Microcontroller-Implemented LED Chaser

Objectives

1. To comprehend the versatility of input and output pins on the PIC 16F628A microcontroller.
2. To configure microcontroller pins for driving external devices.
3. To execute the deployment of a chaser effect on LEDs employing a PIC microcontroller.

Apparatus

Software: MikroC PRO, PROTEUS, MPLAB IPE (V3).

Hardware: PIC 16F628A, PicKit 3, 12 V (~3 A) power adaptor, LM 7805 voltage regulator, breadboard, 7 × LED's, 1 kΩ and 7 × 470 Ω resistors.

Theory

The LED chaser, also known as the Knight Rider circuit, presents an engaging lighting effect widely employed in automotive lighting and decorative displays. This application, comprising a row of LEDs arranged to mimic a scanning motion, underscores the critical role of timing considerations in embedded systems design. Precise control over time intervals is indispensable for achieving desired functionalities, a task facilitated by the delay function "**Delay_ms()**". The delay function regulates timing and synchronization within the Knight Rider circuit and enhances visual appeal by introducing subtle delays between LED shifts, typically spanning a few tens of milliseconds. Furthermore, adjustments of the parameter within the "Delay_ms()" function allow for fine-tuning the LED illumination duration, thereby affording more control over the circuit's behavior.

In microcontroller programming, integers serve as foundational elements for representing numerical values and executing arithmetic operations. In this experiment, the integer variable, 'i', assumes the role of a loop counter, enabling sequential control of LED lights. The illuminated LED can traverse the sequence by iteratively incrementing or decrementing 'i' within a loop structure, highlighting the adaptability of integers in managing iterative processes. This fundamental utilization of integers underscores their pivotal role in directing program flow and managing data within embedded systems.

Disabling the comparator module (CMCON = 7) is crucial to prevent interference with the microcontroller's intended functionality. This preventive measure is imperative, as the comparator module has the potential to introduce extraneous noise or impede the microcontroller's performance. Following the comparator module's deactivation, all pins of PORTB are configured as outputs, priming them for driving LEDs and ensuring seamless operation of the Knight Rider circuit.

Procedure

1. Write a code in the C language to control the LED chaser.
2. Create a schematic representation of the LED ON/OFF circuit shown in Figure 1 within the PROTEUS simulation environment.
3. Validate the code's functionality by integrating the corresponding "HEX" file into the PROTEUS simulator.
4. Employ the Micro PikKit 3 to transfer the compiled program code onto the PIC 16F628A microcontroller.
5. Set up the circuit on a breadboard to physically exemplify its operational dynamics.

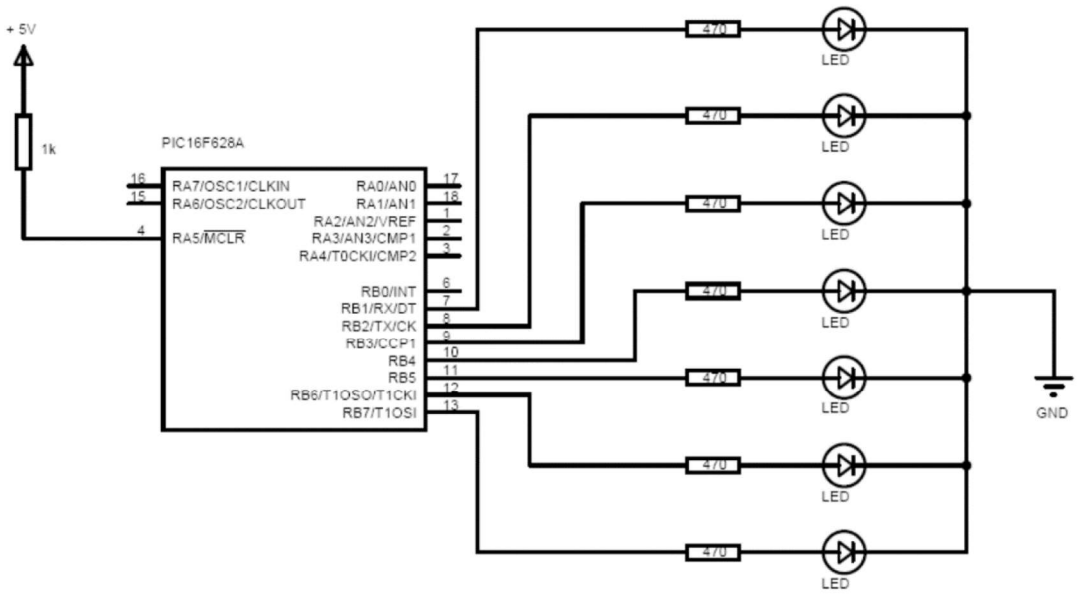

Figure 1. LED Chasing circuit diagram for PIC16F268A.

Template Code

```
void knightrider(void) {
   int i;
  // Step 1: Set all pins of PORTB as outputs
   TRISB = ****; // Hint: Set all pins of PORTB as outputs
  // Step 2: Initialize PORTB with the first LED lit
   PORTB = ****; // Hint: Initialize PORTB to light the first LED (RB0)
  // Step 3: Define the left shift loop
   for (i = ****; i <= ****; i++) {
   PORTB = (PORTB **** 1); // Hint: Shift the lit LED to the left
  // Step 4: Delay for smoother animation
      Delay_ms(****); // Hint: Delay for smoother animation
   }
  // Step 5: Define the right shift loop
   for (i = ****; i >= ****; i--) {
      PORTB = (PORTB ****); // Hint: Shift the lit LED to the right
  // Step 6: Delay for smoother animation
      Delay_ms(****);
   }
}
void main() {
   CMCON = ****; // Hint: Disable comparators
   TRISA = ****; // Hint: Set all PORTA pins as digital I/O
   while (****) { Hint: Enter a condition for the infinite loop
   knightrider(); // Call the knightrider function
   }
}
```

Execute the **** sections in the code to finalize the program.