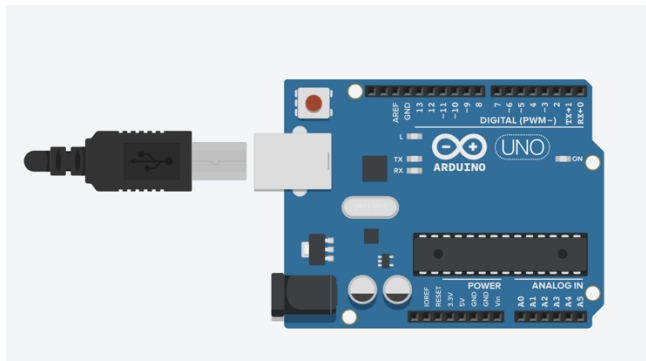
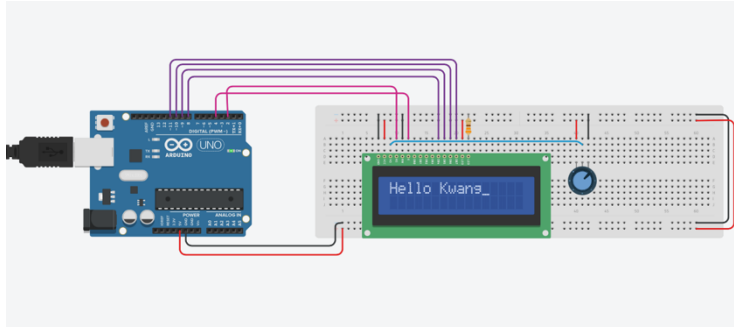


Lab4 Problem1: AVR USART Programming



```
1 // C++ code
2 //4.1
3
4 void USART_Init(unsigned int ubrr){
5 //Set baud rate
6     UBRR0H = (unsigned char)(ubrr >> 8);
7     UBRR0L = (unsigned char)ubrr;
8
9 //Enable receiver & transmitter
10    UCSRB = (1 << RXEN0) | (1 << TXEN0);
11
12 //Set frame format (8 data, 2 stop bits)
13    UCSRC = (1 << UCSZ00)|(3 << UCSZ01);
14 }
15
16 void USART_Transmit(unsigned char data){
17 //Wait for empty transmit buffer
18     while(!( UCSRA & (1 << UDRE0) ));
19
20 //Put data into buffer, sends data (auto)
21     UDR0 = data;
22 }
23
24 unsigned char USART_Receive(){
25 //Wait for data to be received
26     while( !( UCSRA & (1 << RXC0) ) );
27
28 //Get and return received data from buffer
29     return UDR0;
30 }
31
32 void printString(char* str){
33     while(*str != '\0' && *str != '.'){
34         USART_Transmit(*str);
35         str++;
36     }
37 }
38
39
40 void setup()
41 {
42     //Set baud rate
43     USART_Init(207);
44     char buffer[30];
45     int i;
46     while(1){
47         unsigned char receivedChar;
48         receivedChar = USART_Receive();
49         buffer[i++] = receivedChar;
50
51         if(receivedChar == '.'){
52             printString("Hello ");
53             printString(buffer);
54             printString("\n");
55             memset(buffer,0,sizeof(buffer));
56             i=0;
57         }
58     }
59 }
60
61 void loop()
62 {
63
64 }
```

Lab4 Problem2: AVR USART Programming



```
1 // C++ code
2 // 4.2
3
4 void USART_Init(unsigned int ubrr){
5 //Set baud rate
6 UBRRH = (unsigned char)(ubrr >> 8);
7 UBRRL = (unsigned char)ubrr;
8
9 //Enable receiver & transmitter
10 UCSRB = (1 << RXEN0) | (1 << TXEN0);
11
12 //Set frame format (8 data, 2 stop bits)
13 UCSRC = (1 << UCSZ00)|(3 << UCSZ01);
14 }
15
16 void USART_Transmit(unsigned char data){
17 //Wait for empty transmit buffer
18 while(!(UCSRBA & (1 << UDRE0) ));
19
20 //Put data into buffer, sends data (auto)
21 UDR0 = data;
22 }
23
24 unsigned char USART_Receive(){
25 //Wait for data to be received
26 while( !(UCSRBA & (1 << RXC0) ) );
27
28 //Get and return received data from buffer
29 return UDR0;
30 }
31
32 void printString(char* str){
33 while(*str != '\0' && *str != '.'){
34 USART_Transmit(*str);
35 str++;
36 }
37 }
38 //LCD FUNCTION
39
40 void initLCD(){
41 DDRB |= 0x0F; //0000 1111
42 PORTB &= 0xF0; //clear the last 4 bits to be 0
43 DDRD |= (1 << DD02) | (1 << DD04);
44 PORTD &= ~(1 << PORTD2) & ~(1 << PORTD4);
45
46 sendLCDCommand(0x33);
47 sendLCDCommand(0x32);
48 sendLCDCommand(0x28);
49 sendLCDCommand(0x0E);
50 // clear
51 sendLCDCommand(0x01);
52 // back to start
53 sendLCDCommand(0x80);
54 }
55
56 void sendLCDCommand(uint8_t command){
57 //Pull RS Down (Leg D2)
58 PORTD &= ~(1 << PORTD2);
59
60 //Put high nibble(4 bit) of the command
61 PORTB &= 0xF0;
62 PORTB |= command >> 4;
63 commitData();
64
65 //Send low nibble(4 bit) of the command
66 PORTB &= 0xF0;
67 PORTB |= (command & 0x0F);
68
69 commitData();
70 }
71
72 void commitData(){
73 PORTD |= (1 << PORTD4);
74 _delay_us(10);
75 PORTD &= ~(1 << PORTD4);
76 _delay_us(10);
77 }
78
79 void lcdDisplayString(char* str){
80 while(*str != '\0' and *str != '.'){
81 { sendLCDData(*str);
82 str++;
83 }
84 }
85 }
86
87 void sendLCDData(uint8_t command){
88 //Pull RS HIGH (Leg D2)
89 PORTD |= (1 << PORTD2);
90
91 //Put high nibble(4 bit) of the command
92 PORTB &= 0xF0;
93 PORTB |= command >> 4;
94 commitData();
95
96 //Send low nibble(4 bit) of the command
97 PORTB &= 0xF0;
98 PORTB |= (command & 0x0F);
99
100 commitData();
101 }
102
103 void setup()
104 {
105 //Set baud rate
106 USART_Init(207);
107 char buffer[30];
108 int i;
109 while(1){
110
111 unsigned char receivedChar;
112 receivedChar = USART_Receive();
113 buffer[i++] = receivedChar;
114
115 if(receivedChar == '.'){
116 initLCD();
117 sendLCDCommand(0x01);
118 sendLCDCommand(0x80);
119
120 printString("Hello ");
121 lcdDisplayString("Hello ");
122
123 printString(buffer);
124 lcdDisplayString(buffer);
125
126 printString("\n");
127 memset(buffer,0,sizeof(buffer));
128 i=0;
129 }
130 }
131 }
132
133 void loop()
134 {
135
136 }
```