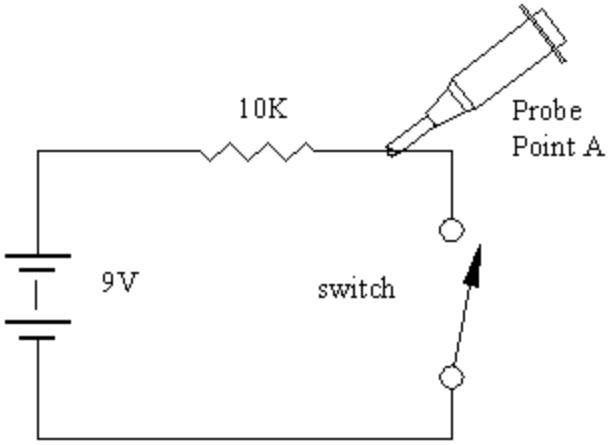
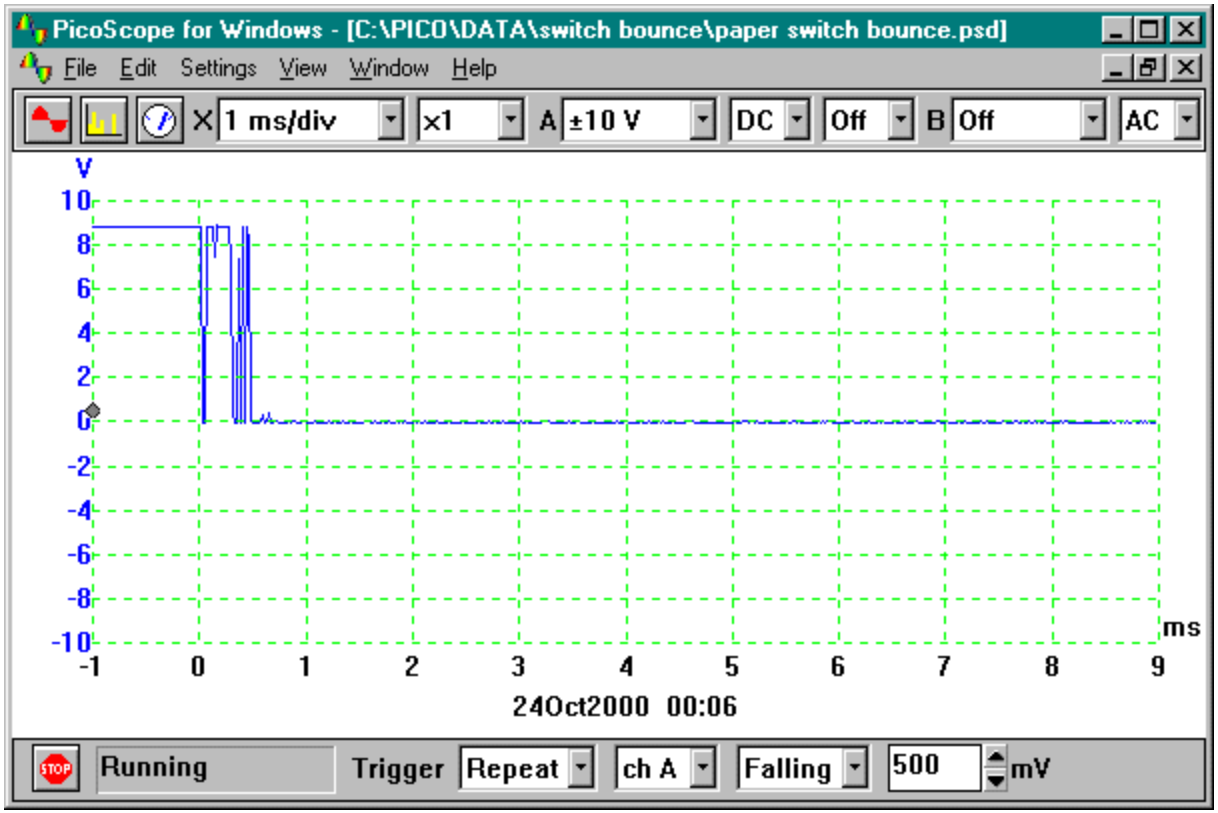
Switch De-bounce Application

Ranga Vemuri

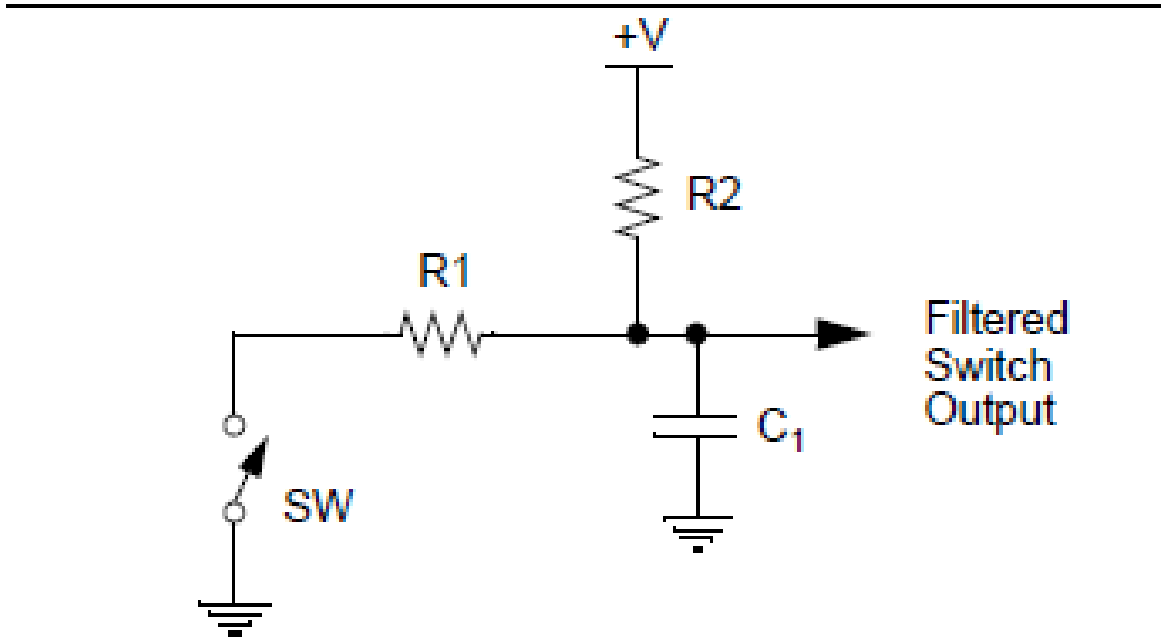
4038C Embedded Systems

Switch Bounce

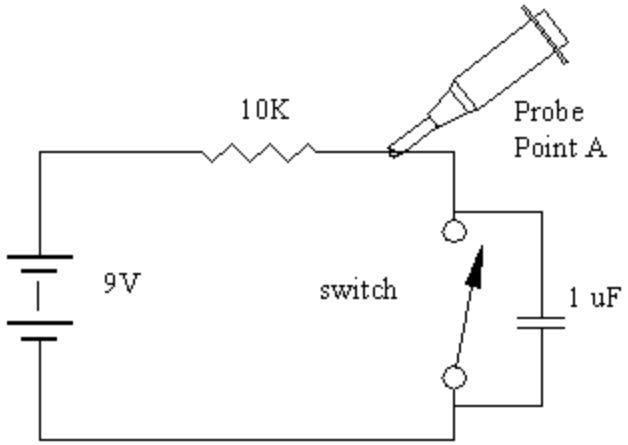




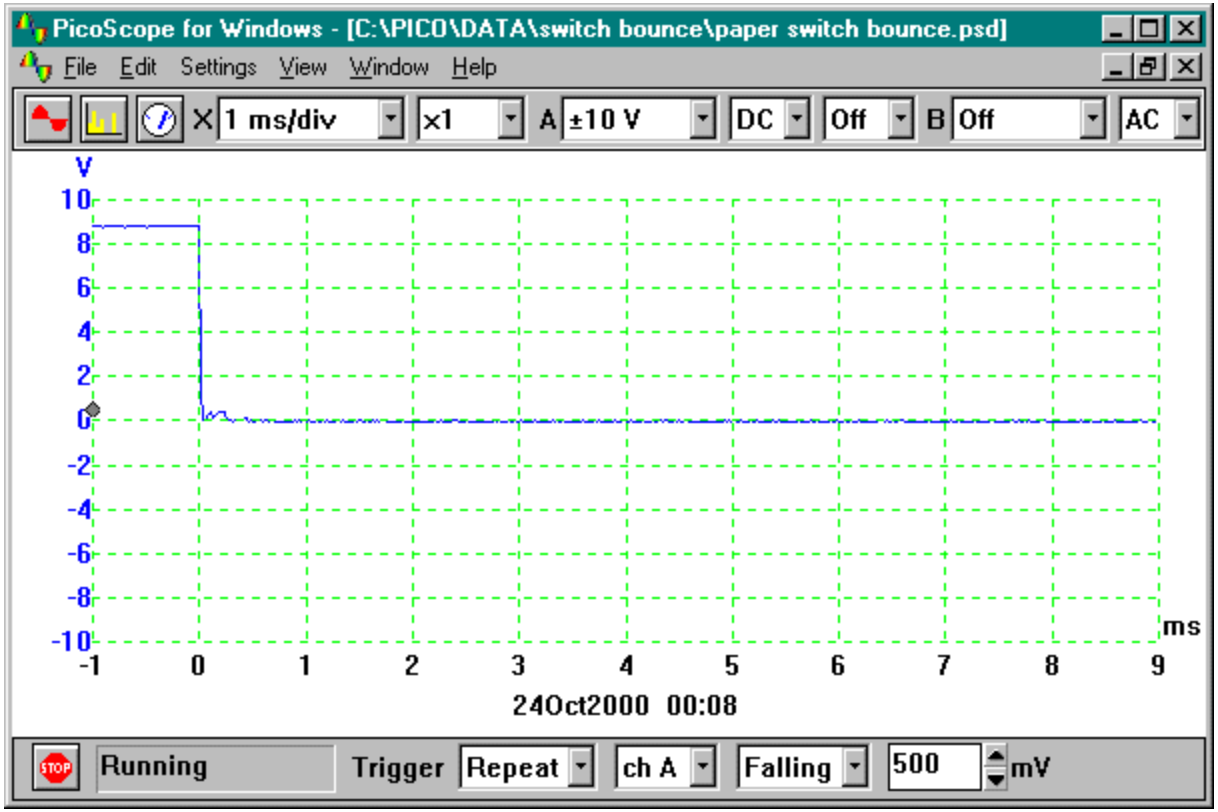
Hardware Debounce through Filtering



V=9v, r2=10kohm, c1=1microfarad, r1= small switch resistance



When switch is off, cap charges to 9v. (2.2RC = 2.2*10 ms time)
When switch is on, cap quickly discharges through small switch resistance to 0v.
Through debounce, cap doesn't get enough time to charge back.



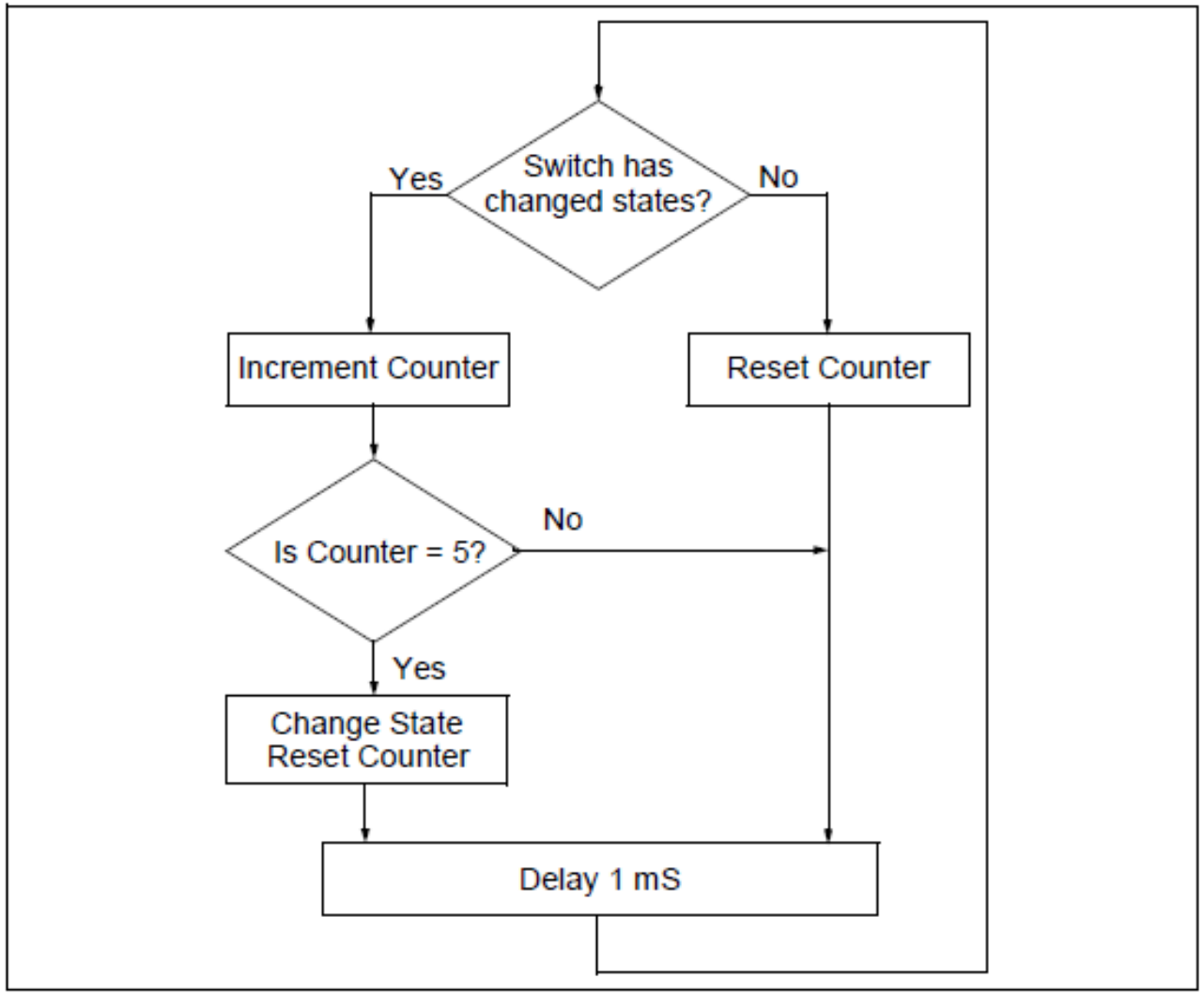
Lesson - 6

- Increment a counter for each press of a push-button switch and display the counter value.

- RB0 = GND when button pushed
  = Vdd when not pushed

Software Debounce

- New state should be stable for at least 5ms for it to register .
- Sample at the rate of 1ms.
- That is, 5 successive samples should yield the same state.



```
LOOP
    Read the button input

    IF button state has changed from previous stable state
      THEN  increment counter
      ELSE    reset counter
    END IF

    IF counter = 5
     THEN register new state and reset  counter
              IF the new state is a down press
                 THEN update the display
              END IF
    END IF

   delay 1 ms;

FOREVER
```

```
 cblock     0x20

Delay
Display                 ; display counter 0-255

LastStableState     ; Flag keep track of switch state
                      ; (open-1; closed-0)
                      ; only the 0th bit used.

Counter                 ; to count 5

endc
```

; Set up

Make PORD all output (for LEDs)

Make RB0 a digital input (push button switch
which connects GND upon push)

Turn all LEDs off initially.

; Assume that the switch is up initially

```
movlw     1
movwf     LastStableState
```

```
LOOP

    IF button state has changed
      THEN  increment counter
      ELSE    reset counter
    END IF

    IF counter = 5
     THEN register new state and reset  counter
            IF the new state is a down press
               THEN update the display
            END IF
    END IF

   delay 1 ms;

FOREVER
```

```
IF current button state is 1
   THEN IF the new button state is 0
              THEN  increment counter       ; 1 to 0
              ELSE    reset counter         ; 1 to 1
         END IF
    ELSE IF the new button state is 1
              THEN  increment counter       ; 0 to 1
              ELSE    reset counter         ; 0 to 0
         END IF
END IF
```

```
MainLoop:
    btfss    LastStableState,0        ; Flag is the 0th bit
    goto     LookingForUp             ; if current state is 0 then look for up

LookingForDown:                       ; If the last stable state was 1
    clrw
    btfss    PORTB,0                  ; test for switch to go down
    incf     Counter,w                ; if it's low, increment the counter (put in w)
    movwf    Counter                  ; store either the 0 or incremented value
    goto     EndDebounce

LookingForUp:                         ; If the last stable state was 0
    clrw
    btfsc    PORTB,0                  ; test for switch to go high
    incf     Counter,w                ;  if it's high, increment the counter (put in w)
    movwf    Counter                  ; store either the 0 or incremented value

EndDebounce:
```

```
LOOP

    IF button state has changed
      THEN  increment counter
      ELSE    reset counter
    END IF

    IF counter = 5
     THEN register new state and reset  counter
            IF the new state is a down press
               THEN update the display
            END IF
    END IF

   delay 1 ms;

FOREVER
```

```
EndDebounce:
    movf     Counter,w          ; have we seen 5 in a row?
    xorlw    5                  ; Z=1 if w = 5, ie. xor(w,101) = 0
    btfss    STATUS,Z           ; if not, goto delay by 1ms
    goto     Delay1mS

    comf     LastStableState,f  ; after 5 straight, reverse the state
    clrf     Counter
    btfss    LastStableState,0  ; Was it a key-down press?
    goto     Delay1mS           ; no: take no action

    incf     Display,f          ; if it's the down direction,
    movf     Display,w          ; take action on the switch
    movwf    PORTD              ; (increment counter and put on display)
```

```
Delay1mS:
    movlw     .71              ; delay ~1000uS
    movwf     Delay
    decfsz    Delay,f          ; ~71*3=213 cycles
    goto      $-1
    decfsz    Delay,f          ;  ~256*3 = 768 cycles
    goto      $-1

    goto      MainLoop
```