第0回　マイクロコンピュータ応用 -導入-

0.1　はじめに

前学期ではアセンブリプログラミングによりZ80マイコンを制御する基礎的な技術を学びました。本学期では, その身につけた技術を用い, スイッチ, LED, ステッピングモータなどを制御することで, マイコンによる制御やプログラミングの考え方を学ぶび身につけることを目標とします。

また, 本実験では前学期の知識を基本としているので, 前学期に用いたテキストを持参するとよいでしょう。

0.2　レポートの書き方

レポートの章立ては次のようにすると良いでしょう。

- 目的

  実験の目的を書く

- 手法

  実験で用いた器具, 手法, 手順などを書く。

- 結果

  実験内容とともに実験の結果を書く。本実験の結果では主にプログラムソースを書く。ソースを書く場合は, 機械語, アセンブリコード, コメントを書く。

- 考察

  考察課題や, 実験で気づいたこと, 分かったことなどを書く。

補足: レポートは, 左上の角をホッチキスでとめて下さい。

0.3　マイコントレーナの使いかた

本実験では, マイコントレーナMT-Zを用います (図1)。MT-Zは8ビットマイクロプロセッサZ-80を搭載しています。レジスタには8ビットデータを記憶することができます。

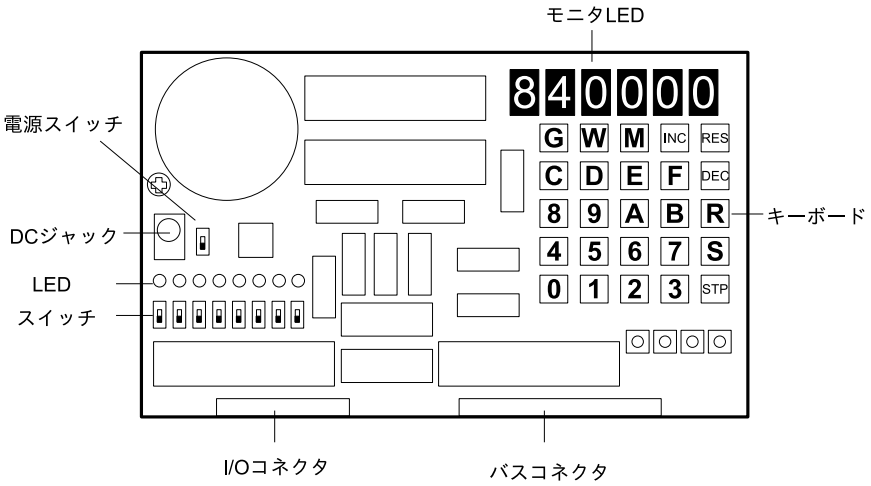

図 1: マイコントレーナ MT-Z。

0.3.1 実験開始の手順

基本的に実験の開始は次のような手順で行います。

- DC ジャックに AC アダプタを繋ぐ。
- AC アダプタをコンセントに繋ぐ
- マイコントレーナの電源を入れる
- モニタ LED が 6 桁とも点灯していることを確認する。
- プログラムを入力する

0.3.2 モニタ LED の見方

メモリに書き込まれているデータを見るために, モニタ LED が存在します。モニタ LED は 6 桁の 16 進の数を表示します。左の 4 桁はメモリのアドレスを表し, 右の 2 桁はそのアドレスに入っているデータを表します (図 2)。

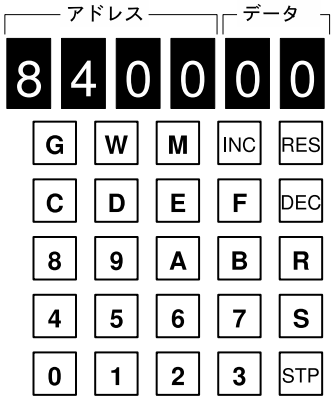

図 2: モニタ LED, キーボード。

0.3.3 メモリ内のデータの読みだしの仕方

メモリの中に入っているデータを読み出すときにはボタン M を用います。8400H 番地のデータを読み出すときには, M を押して 8400 と入力すると 8400H 番地のデータが表示されます。次の番地のデータが見たいときは, INC を押すことで次の番地に進みます。また, DEC を押すことで一つ前の番地のデータを読むことができます。

0.3.4 メモリの書き込みの仕方

メモリにデータを書き込むときは, ボタン W を用います。8400H 番地に 01H というデータを書き込む場合は, まずボタン M を押し, 8400 と入力することで 8400H 番地に移動する。次に, W を押すことで書き込みモードにし, データを入力します。次の番地に書き込みたいときには INC を押すことで, 次の番地に書き込めます。また, DEC を押すことで一つ前の番地のデータに書き込めます。

0.3.5 実験プログラムを入れることのできるメモリの領域

実験プログラムなどを入れることができるメモリの領域は 8400H–EFFFH です。実験で作ったプログラムはその番地に書き込んでください。サブルーティンや数値データはすぐ隣の番地に置かず, 離れた番地に置くと, プログラムを修正する場合や書く加える時に便利です。

0.3.6 プログラムの実行, リセットの仕方

プログラムを実行する場合には, G ボタンを用います。8400H 番地から書き込んだプログラムを実行する場合は, M ボタンを押し, 次に 8400 と入力することで 8400H 番地に移動

します。そして, G ボタンを押すことでプログラムが実行されます。

もしプログラムが暴走したり, 無限ループのプログラムを終了させたいときには, RES ボタンを押すことでプログラムを終了させることができます。

0.4 本実験における作業の流れ

本演習の多くはアセンブリプログラミング作業です。プログラミングを行う作業の流れは次の順番で行うとよいでしょう。

- 処理の流れを考える。

  フローチャートなどをかいてもよい。

- アセンブリコードを考える。

- アセンブリコードを機械語に直す。

  アセンブリコードを機械語に直すとき, テキストに付属するニーモニックと機械語の対応リストを参照してください。

- 機械語を MT-Z に入力し, 実行する。

もしプログラムが動かないときは, 上記の作業の流れのうちどれかが間違っています。よく見直してみましょう。

第1回　マイクロコンピュータ応用

1.1　目的

今回は, 前学期のマイクロコンピュータ基礎で行ったアセンブリ言語の復習を行います。今回復習するのは, データの転送, ループ, 条件分岐, LED の制御です。

1.2　装置

前学期で用いたマイコントレーナ MT-Z を用いる。

1.3　実験

今回は, 前学期習った Z80 アセンブリを復習します。アセンブリプログラミングで必要なニーモニックは, 付属のニーモニックと機械語のリストに機能と機械語が記載されているので, 参考にしてください。

1.3.1　レジスタ, メモリの操作の復習

レジスタとメモリはデータを保存する場所です。ここでは, レジスタやメモリのデータの転送の仕方を復習します。

課題 1 8500H の数値に 5 を足した数値を, 8501H に書き込むプログラムを作りなさい (表 1.1)。

- データを転送するニーモニックは “LD” を使います。“LD” はデータを任意の場所へ送る命令です。基本的な使い方は, “LD データの転送先, データの転送元” です。
- 加算を行うニーモニックは “ADD” です。

課題 2 図 1.1 を参考に, 8500H と 8501H の数値を足した数値を 8502H に書き込むプログラムを作りなさい。

表 1.1: 課題 1 のプログラム

アドレス	機械語	ニーモニック	コメント
8400		ORG 8400H	
8400	__ __ __	LD A, (8500H)	8500H の値を A レジスタに転送
8403	__ __	ADD A, 05H	A レジスタの値に 5 を足す
8405	__ __ __	LD (8501H), A	A レジスタの値を 8501H に転送
8408	__ __ __	JP 0000H	モニタプログラムにジャンプ
8500		ORG 8500H	
8500	__	DB 01H	
8501		END	

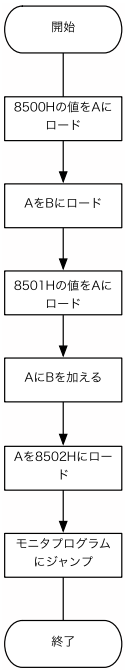

図 1.1: 課題 2 の処理の流れ。

表 1.2: 課題 3 のプログラム

アドレス	機械語	ラベル	ニーモニック	コメント
8400			ORG 8400H	
8400	＿＿＿		LD A, (8500H)	8500H の値を A レジスタに転送
8403	＿		LD B, A	A レジスタの値を B レジスタに転送
8404	＿＿＿		LD A, (8501H)	8501H の値を A レジスタに転送
8407	＿		CP B	B と比較
8408	＿＿＿		JP P, MORE	フラグレジスタが正なら MORE にジャンプ
840B	＿		LD A, B	B レジスタの値を A レジスタに転送
840C	＿＿＿		LD (8502H), A	A レジスタの値を 8502H に転送
840F	＿＿＿		JP 0000H	モニタプログラムにジャンプ
8412	＿＿＿	MORE:	LD (8502H), A	A レジスタの値を 8502H に転送
8415	＿＿＿		JP 0000H	モニタプログラムにジャンプ
8500			ORG 8500H	
8500	＿＿		DB 01H, 03H	
8502			END	

1.3.2 条件分岐, ループ

プログラミング言語として最も重要な機能の一つが条件分岐やループです。ここではアセンブリにおける条件分岐, ループを復習します。

課題 3 8500H と 8501H の数値を比較し, 大きい方を 8502H に書き込むプログラムを作りなさい (表 1.2)。

- データを比較するときには “CP” を使います。比較した結果はフラグレジスタに保存されます。
- プログラムの任意の場所にジャンプするときは “JP” を用います。“JP 番地” で無条件に任意の番地にジャンプし, “JP X 番地” でフラグレジスタが X の時に任意の番地にジャンプします。

課題 4 10 から 1 ずつ引いていき, 5 以下になった場合終了するプログラムを作りなさい (表 1.3)。

課題 5 図 1.2 を参考に, 1–10 の和をメモリの 8500H 番地に書き込むプログラムを作りなさい。

表 1.3: 課題 4 のプログラム

アドレス	機械語	ラベル	ニーモニック	コメント
8400			ORG 8400H	
8400	__ __		LD A, 0AH	値 0AH を A レジスタに転送
8402	__	LOOP:	DEC A	A から 1 を引く
8403	__ __		CP 05H	05H と比較する
8405	__ __ __		JP NZ, LOOP	フラグレジスタが NZ ならば LOOP にジャンプ
8408	__ __ __		LD (8500H), A	A レジスタの値を 8500H に転送
840B	__ __ __		JP 0000H	モニタプログラムにジャンプ
840E			END	

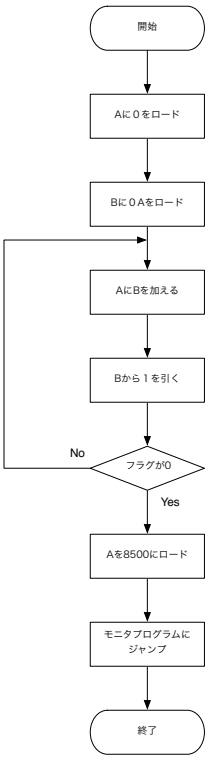

図 1.2: 課題 5 の処理の流れ

表 1.4: 課題 6 の処理の流れ

アドレス	機械語	ニーモニック	コメント
0005		PB EQU 05H	ポート B のアドレス
0007		CTL EQU 07H	コントロールポートのアドレス
0090		CLWD EQU 90H	コントロールワード
8400		ORG 8400H	
8400	__ __	LD A, CLWD	コントロールワードを A に転送
8402	__ __	OUT (CTL), A	コントロールポートに A の値を出力
8404	__ __ __	LD A, (8500H)	8500H の値を A レジスタに転送
8407	__ __	OUT (PB), A	ポート B に A を出力
8409	__ __ __	JP 0000H	モニタプログラムにジャンプ
8500		ORG 8500H	
8500	__	DB 0FFH	
8501		END	

1.3.3 LED の制御の復習

MT-Z にはパラレル入出力 IC の 8255A が搭載されています。8255A には, A, B, C, コントロールのポートがあります。これらのポートを使うには, 各ポートを入力もしくは出力で使うかの設定情報 (コントロールワード) をコントロールポートに出力しなければなりません。今回はポート A は入力, ポート B, C は出力として使うので, コントロールワードは 90H とします。今回使う LED はポート B につながっています。ポート B に適切な信号を出力することで LED を制御します。

課題 6 LED をすべて点灯させなさい (表 1.4)。

- ポートに信号を出力する場合は “OUT” を使います。

課題 7 LED を一つおきに点灯させなさい。

課題 8 LED の点灯が反転を繰り返すプログラムを作りなさい (表 1.5)。

課題 9 図 1.3 を参考に, LED の点灯位置が左にシフトするプログラムを作りなさい。

- 左にシフトさせるには, 左にビットシフトさせる “RLCA” を使う。

課題 10(発展) LED の点灯位置が, 始め左にシフトし, 左端にきたら右にシフト, 右端にきたら左にシフトするようなプログラムを作れ。

表 1.5: 課題 8 のプログラム

アドレス	機械語	ラベル	ニーモニック	コメント
0005			PB EQU 05H	ポート B のアドレス
0007			CTL EQU 07H	コントロールポートのアドレス
0090			CLWD EQU 90H	コントロールワード
8400			ORG 8400H	
8400	__ __		LD A, CLWD	コントロールワードを A レジスタに転送
8402	__ __		OUT (CTL), A	コントロールポートに A レジスタの値を出力
8404	__ __		LD A, 08H	値 08H を A レジスタに転送
8406	__ __	SHIFT:	OUT (PB), A	ポート B に A レジスタの値を出力
8408	__ __		XOR 0FFH	A レジスタと FFH との排他的論理和をとる
840A	__ __ __		CALL TIMER	タイマを呼び出す
840D	__ __ __		JP SHIFT	シフトにジャンプ
8440			ORG 8440H	
8440	21 00 40	TIMER:	LD HL , 4000H	値 4000H を HL レジスタに転送
8443	5F		LD E, A	A レジスタの値を E レジスタに転送
8444	2B	TLOOP:	DEC HL	HL レジスタの値から 1 を引く
8445	7C		LD A, H	H レジスタの値を A レジスタに転送
8446	B5		OR L	A の値と L の値の論理和をとる
8447	20 FB		JR NZ , TLOOP	フラグレジスタが NZ ならば TLOOP にジャンプ
8449	7B		LD A, E	E レジスタの値を A レジスタに転送
844A	C9		RET	ルーティンの終了
844B			END	

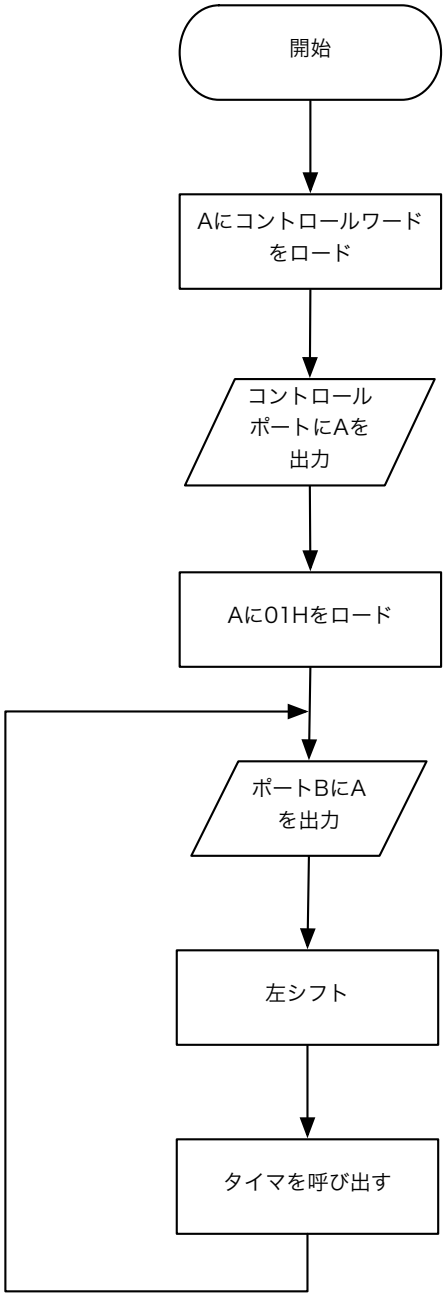

図 1.3: 課題 9 の処理の流れ

1.4 考察課題

考察課題 1 アセンブリ言語は低級言語といわれている。低級言語とは何か報告しなさい。

考察課題 2 プログラミング言語には高級言語と呼ばれるものがある。高級言語とは何か報告しなさい。また, 高級言語の例を 2 つ報告し, その言語が主にどのような用途で使われるかなどの特徴も報告しなさい。

第2回　マイクロコンピュータ応用

2.1　目的

まず, ステッピングモータを理解し, 制御することが今回の目標です。さらに, スイッチの操作の復習も行います。

2.2　装置

2.2.1　ステッピングモータとは

ステッピングモータは, 普通のモータ異なり, パルス信号が入力されるごとに, 一定の角度ずつ回転させることができるモータです。一定の角度回転させることができるので, 装置の位置を制御するのに適しているモータです。

ステッピングモータは周囲に付けられたコイル (ステータ) と, 回転軸に固定された磁石 (ロータ) で構成されます。コイルに電流を流すことで磁界が生じ (このことを励磁と言います), その磁界により磁石が引き寄せられることで, ステッピングモータは一定角度回転します。コイルへの電流の流し方を変えることで, 異なった性質を持つ回転をつくり出すことができます。

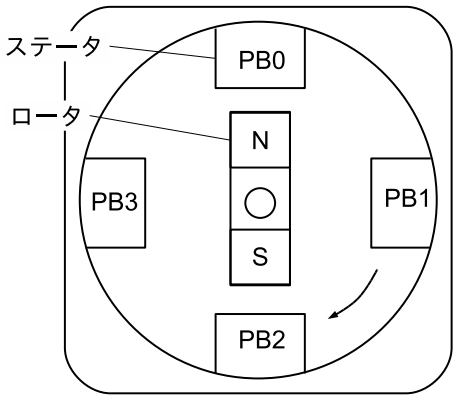

図 2.1: ステッピングモータの概念図。ポート B に電流を流すことで磁界が発生し, 磁石が回る。

2.2.2 装置セッティング

今回はMT-Zの拡張パーツであるステッピングモータを用います。ステッピングモータを使えるようにするには, まず, ステッピングモータはステッピングモータインターフェースボードに接続します。次に, ステッピングモータインターフェースボードにアダプタを接続します。そして, 図2.2のようにステッピングモータインターフェースボードをMT-Zの左にあるIOポートに繋ぎます。以上のように接続することで, オンボードの8255Aのポート Bにつながり, ポート Bに信号を出力することでステッピングモータを制御することができます。

ちなみに, オンボード8255Aのポート BはMT-Z上のLEDと同じポートです。よって, ステッピングモータに送る信号 (ドライブパターン) と同様のLEDが点滅することとなります。LEDの点灯パターンを見ることでドライブパターンがきちんとステッピングモータに送られているか確認することができます。

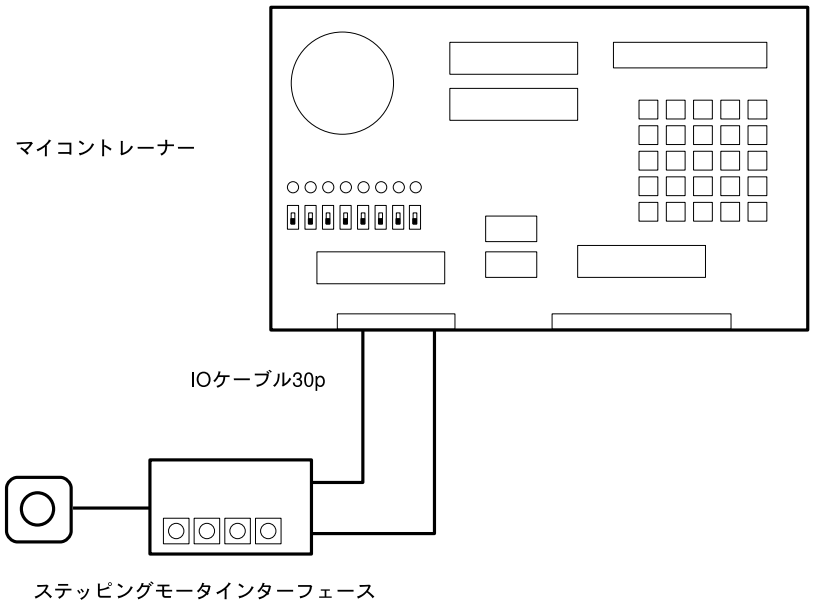

図 2.2: マイコントレーナ MT-Z とステッピングモータの接続図

2.3 実験

ステッピングモータを回転させます。ステッピングモータは, 流すパルス電流のパターン (ドライブパターン) を変えることで, 性質の異なった回転をさせることができます。今回は, 1相励磁回転 (表2.1), 2相励磁回転 (表2.3), 1-2相励磁回転 (表2.4) の3種類のドライブパターンを行います。

表 2.1: 1 相励磁回転ドライブパターン

ステップ	PB0	PB1	PB2	PB3
0	1	0	0	0
1	0	1	0	0
2	0	0	1	0
3	0	0	0	1

課題 1 表 2.1 は 1 相励磁回転のドライブパターンである。1 相励磁回転をさせなさい (表 2.2)。

課題 2 表 2.3 は 2 相励磁回転のドライブパターンである。課題 1 のプログラムを参考に, ステッピングモータを 2 相励磁回転させよ。

課題 3 表 2.4 は 1–2 相励磁回転のドライブパターンである。課題 1 のプログラムを参考に, ステッピングモータを 1–2 相励磁回転させよ。

2.4 スイッチ操作の復習

ここでは, スイッチ操作の復習を行います。スイッチは, MT-Z 上の 8255A のポート A につながっています。ポート A を入力モードにすることでスイッチからの信号を入力することができるようになります。よって, ここでもコントロールワードは 90H を用いることになります。

課題 4 スイッチの状態を読み込んで, それを LED に出力するプログラムを作りなさい (表 2.5)。

- ポートの信号を入力する場合は “IN” を使います。

課題 5 図 2.3 を参考に, スイッチの右端が ON の場合 LED の点灯を左にシフトさせ, OFF の場合停止させるプログラムを作りなさい。

課題 6 2 つの 4 桁の 2 進の数値を入力し, それを加算し結果を LED で表示するプログラムを作りなさい。数値を入力する場合は, スイッチの右 4 つ, 左 4 つそれぞれの状態を 2 つの 4 桁の数としなさい。

表 2.2: 課題 1 のプログラム

アドレス	機械語	ラベル	ニーモニック	コメント
0005			PB EQU 05H	ポート B アドレス
0007			CTL EQU 07H	コントロールポートアドレス
0091			CLWD EQU 90H	コントロールワード
8400			ORG 8400H	
8400	__ __	STPMTR:	LD A , CLWD	コントロールワードを A レジスタに転送
8402	__ __		OUT (CTL) , A	コントロールポートに A レジスタの値を出力
8404	__ __	LOOP:	LD A , 01H	01H を A レジスタに転送
8406	__ __		OUT (PB) , A	A レジスタの値をポート B に出力
8408	__ __ __		CALL TIMER	タイマを呼び出す
840B	__ __		LD A , 02H	02H を A レジスタに転送
840D	__ __		OUT (PB) , A	A レジスタの値をポート B に出力
840F	__ __ __		CALL TIMER	タイマおよび出す
8412	__ __		LD A , 04H	04H を A レジスタに転送
8414	__ __		OUT (PB) , A	A レジスタの値をポート B に出力
8416	__ __ __		CALL TIMER	タイマを呼び出す
8419	__ __		LD A , 08H	08H を A レジスタに転送
841B	__ __		OUT (PB) , A	A レジスタの値をポート B に出力
841D	__ __ __		CALL TIMER	タイマを呼び出す
8420	__ __ __		JP LOOP	ループにジャンプ
8440			ORG 8440H	
8440	21 00 40	TIMER:	LD HL , 4000H	値 4000H を HL レジスタに転送
8443	5F		LD E, A	A レジスタの値を E レジスタに転送
8444	2B	TLOOP:	DEC HL	HL レジスタの値から 1 を引く
8445	7C		LD A, H	H レジスタの値を A レジスタに転送
8446	B5		OR L	A の値と L の値の論理和をとる
8447	20 FB		JR NZ , TLOOP	フラグレジスタが NZ ならば TLOOP にジャンプ
8449	7B		LD A, E	E レジスタの値を A レジスタに転送
844A	C9		RET	ルーティンの終了
844B			END	

表 2.3: 2 相励磁回転ドライブパターン

ステップ	PB0	PB1	PB2	PB3
0	1	1	0	0
1	0	1	1	0
2	0	0	1	1
3	1	0	0	1

表 2.4: 1–2 相励磁回転ドライブパターン

ステップ	PB0	PB1	PB2	PB3
0	1	0	0	0
1	1	1	0	0
2	0	1	0	0
3	0	1	1	0
4	0	0	1	0
5	0	0	1	1
6	0	0	0	1
7	1	0	0	1

表 2.5: 課題 4 のプログラム

アドレス	機械語	ラベル	ニーモニック	コメント
0004			PA EQU 04H	ポート A アドレス
0005			PB EQU 05H	ポート B アドレス
0007			CTL EQU 07H	コントロールポートアドレス
0090			CLWD EQU 90H	コントロールワード
8400			ORG 8400H	
8400	__ __		LD A, CLWD	コントロールワード
8402	__ __		OUT (CTL), A	コントロールポートに A レジスタの値を出力
8404	__ __	LOOP:	IN A, (PA)	ポート A から A レジスタに入力
8406	__ __		OUT (PB), A	A レジスタの値をポート B に出力
8408	__ __ __		JP LOOP	LOOP にジャンプ
840B			END	

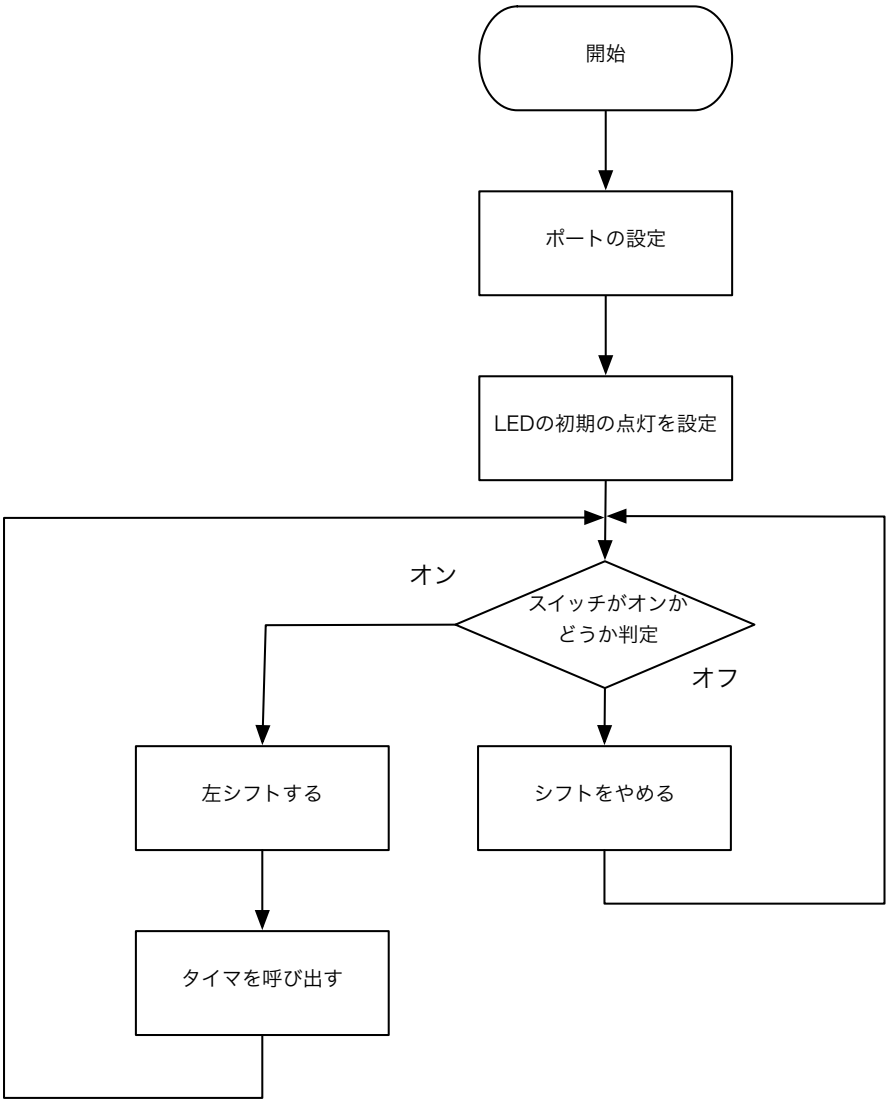

図 2.3: 課題 5 の処理の流れ

2.5 考察課題

考察課題 1 タイマサブルーティンはなぜ必要か答えなさい。

考察課題 2 今回は 3 種類のドライブの仕方を行った。各ドライブパターンの特徴をしらべて報告しなさい。

考察課題 3 課題 1 のプログラムをどのように変えると回転速度や回転方向が変わるか考えなさい。

第3回　マイクロコンピュータ応用

3.1　目的

拡張パラレル IO ボードの使い方を学び, 拡張パラレル IO ボードにステッピングモータをつなぎ制御する。さらにスイッチも同時に制御することを目標とする。

3.2　装置

3.2.1　拡張パラレル IO ボード

マイコントレーナには 1 つの IO ポートが付いています。しかし, 2 つ以上の拡張ボードを接続する必要が生じることがあります。MT-Z では本演習で用いるパラレル IO ボードを接続することで, 2 つの IO ポートを増やすことができます。

3.2.2　装置セッティング

まず図 3.1 のように, MT-Z の左下にあるバスコネクタに拡張パラレル IO ボードを繋ぎます。そして, ステッピングモータとアダプタをあらかじめ接続したステッピングモータインターフェースボードを CH1 に繋ぎます。

拡張パラレル IO ボードは, 前学期用いた 8255A が 2 つ使われています。よって, 拡張パラレル IO ボードに接続した機器を使う場合, 前学期に習ったスイッチ, LED などの制御方法と同じようにコントロールワードを指定し, ポートに信号を送ってやれば動作します。IO ボードを使う場合の各ポートのアドレスは表 3.1, 表 3.2 のようになっています。ここでは, CH1 の 8255A を用いるので CH1 のアドレスを用います。プログラムを組むときには表 3.1 のアドレスを用いてください。今回もコントロールワードは 90H を用います。

表 3.1: 拡張パラレル IO ボード CH1 のポートアドレス

PA	20H
PB	21H
PC	22H
CTRL	23H

表 3.2: 拡張パラレル IO ボード CH2 のポートアドレス

PA	24H
PB	25H
PC	26H
CTRL	27H

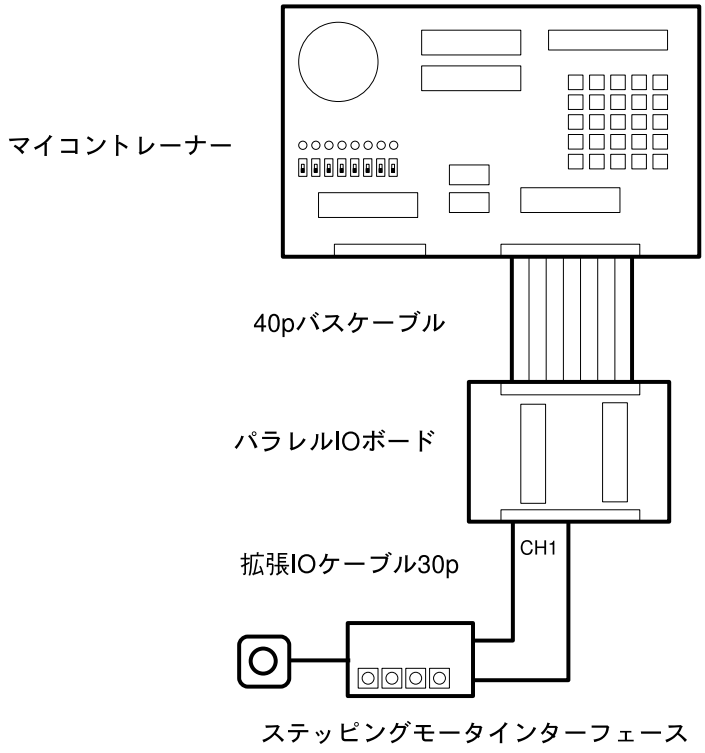

図 3.1: 装置の接続の様子。

表 3.3: 課題 1 のプログラム

アドレス	機械語	ラベル	ニーモニック	コメント
0005			PB EQU 21H	拡張 IO ポート B アドレス
0007			CTL EQU 23H	拡張 IO コントロールポートアドレス
0090			CLWD EQU 90H	拡張 IO コントロールワード
8400			ORG 8400H	
8400	__ __	STPMTR:	LD A , CLWD	コントロールワードを A レジスタに転送
8402	__ __		OUT (CTL) , A	A レジスタの値をコントロールポートに出力
8404	__ __	LOOP:	LD A , 01H	01H を A レジスタに転送
8406	__ __		OUT (PB) , A	A レジスタの値をポート B に出力
8408	__ __ __		CALL TIMER	タイマを呼び出す
840B	__ __		LD A , 02H	02H を A レジスタに転送
840D	__ __		OUT (PB) , A	A レジスタの値をポート B に出力
840F	__ __ __		CALL TIMER	タイマを呼び出す
8412	__ __		LD A , 04H	04H を A レジスタに転送
8414	__ __		OUT (PB) , A	A レジスタの値をポート B に出力
8416	__ __ __		CALL TIMER	タイマを呼び出す
8419	__ __		LD A , 08H	08H を A レジスタに転送
841B	__ __		OUT (PB) , A	A レジスタの値をポート B に出力
841D	__ __ __		CALL TIMER	タイマを呼び出す
8420	__ __ __		JP LOOP	ループにジャンプ
8440			ORG 8440H	
8440	21 00 40	TIMER:	LD HL , 4000H	値 4000H を HL レジスタに転送
8443	5F		LD E, A	A レジスタの値を E レジスタに転送
8444	2B	TLOOP:	DEC HL	HL レジスタの値から 1 を引く
8445	7C		LD A, H	H レジスタの値を A レジスタに転送
8446	B5		OR L	A の値と L の値の論理和をとる
8447	20 FB		JR NZ , TLOOP	フラグレジスタが NZ ならば TLOOP にジャンプ
8449	7B		LD A, E	E レジスタの値を A レジスタに転送
844A	C9		RET	ルーティンの終了
844B			END	

3.3 実験

課題 1 拡張パラレル IO ボードに接続したステッピングモータを 1 相励磁回転で回転させなさい (表 3.3)。プログラムは, 前回のステッピングモータ制御プログラムの, 出力ポートアドレスを表 3.1 を参考に修正することでステッピングモータは動きます。

課題 2 課題 1 で作ったプログラムを改造し, 速度をいろいろと変えてみなさい。

課題 3 課題 1 で作ったプログラムを改造し, 逆回転させなさい。

課題 4 右端のスイッチが ON の場合高速回転, OFF の場合は低速回転するプログラムを作りなさい (表 3.4, 3.5)。

課題 5(発展) 右端のスイッチを ON にすると時計回りに回転し, OFF の時に反時計回りに回転させるプログラムを作りなさい。

表 3.4: 課題 4 のプログラム

アドレス	機械語	ラベル	ニーモニック	コメント
0004			PA EQU 04H	オンボード 8255A ポート A アドレス
0005			PB EQU 05H	オンボード 8255A ポート B アドレス
0007			CTL EQU 07H	オンボード 8255A コントロールポートアドレス
0090			CLWD EQU 90H	オンボード 8255A コントロールワード
0021			PB2 EQU 21H	拡張 IO ポート B アドレス
0023			CTL2 EQU 23H	拡張 IO コントロールポートアドレス
0090			CTLW2 EQU 90H	拡張 IO コントロールワード
8400			ORG 8400H	
8400	__ __		LD A, CLWD	オンボード 8255A 用コントロールワードを A レジスタに転送
8402	__ __		OUT (CTL), A	A レジスタの値をオンボード 8255A コントロールポートに出力
8404	__ __		LD A , CTLW2	拡張 IO コントロールワードを A レジスタに転送
8406	__ __		OUT (CTL2) , A	A レジスタの値を拡張 IO コントロールポートに出力
8408	__ __ __		LD DE, 4000H	4000H を DE レジスタに転送
840B	__ __ __	LOOP:	CALL MOTOR	モータを呼び出す
840E	__ __		IN A, (PA)	ポート A の状態を A レジスタに入力
8410	__ __		AND 01H	01H との論理積をとる
8412	__ __		CP 00H	00 と比較する
8414	__ __ __		JP Z, SLOW	フラグレジスタが Z ならば SLOW にジャンプ
8417	__ __ __		JP FAST	FAST にジャンプ
841A	__ __	MOTOR	LD A , 01H	01H を A レジスタに転送
841C	__ __		OUT (PB2) , A	A レジスタの値を拡張 IO ポート B に出力
841E	__ __ __		CALL TIMER	タイマを呼び出す
8421	__ __		LD A , 02H	02H を A レジスタに転送
8423	__ __		OUT (PB2) , A	A レジスタの値を拡張 IO ポート B に出力
8425	__ __ __		CALL TIMER	タイマを呼び出す
8428	__ __		LD A , 04H	04H を A レジスタに転送
842A	__ __		OUT (PB2) , A	A レジスタの値を拡張 IO ポート B に出力
842C	__ __ __		CALL TIMER	タイマを呼び出す
842F	__ __		LD A , 08H	08H を A レジスタに転送
8431	__ __		OUT (PB2) , A	A レジスタの値を拡張 IO ポート B に出力
8433	__ __ __		CALL TIMER	タイマを呼び出す
8436	__		RET	ルーティン終了
8437	__ __ __	SLOW:	LD DE, 1000H	
843A	__ __ __		JP LOOP	
843D	__ __ __	FAST:	LD DE, 300H	
8440	__ __ __		JP LOOP	

表 3.5: 課題 4 のプログラムの続き

アドレス	機械語	ラベル	ニーモニック	コメント
8600			ORG 8600H	
8600	__	TIMER:	LD H, D	
8601	__		LD L, E	
8602	5F		LD E, A	A レジスタの値を E レジスタに転送
8603	2B	TLOOP:	DEC HL	HL レジスタの値から 1 を引く
8604	7C		LD A, H	H レジスタの値を A レジスタに転送
8605	B5		OR L	A の値と L の値の論理和をとる
8606	20 FB		JR NZ , TLOOP	フラグレジスタが NZ ならば TLOOP にジャンプ
8608	7B		LD A, E	E レジスタの値を A レジスタに転送
8609	C9		RET	ルーティンの終了
860A			END	

3.4 考察課題

考察課題 1 回転速度を速くしたり, 遅くしたりしたときに, うまくモータが回転したか。おそらく, うまく回転しなかった場合があると思われる。そのどのようなときにうまく回転しなかったか報告し, その原因が何か考えよ。

考察課題 2 次回の課題プログラムの流れを考えよ。

第4回　マイクロコンピュータ応用

4.1　目的

マイクロコンピュータ演習の集大成として、これまで習った知識を用いてステッピングモータ制御のシステムを構築する。

4.2　装置

マイコントレーナー MT-Z およびステッピングモータを用いる。

4.3　最終課題

以下の機能を実装するプログラムを作りなさい。細かい仕様は各グループの判断に任せます。レポートではプログラムの仕様 (どのような動作を行うプログラムか)、大まかな処理の流れ、プログラムソースを報告すること。

- ステッピングモータインターフェースをパラレル IO ボードにつなぎ、1 相励磁回転させる。
- スイッチにより、回転方向、回転速度を変えられるようにする。
- LED の点灯を回転に応じて変化させる。(変化のさせかたは任意。例えば回転速度を LED の点灯数で表す、回転に応じて LED の点灯が移動するなど)

第5回　ニーモニックと機械語のリスト(アルファベット順)

ADD(加算)		
機械語	ニーモニック	機能
87	ADD A, A	A に A を加える
80	ADD A, B	A に B を加える
81	ADD A, C	A に C を加える
82	ADD A, D	A に D を加える
83	ADD A, E	A に E を加える
84	ADD A, H	A に H を加える
85	ADD A, L	A に L を加える
C6 n	ADD A, n	A に n を加える

AND(論理積)		
A7	AND A	A と A のビットごとの論理積をとる
A0	AND B	A と B のビットごとの論理積をとる
A1	AND C	A と C のビットごとの論理積をとる
A2	AND D	A と D のビットごとの論理積をとる
A3	AND E	A と E のビットごとの論理積をとる
A4	AND H	A と H のビットごとの論理積をとる
A5	AND L	A と L のビットごとの論理積をとる
E6 n	AND n	A と n のビットごとの論理積をとる

CALL(呼び出し)		
CD n m	CALL mnH	mnH 番地をコールする
DC n m	CALL C, mnH	桁上げ桁下げがあるとき mnH 番地をコールする
C4 n m	CALL NC, mnH	桁上げ桁下げが無いとき mnH 番地をコールする
FC n m	CALL M, mnH	負の数のとき mnH 番地をコールする
F4 n m	CALL P, mnH	正の数のとき mnH 番地をコールする
EC n m	CALL PE, mnH	偶数のとき mnH 番地をコールする
E4 n m	CALL PO, mnH	奇数のとき mnH 番地をコールする
CC n m	CALL Z, mnH	ゼロのとき mnH 番地をコールする
D4 n m	CALL NZ, mnH	ゼロでないときのとき mnH 番地をコールする

CP(比較)		
BF	CP A	オペラントと A を比較する
B8	CP B	
B9	CP C	
BA	CP D	
BB	CP E	
BC	CP H	
BD	CP L	
FE n	CP n	

DEC(デクリメント)		
機械語	ニーモニック	機能
3D	DEC A	指定したレジスタから 1 引く
05	DEC B	
0D	DEC C	
15	DEC D	
1D	DEC E	
25	DEC H	
2D	DEC L	

INPUT(入力)		
DB n	IN A, (n)	n で指定した I/O ポートに A を入力

INC(インクリメント)		
3C	INC A	指定したレジスタに 1 足す
04	INC B	
0C	INC C	
14	INC D	
1C	INC E	
24	INC H	
2C	INC L	

JP(ジャンプ)		
C3 n m	JP mnH	mnH 番地にジャンプする
DA n m	JP C, mnH	桁上げ桁下げフラグが立っているとき mnH 番地にジャンプする
D2 n m	JP NC, mnH	桁上げ桁下げフラグが立っていないとき mnH 番地にジャンプする
FA n m	JP M, mnH	負の数フラグが立っているとき mnH 番地にジャンプする
F2 n m	JP P, mnH	正の数フラグが立っているとき mnH 番地にジャンプする
EA n m	JP PE, mnH	偶数フラグが立っているとき mnH 番地にジャンプする
E2 n m	JP PO, mnH	奇数フラグが立っているとき mnH 番地にジャンプする
CA n m	JP Z, mnH	ゼロフラグ立っているとき mnH 番地にジャンプする
C2 n m	JP NZ, mnH	ゼロフラグがたっていないとき mnH 番地にジャンプする

LD(転送)		
3A n m	LD A, (mnH)	メモリ mnH 番地の値を A に転送する
7F	LD A, A	A を A に転送する
78	LD A, B	B を A に転送する
79	LD A, C	C を A に転送する
7A	LD A, D	D を A に転送する
7B	LD A, E	E を A に転送する
7C	LD A, H	H を A に転送する
7D	LD A, L	L を A に転送する
3E n	LD A, n	値 n を A に転送する
47	LD B, A	A を B に転送する
40	LD B, B	B を B に転送する
41	LD B, C	C を B に転送する
42	LD B, D	D を B に転送する
43	LD B, E	E を B に転送する
44	LD B, H	H を B に転送する
45	LD B, L	L を B に転送する
06 n	LD B, n	値 n を B に転送する
32 nm	LD mn, A	A を mnH に転送する

LD(転送)		
機械語	ニーモニック	機能
4F	LD C, A	A を C に転送する
48	LD C, B	B を C に転送する
49	LD C, C	C を C に転送する
4A	LD C, D	D を C に転送する
4B	LD C, E	E を C に転送する
4C	LD C, H	H を C に転送する
4D	LD C, L	L を C に転送する
0E n	LD C, n	値 n を C に転送する
57	LD D, A	A を D に転送する
50	LD D, B	B を D に転送する
51	LD D, C	C を D に転送する
52	LD D, D	D を D に転送する
53	LD D, E	E を D に転送する
54	LD D, H	H を D に転送する
55	LD D, L	L を D に転送する
16 n	LD D, n	値 n を D に転送する
5F	LD E, A	A を E に転送する
58	LD E, B	B を E に転送する
59	LD E, C	C を E に転送する
5A	LD E, D	D を E に転送する
5B	LD E, E	E を E に転送する
5C	LD E, H	H を E に転送する
5D	LD E, L	L を E に転送する
1E n	LD E, n	値 n を E に転送する
67	LD H, A	A を H に転送する
60	LD H, B	B を H に転送する
61	LD H, C	C を H に転送する
62	LD H, D	D を H に転送する
63	LD H, E	E を H に転送する
64	LD H, H	H を H に転送する
65	LD H, L	L を H に転送する
26 n	LD H, n	値 n を H に転送する
6F	LD L, A	A を L に転送する
68	LD L, B	B を L に転送する
69	LD L, C	C を L に転送する
6A	LD L, D	D を L に転送する
6B	LD L, E	E を L に転送する
6C	LD L, H	H を L に転送する
6D	LD L, L	L を L に転送する
2E n	LD L, n	値 n を L に転送する
01 n m	LD BC, mn	値 mnH を BC に転送する
11 n m	LD DE, mn	値 mnH を DE に転送する
21 n m	LD HL, mn	値 mnH を HL に転送する

OR(論理和)		
機械語	ニーモニック	機能
B7	OR A	A と A のビットごとの論理和をとる
B0	OR B	A と B のビットごとの論理和をとる
B1	OR C	A と C のビットごとの論理和をとる
B2	OR D	A と D のビットごとの論理和をとる
B3	OR E	A と E のビットごとの論理和をとる
B4	OR H	A と H のビットごとの論理和をとる
B5	OR L	A と L のビットごとの論理和をとる
F6 n	OR n	A と n のビットごとの論理和をとる

OUT(出力)		
D3 n	OUT (n), A	A の内容を I/O ポート n に出力する

RET(戻る)		
C9	RET	サブルーティンから戻る
D8	RET C	桁上げ桁下げフラグが立っているときサブルーティンから戻る
D0	RET NC	桁上げ桁下げフラグが立っていないときサブルーティンから戻る
F8	RET M	負の数フラグが立っているときサブルーティンから戻る
F0	RET P	正の数フラグが立っているときサブルーティンから戻る
E8	RET PE	偶数フラグが立っているときサブルーティンから戻る
E0	RET PO	奇数フラグが立っているときサブルーティンから戻る
C8	RET Z	ゼロフラグが立っているときサブルーティンから戻る
C0	RET NZ	ゼロフラグが立っていないときサブルーティンから戻る

RLCA(左ビットシフト)		
07	RLCA	A の内容を左にシフトする

RRCA(右ビットシフト)		
0F	RRCA	A の内容を右にシフトする

SUB(減算)		
97	SUB A	A から A を引く
90	SUB B	A から B を引く
91	SUB C	A から C を引く
92	SUB D	A から D を引く
93	SUB E	A から E を引く
94	SUB H	A から H を引く
95	SUB L	A から L を引く
D6 n	SUB n	A から n を引く

XOR(排他的論理和)		
AF	XOR A	A と A のビットごとの排他的論理和をとる
A8	XOR B	A と B のビットごとの排他的論理和をとる
A9	XOR C	A と C のビットごとの排他的論理和をとる
AA	XOR D	A と D のビットごとの排他的論理和をとる
AB	XOR E	A と E のビットごとの排他的論理和をとる
AC	XOR H	A と H のビットごとの排他的論理和をとる
AD	XOR L	A と L のビットごとの排他的論理和をとる
EE n	XOR n	A と n のビットごとの排他的論理和をとる