

3色12点VU/リニアLEDレベル メーター(品番JVU-0301)説明書

概要

本キットは、マイクロチップ社、8ビットマイクロコントローラ、ミッドレンジ、エンハンスシリーズPIC16F1826を使用した3色12点LEDバーグラフ(棒状)表示VUレベルメーター(品番JVU-0301完成品)です。

通常のVUレベルメーターとして、オーディオ信号などの入力信号を対数(VU)スケールで棒状表示するほか、リニア(直線)表示切替が可能で、DC直流電圧をリニアスケールで3色12点LEDバーグラフ(棒状)表示ができます。

特長および機能

1. 3色(緑、黄、赤)12点LED表示
2. VU/リニア(直線)表示切替可
3. 高速/低速サンプリング速度切替可
4. 各色LEDを直列接続し、定電流ドライブすることにより消費電流の低減化(約60mA 全LED点灯時)
5. PIC内蔵10ビットA/D変換モジュール使用
6. A/D変換値の移動平均によるソフトウェアローパスフィルタ採用
7. 利得4倍(12dB)の全波整流入力アンプ付
8. 推奨電源電圧は12V

電気的特性

1. 動作電源電圧 10V~15V(12V推奨)
2. 動作電流 約60mA @12V、全LED点灯時
約20mA @12V、全LED消灯時
3. 入力アンプゲイン 12dB(4倍)
4. 入力電圧範囲
AC入力 0~0.256Vrms(0~0.717Vp-p)*注1
DC入力 0~0.256V*注1
5. サンプリング速度 1mS/4mS(高速/低速)
6. 表示速度 16mS/64mS(高速/低速)
7. ADC基準電圧 1.024V
8. 分解能 1mV/bit
9. LED定電流源 15mA/LEDセグメント

準備するもの

1. 12V直流電源、100mA以上のもの。弊社12V三端子電源キット JPS-0161+12Vなど
2. 3ピンのピンソケット(メス)2個

使い方

1. VU表示レベルメーターとして使用する場合
SW1スイッチをAC IN側にスライドする。入力にカップリング電解コンデンサ(C5)が入り、交流カップリングとなる。

コネクタCN2の3番ピンに入力信号、2番ピンにグランド(GND)を接続。信号源にはオーディオ信号などの交流信号を使う。
＜調整の仕方＞基準となる信号を入力し、半固定抵抗(TP1)を回し、所定のレベルのLEDを点灯させる。入力には利得4倍(12dB)の全波整流入力アンプが入っているため、本機の最大入力電圧(*注1)は0.256Vrms(0.717Vp-p)となる。このレベルを超える入力は、半固定抵抗(TP1)を左回し、最大入力レベルを超えないようにすること。CPU入力電圧がVdd(5V)を超えると故障の原因となる。12点LED、デシベル(dB)及び10ビットA/D換算値の関係を表1に示す。

2. リニア(直線)表示レベルメーターとして使用する場合
SW1スイッチをDC IN側にスライドする。入力バイポーラ電解コンデンサ(C5)がショートされ、直流カップリングとなる。コネクタCN2の3番ピンに入力信号、2番ピンにグランド(GND)を接続。信号源は、DC直流電圧で、入力電圧をリニアスケールでLEDバーグラフ表示できる。
＜調整の仕方＞基準となる電圧用意し、半固定抵抗(TP1)を回し、所定のレベルのLEDを点灯させる。入力には利得4倍(12dB)の全波整流入力アンプが入っているため、本機の最大入力電圧(*注1)は256mV(0.256V)となる。このレベルを超える入力は、半固定抵抗(TP1)を左回し、最大入力レベルを超えないようにすること。CPU入力電圧がVdd(5V)を超えると故障の原因となる。12点LEDと10ビットA/D換算値の関係を表2に示す。
3. 高速/低速サンプリング速度切替
音声、音楽のように振幅変化の早い入力信号に対しては、速度切換を高速(FAST)にすると表示速度が速くなり、より正確な表示が可能となる。

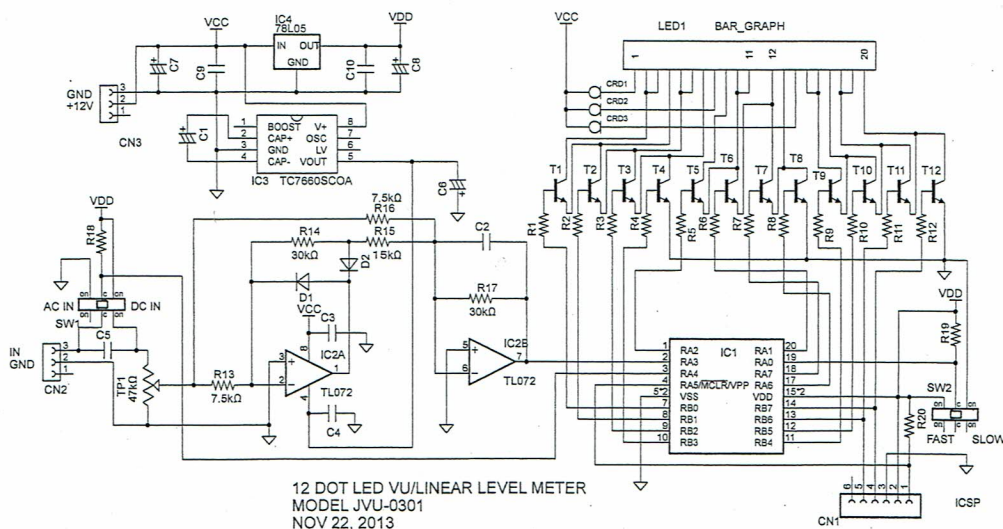
プログラムについて

MPLAB XC8 C コンパイラ ライトモードV1.21を使用。
ROM(プログラムメモリ) 2048ワード中533使用 約26%
RAM(データメモリ) 884バイト中47バイト使用 約5%
参考資料としてソースファイルのリスト(Listing)を添付する。
＜プログラムの特記事項＞
サンプリング周期にウォッチドックタイマー(WDT)を使う。
A/Dサンプル、変換時以外はCPU及びその周辺モジュールがスリープ状態となっている。また、16個のA/D変換値の移動平均を使い、プログラムによるローパスフィルタを採用した。
＜注意＞CONFIGワードのコードプロテクションビットはOFFになっており、ICプログラムメモリの上書き、読み込みが自由にできますが、ICSPを使用しプログラムを変更した場合には、保証できなくなりますから注意してください。また、プログラムに関するご質問はお受けできませんのでご了承ください。

注1: 全LEDを点灯させるために必要な最低入力電圧範囲のこと。

入力電圧範囲の可変は、半固定抵抗(TP1)で調整する。本機の表示は、入力電圧に対し相対的に点灯するので、入力電圧範囲は半固定抵抗(TP1)を使い自由に決めることができる。例えば、基準電圧として6Vを入力し、半固定抵抗(TP1)を左回し、LED12(全LED)を点灯させた時、各LEDあたりの電圧は0.5Vとなる。

3色12点VU/リニアLEDレベルメーター
(品番JVU-0301)回路図



3色12点VU/リニアLEDレベルメーター
(品番JVU-0301)組立図

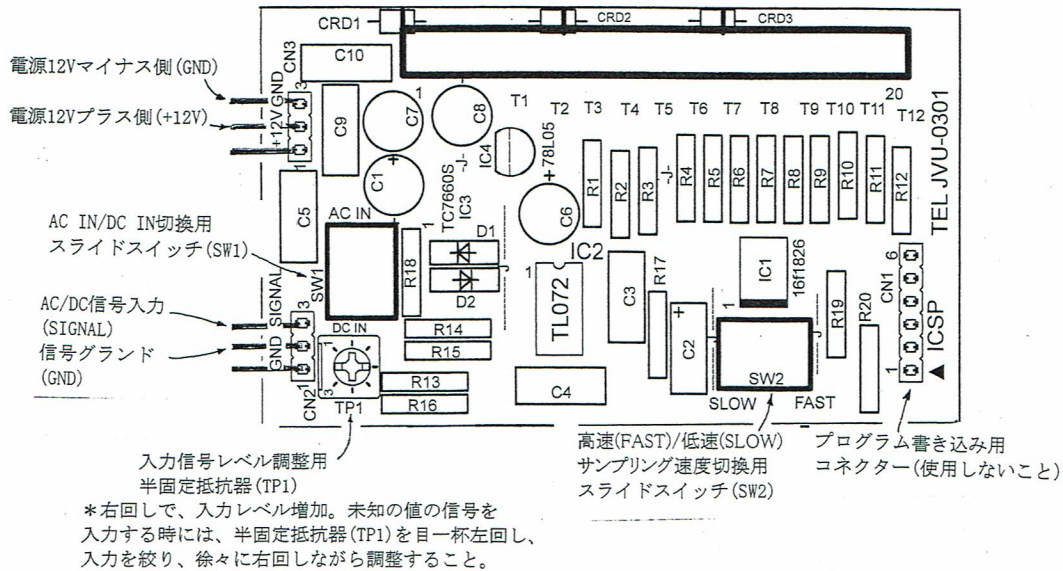


表1: 12点LED、デシベル (dB) 及び10ビットA/D換算値の関係 (VU表示)

$$10\text{ビットA/D換算値} = 408 \times 10^{Gv/20}$$

点灯 LED 番号	0	1	2	3	4	5	6	7	8	9	10	11	12
電圧 ゲイン Gv (dB)	-20	-15	-10	-7	-5	-3	-1	0	+1	+3	+5	+8	
10ビット A/D 換算値	< 20	40	72	129	182	229	288	363	408	457	576	725	1024

表2: 12点LEDと10ビットA/D換算値の関係 (リニア表示)

点灯 LED 番号	0	1	2	3	4	5	6	7	8	9	10	11	12
10ビット A/D 換算値	< 76	155	234	313	392	471	550	629	708	787	866	945	1024

TEL エレクトロニクス・キット (有) 谷岡電子
〒164-0003 東京都中野区東中野 1-51-13
大島ビル第一別館 402 ☎ (03)3366-4552

参考資料

3色12点VU/リニアLEDレベルメーター (品番JVU-0301) Cソースファイルリスト

```

1 //***** 12DOT 3COLOR LED VU/LINEAR LEVEL METER *****
2 * Project File: JVU_0301
3 * Source File : JVU_0301.c
4 * Features:
5 * *PIC enhanced mid-range 16F1826
6 * *12DOT 3COLOR LED BARGRAPH
7 * *SINGLE POWER SUPPLY +12V
8 * *HARDWARE SELECTABLE VU/LINEAR DISPLAY
9 * *FAST/SLOW DISPLAY SPEED (DISPLAY TIME: 16ms/64ms)
10 * *SOFTWARE LOW PASS FILTER REDUCES EFFECTIVELY FLICKERING LIGHT PROBLEM
11 * Pin Assignments:
12 * *FAST/SLOW SWITCH INPUT <RA0>
13 * *SIGNAL INPUT (ANALOG INPUT) <RA3>
14 * *AC/DC SWITCH INPUT <RA4>
15 * *BARGRAPH DRIVING OUTPUT <RA7,RA6> <RA2,RA1> <RB7:RB0>
16 * Compiler: xc8 v1.21
17 * Jan 4,2014: Coded by yuji Tanioka(TEL company)
18 *****/
19 #include <xc.h>
20
21 #pragma config MCLRE = OFF, CP = OFF, BOREN = OFF, WDTE = SWDTEN,
22 #pragma config FVRCON = ON, CPD = OFF, CLKOUTEN = OFF, FCMEN = OFF
23 #pragma config WRT = OFF, STVREN = OFF, BORV = LO, LVP = OFF,
24 #pragma config FOSC = INTOSC, PLEN = OFF
25
26 #define _XTAL_FREQ 4000000 //4MHz
27
28 typedef volatile bit Bool; //flags
29
30 #define TRUE 1
31 #define FALSE 0
32 #define AC_DC_IN RA4
33 #define FAST_SLOW RA0
34 #define NSAMPLE 16
35
36 //function prototype declaration
37 void reg_init(void); //F13
38 void linear_dpy(unsigned int); //F22
39 void vu_dpy(unsigned int); //F23
40 void seg_table(unsigned char); //F24
41
42 //global variables
43 Bool sleep_flag = FALSE;
44
45 /*Main function*/
46 void main(void)
47 {
48     static unsigned int sam_buf[NSAMPLE];
49     static unsigned char pos = 0;
50     static unsigned int adc_sum = 0;
51     static unsigned int adc_ave;
52
53     reg_init(); //F13
54
55     for(;;)
56     {
57         if(FAST_SLOW)
58             WDTCONbits.WDTPS = 0b000000; //FAST INTERVAL 1ms
59         else
60             WDTCONbits.WDTPS = 0b000010; //SLOW INTERVAL 4ms
61
62         if(sleep_flag)
63         {
64             SWDTEN = 1; //WATCHDOG TIMER ENABLED IN SLEEP
65             sleep_flag = FALSE;
66             SLEEP();
67         }
68
69         SWDTEN = 0;
70
71         ADON = 1; //START A SAMPLING
72         _delay(10); //acquisition time(10us)
73         GO_DONE = 1; //START AN AD CONVERSION
74         while(!GO_DONE); //WAIT UNTIL ADC COMPLETES
75
76         adc_sum -= sam_buf[pos]; //SUBTRACT OLD SAMPLE FROM TOTAL
77         sam_buf[pos] = (unsigned int)ADRESH << 8 | ADRESL;
78         adc_sum += sam_buf[pos]; //SAVE NEW SAMPLE
79         //ADD IT TO RUNNING TOTAL
80
81         if(++pos == NSAMPLE) //DISPLAY INTERVAL 16ms/64ms
82         {
83             pos = 0;
84             adc_ave = adc_sum >> 4; //DIVIDED BY 16
85             if(AC_DC_IN)
86                 linear_dpy(adc_ave); //LINEAR display, F22
87             else
88                 vu_dpy(adc_ave); //VU display, F23
89
90             sleep_flag = TRUE;
91         }
92     }
93 }
94
95 /*function definition F13, register initialization*/
96 void reg_init(void)
97 {
98     OSCCON = 0b01101000; //4Mhz,IRCF=01101
99     OSTUNE = 0x00;
100     TRISA = 0b00011001; //RA4:RA3:RA0>INPOT
101     TRISB = 0b00000000; //RB7:RB0> OUTPUT
102     ANSELA = 0b00001000; //RA3> ANALOG
103     ANSELB = 0x00; //DIGITAL I/O
104     WPUA5 = 1; //WAKE PULL-UP RA5
105     WPUB = 0x00; //WAKE PULL-UP DISABLED
106     LATA7 = 1; LATA6 = 0; //PORTA ALL LED OFF
107     LATA2 = 1; LATA1 = 1; //PORTA ALL LED OFF
108     LATB = 0b01110111; //PORTB ALL LED OFF
109
110     //configure ADC
111     ADCON0 = 0b00001100; //ANALOG CH SELECT CHS=00011 AN3
112     //00011--
113     //-----0
114     ADCON1 = 0b10010011; //ADFM=1, Right justified
115     //1-----
116     //001----
117     //-----0--
118     //-----11
119     //-----
120     //-----
121     //-----
122     //-----

```

```

123 //configure FVR(fix voltage reference control register)
124 FVRCON = 0b11000001;
125 //1-----
126 //1----- FVREN=1, enabled
127 //1----- always '1' with LDO (16F1826)
128 //-----00-- off
129 //-----01 ADFVR=01, 1.024V
130 }
131
132 /*function definition F22, LINEAR SCALE DISPLAY*/
133 void linear_dpy(unsigned int val) //F22
134 {
135     unsigned char led_seg;
136
137     if(val < 76) led_seg = 0;
138     else if(val < 155) led_seg = 1;
139     else if(val < 234) led_seg = 2;
140     else if(val < 313) led_seg = 3;
141     else if(val < 392) led_seg = 4;
142     else if(val < 471) led_seg = 5;
143     else if(val < 550) led_seg = 6;
144     else if(val < 629) led_seg = 7;
145     else if(val < 708) led_seg = 8;
146     else if(val < 787) led_seg = 9;
147     else if(val < 866) led_seg = 10;
148     else if(val < 945) led_seg = 11;
149     else if(val <= 1024) led_seg = 12;
150
151     seg_table(led_seg); //F24
152 }
153
154 /*function definition F23, LOG/UV SCALE DISPLAY*/
155 void vu_dpy(unsigned int val) //F23
156 {
157     unsigned char led_seg;
158
159     if(val < 20) led_seg = 0;
160     else if(val < 40) led_seg = 1;
161     else if(val < 72) led_seg = 2;
162     else if(val < 129) led_seg = 3;
163     else if(val < 182) led_seg = 4;
164     else if(val < 229) led_seg = 5;
165     else if(val < 288) led_seg = 6;
166     else if(val < 363) led_seg = 7;
167     else if(val < 408) led_seg = 8;
168     else if(val < 457) led_seg = 9;
169     else if(val < 576) led_seg = 10;
170     else if(val < 725) led_seg = 11;
171     else if(val <= 1024) led_seg = 12;
172
173     seg_table(led_seg); //F24
174 }
175
176 /*function definition F24, BARGRAPH SEGMENT TABLE*/
177 void seg_table(unsigned char seg_num) //F24
178 {
179     switch(seg_num)
180     {
181         case 0:
182             LATA7 = 1; LATA6 = 0; LATA2 = 1; LATA1 = 1;
183             LATB = 0b01110111; break;
184
185         case 1:
186             LATA7 = 1; LATA6 = 0; LATA2 = 1; LATA1 = 1;
187             LATB = 0b01111111; break;
188
189         case 2:
190             LATA7 = 1; LATA6 = 0; LATA2 = 1; LATA1 = 1;
191             LATB = 0b01111110; break;
192
193         case 3:
194             LATA7 = 1; LATA6 = 0; LATA2 = 1; LATA1 = 1;
195             LATB = 0b01111100; break;
196
197         case 4:
198             LATA7 = 1; LATA6 = 0; LATA2 = 1; LATA1 = 1;
199             LATB = 0b01111000; break;
200
201         case 5:
202             LATA7 = 1; LATA6 = 1; LATA2 = 1; LATA1 = 1;
203             LATB = 0b01111000; break;
204
205         case 6:
206             LATA7 = 1; LATA6 = 1; LATA2 = 0; LATA1 = 1;
207             LATB = 0b01111000; break;
208
209         case 7:
210             LATA7 = 1; LATA6 = 1; LATA2 = 0; LATA1 = 0;
211             LATB = 0b01111000; break;
212
213         case 8:
214             LATA7 = 0; LATA6 = 1; LATA2 = 0; LATA1 = 0;
215             LATB = 0b01111000; break;
216
217         case 9:
218             LATA7 = 0; LATA6 = 1; LATA2 = 0; LATA1 = 0;
219             LATB = 0b11110100; break;
220
221         case 10:
222             LATA7 = 0; LATA6 = 1; LATA2 = 0; LATA1 = 0;
223             LATB = 0b11101000; break;
224
225         case 11:
226             LATA7 = 0; LATA6 = 1; LATA2 = 0; LATA1 = 0;
227             LATB = 0b11001000; break;
228
229         case 12:
230             LATA7 = 0; LATA6 = 1; LATA2 = 0; LATA1 = 0;
231             LATB = 0b10001000; break;
232
233         default: break;
234     }
235 }

```

2014.01.04