Report on the Experiment

No. 8

Subject マイコン実習 (2)

Date 2019. 05. 06

Weather 曇り Temp 22.2 ℃ Wet 58 %

Class	**E3**	
Group	**6**	
Chief		
Partner	大橋	りさ
	二重谷	光輝
	森	和也
	DANDAR	**TUGULDUR**

No **15**
Name 小畠 一泰

Kure National College of Technology

1　目的

組み込みマイコン dsPIC を用いて, 非接触距離計や電光掲示板の製作を行うことで, マイコンの応用について理解するとともに, C 言語に習熟することを目的とする.

2　解説

2.1　赤外線距離センサ

シャープの距離モジュール GP2Y0A02YK は, 赤外線と PSD(position secsitive detector) を使用して非接触で距離を測定できる. 20〜150cm まで測定可能で, アナログ出力 (電圧出力) である (図. 1 参照). この出力をマイコンのアナログ入力に接続し, A/D 変換してディジタル値として取得する.

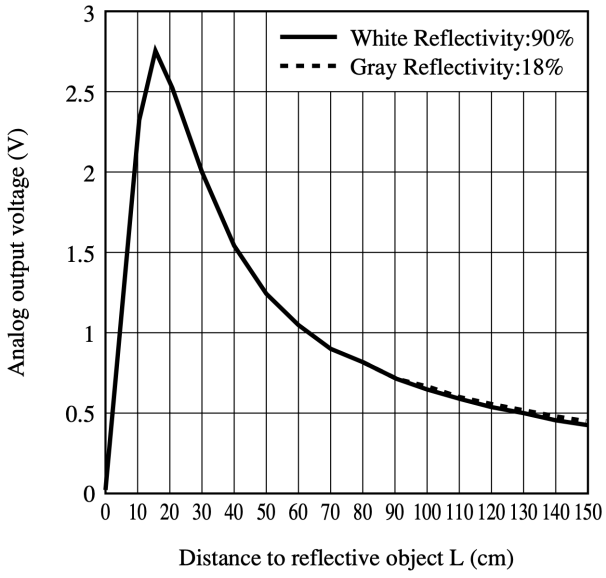

図.1: 反射物と出力電圧の関係

2.2　ドットマトリックス LED

8×8 ドットマトリックス LED(C/A-3889EG) は, 64 個の LED を発光できる部品である. これに LED ディスプレイドライバ (MAX7219) を用いてダイナミック点灯制御を行う. MAX7219 はシリアル入力 / 出力顧問カソードディスプレイドライバで, 最大 8 桁の 7 セグメント LED ディスプレイ, バーグラフディスプレイ, または 64 個の LED を制御できる.

3　使用器具

1. PC (No. 4)
2. 実験キット (No. 6)

4　プログラム実習

4.1　距離センサ

1. smp_INT() を用いてセンサが出力する電圧値を LCD に表示せよ．ただし, 関数が返す値は 0 から 4095 の整数で, 5V が 4095 に対応する．また, 一定周期で測定と表示を繰り返すようにすること．その際前回の表示を消してから表示すること．実行したら反射物の距離によって出力値が変わることを確認せよ．

```
#include "c:\work\e3exp.h"

int main(void) {
  init_LCD();
  init_ADC();

  char msg[10];
  int sensing_position = 0;

  while (1) {
    sensing_position = smp_INT();
    sprintf(msg, "%4d", sensing_position);
    clr_LCD();
    put_str(msg);
    delayms(200);
  }
}
```

2. (1) のプログラムを改良して, 電圧値が表示されるようにせよ. 電圧値は実数なので, smp_INT() の結果を 0.001221 倍して単精度実数に置き換え, sprintf() を用いて文字に変換し表示せよ. 実行したら反射物の距離によって出力電圧が図 8.1 のようになることを確認せよ.

結果のグラフを図. 2 に示す.

```
#include "c:\work\e3exp.h"

int main(void) {
  init_LCD();
  init_ADC();

  char msg[10];
  int sensing_position = 0;
  double converted_voltage = 0.0;

  while (1) {
    sensing_position = smp_INT();
    converted_voltage = (double)sensing_position * (double)(5. / 4096.);
    sprintf(msg, "%8.6f", converted_voltage);
    clr_LCD();
    put_str(msg);
    delayms(200);
  }
}
```

3. (1) のプログラムを改良して, 距離情報に変換することで非接触距離計を作れ. 曲線の関数をピッタリ当てはめるのは難しいので, 次の近似式を用いよ.

$$V = 0.49 \sim 1.99 \text{のとき} d = \frac{56}{v - 0.12}$$

$$V = 1.99 \sim 2.80 \text{のとき} d = \frac{30}{v - 1.0}$$

$$\text{それ以外のとき} d = 0$$

結果のグラフを図. 2 に示す.

```c
#include "c:\work\e3exp.h"

double voltage_to_distance(double voltage);

int main(void) {
  init_LCD();
  init_ADC();

  char msg[10];
  int sensing_position = 0;
  double converted_voltage = 0.0;
  double distance = 0.0;

  while (1) {
    sensing_position = smp_INT();
    converted_voltage = (double)sensing_position * (double)(5. / 4096.);
    distance = voltage_to_distance(converted_voltage);
    sprintf(msg, "%8.6f", distance);
    clr_LCD();
    put_str(msg);
    delayms(200);
  }
}
```

```
24
25
26 double voltage_to_distance(double voltage) {
27   if (0.49 < voltage && voltage <= 1.99)
28     return (56.0 / voltage - 0.12);
29
30   if (1.99 < voltage && voltage <= 2.80)
31     return (30.0 / voltage - 1.00);
32
33   return 0;
34 }
```

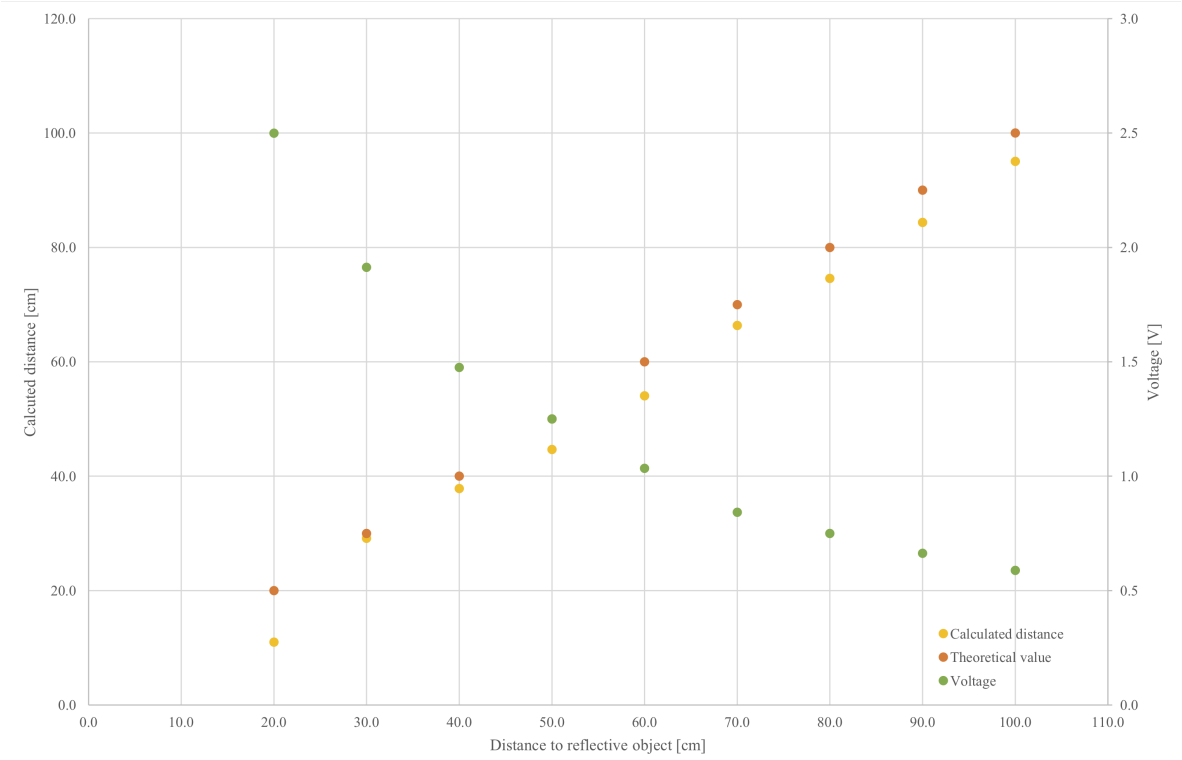

図.2: 出力電圧と非接触距離計

4.2 8×8 ドットマトリックス LED

1. 8×8 ドットマトリックス LED に 1 個分のデータを表示させよ. LED を初期化し, 文字配列に表示するデータ (例: `char a[] = { 0x1f, 0x24, 0x44, 0x24, 0x1f, 0x21, 0x7f, 0x01 }`) をセットし, for 文を使って i を 0 から 7 まで変化させ, アドレス 1 から 8 に `a[0]` から `a[7]` の値を `sendone()` で順次転送する.

```
#include "c:\work\e3exp.h"

int main(void) {
  init_MTLED();

  char a[] = { 0x1f, 0x24, 0x44, 0x24, 0x1f, 0x21, 0x7f, 0x01 };

  int i = 0;
  for (i = 0; i < 8; i++) {
    send_one(i + 1, a[i]);
  }

  while(1)
    ;
}
```

2. 16×8 ドットマトリックス LED にデータを表示せよ. LED を初期化し, 文字配列に表示するデータを 16 個セットする. Send_dual() を使って, アドレス 1 に a[0] , アドレス 1 に a[8] を転送 アドレス 2 に a[1] , アドレス 2 に a[9] を転送 のようにデータを 16 個順次転送する.

```
#include "c:\work\e3exp.h"

int main(void) {
  init_MTLED();

  char a[] = {
    0x1f, 0x24, 0x44, 0x24, 0x1f, 0x7f, 0x49, 0x49,
    0x49, 0x36, 0x21, 0x7f, 0x01, 0x21, 0x7f, 0x01
  };

  int i = 0;
  for (i = 0; i < 8; i++) {
    send_dual(i + 1, a[i], i + 1, a[i + 8]);
  }

  while(1)
    ;
}
```

3. 8×8 ドットマトリックス LED にデータをスクロールさせながら表示するプログラムを作成せよ．LED を初期化したあと, 表示させるデータを a[0] ～a[23] とする．図のように先頭の配列の添え字は $0, 1, ..., 23$ まできたら再び 0, 末尾の配列の添え字は先頭から $+7$ となる．

```c
#include <stdio.h>
#include "c:\work\e3exp.h"

int main(void) {
  init_MTLED();

  char char_set[] = {
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x1f, 0x24,
    0x44, 0x24, 0x1f, 0x00, 0x7f, 0x49, 0x49, 0x49,
    0x36, 0x00, 0x7f, 0x08, 0x14, 0x22, 0x41
  };

  int offset = 0;
  int i = 0;

  while (1) {
    for (i = 0; i < 8; i++) {
      send_one(i + 1, char_set[(i + offset) % 24]);
    }
    delayms(200);
    offset++;
  }
}
```

4. 16×8 ドットマトリックス LED にデータをスクロールさせながら表示するプログラムを作成せよ.

   (6) のデータを 40 個に増やす. 0 回目は (5) のように a[0] ～a[7] と a[8] ～a[15] を転送し, (6) のように繰り返すこと.

```
#include <stdio.h>
#include "c:\work\e3exp.h"

int main(void) {
  init_MTLED();

  char char_set[] = {
    0x00, 0x02, 0x25, 0x21, 0x21, 0x21, 0x01, 0x00,  // こ
    0x7f, 0x00, 0x12, 0x15, 0x55, 0x3e, 0x51, 0x00,  // ば
    0x21, 0x2e, 0x70, 0x22, 0x15, 0x11, 0x11, 0x00,  // た
    0x7e, 0x00, 0x10, 0x11, 0x7e, 0x10, 0x10, 0x00   // け
  };

  int offset = 0;
  int i = 0;

  while (1) {
    for (i = 0; i < 8; i++) {
      int address = i + 1;
      int left = i + offset;
      int right = left + 8;
      int text_length = 4

      send_dual(
        address,
        char_set[left % (text_length * 8)],
        address,
        char_set[right % (text_length * 8)]
      );
```

```
30      }
31      delayms(200);
32      offset++;
33    }
34  }
```

5　その他

5.1　8×8 ドットマトリックス LED の転送データの作成の困難性と解決法について

本実験では各自が点灯させたい LED を手作業でデータ化していたが, テキストを表示させる場合, 特に日本語の場合は形が複雑で作成が困難であると考える. そこで 8×8 ドットフォントの美咲フォントゴシックを使用することでより簡単にデータ作成をしようと思う. 方法としては, テキストを一語一語に分解, それぞれを画像化し読み込んで 2 値化する. 次にそれを 1 列ごとに 16 進数に変換し出力する. すると下のように「こばたけ」というテキストを変換できる.

```
$ python create_8x8_dot_matlix_data.py
> こばたけ
>
> ['こ', 'ば', 'た', 'け']
> ['0x00', '0x02', '0x25', '0x21', '0x21', '0x21', '0x01', '0x00']
> ['0x7f', '0x00', '0x12', '0x15', '0x55', '0x3e', '0x51', '0x00']
> ['0x21', '0x2e', '0x70', '0x22', '0x15', '0x11', '0x11', '0x00']
> ['0x7e', '0x00', '0x10', '0x11', '0x7e', '0x10', '0x10', '0x00']
```

このプログラムを応用すれば, どんなに大きなマトリックスになっても画像やテキストをデータ化することが一瞬でできる.

以下リンクより作成したプログラムの詳細やダウンロードを行うことができる.

https://github.com/kobakazu0429/text-to-8x8-dot-matlix

参考文献

- 山口晶大: 特集 PIC で体験するマイコンの世界, トランジスタ技術 8 月号, p.97-158 (2007)
- 8×8 ドットマトリックス LED 16 進数変換ツールを作ってみた - 自分成長日記 [https://koniko2.blog.fc2.com/blog-entry-2458.html]