IO Lab

Example 1:

Write Embedded C code using ATmega328P µC to control a led.

Requirements:

1. Configure the µC control with internal 16MHz Clock.
2. The LED is connected to pin 6 in PORTD.
3. Connect the LED using Negative Logic configuration. Flash the led every 1 second.

Steps Main:

1. Include port and delay library <avr/io.h>, <util/delay.h>
2. Configure pin 0 in PORTD as output pin
3. Initiate LED as OFF
4. In infinite loop: Make LED ON then wait 1s then make LED OFF then wait 1s. using (_delay_ms(1000))

Example 2:

Write Embedded C code using ATmega328p µC to control two LEDs using two push buttons.

Requirements:

1. Configure the µC control with internal 16MHz Clock.
2. The switch 1 & 2 are connected to pin 0 & 1 in PORTB.
3. Connect both switches using Pull Down configuration.
4. The LEDs 1 & 2 is connected to pin 0 & 1 in PORTC.
5. Connect both LEDs using Positive Logic configuration.
6. If switch1 is pressed just turn on the first LED1 only and if switch2 is pressed just turn on LED2 only.
7. In case both switches are pressed both LEDs are on.
8. In case no switches are pressed both LEDs are off.

Steps Main:

1. Configure pin 0 and 1 of PORTB to be input pins.
2. configure pin 0,1 of PORTC to be output pins.
3. initialize LEDs: LED1 (pin 0 in PORTC) is off at the beginning and LED2 pin 1 in PORTC) is off at the beginning.
4. In infinite loop:
   - ✓ Check if the first switch is pressed. If yes, Make LED1 ON. If No, Make LED1 OFF.
   - ✓ Check if the second switch is pressed. If yes, Make LED2 ON. If No, Make LED2 OFF.

Example 3:

Write Embedded C code using ATmega328p µC to control a LED using a push button.

Requirements:

1. Configure the µC control with internal 16MHz Clock.
2. The switch is connected to pin 0 in PORTB.
3. Connect the switch using Internal Pull Up configuration.
4. The LED is connected to pin 0 in PORTC.
5. Connect the LED using Negative Logic configuration.
6. If the switch is pressed just toggle the LED.
7. Consider switch debouching problem.

Steps Main:

1. Configure pin 0 of PORTB to be input pin.
2. Activate the internal pull up resistor of PB0.
3. Configure pin 0 of PORTC to be output pin.
4. Make LED is off at the beginning.
5. Check if the push button is pressed or not. If yes, wait 30ms then check if the button is still pressed due to switch de-bouncing, If yes, so the Button is confirm to be pushed so toggle the LED.
6. Make sure that the LED toggle once for every button pressed so the LED will not toggle if we still pressed the button

Example 4:

Write Embedded C code using ATmega328p µC to control a 7-segment using a push button.

Requirements:

1. Configure the µC control with internal 16MHz Clock.
2. The push button is connected to pin 4 in PORTD.
3. Connect the button using Pull Down configuration.
4. The 7-segment is connected to first 4-pins of PORTC.
5. If the button is pressed just increase the number appeared in the 7 segments display, and if we reach the maximum number (9) overflow occurs.

Steps Main:

1. Configure pin 4 of PORTD to be input pin.
2. Configure all pins of PORTC as output pins.
3. Initialize a counter to 0 and display it on the 7-segment.
4. Check if the push button is pressed or not. If pressed, increase the counter and display it in 7 segments until overflow happens.
5. In case of overflow happens, clear the counter.
6. Make sure that the 7-segments increase once for every button pressed so the 7-segments will not toggle if we still pressed the button

Example 5:

Write Embedded C code using ATmega328p µC to control a 7-segment using two push buttons.

Requirements:

1. Configure the µC control with internal 16MHz Clock.
2. The two switches are connected to pin 0 & 1 in PORTA.
3. Connect both switches using Pull Down configuration.
4. The 7-segment is connected to first 4-pins of PORTD.
5. If the switch1 is pressed just increase the number appeared in the 7 segments display, and if the number reach the maximum number (9) do nothing.
6. If the switch2 is pressed just decrease the number appeared in the 7 segments display, and if the number reach the minimum number (0) do nothing.