# Lab 6: Gregor Karetka

Link to your `Digital-electronics-2` GitHub repository:

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

# Lab 6: Display devices, LCD display

### Learning objectives

After completing this lab you will be able to:

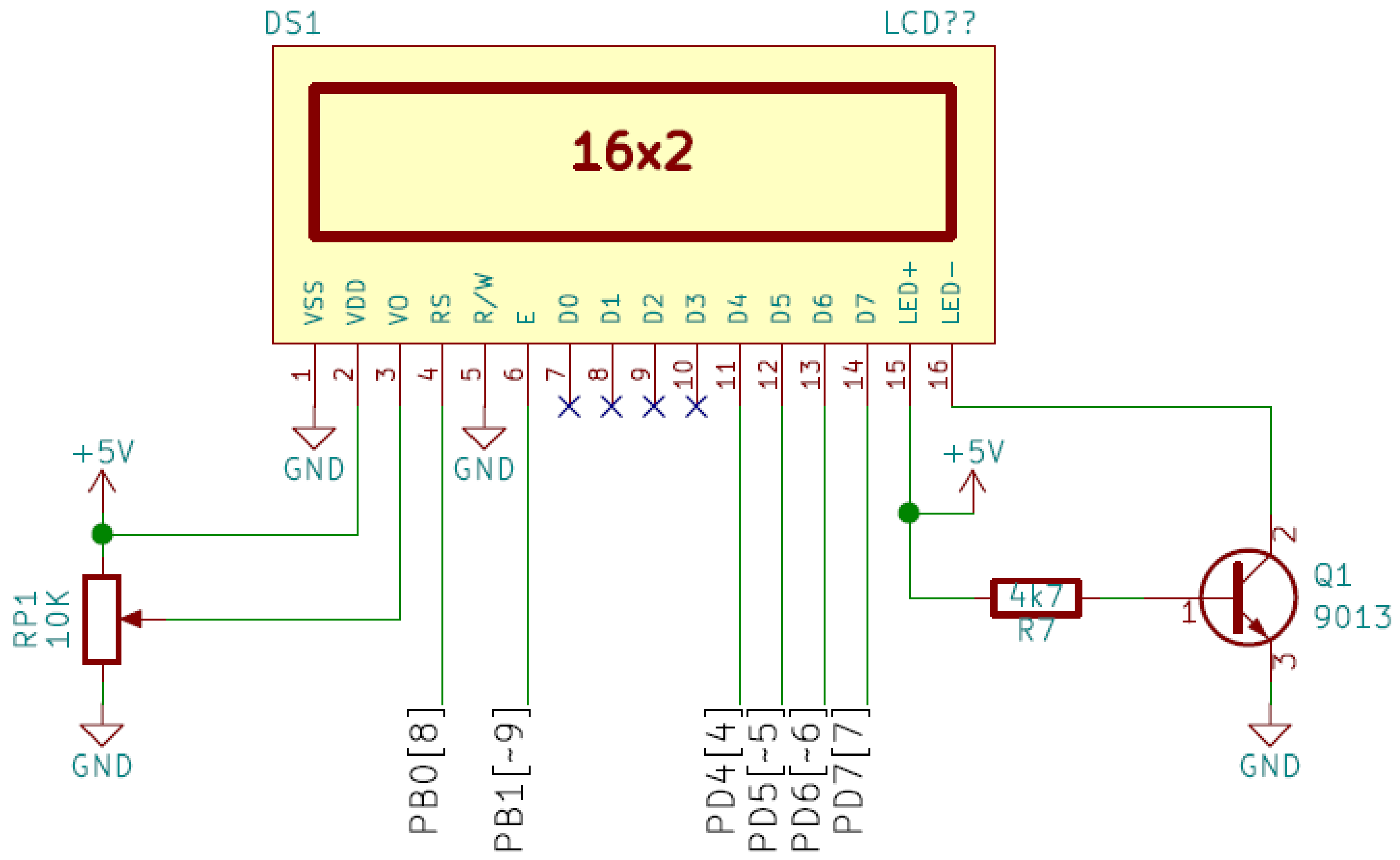
- Use text-based LCD
- Understand the digital communication between MCU and HD44780
- Use library functions for LCD
- Generate custom characters on LCD

The purpose of the laboratory exercise is to understand the serial control of Hitachi HD44780-based LCD character display and how to define custom characters. Another goal is to learn how to read documentation for library functions and use them in your own project.

## Preparation tasks (done before the lab at home)

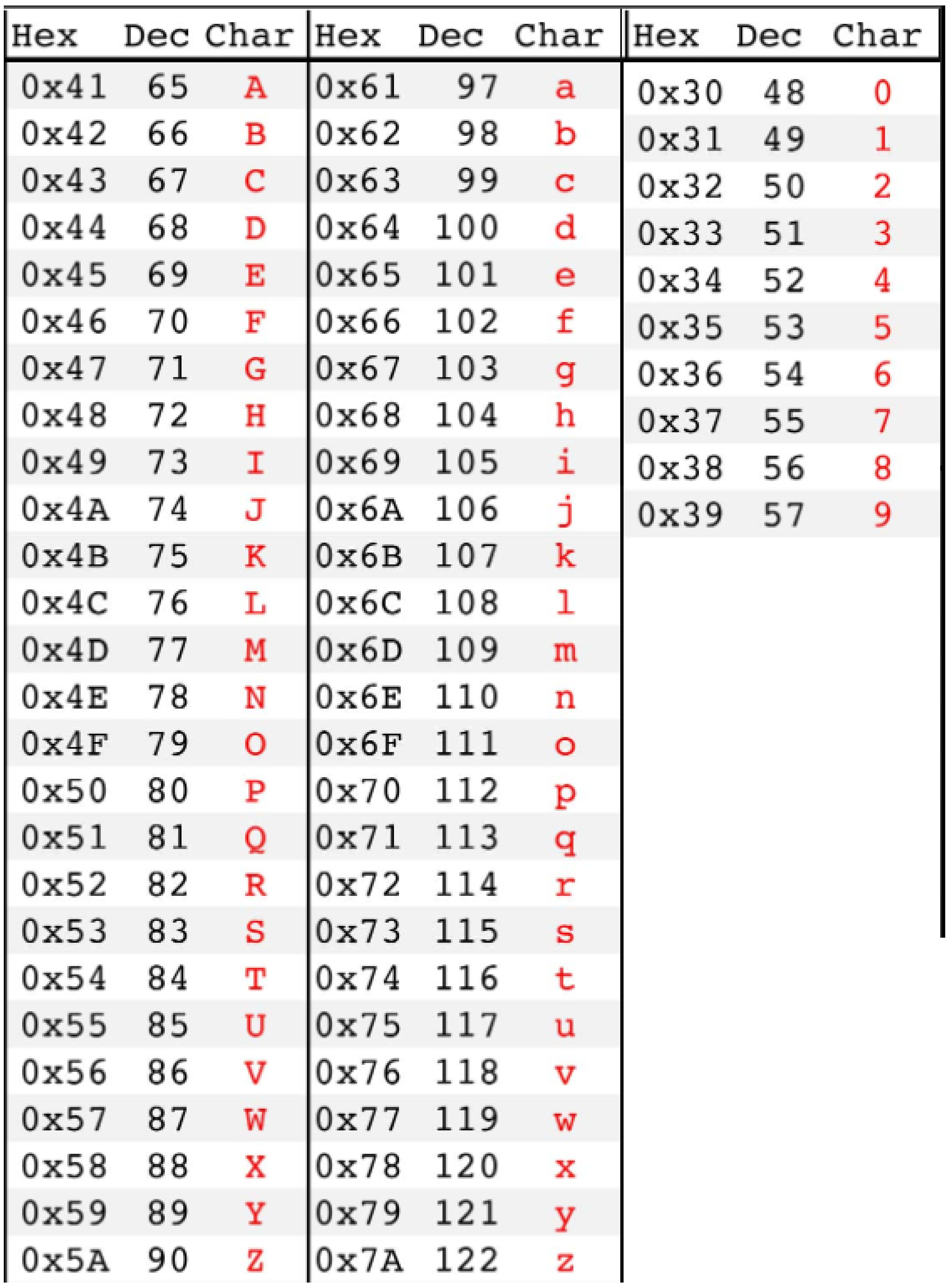
Use schematic of the LCD keypad shield and find out the connection of LCD display. What data and control signals are used? What is the meaning of these signals?

# LCD



| 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/write (active low) selector |
| E | PB1 | Enable: Starts data read/write |
| D[3:0] | N/C (we use 4-bit mode) | Four low order bidirectional tristate data bus pins. Used for data transfer and receive between the MPU and the HD44780U. These pins are not used during 4-bit operation |
| D[7:4] | PD4, PD5, PD6, PD7 | Four high order bidirectional tristate data bus pins. Used for data transfer and receive between the MPU and the HD44780U. DB7 can be used as a busy flag. |

What is the ASCII table? What are the codes/values for uppercase letters `A` to `Z`, lowercase letters `a` to `z`, and numbers `0` to `9` in this table?



| Hex | Dec | Char | Hex | Dec | Char | Hex | Dec | Char |
|---|---|---|---|---|---|---|---|---|
| 0x41 | 65 | A | 0x61 | 97 | a | 0x30 | 48 | 0 |
| 0x42 | 66 | B | 0x62 | 98 | b | 0x31 | 49 | 1 |
| 0x43 | 67 | C | 0x63 | 99 | c | 0x32 | 50 | 2 |
| 0x44 | 68 | D | 0x64 | 100 | d | 0x33 | 51 | 3 |
| 0x45 | 69 | E | 0x65 | 101 | e | 0x34 | 52 | 4 |
| 0x46 | 70 | F | 0x66 | 102 | f | 0x35 | 53 | 5 |
| 0x47 | 71 | G | 0x67 | 103 | g | 0x36 | 54 | 6 |
| 0x48 | 72 | H | 0x68 | 104 | h | 0x37 | 55 | 7 |
| 0x49 | 73 | I | 0x69 | 105 | i | 0x38 | 56 | 8 |
| 0x4A | 74 | J | 0x6A | 106 | j | 0x39 | 57 | 9 |
| 0x4B | 75 | K | 0x6B | 107 | k | | | |
| 0x4C | 76 | L | 0x6C | 108 | l | | | |
| 0x4D | 77 | M | 0x6D | 109 | m | | | |
| 0x4E | 78 | N | 0x6E | 110 | n | | | |
| 0x4F | 79 | O | 0x6F | 111 | o | | | |
| 0x50 | 80 | P | 0x70 | 112 | p | | | |
| 0x51 | 81 | Q | 0x71 | 113 | q | | | |
| 0x52 | 82 | R | 0x72 | 114 | r | | | |
| 0x53 | 83 | S | 0x73 | 115 | s | | | |
| 0x54 | 84 | T | 0x74 | 116 | t | | | |
| 0x55 | 85 | U | 0x75 | 117 | u | | | |
| 0x56 | 86 | V | 0x76 | 118 | v | | | |
| 0x57 | 87 | W | 0x77 | 119 | w | | | |
| 0x58 | 88 | X | 0x78 | 120 | x | | | |
| 0x59 | 89 | Y | 0x79 | 121 | y | | | |
| 0x5A | 90 | Z | 0x7A | 122 | z | | | |

## Part 3: Library for HD44780 based LCDs

In the lab, we are using LCD library for HD44780 based LCDs developed by Peter Fleury. Use the online manual of LCD library and add the input parameters and description of the following functions.

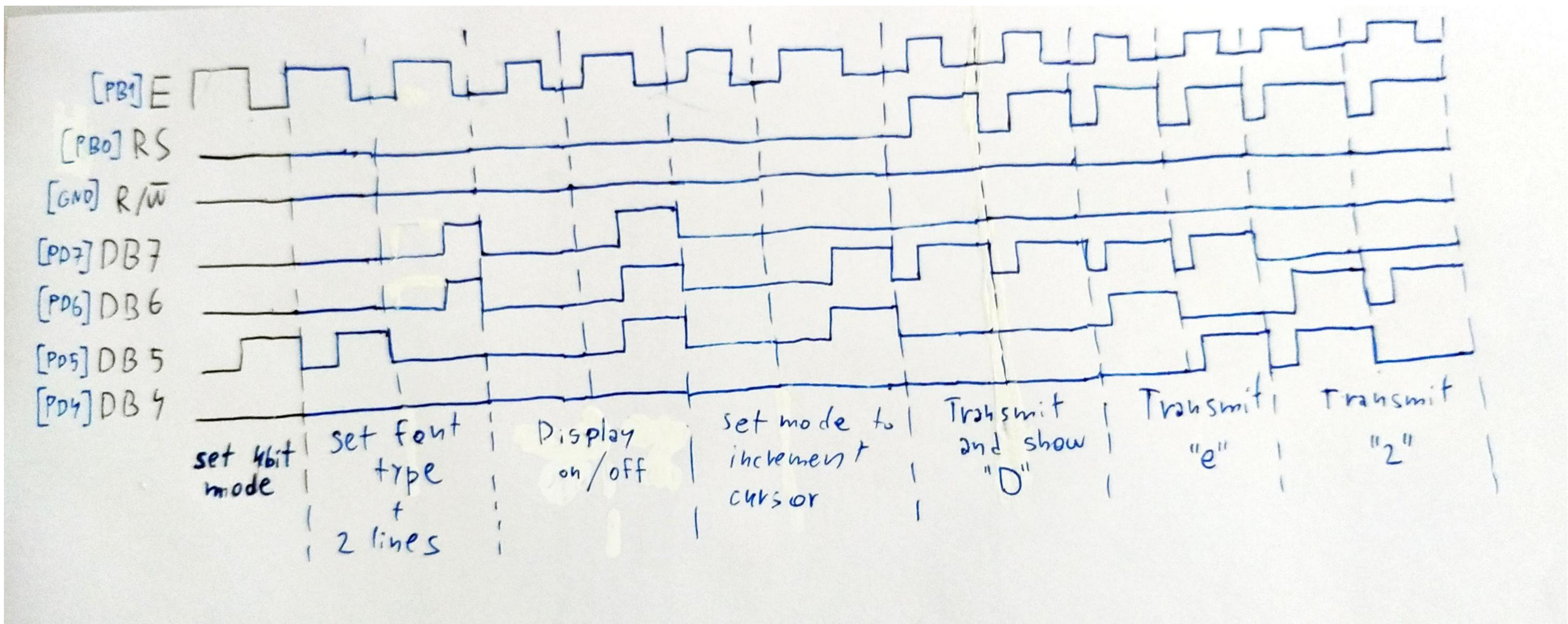
| Function name | Function parameters | Description | Example |
|---|---|---|---|
| `lcd_init` | `LCD_DISP_OFF`<br>`LCD_DISP_ON`<br>`LCD_DISP_ON_CURSOR`<br>`LCD_DISP_ON_CURSOR_BLINK` | Display off<br>Display on, cursor off<br>Display on, cursor on<br>display on, cursor on, blink char | `lcd_init(LCD_DISP_ON_CURSOR)` |
| `lcd_clrscr` | none | Clear display and set cursor to home position. | `lcd_clrscr();` |
| `lcd_gotoxy` | uint8_t x position, uint8_t y position | Set cursor to specified position. | `lcd_gotoxy(0, 1);` |
| `lcd_putc` | char c, to be displayed | Display character at current cursor position. | `lcd_putc('x');` |
| `lcd_puts` | char *s, array of chars | Display string without auto linefeed. | `lcd_puts("msg");` |
| `lcd_command` | LCD_CLR<br>LCD_HOME<br>etc ... (see) | Send LCD controller instruction command. | `lcd_command(LCD_CLR);` |
| `lcd_data` | uint8_t data, to be sent | Send data byte to LCD controller. | `lcd_data('c');` |

### LCD display module

1. In your words, describe what ASCII table is.

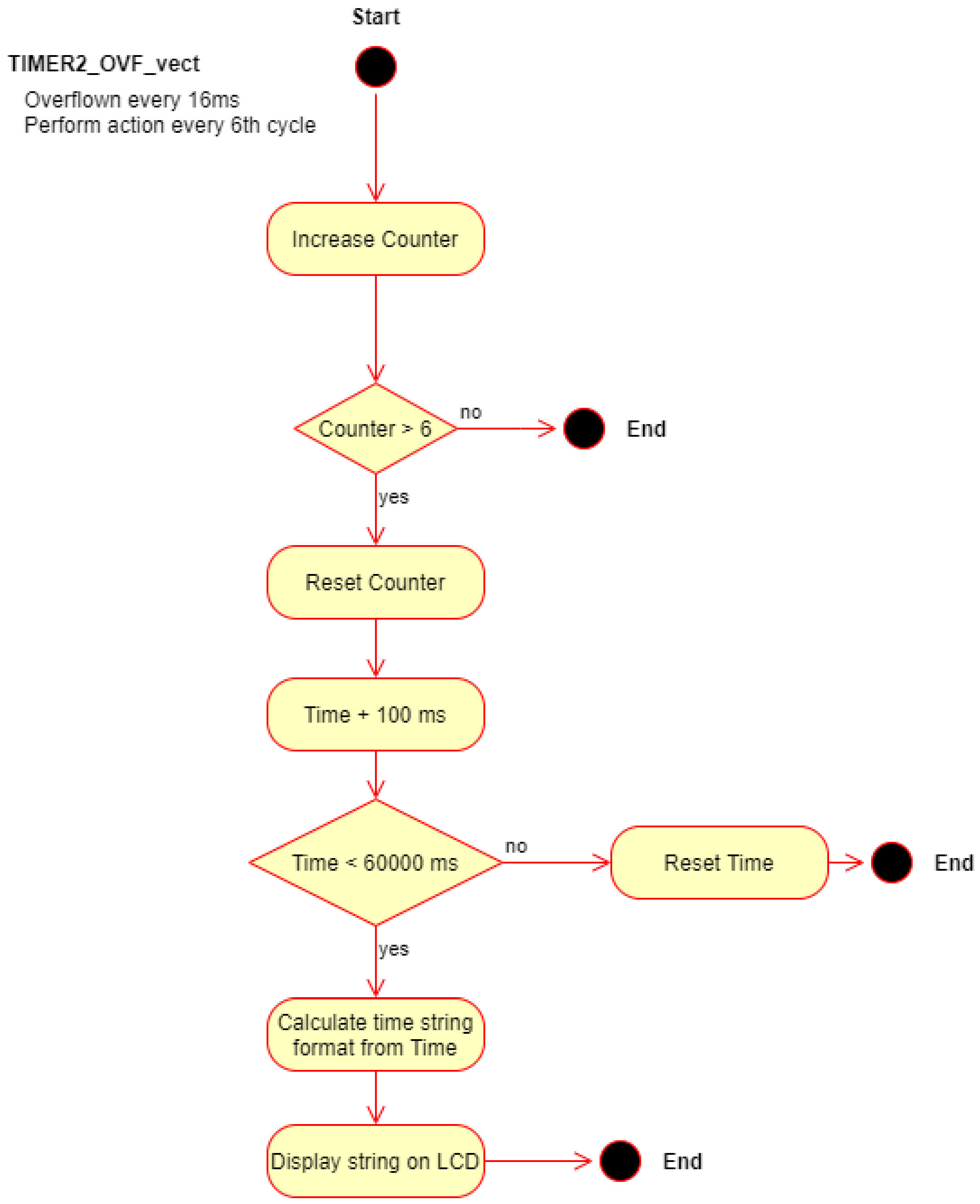
    - ASCII - is table that defines number value for standard characters (letters, numbers, control characters like new line/delete etc.)

2. (Hand-drawn) picture of time signals between ATmega328P and LCD keypad shield (HD44780 driver) when transmitting three character data `De2`.



## Stopwatch

1. Flowchart figure for `TIMER2_OVF_vect` interrupt service routine which overflows every 16 ms but it updates the stopwatch LCD approximately every 100 ms (6 x 16 ms = 100 ms). Display tenths of a second and seconds `00:seconds.tenths`. Let the stopwatch counts from `00:00.0` to `00:59.9` and then starts again. The image can be drawn on a computer or by hand. Use clear descriptions of the individual steps of the algorithms.



## Custom characters

1. Code listing with syntax highlighting of two custom character definition:

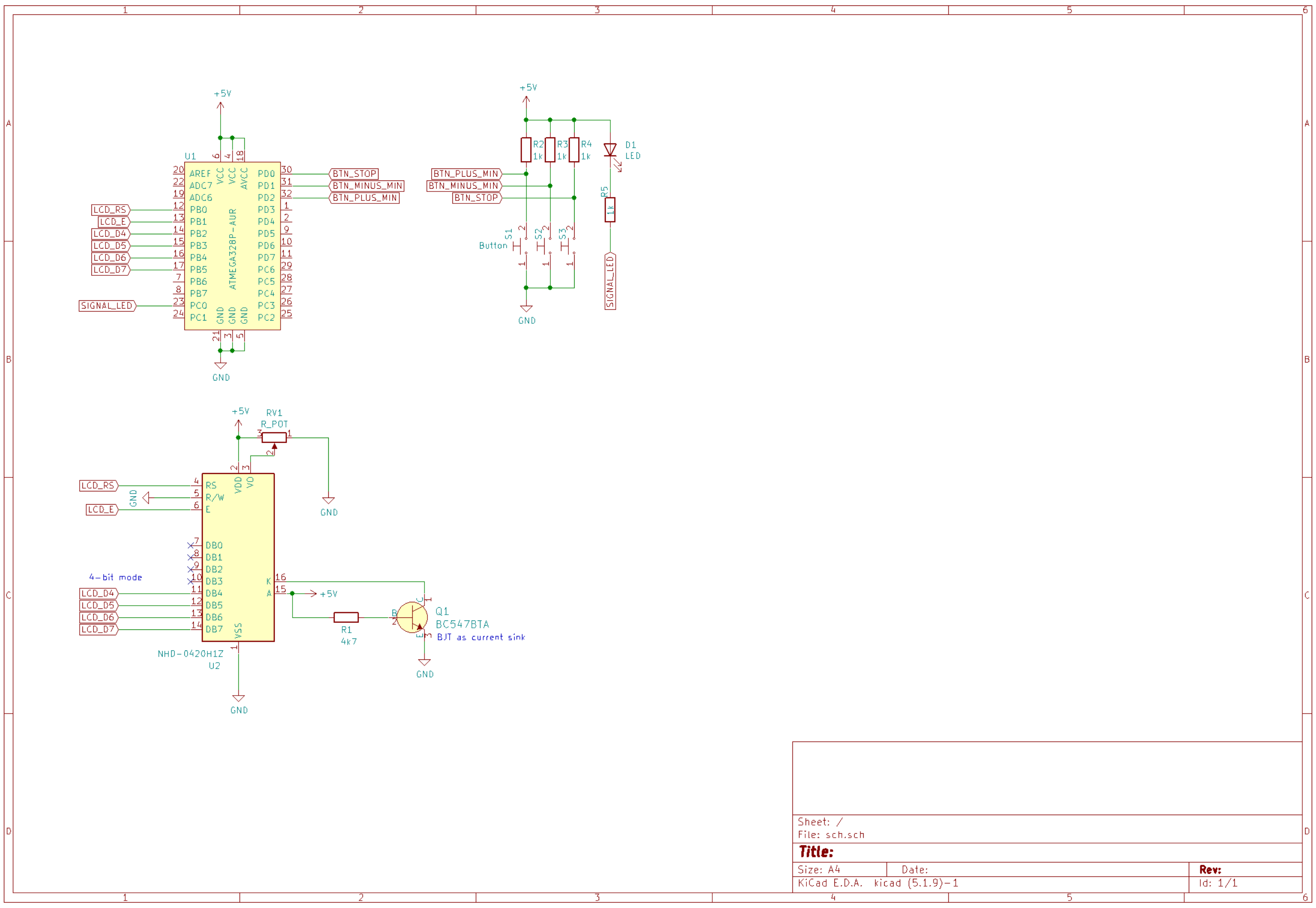
```
/* Variables ---------------------------------------------------------*/
// Custom character definition
uint8_t custom_char_1[8] = {
    0b00111,
    0b01110,
    0b11100,
    0b11000,
    0b11100,
    0b01110,
    0b00111,
    0b00011
};

uint8_t custom_char_2[8] = {
    0b01100,
    0b01100,
    0b01100,
    0b01100,
    0b01100,
    0b00000,
    0b01100,
    0b01100
};
```

## Kitchen alarm

Consider a kitchen alarm with an LCD, one LED and three push buttons: start, +1 minute, -1 minute. Use the +1/-1 minute buttons to increment/decrement the timer value. After pressing the Start button, the countdown starts. The countdown value is shown on the display in the form of mm.ss (minutes.seconds). At the end of the countdown, the LED will start blinking.

1. Scheme of kitchen alarm; do not forget the supply voltage. The image can be drawn on a computer or by hand. Always name all components and their values.



+5V
U1
ATMEGA328P-AUR
AREF
ADC7
ADC6
PB0
PB1
PB2
PB3
PB4
PB5
PB6
PB7
PC0
PC1
VCC
VCC
AVCC
GND
GND
GND
PD0
PD1
PD2
PD3
PD4
PD5
PD6
PD7
PC6
PC5
PC4
PC3
PC2
LCD_RS
LCD_E
LCD_D4
LCD_D5
LCD_D6
LCD_D7
SIGNAL_LED
BTN_STOP
BTN_MINUS_MIN
BTN_PLUS_MIN
GND
+5V
R2 1k
R3 1k
R4 1k
D1 LED
R5 1k
S1
S2
S3
Button
BTN_PLUS_MIN
BTN_MINUS_MIN
BTN_STOP
GND
SIGNAL_LED
+5V
RV1
R_POT
GND
LCD_RS
GND
LCD_E
RS
R/W
E
VDD
VO
DB0
DB1
DB2
DB3
DB4
DB5
DB6
DB7
K
A
VSS
4-bit mode
LCD_D4
LCD_D5
LCD_D6
LCD_D7
+5V
NHD-0420H1Z
U2
GND
R1
4k7
Q1
BC547BTA
BJT as current sink
GND
Sheet: /
File: sch.sch
Title:
Size: A4
Date:
Rev:
KiCad E.D.A. kicad (5.1.9)-1
Id: 1/1