

無線リモートコントローラ受信部 (品番RF-0299-RX)説明書

概要

本キットは、マイクロチップ社ナノワット超低電力(XLP)テクノロジー、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PIC 16F1826を使用した無線リモートコントローラ(ラジコン)受信部(品番RF-0299-RX)(完成品)です。
無線リモートコントローラ(ラジコン)送信部(品番RF-0299L-TX)と組み合わせて、315メガ帯での無線リモコン制御ができます。
本機は送信側キースイッチ対応したキーコードを16進で出力します。
また、本機の出力に10進変換アダプター(品番HtoD-0297)を繋げば出力信号が保持され、種々の機器の制御が簡単になります。

特長

- 消費電力が小さい。約2.5mA@5V
- 小型でローコスト
- 16個のキースイッチを使って最大26種類のキーコード(5ビット)を受信。
- 16個のキースイッチ中、二度押しスイッチが4個、四度押しスイッチが2個、単発スイッチが10個ある。
- デバイスコードは、4ビットDIPスイッチを使って最大14個のデバイスを設定できる。
- 通信可能距離は直線で約50m(注1)
- 動作電源電圧は4.0V-5.0V

準備するもの

- 無線リモートコントローラ(ラジコン)受信部本体(品番RF-0299-RX)
完成品、ICはプログラム済み、動作確認済み
- 無線リモートコントローラ(ラジコン)送信部(品番RF-0299L-TX)
- 5V電源(単一電池3本直列接続可能)
- 3ピンのピンソケット(メス)1個
- キーパッド(品番ACC-0295-KEY4x4)1個
- 10cm程の線材(アンテナとして使用する)。
- 10進変換アダプター(品番HtoD-0297)(オプション)

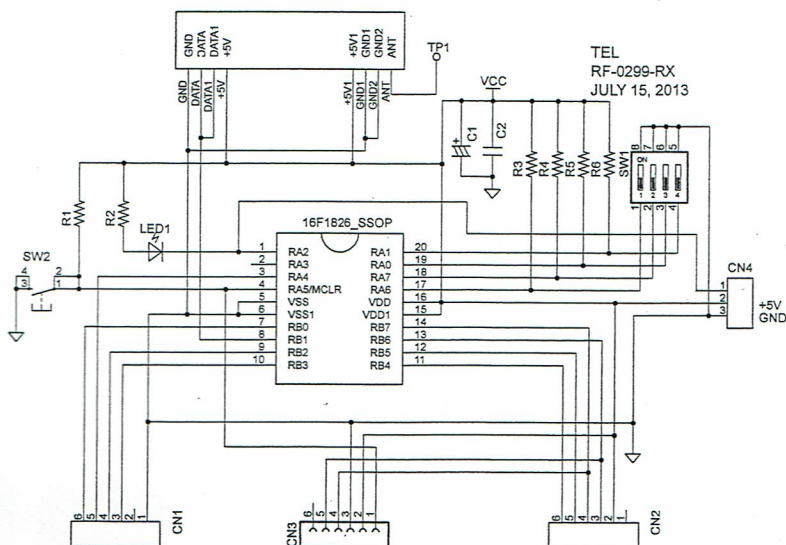
組み立て

- キーパッドのピンヘッダCN1,CN2と本体のピンソケットCN1,CN2のコンネクターを差し込む。
- 2P電池ケースのプラス側を本体(CN4)5V、マイナス側をグラウンドGNDにピンソケットを使ってCN4に繋ぐ。
極性に十分注意すること。
- 本機に電源が入れば直ちにラジコン受信機として動作する。

キーパッド各スイッチと出力(5ビットキーコード)の関係

送信側のキーパッド各スイッチと受信側の出力の関係を表1に示す。
また10進変換アダプター(品番HtoD-0297)を接続すれば送信側キーパッドの各スイッチに対応した発光ダイオード(LED)が点灯する。デバイスコードのLED表示は下三桁のみ。例えば、デバイスコードが"1110"の場合、"110"と表示される。

無線リモートコントローラ受信部 (品番RF-0299-RX) 回路図



単発パルス出力について

本機は受信の度にLED1が点灯し、コネクターCN4の1番ピンに約65mSの負論理パルス信号が出力されている。つまり、1番ピンは通常Hiであるが、送信側のキーパッドの何れかのスイッチが押されると発光ダイオードLED1と連動してLoの短いパルス信号が出る。特定のスイッチに対する単発パルス出力が必要なときは、1番ピンと10進変換アダプター(品番HtoD-0297)の特定のスイッチのコネクター出力ピンとNORゲートをとること。(後述する使用例参照)

キースイッチ機能

- 本機にキーパッドは必要なし。キーパッドを接続しないこと。
送信側キーパッドのスイッチ機能を下記する。
- 二度押しスイッチはトグルスイッチとして使用できる。
例えば、パワースイッチなどON/OFFスイッチ。
 - 四度押しスイッチは4個の設定が必要な制御に使用できる。
例えば、音量調整(切る、小、中、大)、風速調整(切る、弱、中、強)。
 - 単発キーは十連のロータリースイッチとして使える。
また、スライドスイッチ(2個以上のスイッチを使う)としても使用できる。
初期値(デフォルト値)は10進変換アダプターで設定しSW7とした。
 - リセット(RESET SW1)、電源投入時の二度押し四度押しキーのデフォルト値は、SW1-0, SW2-0, SW3-0, SW4-0, SW5-0, SW6-0である。

デバイスコードの変更

デバイスコードは4PDIPスイッチSW1を使って設定する。1番がLSB(LEAST-SIGNIFICANT-BIT)、ONにするとTRUE(負論理で'0')となる。4ビット使用で"0001"から"1111"まで14種類の設定が可能。
"0001" "0010" "0011" "0100" "0101" "0110" "0111" "1001" "1010" "1011" "1100" "1101" "1110"(デフォルト) "1111"
但し、"0000","1000"(DIPスイッチ1,2,3番ON)は使用禁止。
例えば"1110"の場合DIPスイッチの1をON、2,3,4をOFFとする。
デバイスコードは送信側と必ず一致させる。

通信形式について

本機はEUSART(Enhanced Universal Synchronous Receiver Transmitter)周辺回路モジュールを使用し、無線でシリアル受信を行う。
一個のデータは8ビットで、LSBからMSBの順にボーレートジェネレーターのクロック信号に同期して自動的にサンプリングされRCREGに格納される。
まず、8個のデータを一回分として、最初に文字データ'T','E','L'を受信し、続いて、デバイスコード、データコード、反転デバイスコード、反転データコード、最後に文字データ'E'を受信し一回分を終了する。
次に同じ8個のデータを三回、1msの間隔で連続受信し、一回の通信を完了する。

プログラムについて

MPLAB XC8 C コンパイラ ライトモードV1.20を使用。
ROM(プログラムメモリ) 2048ワード中1172使用 約57%
RAM(データメモリ) 884バイト中40バイト使用 約5%
参考資料としてCソースファイルのリスト(Listing)を添付する。
(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:通信可能距離は直線は障害物、アンテナの長さ、形状などによって大きく変わってくる。受信側のアンテナについては制限がないが、送信側のアンテナを余り大きくすると電波法に抵触する可能性があるので注意すること。

T.E.L. キット製造販売

(有) 谷岡電子

〒164 東京都中野区東中野1-51-13
-0003 大島ビル第一別館402
☎ 03-3366-4552

無線リモートコントローラ受信部 (品番RF-0299-RX) 本体

無線リモートコントローラ受信部
(品番RF-0299-RX) と
10進変換アダプター(品番HtoD-0297)
を組み合わせて使用した例

(注意)
デバイスコード"0000","1000"
(DIPスイッチ1,2,3番ON)は
使用禁止。動作しません。

ICSP(In-Circuit
Serial Programming)
用コネクター。
使用しないこと。

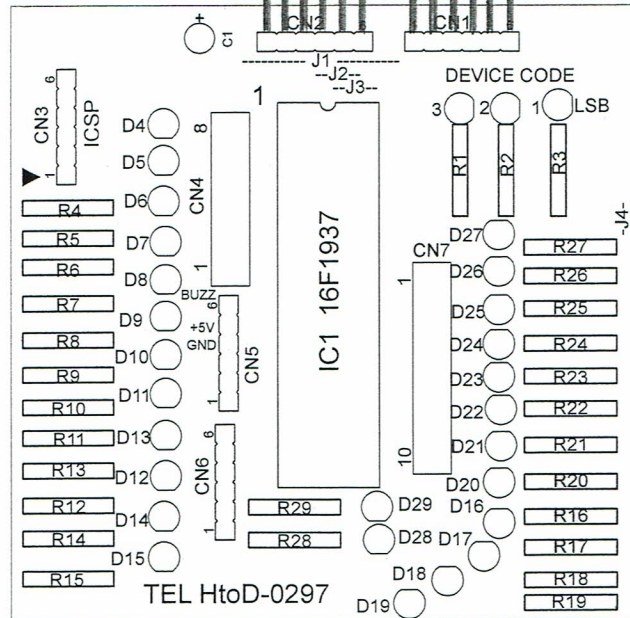
DIPスイッチ
4ビットデバイス
コードを設定。
例えば"1110"は
1番をON, 2,3,4番
をOFFとする。
送信側も"1110"と
設定すること。

LED1
受信ごとに
約65mS点灯

ANT
10cm程の線材を
アンテナとして
ここに接続する。

電源グラウンドGND
5V電源プラス側
単発パルス出力
(約65mS)

RESET(SW2)
本機をリセットし
デフォルト値に
戻す。
10進変換アダプター
(品番HtoD-0297)
にリセットをかけ
デフォルト(初期値)
に戻す。



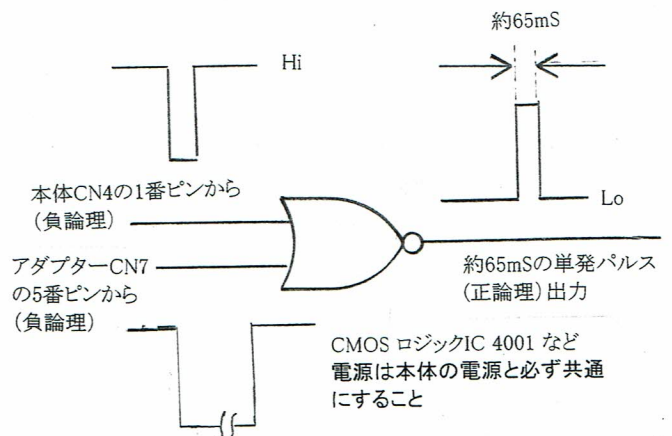
10進変換アダプター
(品番HtoD-0297)

10進変換アダプターの詳細
については10進変換アダプター
の説明書参照

表1 キーパッド各スイッチと出力(5ビットキーコード)の関係

コネクター番号 ポート	CN2-3 RB6	CN2-4 RB7	CN2-5 RB5	CN2-6 RB4	CN1-3 RB3	CN1-4 RB2	CN1-5 RA4	CN1-6 RB0
	3ビットデバイスコード(負論理) デフォルト値='110'				5ビットキーコード(正論理)			
SW1-0	1	1	0	0	0	0	0	0
SW1-1	1	1	0	0	0	0	0	1
SW2-0	1	1	0	0	0	0	1	0
SW2-1	1	1	0	0	0	0	1	1
SW3-0	1	1	0	0	0	1	0	0
SW3-1	1	1	0	0	0	1	0	1
SW4-0	1	1	0	0	0	1	1	0
SW4-1	1	1	0	0	0	1	1	1
SW5-0	1	1	0	0	1	0	0	0
SW5-1	1	1	0	0	1	0	0	1
SW5-2	1	1	0	0	1	0	1	0
SW5-3	1	1	0	0	1	0	1	1
SW6-0	1	1	0	0	1	1	0	0
SW6-1	1	1	0	0	1	1	0	1
SW6-2	1	1	0	0	1	1	1	0
SW6-3	1	1	0	0	1	1	1	1
SW7	1	1	0	1	0	0	0	0
SW8	1	1	0	1	0	0	0	1
SW9	1	1	0	1	0	0	1	0
SW10	1	1	0	1	0	0	1	1
SW11	1	1	0	1	0	1	0	0
SW12	1	1	0	1	0	1	0	1
SW13	1	1	0	1	0	1	1	0
SW14	1	1	0	1	0	1	1	1
SW15	1	1	0	1	1	0	0	0
SW16	1	1	0	1	1	0	0	1

単発パルスの作り方の一例
単発キーSW7と単発パルス出力とのNOR ゲート
から単発パルスを得る方法。



無線リモートコントローラ受信部 (品番RF-0299-RX) Cソースファイルリスト(Listing)

```
1 /***** RF Wireless Remote Controller Receiver *****/
2 * Project name: RF0299RX
3 * File name: RF0299RX.c
4 * Features:
5 * "nano watt extreme low power(XLP), 1.8uWatt typical,
6 * "Micro controller PIC enhanced mid-range 16F1826.
7 * "3 bits device code
8 * "Wide operating voltage, 4.0V-5.0V.
9 * Pin assignments:
10 * "Device code input <RA6:RA7>, <RA1:RA0>
11 * "Data Input <RB1>
12 * "Device Code Output <RB7:RB4>
13 * "Key Code Output(Data Output) <RB3:RA4> <RB2:RB0>
14 * "Led Indicator Output <RA2>
15 * Compiler: xc8 v1.12
16 * AUG 10, 2013. Coded by yuji Tanioka (TEL company)
17 *****/
18 #include <xc.h>
19
20 #pragma config MCLRE = ON, CP = OFF, BOREN = OFF, WDT = OFF,
21 #pragma config PWRTE = ON, CPD = OFF, CLKOUTEN = OFF, FCMEN = OFF
22 #pragma config WR = OFF, STVREN = OFF, BORV = LO, LVF = OFF,
23 #pragma config FOSC = INTOSC, FLEN = OFF
24
25 #define XTAL_FREQ 8000000 //8MHZ
26 #define STR_LEN 8
27 #define ERROR 0xFF
28 #define TRUE 1
29 #define FALSE 0
30
31 /*function prototype declaration*/
32 void reg_init(void); //F13
33 void rf_receive(void); //F7
34 unsigned char getch(void); //F10
35 void output_table(unsigned char pin, unsigned char key); //F11
36 void led_on(void); //F18
37 void interrupt_isr(void); //F17
38
39 typedef bit bool;
40
41 /*GLOBAL VARIABLES*/
42 unsigned char str_buf[STR_LEN + 1];
43 unsigned char str_pos = 0;
44 bool str_flag = FALSE;
45
46 /*Main function*/
47 void main(void)
48 {
49     reg_init(); //F13
50     for(;;)
51     {
52         rf_receive();
53     }
54 }
55
56 /*function definition F13, register initialization*/
57 void reg_init(void)
58 {
59     OSCCON = 0b01110000; //8MHz, HF:IRCF=01110
60     OSCUNE = 0x00;
61     TRISA = 0b11000011; //input <RA6:RA7>, <RA1:RA0>
62     TRISB = 0b00000010; //RB1 INPUT
63     PORTB = 0;
64     RA4 = 0;
65     ANSEL = 0x00; ANSELB = 0x00; //DIGITAL I/O
66     WPUA5 = 0; //WEAK PULL-UP RA5 DISABLED
67     WPUB = 0x00; //WEAK PULL-UP DISABLED
68 }
69
70 nWPEN = 1;
71 LATA2 = 1;
72 BAUDCON = 0b00000000; //0x08
73 //---0--- //CLOCK POLARITY SELECT SCKP=0
74 //---0--- //BRG16=0
75 //SPBRGL = 0x40; //BAUD RATE=2400, BRGH=1, BRG16=1
76 //SPBRGH = 0x03; //832(0x0340), ERROR 0.04%
77 //SPBRGL = 0x0A; //BAUD RATE=300, BRGH=1, BRG16=1
78 //SPBRGH = 0x1A; //6666(0x1A0A), ERROR 0%
79 SPBRGL = 103; //BAUD RATE=1200, BRGH=0, BRG16=0
80 SPBRGH = 0; //103 ERROR 0.16%
81
82 TXIE = 0; //TX1 INTERRUPT DISABLED
83
84 TXSTA = 0b00000000; //0x00
85 //---0--- TXS=0
86 //---0--- TXEN=0
87 //---0--- SYNC=0
88 //---0--- BRGH=0
89
90 RCIE = 1; //RC INTERRUPT ENABLED
91
92 RCSTA = 0b10010000; //0x90
93 //1----- //Serial Port enable (SPEN)
94 //---0--- //use ninth bit (RX9)
95 //---1--- //CREN=1
96 //---0--- //ADDEN=0
97
98 ADDEN = 0;
99 SYNC = 0;
100
101 T2CON = 0b00111011; //0x0B
102 //---0111--- 1:8 POSTSCALER T2OUTPS<3:0>
103 //---0--- TIMER2 OFF
104 //---11--- PRESCALER IS 64 T2CKPS<1:0>
105 TMR2 = 0x00; //TIMER 2 CLEAR
106 PR2 = 0xFF; //TOTAL LED ON TIME:
107 //0.5uSx64x256x8=65.536ms
108 TMR2IE = 1; //TIMER 2 ENABLED
109
110 PEIE = 1;
111 ei();
112 }
113
114 void rf_receive(void) //F7
115 {
116     unsigned char pin_code, key_code, n_key_code, pin_data, n_pin_data;
117
118     pin_code = PORTA;
119     pin_code &= 0xC3; //CLEAR <RA5:RA2>
120     pin_code = pin_code << 2 | pin_code >> 6;
121
122     CREN = 1;
123
124     if(getch() == 'T')
125     {
126         str_pos++;
127     }
128     else
129     {
130         str_pos = 0;
131         return;
132     }
133
134     if(getch() == 'E')
135     {
136         str_pos++;
137     }
138     else
139     {
140         str_pos = 0;
141         return;
142     }
143     else
144     {
145         str_pos = 0;
146         return;
147     }
148 }
149
150 pin_data = getch(); //read device code
151 pin_data &= 0x0F;
152 if(pin_code == pin_data) //device code test
153 {
154     str_pos++;
155 }
156 else
157 {
158     str_pos = 0;
159     return;
160 }
161
162 key_code = getch(); //read key code
163 str_pos++;
164 n_pin_data = ~getch(); //read compliment device code
165 n_pin_data &= 0x0F;
166 if(pin_code == n_pin_data) //pin code compliment test
167 {
168     str_pos++;
169 }
170 else
171 {
172     str_pos = 0;
173     return;
174 }
175
176 n_key_code = ~getch(); //read compliment key code
177 if(key_code == n_key_code) //key code test
178 {
179     str_pos++;
180 }
181 else
182 {
183     str_pos = 0;
184     return;
185 }
186
187 if(!OERR && !FERR)
188 {
189     output_table(pin_code, key_code);
190     led_on();
191 }
192 if(OERR)
193 {
194     CREN = 0;
195     str_pos = 0;
196 }
197
198 void interrupt_isr(void)
199 {
200     if(RCIF)
201     {
202         str_flag = TRUE;
203         str_buf[str_pos] = RCREG;
204     }
205 }
206
207 if(TMR2IE && TMR2IF)
208 {
209     TMR2IF = 0;
210     TMR2ON = 0;
211     LATA2 = 1; //LED OFF
212 }
213
214 /*function definition F18, led on */
215 void led_on(void) //F18
216 {
217     PR2 = 0xFF;
218     TMR2IE = 1;
219     TMR2ON = 1;
220     LATA2 = 0; //LED ON
221 }
222
223 unsigned char getch(void) //F10
224 {
225     while(str_flag == FALSE)
226     {
227         str_flag = FALSE;
228         return str_buf[str_pos];
229     }
230 }
231
232 /*function definition F11, OUTPUT TABLE*/
233 void output_table(unsigned char pin, unsigned char key) //F11
234 {
235     unsigned char sPORTA, sLATB;
236
237     sPORTA = pin;
238     sLATB = LATB;
239     sPORTA &= 0x0F;
240     sLATB &= 0x1F;
241     sPORTA <= 5;
242     sLATB |= sPORTA;
243     LATB = sLATB; //PIN CODE OUTPUT
244
245     switch(key) //KEY CODE OUTPUT
246     {
247     case 0:
248         sLATB = LATB;
249         sLATB &= 0xE2;
250         sLATB |= 0x00;
251         LATB = sLATB;
252         LATA4 = 0;
253         break;
254     case 1:
255         sLATB = LATB;
256         sLATB &= 0xE2;
257         sLATB |= 0x01;
258         LATB = sLATB;
259         LATA4 = 0;
260         break;
261     case 2:
262         sLATB = LATB;
263         sLATB &= 0xE2;
264         sLATB |= 0x00;
265         LATB = sLATB;
266         LATA4 = 1;
267         break;
268     case 3:
269         sLATB = LATB;
270         sLATB &= 0xE2;
271         sLATB |= 0x01;
272         LATB = sLATB;
273         LATA4 = 1;
274         break;
275     case 4:
276         sLATB = LATB;
277         sLATB &= 0xE2;
278         sLATB |= 0x01;
279         LATB = sLATB;
280         LATA4 = 1;
281         break;
282     }
283 }
```

```
137     str_pos = 0;
138     return;
139 }
140
141 if(getch() == 'L')
142 {
143     str_pos++;
144 }
145 else
146 {
147     str_pos = 0;
148     return;
149 }
150
151 pin_data = getch(); //read device code
152 pin_data &= 0x0F;
153 if(pin_code == pin_data) //device code test
154 {
155     str_pos++;
156 }
157 else
158 {
159     str_pos = 0;
160     return;
161 }
162
163 key_code = getch(); //read key code
164 str_pos++;
165 n_pin_data = ~getch(); //read compliment device code
166 n_pin_data &= 0x0F;
167 if(pin_code == n_pin_data) //pin code compliment test
168 {
169     str_pos++;
170 }
171 else
172 {
173     str_pos = 0;
174     return;
175 }
176
177 n_key_code = ~getch(); //read compliment key code
178 if(key_code == n_key_code) //key code test
179 {
180     str_pos++;
181 }
182 else
183 {
184     str_pos = 0;
185     return;
186 }
187
188 if(!OERR && !FERR)
189 {
190     output_table(pin_code, key_code);
191     led_on();
192 }
193 if(OERR)
194 {
195     CREN = 0;
196     str_pos = 0;
197 }
198
199 void interrupt_isr(void)
200 {
201     if(RCIF)
202     {
203         str_flag = TRUE;
204         str_buf[str_pos] = RCREG;
205     }
206 }
```

```
205 if(TMR2IE && TMR2IF)
206 {
207     TMR2IF = 0;
208     TMR2ON = 0;
209     LATA2 = 1; //LED OFF
210 }
211
212 /*function definition F18, led on */
213 void led_on(void) //F18
214 {
215     PR2 = 0xFF;
216     TMR2IE = 1;
217     TMR2ON = 1;
218     LATA2 = 0; //LED ON
219 }
220
221 unsigned char getch(void) //F10
222 {
223     while(str_flag == FALSE)
224     {
225         str_flag = FALSE;
226         return str_buf[str_pos];
227     }
228 }
229
230 /*function definition F11, OUTPUT TABLE*/
231 void output_table(unsigned char pin, unsigned char key) //F11
232 {
233     unsigned char sPORTA, sLATB;
234
235     sPORTA = pin;
236     sLATB = LATB;
237     sPORTA &= 0x0F;
238     sLATB &= 0x1F;
239     sPORTA <= 5;
240     sLATB |= sPORTA;
241     LATB = sLATB; //PIN CODE OUTPUT
242
243     switch(key) //KEY CODE OUTPUT
244     {
245     case 0:
246         sLATB = LATB;
247         sLATB &= 0xE2;
248         sLATB |= 0x00;
249         LATB = sLATB;
250         LATA4 = 0;
251         break;
252     case 1:
253         sLATB = LATB;
254         sLATB &= 0xE2;
255         sLATB |= 0x01;
256         LATB = sLATB;
257         LATA4 = 0;
258         break;
259     case 2:
260         sLATB = LATB;
261         sLATB &= 0xE2;
262         sLATB |= 0x00;
263         LATB = sLATB;
264         LATA4 = 1;
265         break;
266     case 3:
267         sLATB = LATB;
268         sLATB &= 0xE2;
269         sLATB |= 0x01;
270         LATB = sLATB;
271         LATA4 = 1;
272         break;
273     case 4:
274         sLATB = LATB;
275         sLATB &= 0xE2;
276         sLATB |= 0x01;
277         LATB = sLATB;
278         LATA4 = 1;
279         break;
280     }
```

```

273     sLATB = LATB;
274     sLATB &= 0xE2;
275     sLATB |= 0x04;
276     LATB = sLATB;
277     LATA4 = 0;
278     break;
279 case 5:
280     sLATB = LATB;
281     sLATB &= 0xE2;
282     sLATB |= 0x05;
283     LATB = sLATB;
284     LATA4 = 0;
285     break;
286 case 6:
287     sLATB = LATB;
288     sLATB &= 0xE2;
289     sLATB |= 0x04;
290     LATB = sLATB;
291     LATA4 = 1;
292     break;
293 case 7:
294     sLATB = LATB;
295     sLATB &= 0xE2;
296     sLATB |= 0x05;
297     LATB = sLATB;
298     LATA4 = 1;
299     break;
300 case 8:
301     sLATB = LATB;
302     sLATB &= 0xE2;
303     sLATB |= 0x08;
304     LATB = sLATB;
305     LATA4 = 0;
306     break;
307 case 9:
308     sLATB = LATB;
309     sLATB &= 0xE2;
310     sLATB |= 0x09;
311     LATB = sLATB;
312     LATA4 = 0;
313     break;
314 case 10:
315     sLATB = LATB;
316     sLATB &= 0xE2;
317     sLATB |= 0x08;
318     LATB = sLATB;
319     LATA4 = 1;
320     break;
321 case 11:
322     sLATB = LATB;
323     sLATB &= 0xE2;
324     sLATB |= 0x09;
325     LATB = sLATB;
326     LATA4 = 1;
327     break;
328 case 12:
329     sLATB = LATB;
330     sLATB &= 0xE2;
331     sLATB |= 0x0C;
332     LATB = sLATB;
333     LATA4 = 0;
334     break;
335 case 13:
336     sLATB = LATB;
337     sLATB &= 0xE2;
338     sLATB |= 0x0D;
339     LATB = sLATB;
340     LATA4 = 0;

```

Page 5 OF 7

2013.08.10

```

409     LATB = sLATB;
410     LATA4 = 1;
411     break;
412 case 24:
413     sLATB = LATB;
414     sLATB &= 0xE2;
415     sLATB |= 0x18;
416     LATB = sLATB;
417     LATA4 = 0;
418     break;
419 case 25:
420     sLATB = LATB;
421     sLATB &= 0xE2;
422     sLATB |= 0x19;
423     LATB = sLATB;
424     LATA4 = 0;
425     break;
426 default: break;
427
428 }
429
430 }

```

Page 7 OF 7

2013.08.10

```

341     break;
342 case 14:
343     sLATB = LATB;
344     sLATB &= 0xE2;
345     sLATB |= 0x0C;
346     LATB = sLATB;
347     LATA4 = 1;
348     break;
349 case 15:
350     sLATB = LATB;
351     sLATB &= 0xE2;
352     sLATB |= 0x0D;
353     LATB = sLATB;
354     LATA4 = 1;
355     break;
356 case 16:
357     sLATB = LATB;
358     sLATB &= 0xE2;
359     sLATB |= 0x10;
360     LATB = sLATB;
361     LATA4 = 0;
362     break;
363 case 17:
364     sLATB = LATB;
365     sLATB &= 0xE2;
366     sLATB |= 0x11;
367     LATB = sLATB;
368     LATA4 = 0;
369     break;
370 case 18:
371     sLATB = LATB;
372     sLATB &= 0xE2;
373     sLATB |= 0x10;
374     LATB = sLATB;
375     LATA4 = 1;
376     break;
377 case 19:
378     sLATB = LATB;
379     sLATB &= 0xE2;
380     sLATB |= 0x11;
381     LATB = sLATB;
382     LATA4 = 1;
383     break;
384 case 20:
385     sLATB = LATB;
386     sLATB &= 0xE2;
387     sLATB |= 0x14;
388     LATB = sLATB;
389     LATA4 = 0;
390     break;
391 case 21:
392     sLATB = LATB;
393     sLATB &= 0xE2;
394     sLATB |= 0x15;
395     LATB = sLATB;
396     LATA4 = 0;
397     break;
398 case 22:
399     sLATB = LATB;
400     sLATB &= 0xE2;
401     sLATB |= 0x14;
402     LATB = sLATB;
403     LATA4 = 1;
404     break;
405 case 23:
406     sLATB = LATB;
407     sLATB &= 0xE2;
408     sLATB |= 0x15;

```

Page 6 OF 7

2013.08.10