 0;              // 右側 7 セグメント LED の値
23 :
24 :     init();                            // 内蔵周辺機能の初期化
25 :     set_ccr( 0x00 );                   // 全体割り込み許可
26 :
27 :     while( 1 ) {
28 :         if( cnt0 >= 10000 ) cnt0 = 0;
29 :         i = cnt0 / 1000;
30 :         seg_right = i;
31 :
32 :         switch( mode ) {
33 :             case 0:
34 :                 // 右側 7 セグメント LED　停止中
35 :                 if( ts_onflag == 1 ) {
36 :                     // タクトスイッチ ON なら
37 :                     cnt0_flag = 1;      // カウンタのカウントアップ開始
38 :                     ts_onflag = 0;
39 :                     mode = 1;
40 :                 }
41 :
42 :                 if( i >= 0 && i <= 4 && ps_syadanflag == 1 ) {
43 :                     // 表示が 0～4 で、ホトインタラプタ遮断なら
44 :                     ps_syadanflag = 0;
45 :                     mode = 2;
46 :                 }
47 :
48 :                 if( i >= 5 && i <= 9 && ps_syadanflag == 1 ) {
49 :                     // 表示が 0～4 で、ホトインタラプタ遮断なら
50 :                     ps_syadanflag = 0;
51 :                     mode = 3;
52 :                 }
53 :                 break;
54 :
55 :             case 1:
56 :                 // 右側 7 セグメント LED　動作中
57 :                 if( ts_onflag == 1 ) {
58 :                     // タクトスイッチ ON なら
59 :                     cnt0_flag     = 0;  // カウンタのカウントアップ停止
60 :                     ts_onflag     = 0;
61 :                     ps_syadanflag = 0;
62 :                     mode = 0;
63 :                 }
64 :                 break;
65 :
```

```
 66 :                case 2:
 67 :                    // ステッピングモータ回転処理
 68 :                    if( tgs == 0 ) {        // ﾄｸﾞﾙｽｲｯﾁによって回転方向を替える
 69 :                        stepper_speed = 20; // ｽﾃｯﾋﾟﾝｸﾞﾓｰﾀ 励磁を替える間隔[ms]
 70 :                        stepper_motor = MOTOR_TOKEI;
 71 :                    } else {
 72 :                        stepper_speed = 20; // ｽﾃｯﾋﾟﾝｸﾞﾓｰﾀ 励磁を替える間隔[ms]
 73 :                        stepper_motor = MOTOR_HANTOKEI;
 74 :                    }
 75 :
 76 :                    if( ps_syadanflag == 1 ) {
 77 :                        // ホトインタラプト遮断なら
 78 :                        ts_onflag     = 0;
 79 :                        ps_syadanflag = 0;
 80 :                        stepper_motor = MOTOR_STOP;
 81 :                        mode = 0;
 82 :                    }
 83 :                    break;
 84 :
 85 :                case 3:
 86 :                    // DC モータ回転処理
 87 :                    if( tgs == 0 ) {        // ﾄｸﾞﾙｽｲｯﾁによって回転方向を替える
 88 :                        dc_motor = MOTOR_TOKEI;
 89 :                    } else {
 90 :                        dc_motor = MOTOR_HANTOKEI;
 91 :                    }
 92 :
 93 :                    if( ps_syadanflag == 1 ) {
 94 :                        // ホトインタラプト遮断なら
 95 :                        ts_onflag     = 0;
 96 :                        ps_syadanflag = 0;
 97 :                        dc_motor = MOTOR_STOP;
 98 :                        mode = 0;
 99 :                    }
100 :                    break;
101 :            }
102 :        }
103 :    }
104 :
105 :    ////////////////////////////////////////////////////////////////////////
106 :    // End of File
107 :    ////////////////////////////////////////////////////////////////////////
```

3.17.4 プログラムの解説

行	詳細
28～30	右側 7 セグメント LED に"0"～"9"の値を表示します。 cnt0 変数は、1ms ごとにカウントアップする変数です。カウントアップは、割り込みプログラム内で自動的に行われます。 課題は、1 秒ごとにカウントアップさせるので cnt0 変数の値を 1000 で割り、その値を 7 セグメント LED に表示しています。 "9"の次は"0"にするので、cnt0 変数としては、10000(＝10 秒)以上になったなら 0 にしています。 cnt0 変数は、cnt0_flag 変数が 1 なら(正確には 0 以外なら)1ms ごとにカウントアップ、0 なら停止します。
35～53	番号 0 の処理です。 7 セグメント LED を"0"～"9"の間で表示させながら、ホトインタラプタが遮断された瞬間を検出する部分です。
35	ts_onflag 変数は、タクトスイッチが OFF から ON になった瞬間に 1 になる変数です。この変数が 1 になったら、タクトスイッチが押されたと判断してします。 if 文などで ts_onflag 変数が 1 になったことを検出したら、プログラムで ts_onflag 変数を 0 にしておきます。0 にしないと 1 のままなので、ずっと「ON になった瞬間」となってしまいます。今回は、38 行で 0 にしています。

37	cnt0_flag 変数を 1 にして cnt0 変数を 1ms ごとにカウントアップさせます。この値を 1/1000 にして 7 セグメント LED に表示させます。
42	7 セグメント LED の表示値が"0"～"4"の間で、かつホトインタラプタが遮断された瞬間を検出したなら、カッコの中を実行します。処理番号を 2 番にします。 ps_syadanflag 変数は、ホトインタラプタが遮断された瞬間に 1 になる変数です。 if 文などで ps_syadanfla 変数が 1 になったことを検出したら、プログラムで ps_syadanfla 変数を 0 にしておきます。0 にしないと 1 のままなので、ずっと「遮断された瞬間」となってしまいます。今回は、44 行で 0 にしています。
48	7 セグメント LED の表示値が"5"～"9"の間で、かつホトインタラプタが遮断された瞬間を検出したなら、カッコの中を実行します。処理番号を 3 番にします。
57～64	番号 1 の処理です。 7 セグメント LED を"0"～"9"に 1 秒ごとにカウントアップ表示する部分です。ただし、カウントアップ処理は割り込みプログラムで行われているので、番号 1 はタクトスイッチが ON になった瞬間かどうかチェックしているだけです。
57	タクトスイッチが ON になった瞬間かチェックします。ON なら cnt0_flag 変数を 0 にして、cnt0 変数のカウントアップを停止、番号 0 に移ります。
68～83	番号 2 の処理です。 ステッピングモータを回します。
68	トグルスイッチをチェックして、"L"ならステッピングモータを時計回りに、"H"なら反時計回りに回します。
76	ホトインタラプタが遮断されたことを検出したならステッピングモータの動作を停止させ、番号 0 の処理に移ります。
87～100	番号 3 の処理です。 DC モータを回します。
87	トグルスイッチをチェックして、"L"なら DC モータを時計回りに、"H"なら反時計回りに回します。
93	ホトインタラプタが遮断されたことを検出したなら DC モータの動作を停止させ、番号 0 の処理に移ります。

第 10 回高校生ものづくりコンテスト全国大会　電子回路組立部門
プログラム解説マニュアル

発行年月日	2010 年 12 月 28 日	第 1.00 版
	2011 年 2 月 14 日	第 1.01 版

発行　(株)ルネサスソリューションズ　ルネサスマイコンカーラリー事務局
〒162-0824　東京都新宿区揚場町 2-1　軽子坂MNビル