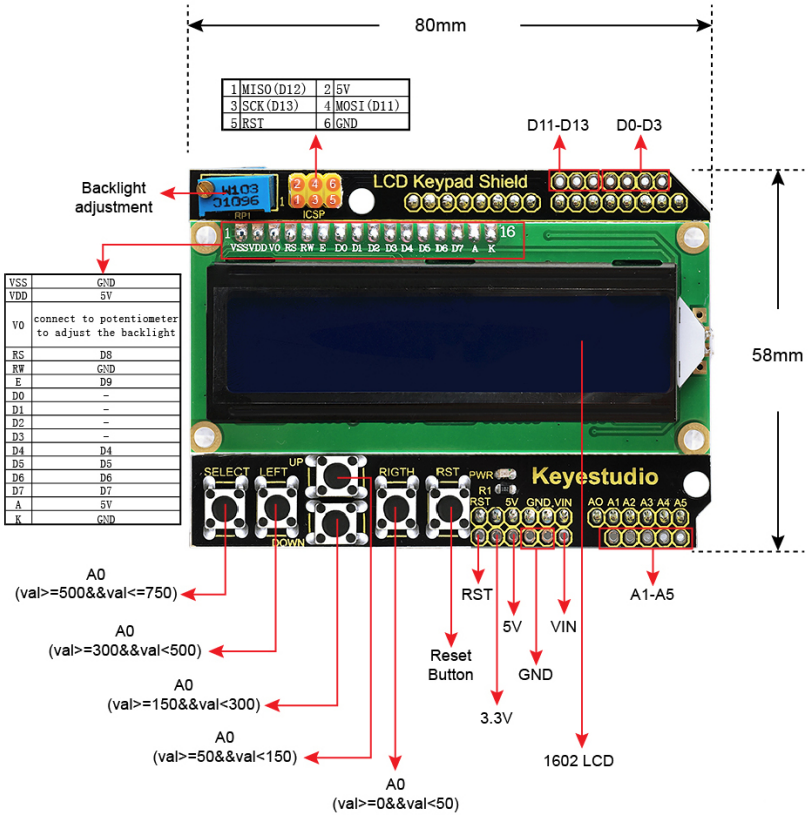
Digital Electronics 2 (Brno University of Technology)

Assignment of lab #7

2019
November

LCD Keypad Shield

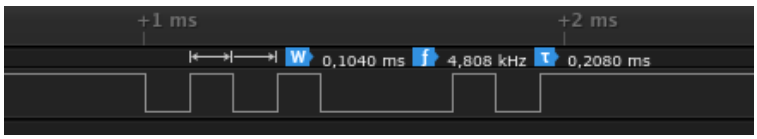


- In `docs/hw/` folder, see schematic of LCD Keypad Shield and find out the connection of Select, Left, Up, Down, and Right push buttons. How do you find out that one of the push buttons has been pressed?
- Calculate the voltage values for all voltage dividers.
- According to [1], which IO registers and which bits configure operations of internal ADC module?

Operation	Register(s)	Bit(s)
Voltage reference selection for the ADC	ADMUX	REFS1, REFS0
Selection which analog input is connected to the ADC		
Enable analog to digital converter		
Start analog to digital conversion		
Enable ADC conversion complete interrupt		
Division factor between f_{CPU} and input clock to the ADC		
ADC conversion result		

UART (Universal asynchronous receiver-transmitter) communication

- What ASCII code/character is transmitting in UART 8N1 mode? According to bit period (one bit duration), calculate the symbol rate. What is the duration of one UART frame in 8N1 mode?



Synchronize Git and create a new project

- In VS Code open your Digital-electronics-2 working directory and synchronize the contents of a repository with a single git command (`git pull`) or a sequence of two commands (`git fetch` followed by `git merge`).
- Create a new folder `project/07-uart` and copy three files from the last project: `main.c`, `Makefile`, `README.md`.

UART library

In the lab, we are using UART library developed by Peter Fleury [2]

- Use online manual of UART library and add the input parameters (output values) and description of the functions to the following table.

Function	Parameter(s)	Description
`uart_init`	`UART_BAUD_SELECT(9600, F_CPU)`	Initialize UART to 8N1 and set baudrate to 9600 Bd
`uart_getc`		
`uart_putc`		
`uart_puts`		

- Use template `project/07-uart/main.c` [3] and test functions to transmit character and string from ATmega328P to PuTTY SSH Client. Comment (or uncomment) source files you need within the list of compiled files in `07-uart/Makefile`.
- Use PuTTY SSH Client to communicate between computer and Arduino board. Setup this application as follows:

Category	Parameter	Value/Description
Session	Connection type	Serial
Serial	Serial line to connect to	`/dev/ttyUSB0` (You can use `dmesg` Linux command to verify the port)
	Speed (baud)	`9600`
	Data bits	`8`
	Stop bits	`1`
	Parity	None
	Flow control	XON/XOFF
Session	Saved Sessions	`usb0`
	button Save	(Save all your settings)
	button Load	(Load your saved settings)
	button Open	Open UART connection

WARNING: Before Arduino board re-programming process, PuTTY SSH Client must be closed!

Analog-to-Digital Conversion

- Configure ADC (AVcc with external capacitor voltage reference, channel ADC0, prescaler 128, enable interrupt), Timer1 (start ADC conversion 4 times per second), and UART (mode 8N1, baud rate 9600) modules. Read voltage level of push buttons, show the value on LCD display, and transmit to UART.

  Identify which push button was pressed according to the ADC value.

Clean folder and synchronize git

- Remove all binaries and object files from the working directory. Then use git commands [4], commit all modified/created files to your local repository and push them to remote repository. Use VS Code editor options to perform these operations.

Ideas for other tasks

- Use *ANSI escape sequences* [5] and modify color and format of transmitted strings according to the following Listing. Try other formatting styles.

```
/* Color/formatting sequence always starts by "\033[" and ends by "m" strings.
 * One or more formatting codes "#", separated by ";" can be used within
 * one line, such as:
 *    \033[#m      or
 *    \033[#;#m    or
 *    \033[#;#;#m  etc. */
uart_puts("\033[4;32m");        // 4: underline style; 32: green foreground
uart_puts("This is all Green and Underlined.");
uart_puts("\033[0m");           // 0: reset all attributes
uart_puts("This is Normal text again.");
```

- Program an interactive console that communicates between ATmega328P and the computer (PuTTY application) via UART. Let the main screen of the console is as follows:

```
--- Interactive UART console ---
1: read current Timer/counter1 value
2: reset Timer/counter1
>
```

After pressing the '1' key on computer keyboard, ATmega328P receives ASCII code of the key and sends the current Timer1 value back to PuTTY. After pressing the '2' key, ATmega328P resets Timer1 value, etc. Use ANSI escape sequences to highlight information within PuTTY console.

```
    uint8_t c;
    ...

    c = uart_getc();
    if (c != '\0') {        // Data available from UART
        if (c == '1') {     // Key '1' received
            ...
        }
    }
```

- Complete `README.md` file.

References

[1] ATmega328P - 8-bit AVR Microcontrollers,
`https://www.microchip.com/wwwproducts/en/ATmega328p`

[2] AVR-GCC Source Examples,
`http://homepage.hispeed.ch/peterfleury/avr-software.html`

[3] AVR repository for DE2 course at Brno University of Technology,
https://github.com/tomas-fryza/Digital-electronics-2

[4] Git Commands,
https://github.com/joshnh/Git-Commands

[5] ANSI escape code,
https://en.wikipedia.org/wiki/ANSI_escape_code