František Konečný – ID: 211313 – BPC-DE2

Link to my repository in GitHub

https://github.com/Konecny343/Digital-electronics-2

1) Preparation tasks

Table with LCD signals

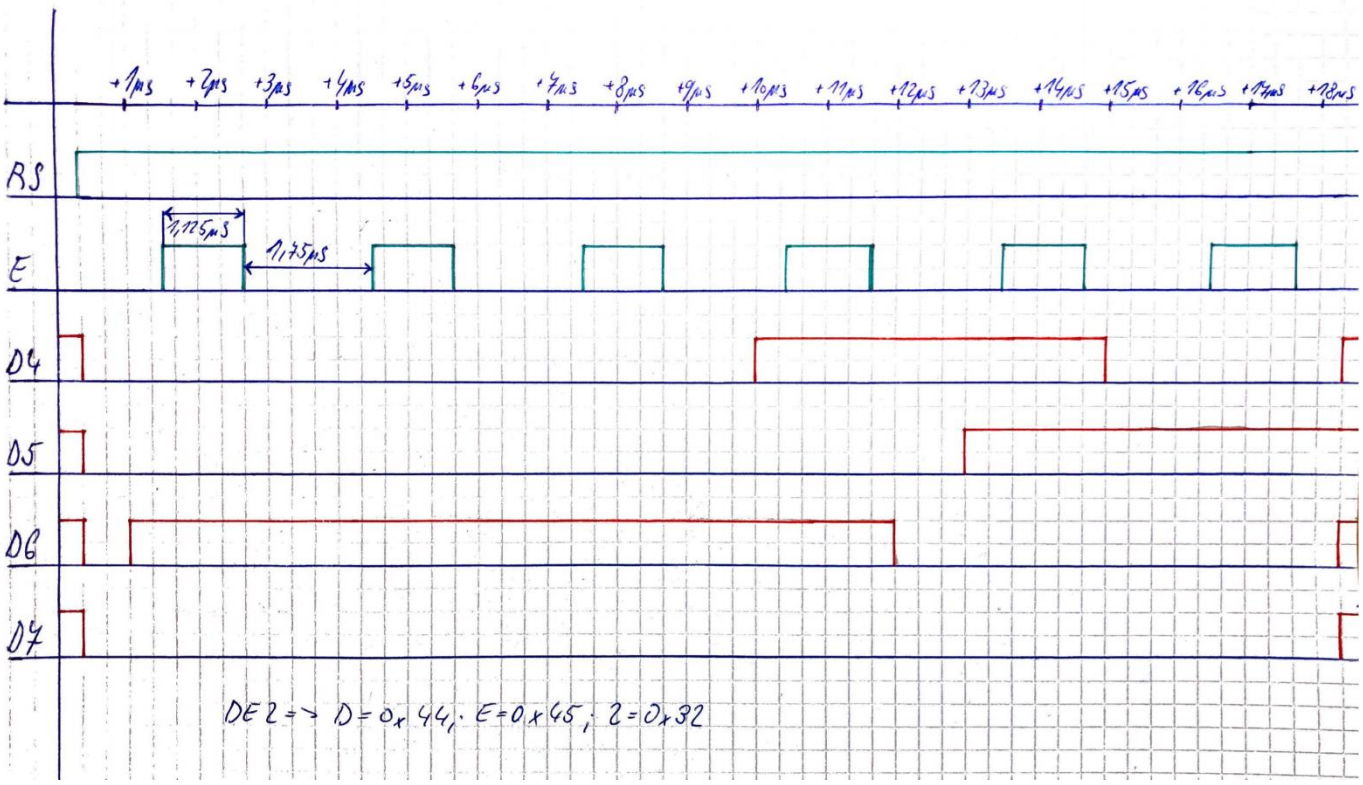
LCD signal(s)	AVR pin(s)	Description
RS	PB0	Register selection signal. Selection between Instruction register (RS=0) and Data register (RS=1)
R/W	GND	Read and write data
E	PB1	Enable data signals
D[3:0]		Data signals
D[7:4]	PD7 to PD4	Data signals

ASCII values

Dec	Hx	Char	Dec	Hx	Char	Dec	Hx	Char
48	30	0	65	41	A	97	61	a
49	31	1	66	42	B	98	62	b
50	32	2	67	43	C	99	63	c
51	33	3	68	44	D	100	64	d
52	34	4	69	45	E	101	65	e
53	35	5	70	46	F	102	66	f
54	36	6	71	47	G	103	67	g
55	37	7	72	48	H	104	68	h
56	38	8	73	49	I	105	69	i
57	39	9	74	4A	J	106	6A	j
ASCII American Standard Code for Information Interchange			75	4B	K	107	6B	k
			76	4C	L	108	6C	l
			77	4D	M	109	6D	m
			78	4E	N	110	6E	n
			79	4F	O	111	6F	o
			80	50	P	112	70	p
			81	51	Q	113	71	q
			82	52	R	114	72	r
			83	53	S	115	73	s
			84	54	T	116	74	t
			85	55	U	117	75	u
			86	56	V	118	76	v
			87	57	W	119	77	w
			88	58	X	120	78	x
			89	59	Y	121	79	y
			90	5A	Z	122	7A	z

2) HD44780 communication

- transmitting data DE2



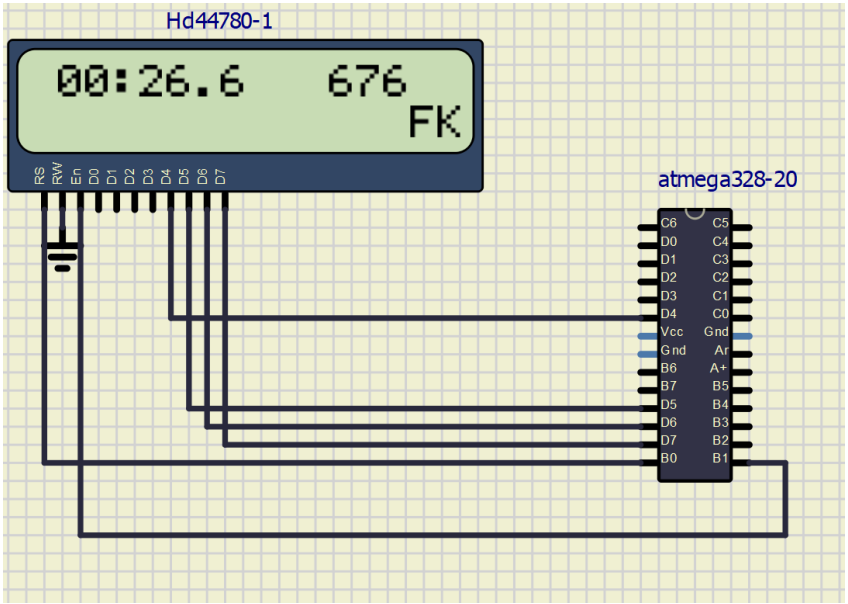
3) Stopwatch

Listing of `Timer2_OVF_vect` from file `main.c`

```
ISR(TIMER2_OVF_vect)
{
	static uint8_t number_of_overflows = 0;
	static uint8_t tens = 0;                        // Tenths of a second
	static uint8_t secs = 0;                        // Seconds
	static uint8_t minut = 0;                       // Minutes
	static uint16_t secsSquare = 0;                 // secs * secs
	char lcd_string[2] = "  ";                      // String for converting numbers by itoa()

	number_of_overflows++;
	if (number_of_overflows >= 6)
	{
		// Do this every 6 x 16 ms = 100 ms
		number_of_overflows = 0;

		tens++;
		if (tens > 9)
		{
			tens = 0;
			secs++;

			secsSquare = secs * secs;
			itoa(secsSquare, lcd_string, 10);
			lcd_gotoxy(11, 0);
			// Display secsSquare
			lcd_puts(lcd_string);
		}

		itoa(tens, lcd_string, 10);
		lcd_gotoxy(7, 0);
		// Display tenths of a second
		lcd_puts(lcd_string);

		if (secs > 59)
		{
			minut++;
			secs = 0;
			lcd_gotoxy(4, 0);
			lcd_putc('0');
			secsSquare = 0;
			// Deleting values on positions secsSquare
			lcd_gotoxy(11, 0);
			lcd_putc(' ');
			lcd_gotoxy(12, 0);
			lcd_putc(' ');
			lcd_gotoxy(13, 0);
			lcd_putc(' ');
			lcd_gotoxy(14, 0);
			lcd_putc(' ');
		}
```

```
                itoa(secs, lcd_string, 10);
                if (secs > 9)
                {
                        lcd_gotoxy(4, 0);
                }
                else
                {
                        lcd_gotoxy(5, 0);
                }
                // Display seconds
                lcd_puts(lcd_string);

                if (minut > 59)
                {
                        minut = 0;
                        lcd_putc('0');
                        lcd_gotoxy(2, 0);
                }
                itoa(minut, lcd_string, 10);

                if (minut > 9)
                {
                        lcd_gotoxy(1, 0);
                }
                else
                {
                        lcd_gotoxy(2, 0);
                }
                // Display minutes
                lcd_puts(lcd_string);
        }
}
```

Screenshot of SimulIDE circuit when ¨Power Circuit¨ is applied



4) Progress bar

Listing of `Timer0_OVF_vect` from file `main.c`

```
ISR(TIMER0_OVF_vect)
{
	static uint8_t symbol = 0;
	static uint8_t position = 0;

	lcd_gotoxy(1 + position, 1);
	lcd_putc(symbol);

	symbol++;
	if (symbol > 5)
	{
		position++;
		symbol = 0;
		if (position == 9)
		{
			position = 0;
			lcd_gotoxy(9,1);
			lcd_putc(0xff);
			lcd_gotoxy(8,1);
			lcd_putc(0xff);
			lcd_gotoxy(7,1);
			lcd_putc(0xff);
			lcd_gotoxy(6,1);
			lcd_putc(0xff);
			lcd_gotoxy(5,1);
			lcd_putc(0xff);
			lcd_gotoxy(4,1);
			lcd_putc(0xff);
			lcd_gotoxy(3,1);
			lcd_putc(0xff);
			lcd_gotoxy(2,1);
			lcd_putc(0xff);
			lcd_gotoxy(1,1);
			lcd_putc(0xff);
		}
	}
}
```

Screenshot of SimulIDE circuit when ¨Power Circuit¨ is applied