Question A.1

```
/*// --Question A.1--


void Initialize()
{
    DDRB |= (1<<DDB1);
    DDRB |= (1<<DDB2);
    DDRB |= (1<<DDB3);
    DDRB |= (1<<DDB4);
    PORTB |= (1<<PORTB1);
    PORTB |= (1<<PORTB2);
    PORTB |= (1<<PORTB3);
    PORTB |= (1<<PORTB4);
}

int main(void)
{
    Initialize();
    while(1);
}

*/
```

Question A.2

```
/*// --Question A.2--

void Initialize()
{
    DDRD &= ~(1<<DDD7);
    DDRB |= (1<<DDB1);
}


int main(void)
{
    Initialize();
    UART_init(BAUD_PRESCALER);
    UART_putstring("Test Serial Print");
    while(1)
    {
        if (PIND & (1<<PIND7))
        {
            PORTB |= (1<<PORTB1);
            UART_putstring("The Light is On (button pressed)\n");
            _delay_ms(20);
        }
        else
        {
            PORTB &= ~(1<<PORTB1);
            UART_putstring("The Light is Off (button not-pressed)\n");
            _delay_ms(20);
        }
    }
}
*/
```

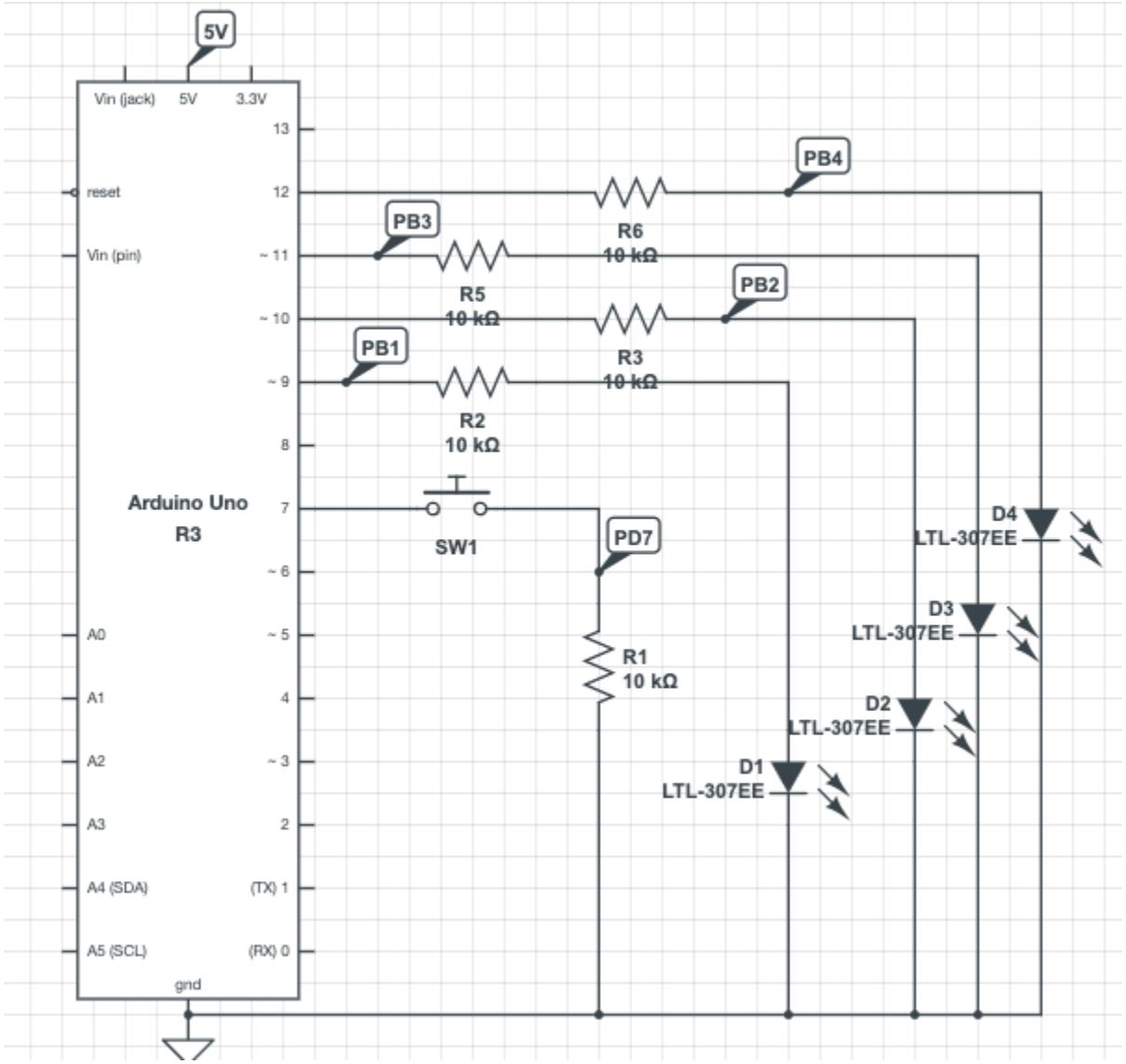
Question A.3

```
/*// --Question A.3--

void Initialize()
{
    DDRD &= ~(1<<DDD7);
    DDRB |= (1<<DDB1);
    DDRB |= (1<<DDB2);
    DDRB |= (1<<DDB3);
    DDRB |= (1<<DDB4);
    PORTB |= (1<<PORTB1);
}

int main(void)
{
    Initialize();
    UART_init(BAUD_PRESCALER);
    UART_putstring("Test Serial Print");
    while(1){
        if ((PIND & (1<<PIND7)) && (PINB & (1<<PORTB1))){
            PORTB &= ~(1<<PORTB1);
            PORTB |= (1<<PORTB2);
            _delay_ms(200);
            UART_putstring("Light1\n");
        }
        else if ((PIND & (1<<PIND7)) && (PINB & (1<<PORTB2))){
            PORTB &= ~(1<<PORTB2);
            PORTB |= (1<<PORTB3);
            _delay_ms(200);
            UART_putstring("Light2\n");
        }
        else if ((PIND & (1<<PIND7)) && (PINB & (1<<PORTB3))){
            PORTB &= ~(1<<PORTB3);
            PORTB |= (1<<PORTB4);
            _delay_ms(200);
            UART_putstring("Light3\n");
        }
        else if ((PIND & (1<<PIND7)) && (PINB & (1<<PORTB4))){
            PORTB &= ~(1<<PORTB4);
            PORTB |= (1<<PORTB1);
            _delay_ms(200);
            UART_putstring("Light4\n");
        }
    }
}
*/
```

Question A.4



Question A.5: The advantage of using interrupts in this situation, as is consistent with most applications in general, the interrupt doesn't require constant attention with the CPU, which is more efficient and allows the CPU to process other tasks. Unfortunately, interrupts require added complexity of prescaling and handling overflow, as well as requiring priority ordering in the case of simultaneous tasks .

Question A.6:

$16M\ ticks/s\ *\ 1\ /\ 256\ ticks\ *\ 1s/1000ms\ *\ 30ms\ =\ 1875ticks$

$16M\ ticks/s\ *\ 1\ /\ 256\ ticks\ *\ 1s/1000ms\ *\ 200ms\ =\ 12500ticks$

$16M\ ticks/s\ *\ 1\ /\ 256\ ticks\ *\ 1s/1000ms\ *\ 400ms\ =\ 25000ticks$

Question A.7:
A prescaler simply uses integer division to lower the frequency of overflow. By dividing each tick by an integer, it takes a relatively longer amount of time to reach 255, allowing operation with low frequency devices. As the in class example with the LED explained, using a prescaler allows us to utilize interrupts at frequencies that suit many different needs, like being visible to the human eye.