The Original Problem to Solve was...

...modify the code so that the S0 push button increments the count as before but the count is displayed as two BCD coded numbers on the 8 LEDs. The four lower and upper LEDs will form each digit. i.e., the LSB digit is displayed on LEDs L1-L4, MSB digit on LED.... The count will "roll over" to 0 after 99.
Coding Styles, The Good, The Bad, The Ugly

int main()
{
    DDRB = 0xFF; // set port B to all outputs
    DDRD = 0x00; // by default, but being explicit here
    while(1){
        if(debounce_switch()) {
            PORTB++; // if the lower digit is 10, reset it and roll over to next digit.
            if ((PORTB & 0x0F) == 0x0A) {
                PORTB += 0x10; // increase the upper digit.
                PORTB &= 0xF0;
            }
            // if upper digit is 10 reset it.
            if ((PORTB & 0xF0) == 0xA0){PORTB &= 0x0F;}
        }
        _delay_ms(2); // keep in loop to debounce 24ms
    }
}

The Good: 365 bytes, 18 lines, clean, simple
Coding Styles, The Good, The Bad, The Ugly

int main() {
    DDRB = 0xFF;  //set port B to all outputs
    while(1){    //do forever
        if(debounce_switch()) {
            //Recreate the number in PORTB as a decimal number and add 1
            //The 10’s place is created by dividing the recreated
            //number by 10 and shifting by 4
            //The 1’s place is created by modding the recreated
            //number by 10
            //These two values are OR’d together
            //To ensure PORTB doesn’t go past 99, the entire number
            //is modded by 160
            //Which is 10100000, which represents 100 in BCD
            PORTB = (((((PORTB >> 4) * 10 + (PORTB & 0x0F) + 1) / 10 << 4) |
                      ((PORTB >> 4) * 10 + (PORTB & 0x0F) + 1) % 10) % 160;
            _delay_ms(2);  //keep in loop to debounce 24ms
        }
    }  //while
}  //main

The Bad: 538 bytes, 18 lines, complicated, magic numbers
#define F_CPU 16000000UL
#include <avr/io.h>
#include <util/delay.h>

int main(){
    uint16_t state = 0;
    DDRB = 0xFF;
    while(1) {
        if ((state = (state << 1) | (PIND & 1) | 0xE000) == 0xF000)
            PORTB = ((EICRA = (EICRA + 1)%100) % 10) | ((EICRA / 10) << 4);
        _delay_ms(5);
    }
}

The Ugly: 416 bytes, 9 lines, clever, comments?, magic register