



### Controller Provides Display of Monitored Process Deviation

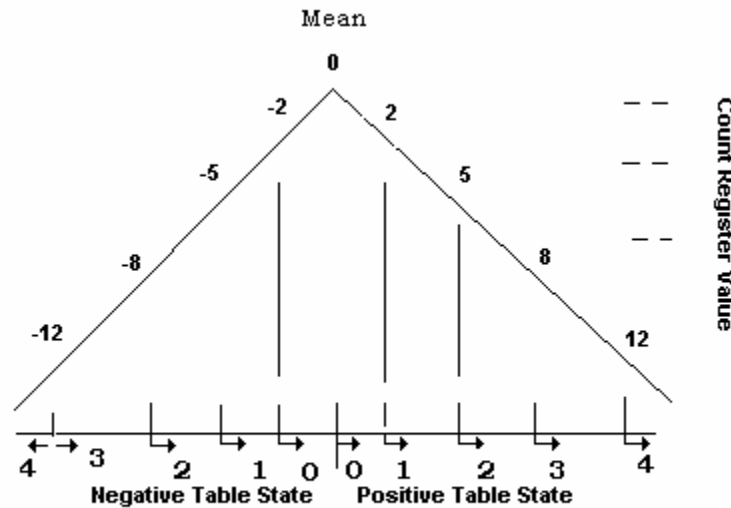
The gadget, shown in Figure 1.0, is implemented with a controller based process which continuously monitors two, discrete, momentary switch inputs and provides a simple indication of totalized variation from a desired mean.

Intended to monitor and evaluate manually entered observations indicating the number of 2 different cell types, it was pointed out to me it could be also used to count cards. In either case the process algorithm is identical.

Manually entered event counts are added, with 1 button, while subtracted with the 2nd button to a common, 8 bit, counter register. This counter register value is subtracted from defined mean to generate the error or variation about the mean.

Pursuing the card counter example, a high card entry is added to the register, while a low card entry is subtracted from the register. The mean is set to zero and would correspond to when the number of high card entries equals the number of low card entries.

The controller's processes saves an 'index' value corresponding to which tabled range the resulting error falls into. These predefined lookup tables set boundary values for the totalized counter based variation. In both counting applications the distribution is symmetrical and the boundary tables are described in the figure below.



**Figure 1 Error/Variation About Mean Mapping to Display State**

In the card counting example, Figure 2 is read as: when 2 to 4 high cards more the low cards have been tallied the index value is +1. When 7 to 11 low cards more the high cards have been tallied the index value is -3.

Using positive and negative measurement tables allow this accumulated sample error to be interpreted using nonsymmetrical distributions about the parameter's mean.

This error 'index' selects a display state sequence, which is continuously scheduled to the display. Using 2 colored LEDs to indicate direction and single and dual flashes to indicate the magnitude, 10 index states are mapped into 10 flashing combination patterns.

#### Positive table states

State 0,	single green LED flash
State 1,	single green LED flash, single red LED flash
State 2,	double green LED flash
State 3,	double green LED flash , single red LED flash
State 4,	double green LED flash, double red LED flash

#### Negative table states

State 0,	single red LED flash
State 1,	single red LED flash, single green LED flash
State 2,	double red LED flash

State 3,                   double red LED flash , single green LED flash  
State 4,                   double red LED flash, double green LED flash

The gadget's form factor, shown in the figure, was set up to be held in the palm of the hand and includes the 2-AAA batteries. The LED's are mounted at what would be the top of the pistol grip between the forefinger and thumb.

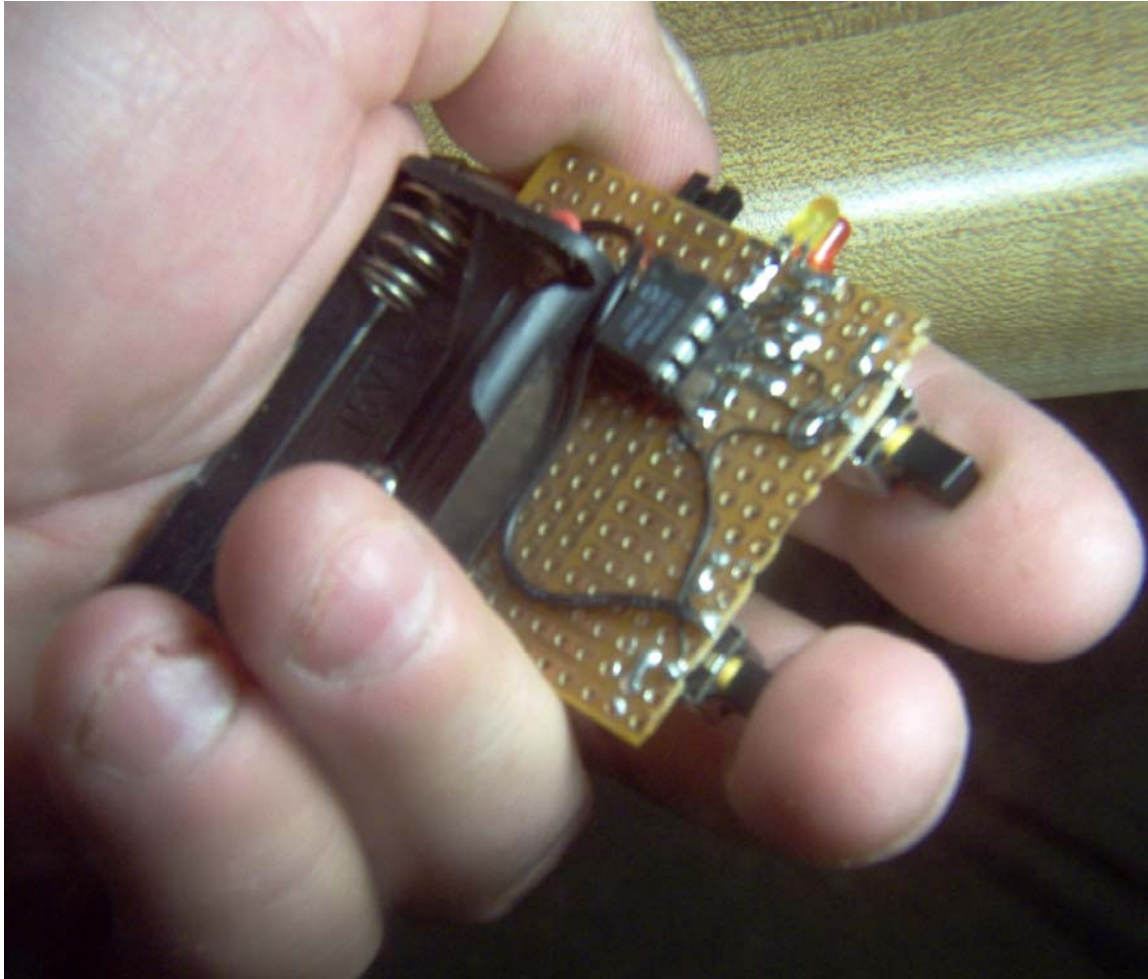


Figure 2

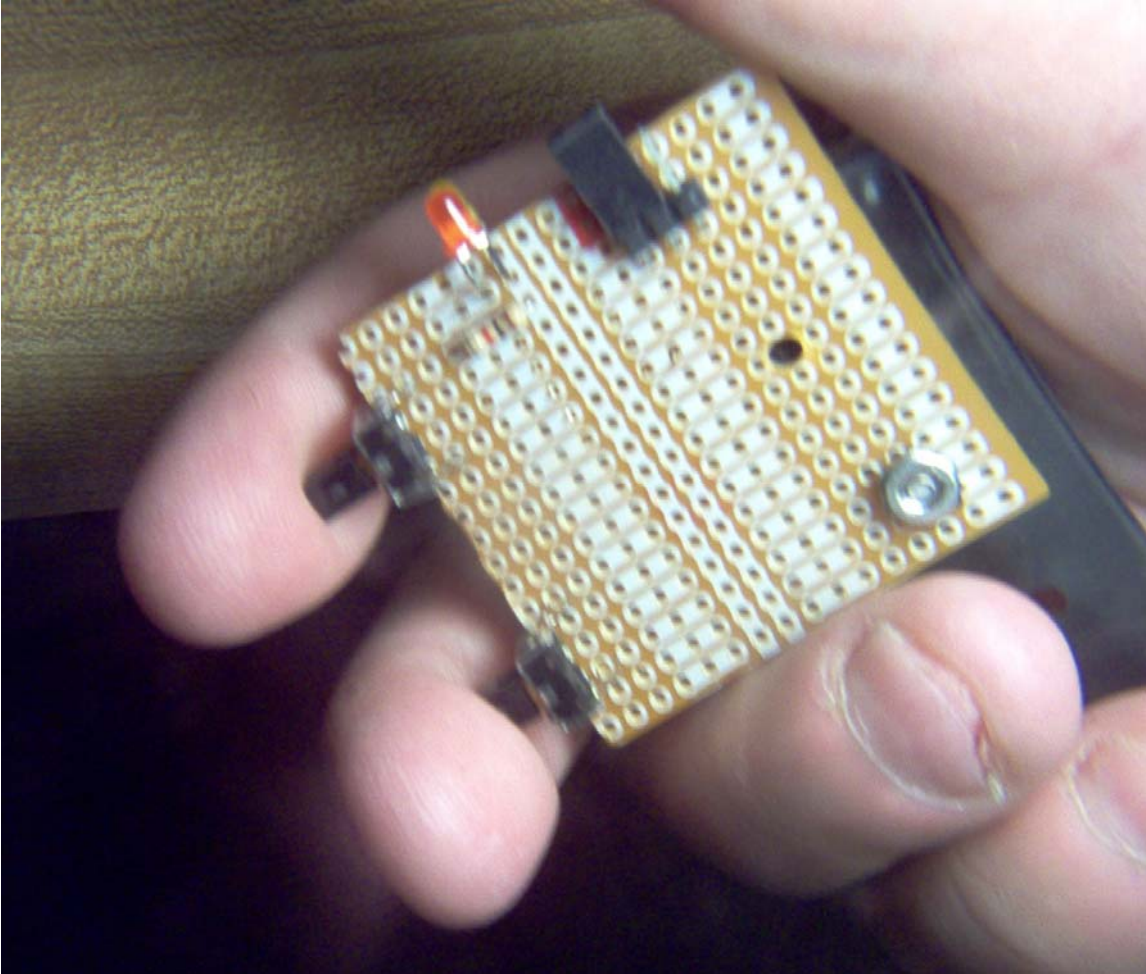
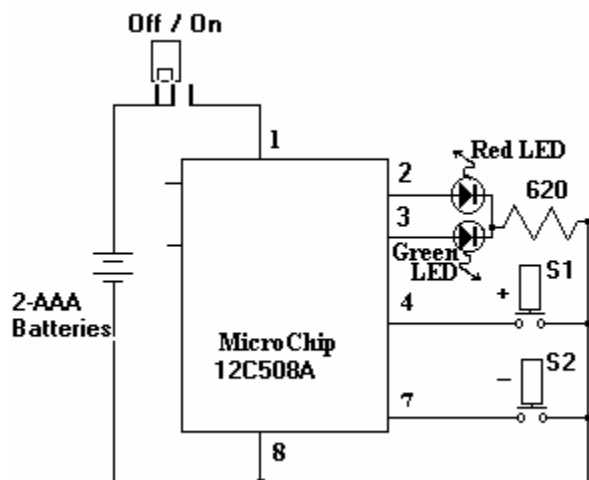


Figure 3

Shown in the schematic, exploiting several features of the Microchip 12C508A controller, this application uses only a few active parts. Coded in assembly language, it Figure 4



uses under 200 bytes of program memory space. Several compatible parts may be substituted including the 10F2x flash based, parts packaged in a SOT23.

The 10F220 with an A/D may also be used to adapt the gadget to a simple process monitor. Monitoring temperature, such as a solder bath, hooded air flow or a host of workspace related parameters the flash based parts, the tables and sequences are reprogrammable to suit the other applications.

## BOM

Battery	2 AAA	Energizer	1212
Battery holder	dual AAA	Memory Protection Devices	BCAAAL
Green	LED	Litton	LT4234
Red	LED	Litton	LT4224
620ohm	.25wresistor 5%	Panasonic	ERD-S2TJ620V
Momontary button right angle		OMRON	B3F-3152
Button Cap	black	OMRON	B32-1010
Controller	8pin	MicroChip	PIC12C508A
Header	3 pin,.1" centers, rt angle	Molex	22-288033
Header	jumper	3M	929955-06
Perf board	.1"centers pre-punched	Radio Shack	276-1588
8 pin	socket	Mil-Max	11-93-308-41-001
Hookupwire	30Gauge/solid/insulated	OK Industries	KSW30WR-0100

Assemble needs only a few required procedures. The perf board is cut to the 1.8"x1.7". The red LED, the resistor, the 3 pin right angle header are mounted on what will be the far side of the palm. The right angle momontary switches, also on this side are placed along the edge of the board, seperated apart by the relative distance between the center of the index and forefinger.

On the palm side the battery holder is securely mounted with small hardware so that the holder provides room for the stubs from the 3 pin header, it's leads are toward the top and the holder hangs over the edge about 3/4". The Yellow LED and 8 pin IC socket are then soldered in place on the palm side. All connections were done with 30 gauge solid insulated wire

The controller is programmed with the a compiled source code, included below. The .hex filed, included in the projects file list, can also be used directly to program the controller.

```

; Gadget.hex
; 508/509 version
; 2 keys payin,payout count
; 2 outputs in single and double flashing and bounce sequences/ rotate
DIRECTION indicates direction ;
; per 3/23/02 using tables below
; delta count    dsplbased on 50==0    mode    out display
; <=-14          <=-20          -5      rr gg
; -10-13         0--15          -4      rr g
; -6-9           20-5           -3      rr
; -3-5           35-25          -2      rg
; -1-2           45-40          -1      r
; 0-1            50-55          +0      g
; 2-4            60-70          +1      gr
; 5-7            75-85          +2      gg
; 8-11           90-105         +3      gg r
; >=12           >=110          +4      gg rr
;
; conversion : sleep to fsleep
; set option references to single call with wakeup on pin change enabled
;

GPIO equ 6      ; ***** return ot 5 for real processor *****
; GPIO, 0,3 are buttons
; GPIO, 4,5 are led outputs
PCL equ 2
SFR equ 4
STATUS equ 3
#define ZERO STATUS,2
#define CARRY STATUS,0
#define PUP STATUS,3
#define TUP STATUS,4
#define PIND STATUS,7

;
OPTIONR equ 0x81
dsplimage equ 0x8
temp equ 0xB
dsplpattern equ 0xC
;payin equ 0xD
keyimage equ 0xE
bits equ 0x10
#define flag bits,1
#define flag1 bits,2
#define direction bits,3
#define debounce bits,4
#define lastgreen bits,5
payinout1 equ 0x14
temp1 equ 0x15
displayimage equ 0x16
keydelta equ 0x17
absol equ 0x18
optimage equ 0x19
; ***** Beginning *****
goto start
;
pdspltable: ; x. x. fr. r. x. x. fg. g. : f =double
flash,r=red,g=green
addwf PCL

```

```

    retlw 0x33 ; double g and r
    retlw 0x13 ; double g/w r
    retlw 3    ; double g
    retlw 0x11 ; g/w r
    retlw 1    ; g
ndspltable: ; x. x. fr. r. x. x. fg. g. : f =double
flash,r=red,g=green
    addwf PCL
    retlw 0x33 ; double r and g
    retlw 0x31 ; double r/w g
    retlw 0x30 ; double r
    retlw 0x11 ; r/w g
    retlw 0x10 ; r
tablepos: ; define trip points to display mode
    addwf PCL ; 0th element(5)is number of defined display states
    retlw 5 ; 12 and greater
    retlw .12 ; 8.9.10.11
    retlw 8 ; 5.6.7
    retlw 5 ; 2.3.4
    retlw 2 ; value is start of index up 0.1
;
tableneg: ; 0th element(5)is number of defined display states
    addwf PCL
    retlw 5 ; -14 or less
    retlw .14 ; -10.11.12.13
    retlw .10 ; -6.7.8.9
    retlw 6 ; -3.4.5
    retlw 3 ; -1.2
;
start:
    btfss PUP ; test power
    goto starta
;
; initialize registers and controls
startpw:
    movlw 0
    movwf GPIO
    movlw 0xF ; 00 1111
    TRIS GPIO
    clrf payinoutl
    movlw 1
    movwf displayimage
    movlw 0x19
    movwf dsplimage
    clrf dsplpattern
    clrf bits
    clrf SFR
    movlw 8D ; disable on pin change for 32*.018sec and enable
pullups
    OPTION
    movlw 9
    movwf keyimage
    sleep
;
; register initalization
starta:
    movfw dsplimage ; clear any outputs
    andlw 0xF
    movwf GPIO
    movlw 0xF ; (re)establish TRIS register
    TRIS GPIO
    btfss TUP ; test WDT

```

```

    goto    startb
;          must be due to pin change
    goto    checkkeya ; would be an error
fsleep:
    movfw  optimage
    OPTION
    sleep
;          ***** begin processes *****
startb:    ; wakeup due to wdt timeout
    movlw  0x8F      ; reload wdt with long delay & disable pin delta
    OPTION
;          ***** begin display related processes
    movfw  dsplimage ; if display was in on state then
    andlw  0x30
    btfsc  ZERO
    goto   alloff
    movfw  keyimage
    andlw  0xF
    movwf  dsplimage ; set appropriate gap
    btfss  direction
    goto   startf    ; positive direction
;          negative direction
    movlw  0xB      ; 0000 1011 ** is set flash gap
    movwf  optimage
    btfsc  flag     ; if in flash set short gap
    goto   starte
;
    btfsc  lastgreen
    goto   startc
    btfss  displayimage,0 ; is red scheduled
    goto   startc
    movlw  0xA      ; 0000 1 010 ** is flash pulse
    movwf  optimage
    goto   starte
;
startf:
    movlw  0xA      ; 0000 1010 ** is flash pulse
    movwf  optimage
    btfsc  flag     ; if in flash set short gap
    goto   starte
;
    btfss  lastgreen
    goto   startc
    btfss  displayimage,4 ; is red scheduled
    goto   startc
    goto   starte
startc:
    movlw  0xD      ; 0000 1 101 ** is normal gap
    movwf  optimage
starte:
    movfw  dsplimage
    movwf  GPIO     ; assert output
;
ckd:
    btfss  debounce
    goto   fsleep   ;
    goto   checkkeya
;
alloff:    ; else WAS in OFF time
    btfsc  flag

```

```

goto    flagon
btfss  lastgreen
goto    dogreen
bcf    lastgreen ;
movfw  displayimage
andlw  0x30
btfsc  ZERO
goto    setgap    ; normal between display update gap
movfw  keyimage
iorlw  0x20
movwf  dsplimage    ; set red
btfsc  displayimage,5 ; test flash flag
bsf    flag
goto    dsplc
setgap:    ; 0000 1 101  ** is normal gap
movlw  0xD
movwf  optimage
movfw  dsplimage
movwf  GPIO    ; assert output
goto   ckd    ; goto  fsleep
dogreen:
bsf    lastgreen ; set last green
movfw  displayimage
andlw  0x3
btfsc  ZERO
goto    setgap    ; normal between display update gap
movfw  keyimage
iorlw  0x10
movwf  dsplimage    ; set green
btfsc  displayimage,1 ; test flash flag
bsf    flag
dsplc:
movfw  dsplimage
movwf  GPIO    ; assert output
movlw  0x9    ; 0000 1 001 fixed on time
movwf  optimage
goto   ckd    ; goto  fsleep
flagon:    ; else (flag set) then set flash on duration
bcf    flag
movlw  0x10
btfss  lastgreen
movlw  0x20
iorwf  keyimage,w
movwf  dsplimage
goto   dsplc
; ***** begin key related processes
checkkeya:    ; service for key test
movfw  GPIO
movwf  temp
xorwf  keyimage,w ; qualify a change
andlw  0x9
btfsc  ZERO    ; if new key image == old image
goto   checka  ; no changes
ckb:
movwf  keydelta ; store copy of changes
movfw  temp
movwf  keyimage ; save image
bsf    debounce
goto   chkkeydd
checka:

```

```

    btfss  debounce
    goto   nonactive
;
    movfw  temp          hook for reset
    andlw  9
    btfsc  ZERO
    goto   stop
    btfss  keydelta,3 ; debounce was set
    goto   bbb
    btfss  temp,3      ; if key is active
    goto   dopayout   ; to add
nonactive:
    bcf    debounce
    movfw  keyimage
    andlw  9
    movwf  dsplimage
    movlw  0xC
    movwf  optimage
    goto   endckkey
bbb:
    btfss  temp,0      ; assume othe key
    goto   dopayin
    goto   nonactive
dopayout:
; *****
;           perform simple bit add of +1 ; payout key depressed
    bcf    debounce
    movlw  0x7F
    xorwf  payinoutl,w
    btfsc  ZERO
    goto   checkkeyc  ; overflow
    incf  payinoutl
    goto   checkkeyc
; *****
dopayin:
;           ; perform simple 8 bit subtract of -1
    bcf    debounce
    movlw  0x80
    xorwf  payinoutl,w
    btfsc  ZERO
    goto   checkkeyc  ; overflow
    decf  payinoutl
; *****
checkkeyc: ; process
; done explicitly subtract (payout - paidin) to payinout
    movfw  payinoutl
    movwf  temp
    bcf    direction
    btfss  payinoutl,7 ; if <0 then
    goto   checkkeyd
; payout is negative
    comf  temp ; create positive value of negiive register
    incf  temp
    bsf  direction
checkkeyd:
    movfw  temp
    movwf  absol
; temp is (now) abs value of delta(-128 to 127) with direction flag
aserted correctly
; parse based on positive or negative count in temp
    btfss  direction

```

```

    goto    positive
;          count was minus
    clrw
    call    tableneg
    movwf  temp1
    movwf  dsplpattern
chkkeyda:
    decfsz dsplpattern
    goto   chkkeydb
    goto   chkkeydf
chkkeydb:
    movfw  dsplpattern
    call   tableneg
    subwf  temp,w
    btfsc  CARRY
    goto   chkkeyda
    goto   chkkeydf
;
positive:
    clrw
    call   tablepos
    movwf  temp1
    movwf  dsplpattern
chkkeydg:
    decfsz dsplpattern
    goto   chkkeyde
    goto   chkkeydf
chkkeyde:
    movfw  dsplpattern
    call   tablepos
    subwf  temp,w
    btfsc  CARRY
    goto   chkkeydg
chkkeydf:      ; dsplpattern has display pattern index
    movfw  dsplpattern
    btfsc  direction
    goto   ckc
    call   pdspltable
    movwf  displayimage
chkkeydd:
    movlw  0x8
    movwf  optimage      ; enable wakeup
endckkey:
    goto   fsleep
ckc:
    call   ndspltable
    movwf  displayimage
    goto   chkkeydd
stop:
    goto   startpw
end

```