```
; 8-CHANNEL PSEUDO RANDOM LED FLASHER
;
; John Baraclough
; 2010-05-30
;
; The flasher uses a 64 byte look-up table of pseudo random bits. Each byte of the table
; comprises four bits from a pseudo-random sequence each followed immediately by a 1.
; This ensures that the LEDs are on most of the time and only flicker off occasionally.
;
; The PIC16F88 processor uses the internal oscillator at 125kHz. This results in an
; instruction clock of 31.25kHz. Timer 0 runs with a prescalar of 32 and overflows at
; 3.8Hz. The timer 0 ISR does all the work. It determines the next bit for each of
; 8 output pins.
;
;
;           PROCESSOR   P16F88A
;           LIST        P=P16F88A

            ERRORLEVEL  -302            ; Turns off "Bank" warning messages

            #INCLUDE    "P16F88.INC"

            __CONFIG    _CONFIG1, _CP_OFF & _CCP1_RB0 & _DEBUG_OFF & _WRT_PROTECT_OFF & _CPD_OFF & _LVP_OFF & _BODEN_OFF &
            __CONFIG    _CONFIG2, _IESO_OFF & _FCMEN_OFF

YEAR            EQU     H'2010'
MONTH_DAY       EQU     H'0530'
VERSION         EQU     H'0010'


;SETUP VALUES FOR REGISTERS
TRISA_DATA      EQU     H'00'           ; Set RA<0:7> as outputs
TRISB_DATA      EQU     H'00'           ; Set RB<0:7> as outputs
OSCCON_DATA     EQU     H'10'           ; IRCF0 set => 125kHz internal clock =>31.25kHz instruction clock
OPTION_DATA     EQU     H'81'           ; RBPU & PS0 set => No pull-ups and Timer0 prescalar /4

            CBLOCK  0X20

            BIT_0_INDEX                 ; Byte pointer
            BIT_0_DATA                  ; Bit data

            BIT_1_INDEX                 ; Byte pointer
            BIT_1_DATA                  ; Bit data

            BIT_2_INDEX                 ; Byte pointer
            BIT_2_DATA                  ; Bit data

            BIT_3_INDEX                 ; Byte pointer
            BIT_3_DATA                  ; Bit data

            BIT_4_INDEX                 ; Byte pointer
            BIT_4_DATA                  ; Bit data

            BIT_5_INDEX                 ; Byte pointer
            BIT_5_DATA                  ; Bit data

            BIT_6_INDEX                 ; Byte pointer
            BIT_6_DATA                  ; Bit data

            BIT_7_INDEX                 ; Byte pointer
            BIT_7_DATA                  ; Bit data

            BIT_COUNT                   ; Bit count

            SAVE_INDEX                  ; Place to save W when entering look-up table

            PORTB_DATA                  ; Copy of data which will be written to PORT B

            ENDC

            ORG     0X00

            GOTO    MAIN

            DW      YEAR                ; Year
            DW      MONTH_DAY           ; Month/Day
            DW      VERSION             ; Version

            ORG     0X04

ISR
            BCF     INTCON, TMR0IF      ; Clear interrupt flag

            BSF     PORTA, 0            ; Set bit A0 as scope trigger

            CLRF    PORTB_DATA          ; Start with all LEDs off

            RLF     BIT_0_DATA, F       ; Get data for Bit 0
            BTFSS   STATUS, C           ; Test CARRY bit
            GOTO    ISR_BIT1            ; Not set so move on

            BSF     PORTB_DATA, 0       ; Switch LED 0 on

ISR_BIT1

            RLF     BIT_1_DATA, F       ; Get data for Bit 1
            BTFSS   STATUS, C           ; Test CARRY bit
            GOTO    ISR_BIT2            ; Not set so move on

            BSF     PORTB_DATA, 1       ; Switch LED 1 on

ISR_BIT2
```

```
103
104             RLF     BIT_2_DATA, F       ; Get data for Bit 2
105             BTFSS   STATUS, C           ; Test CARRY bit
106             GOTO    ISR_BIT3            ; Not set so move on
107
108             BSF     PORTB_DATA, 2       ; Switch LED 2 on
109
110 ISR_BIT3
111
112             RLF     BIT_3_DATA, F       ; Get data for Bit 3
113             BTFSS   STATUS, C           ; Test CARRY bit
114             GOTO    ISR_BIT4            ; Not set so move on
115
116             BSF     PORTB_DATA, 3       ; Switch LED 3 on
117
118 ISR_BIT4
119
120             RLF     BIT_4_DATA, F       ; Get data for Bit 4
121             BTFSS   STATUS, C           ; Test CARRY bit
122             GOTO    ISR_BIT5            ; Not set so move on
123
124             BSF     PORTB_DATA, 4       ; Switch LED 4 on
125
126 ISR_BIT5
127
128             RLF     BIT_5_DATA, F       ; Get data for Bit 5
129             BTFSS   STATUS, C           ; Test CARRY bit
130             GOTO    ISR_BIT6            ; Not set so move on
131
132             BSF     PORTB_DATA, 5       ; Switch LED 5 on
133
134 ISR_BIT6
135
136             RLF     BIT_6_DATA, F       ; Get data for Bit 6
137             BTFSS   STATUS, C           ; Test CARRY bit
138             GOTO    ISR_BIT7            ; Not set so move on
139
140             BSF     PORTB_DATA, 6       ; Switch LED 6 on
141
142 ISR_BIT7
143
144             RLF     BIT_7_DATA, F       ; Get data for Bit 7
145             BTFSC   STATUS, C           ; Test CARRY bit
146             BSF     PORTB_DATA, 7       ; Bit is set so switch LED 7 on
147
148             MOVF    PORTB_DATA, W       ; Get the LED data and ...
149             MOVWF   PORTB               ; ... write it to the port.
150
151             DECFSZ  BIT_COUNT, F        ; Decrement bit count and ...
152
153             GOTO    ISR_END             ; ... finish if not zero
154
155             MOVLW   8                   ; Reload bit counter.
156             MOVWF   BIT_COUNT
157
158             MOVLW   1                   ; Now increment all of the pointers and get new data
159             ADDWF   BIT_0_INDEX, W
160             ANDLW   D'63'
161             MOVWF   BIT_0_INDEX
162             CALL    BYTE_TABLE
163             MOVWF   BIT_0_DATA
164
165             MOVLW   1
166             ADDWF   BIT_1_INDEX, W
167             ANDLW   D'63'
168             MOVWF   BIT_1_INDEX
169             CALL    BYTE_TABLE
170             MOVWF   BIT_1_DATA
171
172             MOVLW   1
173             ADDWF   BIT_2_INDEX, W
174             ANDLW   D'63'
175             MOVWF   BIT_2_INDEX
176             CALL    BYTE_TABLE
177             MOVWF   BIT_2_DATA
178
179             MOVLW   1
180             ADDWF   BIT_3_INDEX, W
181             ANDLW   D'63'
182             MOVWF   BIT_3_INDEX
183             CALL    BYTE_TABLE
184             MOVWF   BIT_3_DATA
185
186             MOVLW   1
187             ADDWF   BIT_4_INDEX, W
188             ANDLW   D'63'
189             MOVWF   BIT_4_INDEX
190             CALL    BYTE_TABLE
191             MOVWF   BIT_4_DATA
192
193             MOVLW   1
194             ADDWF   BIT_5_INDEX, W
195             ANDLW   D'63'
196             MOVWF   BIT_5_INDEX
197             CALL    BYTE_TABLE
198             MOVWF   BIT_5_DATA
199
200             MOVLW   1
201             ADDWF   BIT_6_INDEX, W
202             ANDLW   D'63'
203             MOVWF   BIT_6_INDEX
204             CALL    BYTE_TABLE
```

```
205             MOVWF   BIT_6_DATA
206
207             MOVLW   1
208             ADDWF   BIT_7_INDEX, W
209             ANDLW   D'63'
210             MOVWF   BIT_7_INDEX
211             CALL    BYTE_TABLE
212             MOVWF   BIT_7_DATA
213
214 ISR_END
215
216             BCF     PORTA, 0            ; Clear bit A0 (scope trigger).
217
218             RETFIE                      ; No more to do
219
220 ;
221 ; Main code starts here
222 ;
223 MAIN
224
225             BSF     STATUS, RP0         ; Select Bank 1
226
227             MOVLW   OSCCON_DATA
228             MOVWF   OSCCON              ; Set internal oscillator to 125kHz
229
230             MOVLW   TRISA_DATA          ; Set up ports
231             MOVWF   TRISA
232
233             MOVLW   TRISB_DATA
234             MOVWF   TRISB
235
236             MOVLW   OPTION_DATA
237             MOVWF   OPTION_REG
238
239             BCF     STATUS, RP0         ; Select Bank 0
240
241             CLRF    PORTA               ; Initialize PORTA
242             MOVLW   H'FF'
243             MOVWF   PORTB               ; Initialize PORTB
244
245             MOVLW   D'0'
246             MOVWF   BIT_0_INDEX
247             CALL    BYTE_TABLE
248             MOVWF   BIT_0_DATA
249
250             MOVLW   D'8'
251             MOVWF   BIT_1_INDEX
252             CALL    BYTE_TABLE
253             MOVWF   BIT_1_DATA
254
255             MOVLW   D'16'
256             MOVWF   BIT_2_INDEX
257             CALL    BYTE_TABLE
258             MOVWF   BIT_2_DATA
259
260             MOVLW   D'24'
261             MOVWF   BIT_3_INDEX
262             CALL    BYTE_TABLE
263             MOVWF   BIT_3_DATA
264
265             MOVLW   D'32'
266             MOVWF   BIT_4_INDEX
267             CALL    BYTE_TABLE
268             MOVWF   BIT_4_DATA
269
270             MOVLW   D'40'
271             MOVWF   BIT_5_INDEX
272             CALL    BYTE_TABLE
273             MOVWF   BIT_5_DATA
274
275             MOVLW   D'48'
276             MOVWF   BIT_6_INDEX
277             CALL    BYTE_TABLE
278             MOVWF   BIT_6_DATA
279
280             MOVLW   D'56'
281             MOVWF   BIT_7_INDEX
282             CALL    BYTE_TABLE
283             MOVWF   BIT_7_DATA
284
285             MOVLW   D'8'                ; All bit changes are synchronised
286             MOVWF   BIT_COUNT
287
288             MOVLW   (1 << GIE) | (1 << TMR0IE)
289             MOVWF   INTCON              ; Enable Timer0 interrupts
290
291 RUN_WAIT
292             NOP
293             GOTO    RUN_WAIT            ; Nothing more to do except wait for the timer interrupt
294
295
296             ORG     0x300
297
298 BYTE_TABLE
299
300             MOVWF   SAVE_INDEX
301             MOVLW   HIGH (BIT_DATA)
302             MOVWF   PCLATH
303             MOVF    SAVE_INDEX, W
304             ADDLW   LOW (BIT_DATA)
305             MOVWF   PCL
306
```

```
307  BIT_DATA
308              DT      0x55, 0x57, 0x57, 0xf5, 0x75, 0xdf, 0xd5, 0x57
309              DT      0xd7, 0x5d, 0x7d, 0xfd, 0x75, 0x57, 0x77, 0xdf
310              DT      0x77, 0xd7, 0x7d, 0x57, 0xff, 0x7d, 0xff, 0x77
311              DT      0xf7, 0x57, 0x55, 0xf7, 0xd5, 0xff, 0x5f, 0xd7
312              DT      0xd5, 0xdf, 0x75, 0xd5, 0xdd, 0x77, 0x75, 0xfd
313              DT      0xfd, 0xf5, 0xff, 0x7f, 0xfd, 0xd7, 0xd7, 0xdd
314              DT      0xd5, 0xf5, 0x57, 0xf7, 0x77, 0x7f, 0xf5, 0xdd
315              DT      0x57, 0x5f, 0xff, 0xf5, 0x5d, 0xff, 0x57, 0x37
316
317
318              END
```