Crystal Clear Electronics

The development of the Crystal Clear Electronics curriculum was supported by the European Commission in the framework of the Erasmus + programme in connection with the “Developing an innovative electronics curriculum for school education” project under “2018-1-HU01-KA201-047718” project number.

Erasmus+

The project was implemented by an international partnership of the following 5 institutions:

- Xtalin Engineering Ltd. – Budapest
- ELTE Bolyai János Practice Primary and Secondary Grammar School – Szombathely
- Bolyai Farkas High School – Târgu Mureș
- Selye János High School – Komárno
- Pro Ratio Foundation working in cooperation with Madách Imre High School – Šamorín

XTALIN

Contact http://crystalclearelectronics.eu/en/
info@kristalytisztaelektronika.hu

24 - Control and Regulation

Written by Gergő Herángli, Csaba Apró, Máté Trádler and Barbara Ujvári

English Translation by Xtalin Engineering Ltd.

Revised by Ádám Szabó

INTRODUCTION

Getting control over something is in human nature. In the course of history, this characteristic continued to exhibit itself. The reason behind is that we only believe that something is our own if we have control over it. We can say that the desire of possession and control is the driving force behind the development of control theory (or control engineering) and sciences in its broadest sense. There are, of course, various aspects of this development. Nonetheless, we must not deny that some of the improvements and developments are the direct outcome of human behaviours like laziness. Think about television remote controls.

The steps for implementing a control loop are modelling, design and finally implementation.

By modelling we mean that we are able to create a model that accurately represents the characteristics of a real-world process. The result of a model is a mathematical equation such as Ohm's law. By process, we mean a measurable physical quantity. Taking the example of the DC motor mentioned in the previous chapters, the process might include the torque, angular velocity, or the position of the rotor. Each of these is measurable or derivable. Torque is a vector quantity that can be calculated from the current and angular velocity, while the position of the motor can be determined by a position encoder installed on the axle.

The non-measurable things are not considered as processes. In the design step, we are developing an algorithm, or in other words, a program that keeps the process at a pre-defined value. If we want to keep the speed of a drill bit on exactly 1 000 rpm then we expect this algorithm to set and keep the speed of the drill bit at 1 000 rpm (supposing there is no gear reduction). This is one of the most difficult tasks in which we must make numerous compromises. For example, if the drill is only rotating at 100 rpm, or is stationary, we cannot expect it to speed up to 1 000 immediately. (This would require infinite energy.) In the implementation step, we translate the developed algorithm to microcontroller code.

The above might seem confusing, however we will attempt to make it easier to understand through examples. The purpose of this curriculum is not to provide a comprehensive knowledge but to provide a simple and useful knowledge about the control engineering.

In the following sections, I want to clarify some concepts, explain more accurately what the system modelling means, and finally, I want to introduce a simple, frequently used control technique.

Erasmus+

This project was supported by the European Commission. The content of this publication does not reflect the official opinion of the European Union. Responsibility for the information and views expressed therein lies entirely with the author(s).

CONTROL LOOPS

In the introduction, we went through some of the key terminology, and in the following we would like to introduce control loops more systematically. First, we must bring clarity to the elements that are involved with a control loop.

There is a process that we want to control. Furthermore, we have a firm determination on what value we want to keep the process, this is called the *reference*. The reference can take on broadly any value, which is why a function block is needed that does not allow us to request any given value from the controller, it is usually responsible for restraining changing rate of the reference.

There must be a control device in the loop that enables us to affect the process. This is necessary because the control algorithms are running on the microcontroller. As it has been noted in a previous chapter, the microcontrollers have 3.3 V to 5 V peripherals. Such an output is insufficient for driving an electric motor. The *actuator*'s job is to convert these weak signals of the microcontroller to strong signals. In practice this actuator is some kind of amplifier stage, but if we need to operate a system in an environment where an explosion might occur, we might need to use a pneumatic actuator as the control device. It is evident that we cannot use a relay as a switching device in such an environment since the electric sparks created when the relay gets opened might cause an explosion.

In 99% of the cases, the microcontroller is the control unit in the system. Its function is to run the control algorithm and decide what kind of control signal to generate for the actuator based on the input. There are numerous benefits of using a microcontroller, as they can be easily reconfigured, and we can modify all the parameters even while the program is running. Furthermore, with the increase of their computing power, number and complexity of their peripherals, they can run almost any algorithm. The question may arise, why it is only 99%? There are analog current loops which's control system is not programmable or the signals in the system are not digitalized. Their presence is mainly restricted to special areas. If high speed is required analog circuits are used even today. For example, in the case of a Phase-Locked Loop (PLL).

Since most engineers are visual types, many schematics and illustrations are used in the field of control engineering to help understanding. The diagram containing the control elements and their logical relations is called a *block diagram*, which helps understanding the operation and the function of each block.

In terms of operating principle, there are two types of control loops: open-loop and closed loop.

OPEN-LOOP CONTROL

Let us start with the concept of open-loop control. A control scheme is called open-loop control if the input signal runs through the block chain, the output of the controller drives the actuator, which has an effect on the process. It is called open-loop control, because the block diagram is open, the controller does not know the impact of its intervention because there is no sensor on the process (or it is not taken into account in the controller). Interference signals coming from various sources can alter the outcome, and we need to know their effects in advance to eliminate them.

The following figure shows a general open-loop control block diagram.

This project was supported by the European Commission. The content of this publication does not reflect the official opinion of the European Union. Responsibility for the information and views expressed therein lies entirely with the author(s).

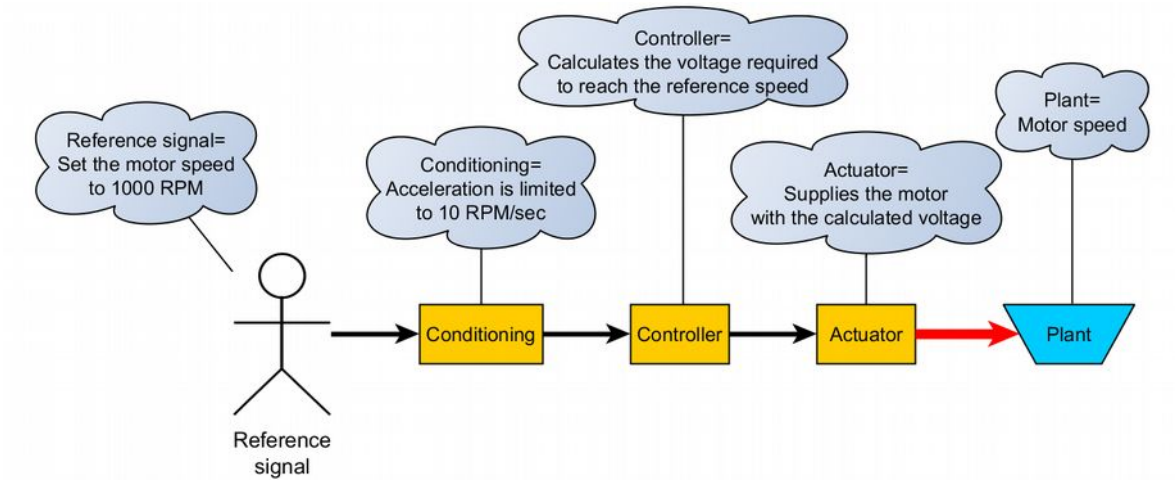

Figure 1 – An example open-loop control block diagram

In the figure above, you can find the blocks that were explained in the previous chapter highlighted in yellow. In the clouds you can see certain points of times or events that the system experiences.

Let's have a look at the elements corresponding to the logical blocks in the example.

At the beginning of the block diagram there are we, giving the reference signal to a variable-speed electric drill. We want to set the speed of the drill bit to 1000 RPM, by adjusting the speed control of the drill to 1000 RPM. Behind the speed control adjuster there is probably a potentiometer which gives an output signal to the rest of the elements in the block diagram.

In the drill example a kind of reference signal conditioning can be that the limits of the potentiometer represent a safe range of possible RPM values which prevents you from setting an arbitrary value.

The controller is an algorithm that runs on a microcontroller, it uses the conditioned reference signal to generate an output signal which represents 1000 RPM based on the data stored in memory.

The actuator is a power electronics circuit that can drive the motor.

The value that we have set is probably valid when the drill is being used (under load). If we turn it on without load, the idle speed will be higher.

If we use the drill improperly, like holding the drill bit down as strongly as we can, this can mean a disturbance signal that the controller cannot compensate as the effect was not known beforehand. Since it was not known, the control logic won't request a higher torque from the actuator, and the speed most certainly won't be 1000 RPM.

Generally speaking, open loop-controlled devices are simpler because they do not contain sensors, so it is not required to select or embed them into the system or to interface their signals to the control equipment. Moreover, an open-loop control logic can be simpler than a complex and fast closed-loop system.

Erasmus+

This project was supported by the European Commission. The content of this publication does not reflect the official opinion of the European Union. Responsibility for the information and views expressed therein lies entirely with the author(s).

CLOSED-LOOP CONTROL

In contrast with open-loop control systems, a closed-loop control system is continuously measuring the output of the process with a sensor. The signal is fed back to the input where it is subtracted from the reference signal resulting in the *error signal*. The error signal becomes the input of the controller. This is called closed-loop control, meaning that the controller measures the effect of its intervention. In addition, the control algorithm aims to decrease the error signal to 0 since we want to measure the output to be equal to the reference signal.

The following figure shows a closed-loop control system for our motor speed example.

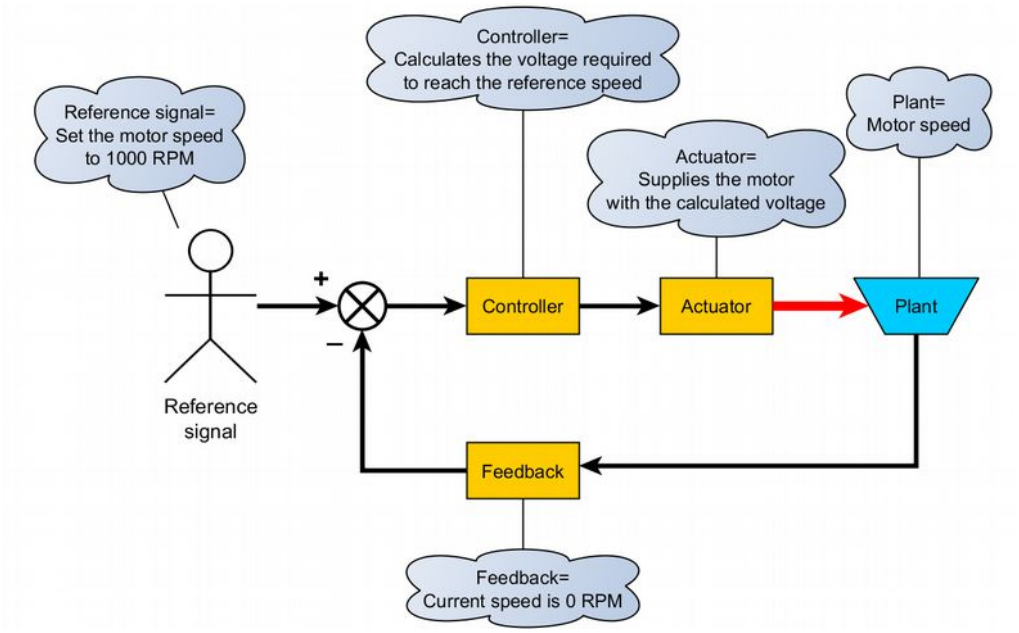

Figure 2 – An example closed-loop block diagram

The block diagram of the closed-loop control system can be complemented with subtractor, amplifier and disturbance blocks but for simplicity, I did not depict them in the figure or only by adding a simple symbol.

Control engineering is perhaps the most important part of industrial control, as most of the industrial systems encompass several types of control systems and associated instrumentations including closed-loop control.

For instance, in a nuclear power plant we cannot use a simple open-loop control system, considering that if the actuator does make sure about that the graphite rods that slow down the reaction are in the right position, then we might notice this error too late. Therefore, we need a closed-loop control system that automatically adjusts the process output. For example, it controls the output of the process with the highest possible accuracy to 10 MW according to the reference which is 10 MW.

PID controller

Nowadays the PID (Proportional-Integral-Derivative) controller is widely used in industrial control systems. It is not an easy task to determine the parameters of a controller but once we find the right parameters,

Erasmus+

This project was supported by the European Commission. The content of this publication does not reflect the official opinion of the European Union. Responsibility for the information and views expressed therein lies entirely with the author(s).

we can very easily implement the PID control algorithm on a microcontroller. The PID controller requires only three basic mathematical operations: multiplication, addition and subtraction.

But what does the PID controller do exactly? The controller responds to the current, past and future value of the error signal, which can be done this with three terms represented by the three letters in its name (P-I-D). The first letter, *P*, denotes the proportional term which means that the error signal input is multiplied by a constant. The term *I* represents the integrating term, which accounts for past values. The error is integrated, which means that the current error signal is added to the sum of the previously measured error signals. Term *D* stands for derivative, it is used to determine the curve of the error signal, simply put, it is the best estimate of the future trend of the reference and the measured process variable error, based on the difference between the past and the current rate of change. The controller will generate the overall control value from these three values. The PID is a relatively simple type of controller and there are more complex controllers than this in use.

COMPARISON OF OPEN-LOOP AND CLOSED-LOOP CONTROL SYSTEMS

Following the brief introduction of the open-loop and closed-loop control systems, we could ask ourselves a lot of questions like: Which group do the microcontroller projects from the previous chapters belong to? In control engineering terms, these were all open-loop control tasks as the sensors (e.g. LDR, NTC thermistor) and the interfering peripherals (e.g. piezo, fan) that we talked about were used separately, or if they were used together, there was no closed control loop designed.

However, if the reference signal of a control process is coming from a sensor (e.g. from an analog temperature sensor) and this signal is used in the control device to operate a peripheral device (e.g. a fan), then it is not yet certain that we can talk about closed-loop control.

For instance, if a control unit is able to turn on the fan located in the same room, when the temperature is above the ideal, then we are talking about an open-loop control system if the two premises are independent from each other, more precisely they are unable to achieve heat transfer. However, if the fan located in one room can circulate cooler air to the other one, then it is a closed-loop control system.

If the fan is operating in the same, closed room, we are also talking about closed-loop control, but with a very poor efficiency. As an effect of the fan (airflow) our skin evaporates more water, therefore we will feel the environment cooler, but its temperature is certainly not dropping (it can even slightly increase because the dissipated power the coils release as heat).

Another example can be a light switch with dimmer controls, where we know that there is usually no light sensor placed in the wall or in the lamp which could measure the intensity of illumination and provide feedback for a closed-loop controller. In fact, this is an open-loop control system and not a closed-loop control one. The potentiometer under the dimmer sets the reference for the power electronics, that is driving the bulb. If we make a device that can control the power electronics of the lamp, in a given variable range, has a light sensor that the device takes into account when running the control algorithm, and it can produce a predetermined luminous intensity, we are talking about closed-loop control system.

Erasmus+

This project was supported by the European Commission. The content of this publication does not reflect the official opinion of the European Union. Responsibility for the information and views expressed therein lies entirely with the author(s).

CONFIGURABLE OPEN-LOOP CONTROL

The following example will be, like most previous tasks were, an open-loop control task. However, we will now implement a control device that is configurable through UART.

We would like to turn our fan on or off depending on temperature. With the help of the ADC peripheral of the microcontroller we will measure the voltage across an NTC thermistor. The output voltage of the voltage divider network will be used to calculate the NTC resistance and then the temperature. The temperature threshold above which the fan needs to be turned on will be received via serial port.

Let's build the circuit below and open the *CE24_1_Configurable_control* project which is available on the website of the curriculum. We already know all the peripherals needed to complete this task from the previous chapters, so their configuration will not be explained here again.

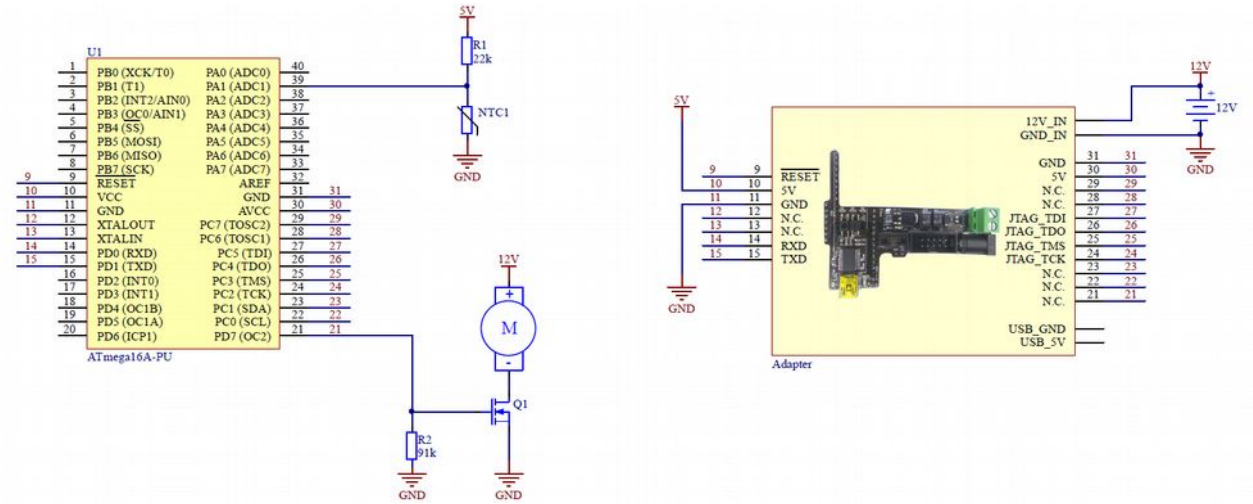

Figure 3 – NTC example schematic

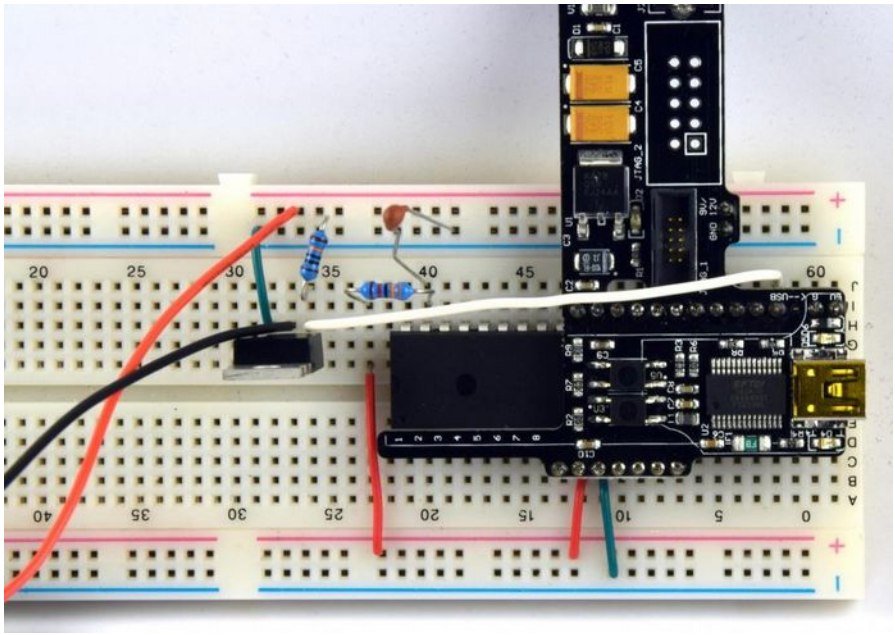

Figure 4a –The assembled circuit

Erasmus+

This project was supported by the European Commission. The content of this publication does not reflect the official opinion of the European Union. Responsibility for the information and views expressed therein lies entirely with the author(s).

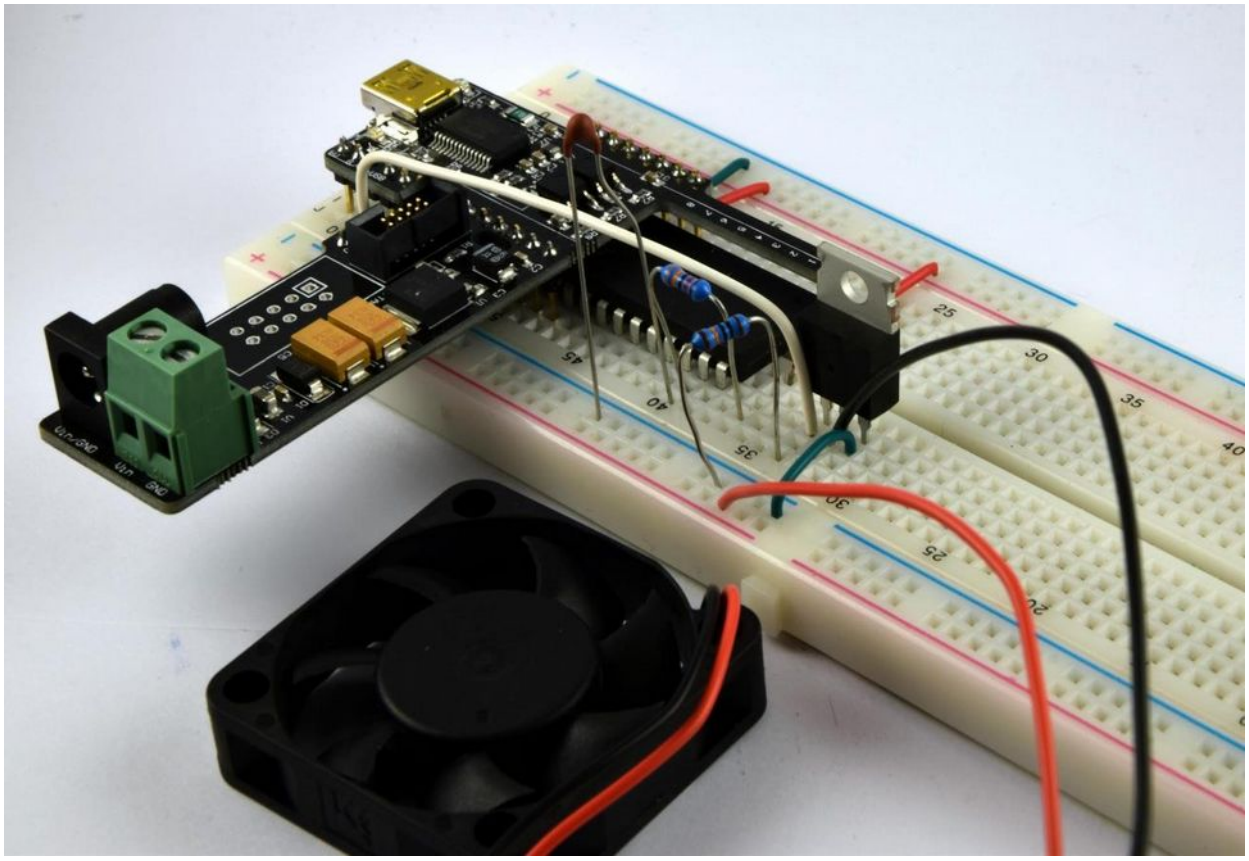

Figure 4b –The assembled circuit

The important things are all in the `while(1)` infinite loop. The AD conversion and the UART receiving are implemented with interrupts. Only the values are saved there, so the processor spends less time executing the interrupt. Calculations and evaluations are executed in an infinite loop with a specific timing, in this case in every second.

```
while (1)
{
	// Converting the ADC value into voltage
	U_NTC = adc_value;
	U_NTC = U_NTC * ADC_CONST;

	// Converting the voltage value into temperature
	// this linear approximation gives a correct value
	// roughly between -10 and 40 °C
	// for the time being, please accept this approximation

	temperature = (-172 * U_NTC + 602530)/10000;

	// If the measured temperature is higher, than the limit set on UART
	if (temperature > limit)
	{
		// Turning on the fan (duty cycle 100%)
		OCR2 = 255;
	 }
```

Erasmus+

This project was supported by the European Commission. The content of this publication does not reflect the official opinion of the European Union. Responsibility for the information and views expressed therein lies entirely with the author(s).

```
        else
        {
                // Turning off the fan (duty cycle 0%)
                OCR2 = 0;
        }
        // Sending of measured voltage and temperature, and the set limit
        printf("measured: %ld mV %ld C limit: %d C\n", U_NTC, temperature, limit);
        _delay_ms(1000);
}
```

At first, the value of the AD conversion is converted to voltage. The `adc_value` variable stores the voltage measured between 0-5 V on 8 bits. This means that the

$$U_{NTC} = \frac{ADC_{value}}{256 \cdot 5\ [\mathrm{V}]}$$

formula determines the measured voltage. If we write this into our program, we will experience that the value of U_NTC will always be 0 V. The reason for this is, that `adc_value` is a number between 0 and 255. Dividing this by 256, the processor will give 0 as a result as they are both integers. When dividing integers, the processor will always return the integer part of the result. Furthermore, if we would like to express the measured voltage in volts, it requires using fractions. There is only a limited support for this in the ATMega16A microcontroller, for this reason we should find another solution. Let's calculate the measured voltage in mV! The formula used is the following:

$$U_{NTC} = ADC_{value} \cdot \frac{5000}{256\ [\mathrm{mV}]}$$

However, we should also pay attention to that if a number is between 0 - 255 multiplied by 5000 is a number between 0 - 1 275 000, which can only be stored in a 32-bit variable, so we also have to execute the operation with 32-bit numbers.

Previously, we already examined how the voltage on an NTC thermistor varies with temperature.

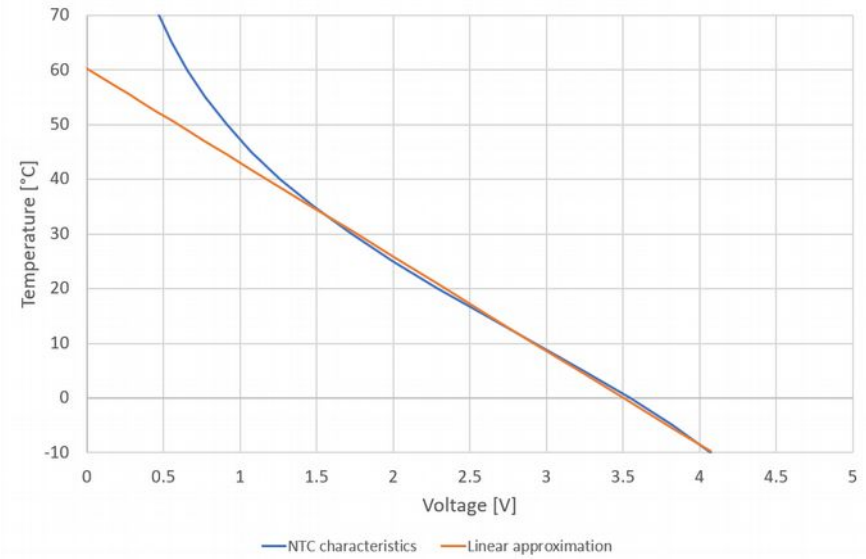

Figure 5 – Curve fitting to NTC characteristic

Erasmus+

This project was supported by the European Commission. The content of this publication does not reflect the official opinion of the European Union. Responsibility for the information and views expressed therein lies entirely with the author(s).

We can see that the blue curve is quite straight between -10 °C and 40 °C. By fitting a line, we can convert the voltage measured in mV to a temperature in °C. Finding the optimal fit is a complicated mathematical process, so please accept without explanation, that we will get the following equation:

$$T = \quad -0.0172 \cdot U_{NTC} + 60.253 \ [°C]$$

Because of the same considerations as above the equation is modified like as follows:

$$T = \frac{-172 \cdot U_{NTC} + 602530}{10000 \ [°C]}$$

Of course, even without a deeper understanding of mathematics, we can fit a line manually to the curve and then read its equation from the graph.

The threshold value read through UART is compared with the temperature calculated with the equation above to set the duty cycle of the PWM, which controls the fan. By default, only 0% or 100% is set.

At the end of the program, the measured/calculated values are displayed together with the threshold, so it is easier to check the correct operation of the program. Program the microcontroller, connect it to your computer via USB and launch the HTerm program used in the UART chapter. Use 4800 baudrate, 8-bit data length, 1 stop bit, and no parity.

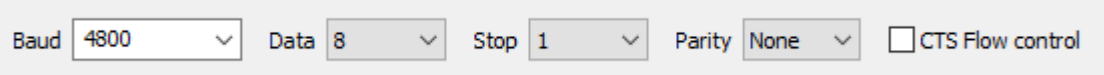

Figure 6 – UART settings in HTerm

After connecting you will see the following:

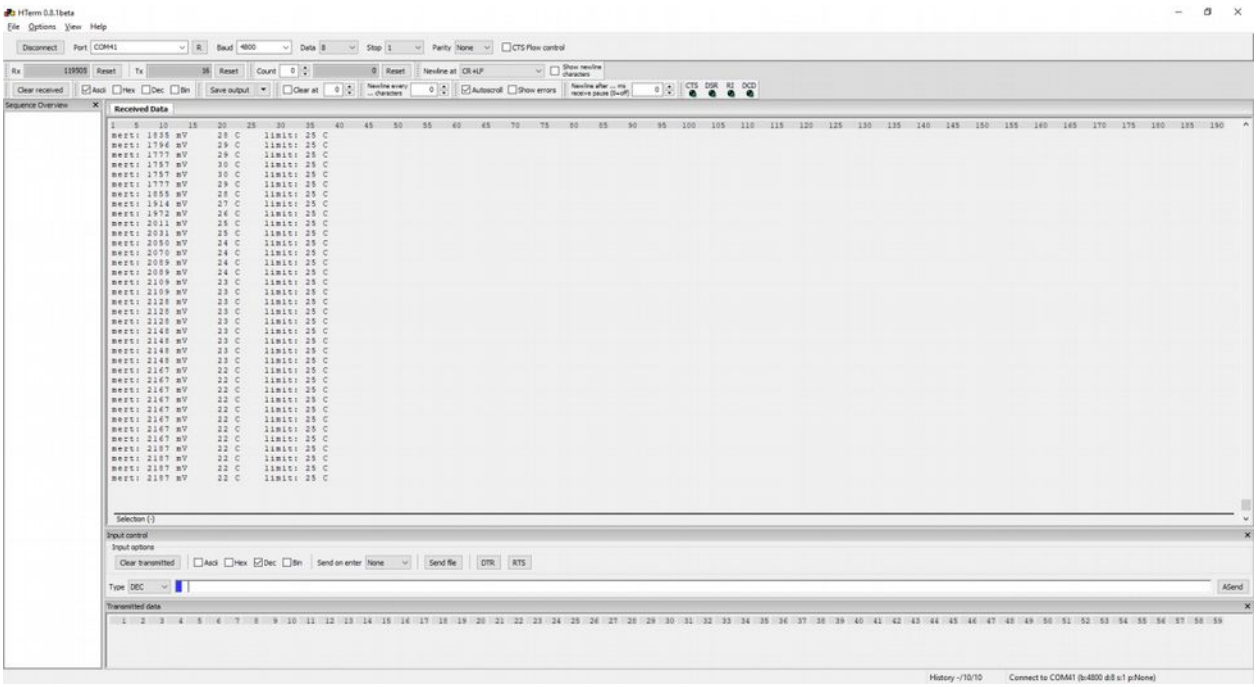

Figure 7 – Output in HTerm

Erasmus+

This project was supported by the European Commission. The content of this publication does not reflect the official opinion of the European Union. Responsibility for the information and views expressed therein lies entirely with the author(s).

To be able to change the limit, change the type of “Input control” from “ASC” to “DEC”. Type in an arbitrary number between 0 and 255 (in the example it is 27) and hit Enter to send it to the microcontroller. This number represents the limit in °C.

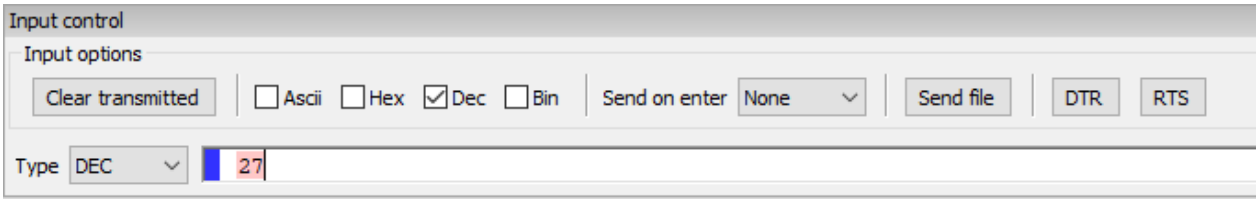

Figure 8 – Modifying the threshold in HTerm

In the output message sent every second, you can see that the limit value has changed to the number you entered.

If you heat the NTC thermistor with your hands, you can see that the voltage on it is decreasing and the temperature is rising at the same time. As soon as this exceeds the limit, the fan will turn on and then turn off when you release the thermometer and it starts to cool down to room temperature.

With this, we implemented the temperature-dependent and configurable control of the fan.

Try to implement the following additions if you want

- The duty cycle of the fan should vary gradually between the set limit and 10 °C above it.
- Through UART, the microcontroller should receive two values and gradually adjust the duty cycle between them.

Erasmus+

This project was supported by the European Commission. The content of this publication does not reflect the official opinion of the European Union. Responsibility for the information and views expressed therein lies entirely with the author(s).

We can encounter microcontroller-based controllers in the most diverse fields: About the Polymerase Chain Reaction

Written by Barbara Ujvári

The history of Polymerase Chain Reaction (PCR) begun in 1985 when Kary Banks Mullis published the principle of the method in a scientific magazine, called Scientific American. The toolbox of molecular biology has been expanded by a simple, but brilliant method. The method is a cyclic DNA synthesis technique which allows to copy the studied DNA fragment outside of the living organism. By repeating the reaction several times, the original DNA sequence can be multiplied to a detectable amount. The significance of this innovation is demonstrated by the fact that Mullis was awarded the Nobel Prize in Chemistry in 1993 for his invention. Nowadays, the PCR device can be considered as a standard equipment of molecular biology laboratories.

The steps of the PCR reaction are repeated cyclically in a temperature range of 50-95 °C. The PCR machine used for this is an automated heating block, controlled by a microcontroller, in which the cooling and heating effects are provided by Peltier modules. The machine is capable of switching between different temperature sections very quickly and accurately. The programmable temperature range is usually between +4 and +99 °C and the heating-cooling rate is between 2-3 °C/second. The machine is also capable of providing equable temperature environment with a precision of one tenth of a Celsius degree that is needed for each step. DNA synthesis is performed in PCR tubes for this purpose. The thermal blocks of this device are made of metal with good thermal conductivity, usually aluminum. The metal blocks can be covered with heated lid, which prevents the reaction mixture from evaporating.

Figure 9 – A picture of a PCR machine

Erasmus+

This project was supported by the European Commission. The content of this publication does not reflect the official opinion of the European Union. Responsibility for the information and views expressed therein lies entirely with the author(s).

Figure 10 – The volume of a PCR tube is 200 µl

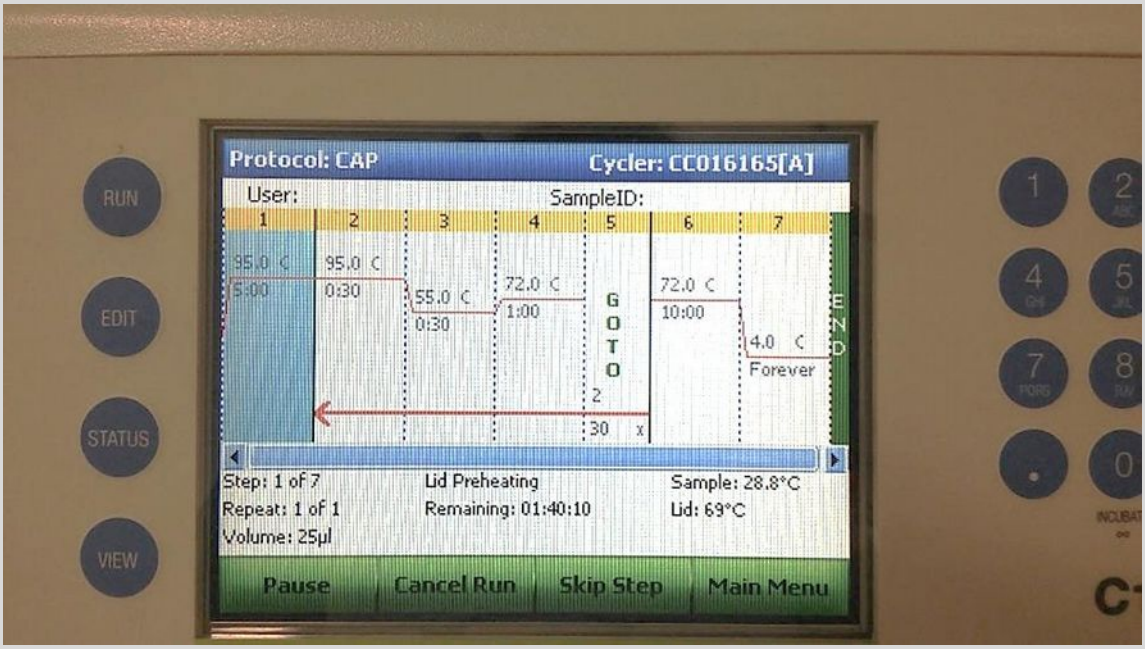

Figure 11 – The diagram of an average PCR protocol

Erasmus+

This project was supported by the European Commission. The content of this publication does not reflect the official opinion of the European Union. Responsibility for the information and views expressed therein lies entirely with the author(s).