LAPPEENRANTA-LAHTI UNIVERSITY OF TECHNOLOGY LUT

School of Energy Systems

Master's Program in Electrical Engineering

Nikita Krasnov

DESIGN OF EMBEDDED CONTROL SYSTEM FOR CONVEYOR LINE APPLICATION USED IN ELECTRONIC COMPONENTS' SURFACE MOUNTING

Master's thesis

Examiners: Professor Olli Pyrhönen

D. Sc. Niko Nevaranta

ABSTRACT

Lappeenranta-Lahti University of Technology LUT
School of Energy Systems
Degree Program in Electrical Engineering

Nikita Krasnov

Design of embedded control system for conveyor line application used in electronic components' surface mounting

Master's thesis
2021
67 pages, 31 figures, 0 tables and 0 appendices
Examiners: Professor Olli Pyrhönen, D. Sc. Niko Nevaranta

Keywords: conveyor line, master-slave, microcontroller, stepper motor control, synchronous control

This study focuses on the process of surface mounting of the electronic components over the printed circuit boards. The system includes multiple operational stages, where printed circuit boards are transported by conveyor lines. The transported board is not fixed on the conveyor belt, but its' accurate positioning is critically important to ensure successful transfer to the receiving stage and to avoid the entire process stop. Thus, the necessity of control of the board position offset during the transportation appears.

The main objective of this study was to develop embedded control system, which can synchronously control multiple stepper motors used in conveyors. In order to fulfill this task, the principles of the synchronous stepper motors' control were studied and based on it the structure of the particular control system was defined. One of the microcontrollers is set as master and translates the reference value to the other system microcontrollers. The controlled variable is the position offset of the printed circuit board location during its transportation by the conveyor. The actual value of the offset is estimated by a position sensor, which is mounted on the conveyor side. The reference value is set using the LCD-keypad shield, which is connected to the master microcontroller. The latter transfers this reference synchronously to all slaves, which derive control signals for the connected stepper motors. One of the motors is directly controlled by the master.

The electrical connections between the control system equipment were designed and after that the corresponding printed circuit boards' topology drawings were developed. Then the programmable code for microcontroller was constructed basing on the common programming structures, and the simplified model of a single conveyor control driven by a stepper motor was created in *MATLAB-Simulink* environment. As a result, an embedded control system capable of multiple synchronous control is developed that can be considered for practical implementation.

ACKNOWLEDGEMENTS

This thesis could not be realized without some people's help. First of all I'd like to thank Professor Olli Pyrhönen for supervising this study and especially D.Sc. Niko Nevaranta who is worthy of the highest and warmest gratitude for his invaluable support, helpful advices and great patience during the entire thesis preparation period. And also I want to thank D. Sc. Katja Hynynen for her help with formal organizational moments of the thesis preparation and making it easier and clearer for me.

As this thesis was prepared as a part of the Double Degree program, I'd like to thank also my supervisor in SPMI-university - Professor Vadim A. Shpenst - and counselor - Dmitrii A. Burylov - for their help in thesis preparation in current circumstance when the communication was a bit difficult due to the isolation restrictions. The administrations of Saint Petersburg Mining University and Lappeenranta-Lahti University of Technology deserve the acknowledgements for giving me the opportunity to participate in this Double Degree program. Special thanks for help with papers in Russia (which significantly helped me to concentrate on the particular thesis research and not to consider about formal paper business) is to Ekaterina Kanygina, whose strong expressions are also a great source of inspiration.

And of course my family, especially my mother Zhanna Krasnova, deserves special appreciation; without your support, motivation and influence on me I would have never been here.

Lappeenranta, May 2021

Nikita Krasnov

CONTENTS

LIST OF ABBREVIATIONS AND SYMBOLS

Abbreviations

AC	Alternative current
ADC	Analog-to-digital converter
DC	Direct current
DC/DC	Direct current to direct current [converter]
EUSART	Enhanced Universal Synchronous/Asynchronous Receiver-Transmitter
I^2C	Inter-Integrated Circuit
LC	Inductor-Capacitor [filter]
LCD	Liquid crystal display
NEMA	National Electrical Manufacturers Association
NVRAM	Non-volatile random access memory
PCB	Printed circuit board
PI	Proportional-integral
PID	Proportional-integral-derivative
PWM	Pulse-width modulation
RAM	Random access memory
RISC	Reduced instruction set computer
ROM	Read-only memory
RS-485	Recommended standard-485
SISO	Single input, single output
SMD	Surface-mounted device
SPI	Serial Peripheral Interface

VCSEL Vertical-cavity surface-emitting laser

Symbols

A	System matrix
a(t)	Reference signal to master
B	Input matrix
b	Belt damping constant
C	Output matrix
e(t)	Error signal
J	Motor inertia, kg·m^2
K	Equivalent spring constant, N/m
K_d	Derivative gain of PID-controller
K_i	Integral gain of PID-controller
K_p	Proportional gain of PID-controller
m	Mass of the transported PCB, kg
r	Conveyor shaft radius, m
r(t)	Reference signal from master to slave
T_e	Drive's torque, Nm
u(t)	Subsystem input signal
x	Position of the transported PCB, m
y(t)	Subsystem output signal

Δh	Distance between the conveyor's side and the nearest PCB's edge, m
Δh_{act}	Actual value of Δh, m
Δh_{ref}	Reference value of Δh, m
φ	Drive shaft angular coordinate, rad

1 INTRODUCTION

Conveyors are one of the most important parts in manufacturing processes that consist of multiple stages. Their role in moving a processed object between different operational stages makes the control of the conveyors' operation and its behavior a very significant and critical part of any system design. The conveyor control system must ensure uninterruptible movement of the delivered object from one operational unit to another without failures. A typical manufacturing process often includes a number of conveyors and this leads to the need for synchronous operation of the conveyors that are parts of a whole process or its branch. Depending on the case, the conveyors should be set up with equal movement speed or other operational parameters.

This thesis focuses on a manufacturing system that consists of multiple operating stages that require conveyors between them. The system under study is located on the enterprise that specializes on electronic devices design and processing. The system realizes full-process surface mounting of electronic components onto the printed circuit board (PCB). The whole operation process combines the following stages:

1. Capture of a single PCB from the stack.

2. The solder paste distribution on the PCB surface according to the preset pattern configuration.

3. Actual surface mounting.

4. Cleaning of the PCB's surface from remains of the solder, flux, dust and other redundant substances.

5. Drying of the PCB.

6. Automated inspection of the mounting quality by means of computer vision and electrical testing.

Between the mentioned stages the PCBs are moved by conveyors. The PCB is given to the conveyor from the previous stage equipment and then goes to the next one by rigidly-fixed rotating teeth chain, located on the stage equipment side. The structure of the transporting lines of the operational stages requires the design of special holes on the PCB; when the teeth are positioned on the holes, the PCB starts to move with help of the gear.

While the object is moved by a conveyor, it is not fixed on the belt. Thus, depending on the speed of the belt motion and acting forces, the position of the object can change after starting the transportation by conveyor. The higher the conveyor motion speed is, then the higher the possibility of exceeding the critical offset value is. Here, the critical (or maximum) offset means that the position of the object with such offset is still applicable for the successful PCB and gear coupling, but after its exceeding the coupling becomes impossible, which results in the system failure. Consequently, the location of the PCB on the conveyor during the whole motion time is critically important to ensure the coupling of the PCB's holes with gear on the border of the conveyor and the next operating stage equipment. Ensuring the latter is important to guarantee the continuous operation of the process..

1.1 Objectives

In the application under study, the PCB's location can be estimated by measuring the distance between one of the conveyor line borders and the nearest PCB's edge. Such measurement is achieved by position sensors that are mounted on the side of each conveyor. The measured signal is read into the processing unit, which compares it to the desired value. Based on the error process control signals are formed and delivered to the driver of the stepper motor, which sets the conveyor into motion. This way the speed of the conveyor movement is controlled and consequently the offset of the mentioned distance is regulated.

The conveyors used in the system are identical; for one operational cycle each conveyor's behavior can be assumed to be the same as others'. This allows implementation of the system communication between processing units by means of the master-slave model and consequently enables setting the offset reference value to one controller, master controller, which gives the command to other controllers. In general, such approach simplifies the overall control system. Thus, the synchronous control of the slave controllers becomes a task that should be solved. The produced control signal by the master microcontroller is used as a reference for the slave microcontrollers and is treated by them according to their own settings in order to derive the control signal for stepper motors which are controlled by them.

Therefore, the general aim of the described study is to design the embedded control system for the finite number of conveyor lines, which operate in one technological process with the same motion speed. The corresponding motion speed of the transportation system is set to

all the conveyor drives synchronously by the corresponding microcontrollers, which are ruled by the main processor unit (master). In general, the problem solved in the thesis can be tightened to solution of two fundamental issues: the implementation of the single stepper motor control and ensuring the synchronous operation of multiple number of identical stepper motors.

In addition, the system should have a convenient user interface for ensuring the possibility of manual changes of the reference value for the maximum position offset and for providing the system operator with information about happened or potential errors during the operation process. Hence, it is necessary to define the speed and acceleration/deceleration values at which the offset of the investigated position is within acceptable limits.

1.2 Structure of the thesis

The thesis starts from a short overview of the conveyors' control issues and discusses approaches for synchronous control in Chapter 2.

After that, the description of the control system design is given in Chapters 3-6, where the design of embedded control system and its components are presented. As the main focus of the thesis is on the development of the overall control system, then the description of the carried work is divided into chapters that discuss the following topics:

1. The general description and the main principles of the whole control system organization are given in Chapter 3.
2. The most important equipment and components related to the designed system highlighted and discussed in Chapter 4.
3. After that the circuitry of the peripherals' connections are given in Chapter 5. Also in this chapter the corresponding printed circuit boards drawings, designed basing on the circuitries, are demonstrated and explained briefly.
4. In Chapter 6 the explanation of the programmable code functionality, realized by microcontrollers, is given.

Because the designed system cannot be tested at the enterprise, in Chapter 7 a simplified simulation of the investigated system operation is carried out, where a simple conveyor line

control with single stepper motor is considered. The selected simulation environment is a MATLAB/Simulink software.

The last chapter of this thesis (Chapter 8) discusses the main results and presents the conclusions of the done work.

2 THEORETICAL REVIEW

This chapter is dedicated to the analysis of the possible techniques of synchronous control of multiple conveyor lines. In this work stepper motors are used for conveyors' driving, therefore it is necessary firstly to understand how the stepper motor operates in general and how it can be controlled. Then the issues of synchronous control of multiple devices are discussed and consequently the specifics of communication between devices are described.

2.1 Stepper motor control

A stepper motor is a specific type of a brushless motor, and depending on the topology, that can be either an alternative current (AC) or direct current (DC) machine. Its main distinction from the traditional rotational machines' functionality is a division of a continuous shaft rotation into sequence of identical steps. In practice, the input signals delivered to the stepper motor are transformed into a turn of the shaft by the fixed angle value. The rotation of the stepper's rotor is done due to the sequential turning on and turning off the stator electromagnets which surround the rotor. This is depicted in Fig. 1. When the previous electromagnet goes to off state and the next one is turning on, the rotor is turned for some particular angle. As the amount of stator electromagnets is usually large, they are divided into phases where each phase includes the same number of electromagnets as the others. The frequency and characteristics of the control pulses that rule electromagnets define the rotation speed of the rotor and the value of the turn angle.

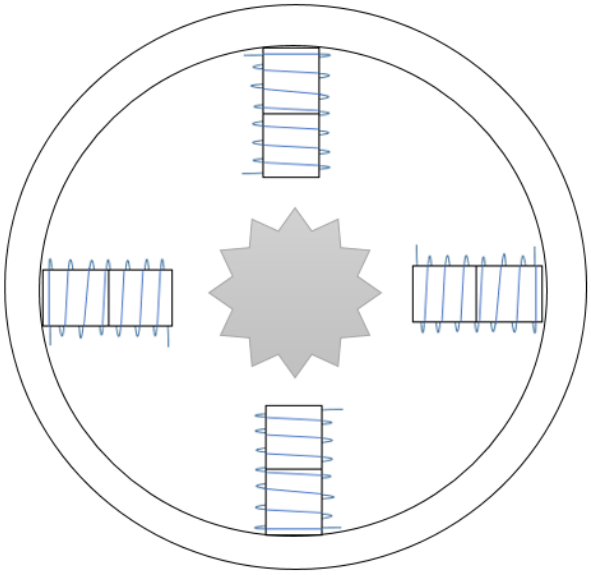

Figure 1. Simplified structure of the stepper motor.

The control of the stepper motor can be realized either by microcontroller, or directly by a driver, or by their combination. There exist some common techniques (often called excitation modes) for the stepper motor control. One of those is the wave technique that is based on the idea that at every instant only one phase is turned on. On the contrary, when the so-called half-step technique is considered, the state varies between one or two phases that turned on. When the full-step technique is applied then two of the phases are turned on at every instant. Finally, when using micro-step technique, the each phase current is not a combination of square-wave pulses, but is similar to a harmonic waveform.

In practice, the most common techniques of these are the half-step and the full-step methods as they result in the best torque performance, although they often cause motor's vibrations (Virgala et al. 2015). For the practical cases while the operating equipment is sensitive to the vibration level or higher accuracy of operation is required, the micro-step technique, which is more complicated in realization comparing to another techniques, is used. If the vibration is not such critical for the application, the designer faces the choice between full- and half-step techniques. The comparison of the implementation of these techniques indicates that the full-step method ensures higher rotor angular velocities and wider range of operation time than the half-step technique. Furthermore, the full-step technique provides higher torque value, but the disadvantage of the full-step based control method is the higher energy consumption ratio compared to the half-step one (Virgala et al. 2015).

The driving circuitry of the stepper motor is defined by the topology of its windings. The unipolar winding topology is a structure where each coil has a center tap that is connected to the voltage source's positive side and the winding terminals are grounded. This kind of winding topology, in general, requires only some basic commutation circuit, which is usually presented by diode bridge or some kind of a transistor in switching mode. The winding structure based on bipolar winding topology is done without any middle tap on any winding. This structure often requires more complicated commutation circuit and is usually realized by implementation of H-bridge.

In past most of the stepper motor control did not use any feedback, i.e. it was based on open-loop control. The open-loop control methods are still in use, especially in some simple practical cases that are not followed by some strict system or process requirements. The open-loop stepper motor control approach is being improved nowadays and the latest results show that open-loop control can also provide the performance with different options for deceleration and acceleration and also ensures a small error value (Zhang et al. 2019).

Although open-loop control shows quite good results in most applications, during the last years research related to the closed-loop stepper motor control have been carried out in order to improve the performance of their usage. The closed-loop control of the stepper motor can be obtained by considering standard cascade PI-control and also by more advanced approaches such as artificial intelligence or fuzzy methods. The controller parameter selection of these approaches are typically based only on the motor characteristics (Marufuzzaman et al. 2010; Son et al. 2015; Jung et al. 2011).

The actual research are focusing on such improvements in the controller parameters selection so that is directed by operational parameters related to the process. Different control structures has been proposed in the covering literature. For instance, by introducing a cascade of current and position control combined with a damping control for hybrid stepper motor, investigation resulted in increase of the motor's operational bandwidth and sufficient decrease in motor's vibration. This kind of structure was discussed in (Le et al. 2017).

The literature dedicated to the stepper motors has also focused on the issues related to the type of connection between stepper motor and its driver. In the most solutions it has been made traditionally by use of wires and cables, but in some cases wireless connection can be successfully implemented (Li et al. 2014). Such solution is ideal in cases where the physical

access to the stepper motor from a driver or a microcontroller is difficult. Naturally, the choice of the control technique and connection method is usually based on the complexity, technical requirements, physical location and other specifics of the investigated system that includes a stepper motor.

2.2 Synchronous control of multiple devices

The task of the multiple devices' synchronous control in a whole comes down to the necessity of the following problem solution: how to translate the control law or some common variable value to peripherals at unit instance and therefore to ensure the simultaneous fulfilment of the transmitted commands or variable processing by all the peripherals. Thus, in general, the specific end point of such regulation (PCB's movement by a conveyor line in the current study case) does not play significant role in the entire synchronization process because the objects which can be controlled directly and synchronously are not the transported PCBs or the conveyors, but only the motors which set the conveyors in motion. In other words, the synchronous control in this case can be studied not in a narrow scope of the particular conveyors synchronization, but in wide terms of the stepper motors synchronization itself, ignoring the particular conveyors' specifics. Moreover, the fact that microcontroller does not communicate with motor directly but does it through the driver, allows not to pay high attention to the particular physics of the stepper motor functionality. This gives the possibility that the stepper motors are not considered as a such and the synchronization techniques developed for another motor types can be analyzed.

The most common solution to introduce the synchronous controlling system is based on standard master-slave communication. This topic is further described in the next sub-chapter. The main idea of the master-slave communication is the simultaneous delivering of the control signals from some central control unit to all the equipment connected to it.

In general, the selected control technique for the entire process naturally depends on the systems' requirements. In the simplest case of the single input and single output (SISO) plant's control, the traditional way of control is obtained by use of feedback from the controlled variable. This way the controlled variable is regulated according to the error between the given reference and the measured actual values.

While considering the synchronization problem with multiple subsystems, the simplest synchronous control is based on traditional technique where the given command is spread to the multiple controlled devices. This approach is depicted in Fig. 2, where the reference value *a(t)* is sent to the master microcontroller from outside (by a line operator or from the external control loop). The master controller (the red block in Fig. 2) transforms the received signal into reference signal *r(t)*, which is then sent synchronously to all the slaves' controllers. After that the error value *e(t)* is calculated as a difference between the reference value *r(t)* (received from the master) and the actual measured output value *y(t)* of the controlled subsystem. For better representation, the unit that calculates this difference value is shown as a block that precedes the controller, but usually this operation is done by the microcontroller itself.

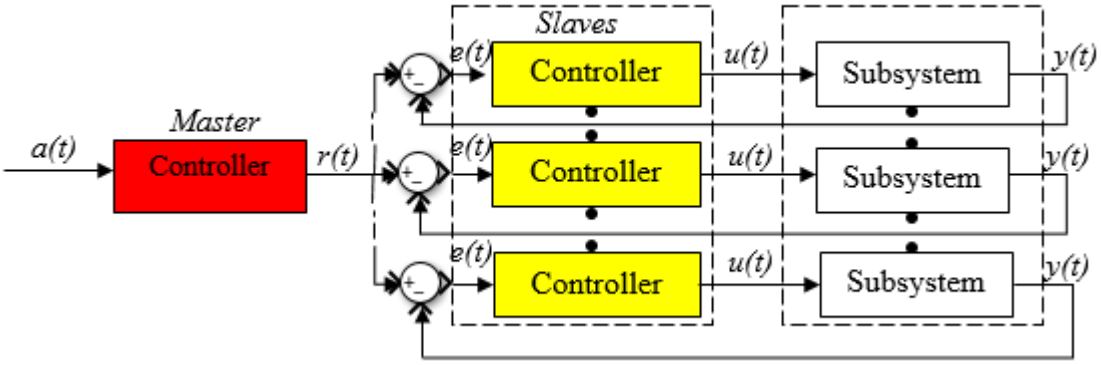

Figure 2. Traditional closed-loop synchronous control implementation.

If identical controlled subsystems are ruled by the control system in the same operational conditions and without strict operational and control requirements, then a more simplified approach can be used where the actual output value is not measured from all subsystems, but only from one of them. It is assumed in such case that other subsystems operate similarly as the one that provides the feedback. Using such approach one output measurement is used to produce the common control law which is relevant for each subsystem. The diagram of such approach is presented in Fig. 3. The control signal *u*(*t*), which is produced by master microcontroller, is adopted then by slaves for their own controlled subsystems.

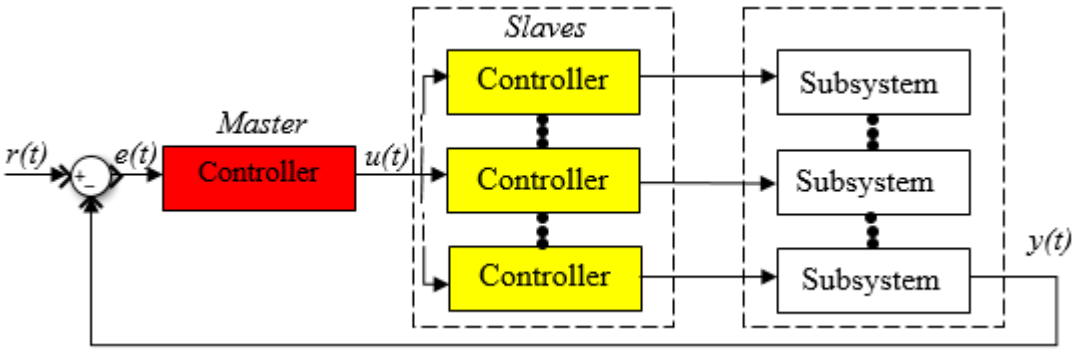

Figure 3. Simplified synchronous control for systems with low accuracy requirements.

The application studied in this thesis requires high accuracy, so the approach described in Fig. 3 with simplified synchronization cannot be considered here. Therefore, it is decided to use as a basis of the control system development the synchronization model that is presented in Fig. 2 as it is also quite simple approach to form the synchronized control signals and at the same time provides higher control accuracy comparing to the Fig. 3 case. In order to optimize this model, the following changes has been considered: the master's microcontroller not only produces the control signals for slaves, but it regulates one of the stepper motor driver directly, without any intermediate slave device. The block diagram of this approach is depicted in Fig. 4, where the red block represents the master controller that regulates one of the motors and at the same time it provides reference signals for the slaves. This solution provides the decrease of the slave microcontrollers needed for system implementation by one unit. Moreover, it also results in more efficient usage of the master microcontrollers computational capacity.

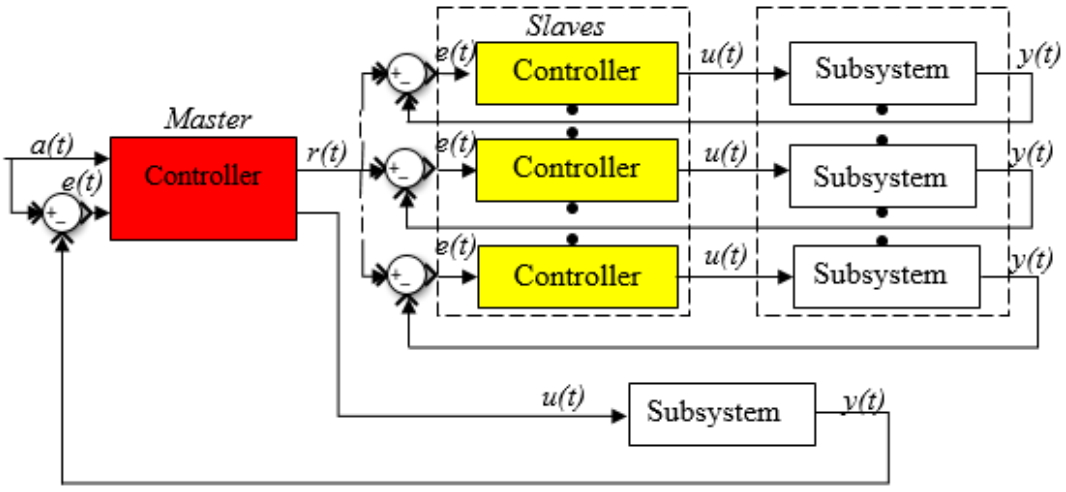

Figure 4. Block diagram of the synchronous control system considered in the thesis.

Generally speaking, the calculated error $e(t)$ is transferred to the controller that derives the control law, which can be for instance a combination of proportional, integral or derivative parts depending on the required control accuracy and other parameters. The gains of these controller parts are defined and set to the controller before the start of the system's operation; usually they cannot be changed during the system exploitation. After operating the error value, the controller forms the control signal $u(t)$ depending on the control law. In this thesis the microcontrollers that regulate the stepper motor drivers and receive the reference value from the master microcontroller are treated as slaves. Therefore, while these slave microcontrollers receive the reference signals from the master, they form their individual control signals based on the calculated error between the reference and the actual measured value as is seen in Fig. 4. Then the slave microcontrollers' output signals are delivered to their peripherals, which are presented by stepper motor drivers in the investigated system.

In the literature, the modified variations of this traditional technique has been proposed. For example, a fuzzy-adaptive method that is illustrated in Fig. 5 provides a possibility of dynamical self-tuning of the master's controller. It is done by additional computation of derivatives of error signals of the slaves. The calculated error derivatives are fed to a fuzzy-logic system that calculates the controller gains based on these. The control gains defined by fuzzy-logic device are appropriate for estimated error changings. After the gains are calculated, they are adapted to the master controller by the fuzzy logic and then are written to the controller as updated settings. In (Biao et al. 2011), the comparison of traditional and fuzzy technique applied for the double-motor synchronous system showed that the fuzzy technique demonstrates faster response speed, better synchronization representation and other beneficial features. In order to increase the accuracy of the entire control system, the slaves' microcontrollers should be also equipped with such fuzzy systems. Naturally the fuzzy rule base and adaption would increase the control systems complexity and might require extensive tuning; therefore, such approach should be implemented only in case of necessity and dissatisfaction in other techniques usage results for the particular control object.

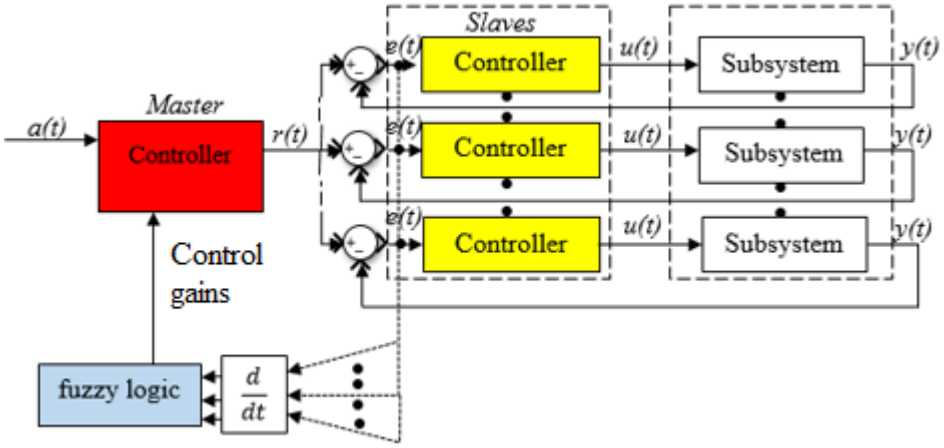

Figure 5. Fuzzy control implementation.

By using microcontrollers and by considering their computational capacity, it is relatively easy to implement both traditional, fuzzy or some other specific control technique. Each control method can be implemented by using combination of standard control parts inside the microcontroller program. The current work does not require complicated control, for instance by the use of fuzzy or other complex techniques, so it is decided to base the control system on the model which is presented in Fig. 4. This approach provides quite good accuracy with rather simple practical implementation requirements and topology.

2.3 Communication models

Different models are defined to implement the tasks' realization and communication between devices that participate in the system's operation. The choice of a model depends on the necessity of hierarchy inside the communication system, the allowed level of data access by each particular device and the need for centralization of the communication process. There exist different classifications of communication models, but it is important firstly to describe the two most commonly distinguished models.

The *producer-consumer* model defines the sequential data processing. The data goes to the operational unit from the previous one; it goes through actions that are prescribed by the current handler and then goes to the next operational stage. The actions of each unit are not defined by another ones and these units have no hierarchical division. The data processing goes from one stage to another; in more complex modifications of such model the processing can

have branches which ensure parallel data processing (for example, some amount of data can be read and stored by different utilities at the same time). Such model is suitable for quite simple data processing that does not require any centralized control and can be divided into small equal-level operations. Sometimes the *producer-consumer* model means a bit different principle than discussed here: the device named producer sent the data to the whole network independently on the presence of requests for it from the consumers' side. But despite these differences in definition, the main feature of this model remains the same. The model implements the low-hierarchical communication inside the network while the consumer devices are not strictly dominated by the producer.

The second model is the *master-slave model* that is also known in some sources – primary-secondary, source-replica, initiator-peripheral etc. model. This approach offers the opportunity of prioritizing of the processing units. The unit that is selected to be a master has a functionality of the centralized control over the whole system. The master is handling the communication process of the system. Other devices in the system are called slaves (often they are presented by drivers of peripheral equipment); they receive specific data chains and tasks from the master. The slaves operate independently from each other (in basic interpretation of this model). Each slave operation is only dependent on the master's commands. The master's behavior itself does not depend on any slaves, in other words the communication becomes asymmetrical. This asymmetry has an influence on the system behavior; for instance if the master has some erroneous settings then the slave will automatically operate with the same inaccuracy. On the other hand, the hierarchical character of such model ensures that any slave settings inexactness would not influence the accuracy of the master settings. The master not only capable of communication with determined slaves but can also set their quantity and change their functionality at any operational instance. The master-slave technology is still the mostly used communication technology in systems of different complexity level because of it centralized and strict-hierarchical nature. Often it is the best variant for the network control and diagnostics; the synchronous control of multiple devices is implemented better using master-slave technique. There are some improved models of the standard master-slave communication. For example, there are approaches with the possibility of slave-to-slave communication or systems where some different devices receive a permit for being a master for a limited time (Biao et al. 2011). Naturally, the decision of what model is used depends on the needs of the process under study.

Apart from these two described models, other classifications for communication exist. Without going into their properties in detail, here are described two important models in short:

- The *simultaneous computation* model as producer-consumer model does not have prioritization of the working units. The data flow character in this model is the following: data is initially send to all the devices that requires it. This implies the simultaneous processing of the same data by different operational units individually. After fulfilment of the processing, all results of operations under initial data come to the common output from where they can be transferred to another processing system or used in cyclic way for new iterations in the current system.
- The *client-server* model can be described as inversed master-slave model: in this case the data transfer starts only after a request for it from the client side. The other communication features resemble the corresponding ones in master-slave case.

Naturally, as in the case of any theoretical classification, the models defined above are rarely met in their standard form and one can say that the definitions of models are often not so strict. It is a common practice that for some complicated practical issues one of the described models is selected as a basis for the whole system or its part implementation, but then it is developed by introducing some features that are specific for another model. It can be easily noticed from described examples of upgraded versions of the above mentioned models that it is sometimes hard to draw a clear difference between such varied models and to determine to which particular model category the finally implemented communication system belongs.

In this thesis and in the application under study the master-slave model is used as a basis for the communication of the system. This selection is motivated by the necessity of centralized control which ensures the reduction of the system's architecture complexity. Another advantage of the master-slave technology usage in the current study framework is based on a fact that the operating conveyor lines are identical and the PCB is expected to be moved through each of them with the equal speed, acceleration/deceleration and step values on a same distance to the conveyor's border. Therefore, only one processing unit is needed to produce the common reference signals to other control devices. This way the other microcontrollers only have to interpret these signals and transfer their own generated control signals to their peripherals.

3 OVERALL DESCRIPTION OF THE INVESTIGATED SYSTEM

The system studied in this thesis includes four conveyor lines. Each conveyor shaft is driven by a stepper motor, which driver receives control signals from the corresponding microcontroller. The control law is formed by comparison of the reference value Δh_{ref} (which is set by the line operator) of desirable distance between the conveyor line's border and the edge of the PCB and its actual value Δh_{act}, which is estimated by the position sensor operation. The difference of the reference and actual signal (error) is calculated by the microcontroller and, based on the error, the microcontroller produces control signal to stepper motor driver in order to decrease or increase the motion speed. As was described before in Fig. 4, one of the stepper motors is directly controlled by the master's microcontroller. Hence the main principle of the single stepper motor control can be presented with a simplified block diagram as is shown in Fig. 6. In other cases when the stepper motor driver is controlled by the sequence of the master and slave microcontrollers, they can also be represented as a single microcontroller.

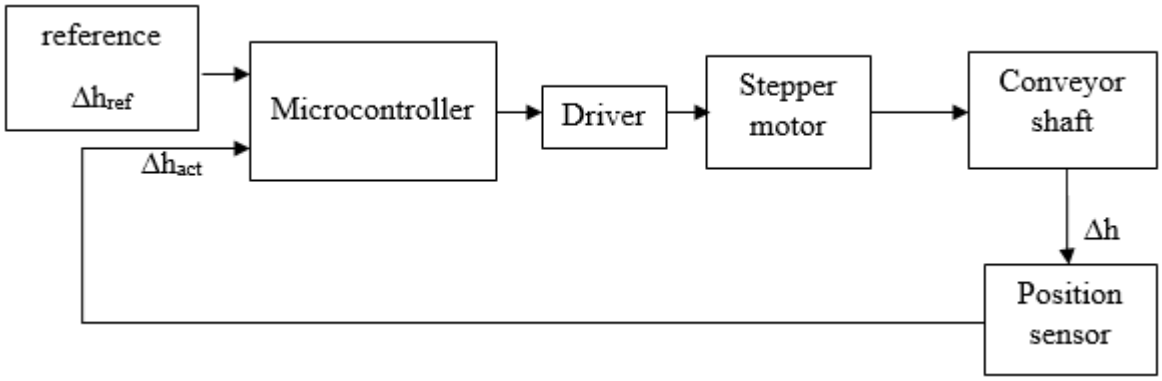

Figure 6. General idea of control law implementation.

The reference value which is received by the master's microcontroller is transmitted to the slave microcontrollers in order to use it in the error calculation and to form basing on it the control signals which are set later to stepper motor drivers. This is depicted in Fig. 7. As was discussed above, the slave microcontrollers uses the feedback information obtained from the position sensors mounted on the conveyor line. To be more specific, the slave receives the measured value from the sensor, compares it to the reference value (which is defined by the master's microcontroller) and forms the control signals for the corresponding stepper motor (which sets the conveyor lines' shafts in motion) through the driver. Also, the master produces the control signals for one of the drivers which is directly connected to the master without any

intermediate slave links. System units are connected to each other by bus as depicted in Fig. 7, where the communication between units fulfill the requirements of RS-485 serial communication standard. The slave devices get the signal from the master and adopt it for their particular control objects. Each operational unit is supplied by 5 V power source that is given by stepping down the input 12 V voltage.

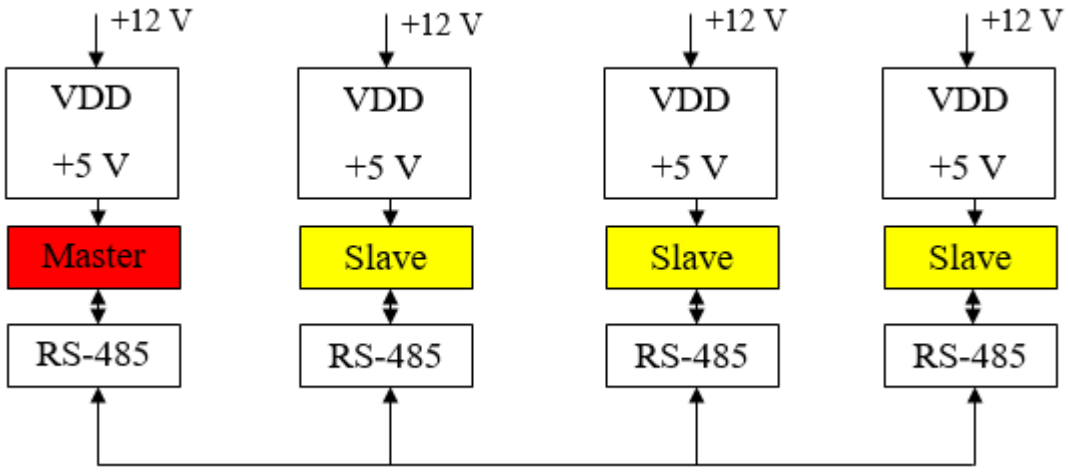

Figure 7. Model of communication between system's microcontrollers.

As all the conveyors participate in moving the same PCB in one operation cycle, then the desirable distance between the conveyor line's border and the edge of the PCB should be more or less equal. This allows the implementation of centralized control for the system and slaves' microcontrollers operation with the same software.

The master microcontroller has some peripherals for communication with external environment in addition for equipment that is used for master-slave communication (RS-485 transceiver) and connection to one of the stepper motor drivers. This is shown in Fig. 8. In this case, the actual value of the measured distance goes to the microcontroller from the position sensor. As was discussed before, the difference between the reference and actual value is a starting point for control law formation for the slaves. Moreover, in order to set a reference value that is valid for each microcontroller used in the system, the keypad consisting of five active buttons is connected to the master microcontroller. For visual interface of the inserted values a liquid crystal display (LCD) is connected to the system. It can be also optionally used for indication of error or failure warnings or displaying the results of the position sensor measurements.

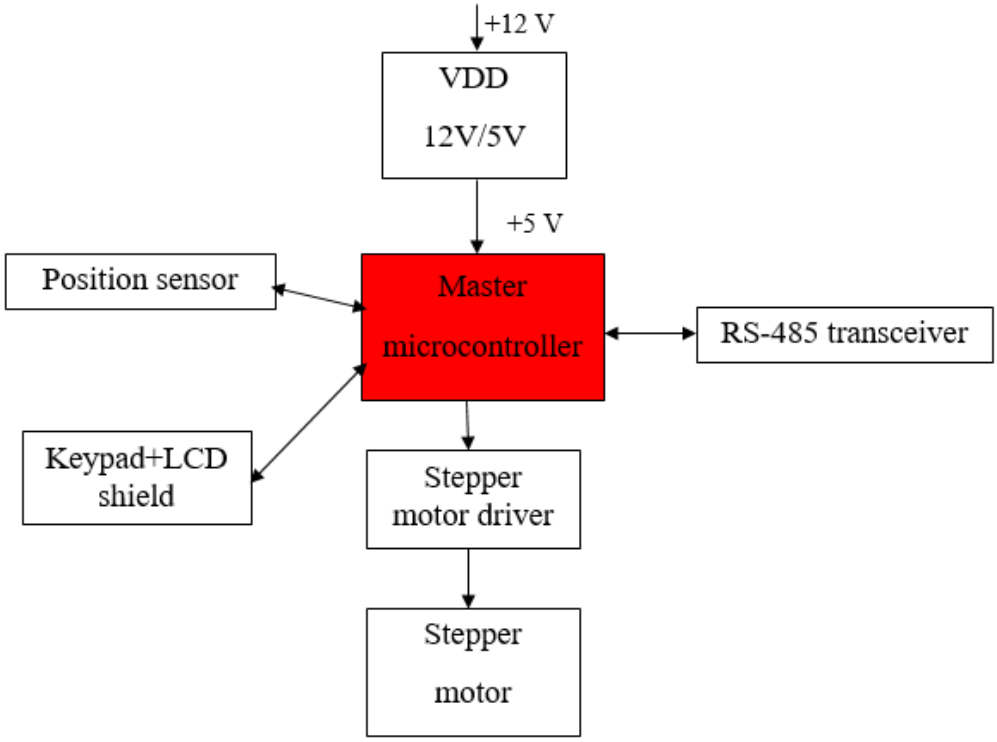

Figure 8. A block diagram of master's peripherals.

To be more specific, in the application under study a manufactured shield that combines the LCD and keypad is in use. Besides promoting more compact dimensions of the resulting control system by reducing the number of connections between the master's microcontroller and its peripherals, decreasing their length and saving the space by using of such combined module instead of separate units, it also makes the resulting project more cost-efficient. This is due to the fact that the manufactured shield is cheaper than the sum cost of the unitary pieces that are used in this module. Moreover, supported by the fact that the module is already prepared for installation and requires only connection of the necessary bonds with the microcontroller, the reduction of the time and cost consumption for design and implementation of the whole control system is performed.

The slave microcontroller peripherals are reduced in comparison to master's to only communication with the master by the RS-485 bus, with the position sensor and with the stepper motor through the corresponding driver. The block diagram is illustrated in Fig. 9. All the microcontrollers are supplied with the 5 V voltage that is stepped down and stabilized from the 12 V input voltage.

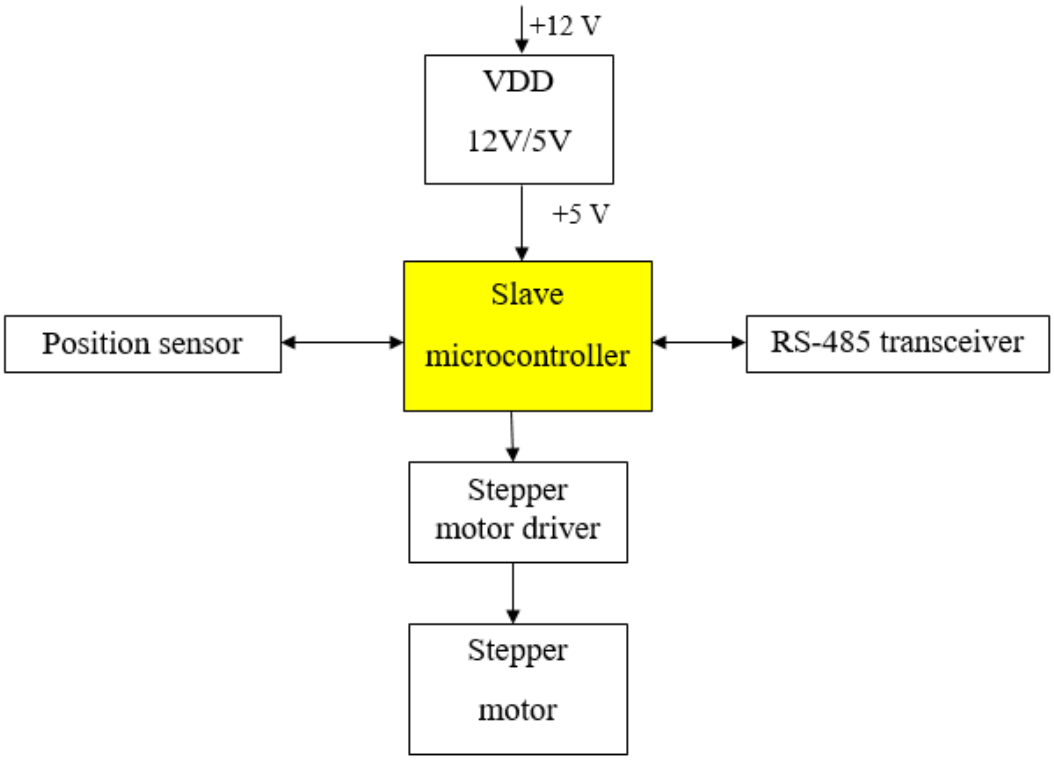

Figure 9. The block diagram of the slaves' peripherals.

4 MANUFACTURED DEVICES USED IN THE STUDY

The main electronic devices and components that are used in the designed control system are briefly described in this chapter. The description is focused on the general parameters of the equipment and the main reasons for their selection.

4.1 Microcontroller

To deal with the microcontrollers tasks the *PIC16LF15376* manufactured by *Microchip* was selected. Considering its suitability for industrial environment, its working temperature ranges lies between -40 °C and 85 °C, which matches the normal operational conditions of industrial system studied in this thesis. In order to measure the temperature of the environment, the microcontroller has also temperature-metering circuit. The operating input voltage of the microcontroller can be varied between *2.3* and *5.5 V*.

The selected microcontroller operates with a reduced instruction set computer (RISC) core, containing 49 instructions – thus providing high speed of instructions' execution with minimum instruction cycle of 125 ns (PIC16(L)F15356/75/76/85/86 2011). The core architecture is presented in Fig 10. The programming of the microcontroller is simplified by C-compiler optimized core architecture. The presence of internal timers (one 8-bit, two 16-bit and window watchdog timers) provides the usage of interrupt service routine and other timer-based operations. The implementation of pulse-width modulation (PWM) is simplified by introducing four ready 10-bit PWM modules.

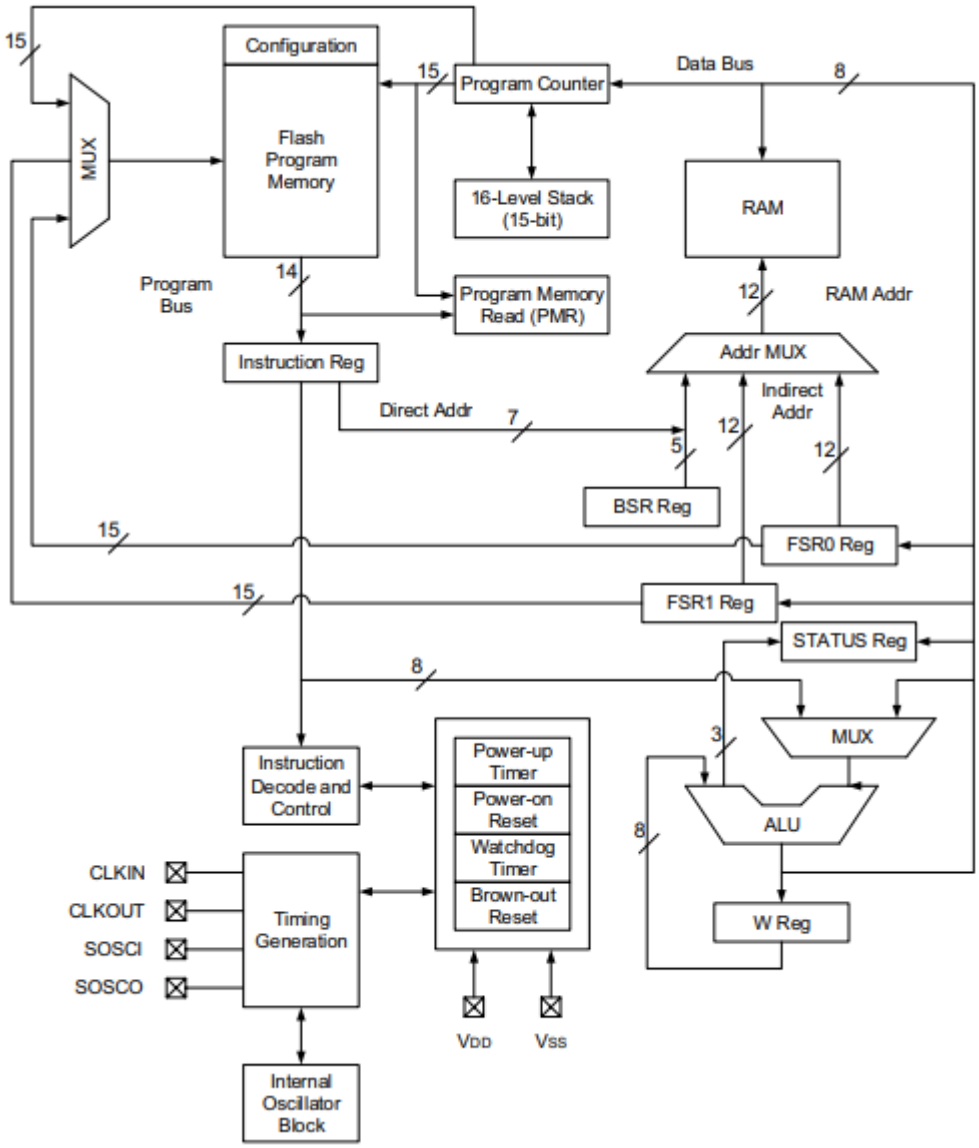

Figure 10. PIC16LF15376 core architecture (modified from (PIC16(L)F15356/75/76/85/86 2011).

For ensuring safety and security, the core is equipped with low-current, low power and brown-out reset features as well as with programmable code protection. Moreover, the large volume of the internal memory provides operations with high data amount. Mainly, the program flash memory has 28 KB volume which is the maximum value in comparison to other *PIC* microcontrollers of this series. Furthermore, the selected microcontroller guarantees the communication with peripherals based on different protocols, including RS-485, Enhanced Universal Synchronous/Asynchronous Receiver-Transmitter (EUSART), Serial Peripheral Interface (SPI) and Inter-Integrated Circuit (I^2C). For communication, the protocol can be set both to master and slave modes which is necessary in the current work. The microcontroller

provides the possibility of two parallel connections by EUSART protocol. In the current work only one of them is used to connect with RS-485 transceiver, but it is possible to reconfigure the circuitry and to make, for instance, the master microcontroller a transfer nod with connections for two RS-485 buses.

4.2 Stepper motor and corresponding driver

A stepper motor type *42HS4813A4* manufactured by *Sumtor* is used in this project. It has been designed according to the unified requirements for stepper motors, defined by NEMA17 standard. Its compact size (42 x 42 x 48 mm) allows to introduce it successfully into the conveyor line system. Moreover, the control through the driver ensures wide possibilities of setting changes.

The step of the selected motor is equal to 1.8° with allowed 5% deviation. The rated torque produced by the motor is up to 52 N·cm. The shafts' diameter is 4.5 mm, which ensures the direct mechanical coupling of the stepper motor with the conveyors' shafts that are in use in the investigated system. The stepper motor's weight is 350 g, which is quite light and guarantees the preservation of the conveyors balance after installation of the motor. The appropriate cost of the motor in comparison to other motors was another reason for its selection in order to ensure the economically successful decision.

Moreover, driver *TB6600* produced by *Toshiba* is selected for stepper motor driver purposes. The driver implements the micro-step control technique, which is the most precise one as was described in Chapter 2. The main functional part of the driver is the integrated circuit *TB6600HG*, which block diagram is presented in Fig. 11.

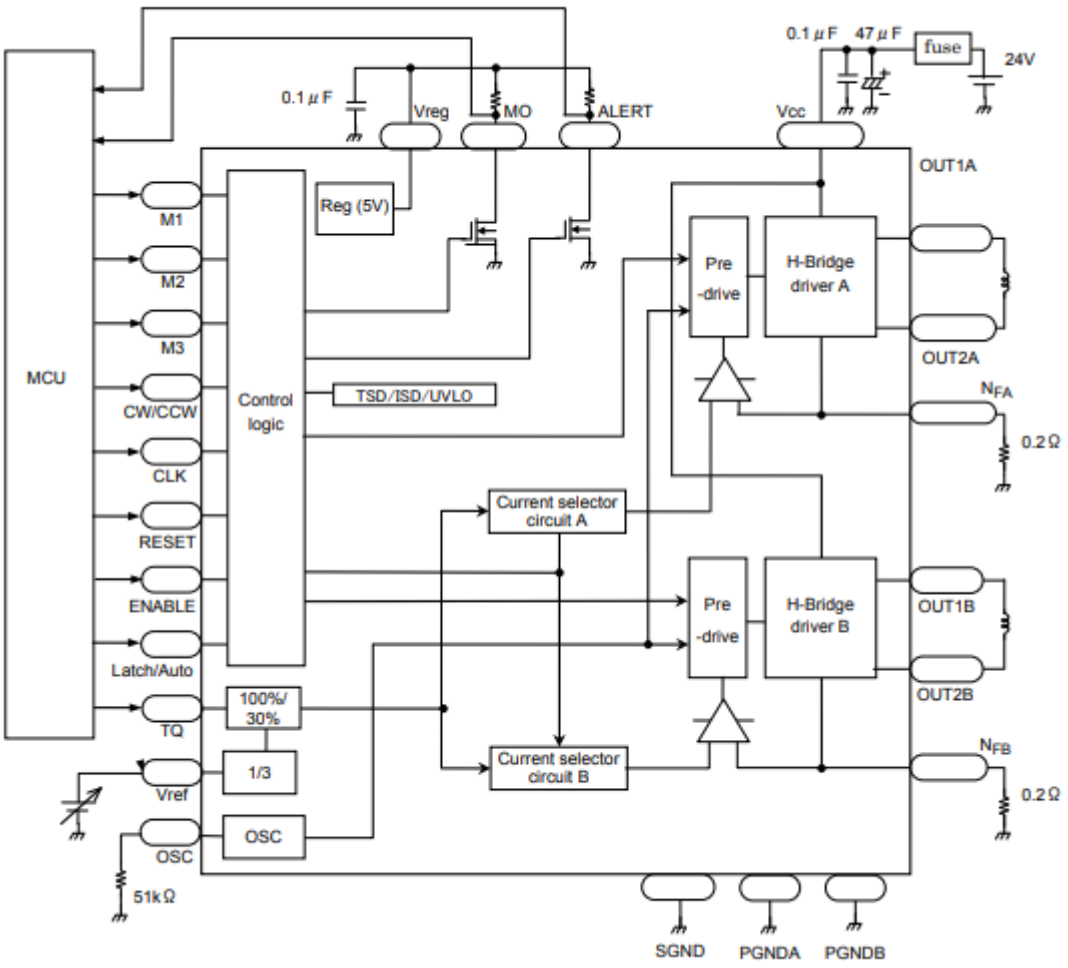

Figure 11. Block diagram of the stepper motor drivers' inner structure (modified from (TB6600HG 2016).

There is large number of ports in the driver, but only few of them are required for basic connection to the microcontroller and the stepper motor. The communication to the driver also does not require any intermediate components. As was previously mentioned for the stepper motor, also the light weight of the driver (125 g) is one of the key reasons for its particular selection. The driver is also equipped with the inner circuits for shutdown at high heating or low voltage situations as well as with the protection circuit against the current peaks (TB6600HG 2016).

4.3 LCD and keypad combined module

For error and warning messages indication and displaying the changeable reference value of the distance between the PCB's board and conveyor's edge during configuration process, the module that combines keypad and LCD was selected. Such combined shield ensures the compact and convenient allocation of the elements. Moreover it is inexpensive component for the application. Its' active buttons (which number is five; the sixth one is used for rebooting the shield) are used to increase or decrease reference values or make some other configuration changes. It is developed for convenient usage with Arduino boards, but can be adopted for work with other boards without major issues. Parallel plugs ensure the various connection topologies to make it more compatible with other equipment.

The shield is equipped with 2 x 16 indication matrix LCD; each matrix element consists of 8 x 5 elementary cells. This LCD is supplied with 5 V voltage as other equipment which is used in this work. Another feature of this LCD selection is that it does not require any complex interface to be connected to the microcontroller's outputs. The viewing area of the LCD - 66 x 16 mm without frame - ensures high visual performance of the displayed data and convenient representation for the line operator's estimation. The displayed data can be presented both by latin and cyrillic characters, making its usage universal and adaptive for utilization in different countries (WH1602A 2008). The module is also equipped with the potentiometer in order to control the contrast of the LCD.

4.4 Position sensor

The distance Δh between the PCB edge and the conveyor line's side is measured by the position sensor. The main principle of the measurement is given in Fig. 12. To realize this function the time-of-flight sensor *VL53L1X* by *STMicroelectronics* was selected. Here it is worth remarking that the time-of-flight means an approach for measuring distance between the moved object and sensor; the measuring principle is based on the speed of the light. The sensor's operation is based on the common optical sensor principles: the difference between instances of the laser beam emission and receiving by the sensitive lens is measured and then the distance between the sensor and the object is calculated with help of the constant laser beam

speed value. The sensor is mounted on one of the conveyor line's side, so the distance between the conveyor side and the PCB is equal to the distance between the sensor and the PCB.

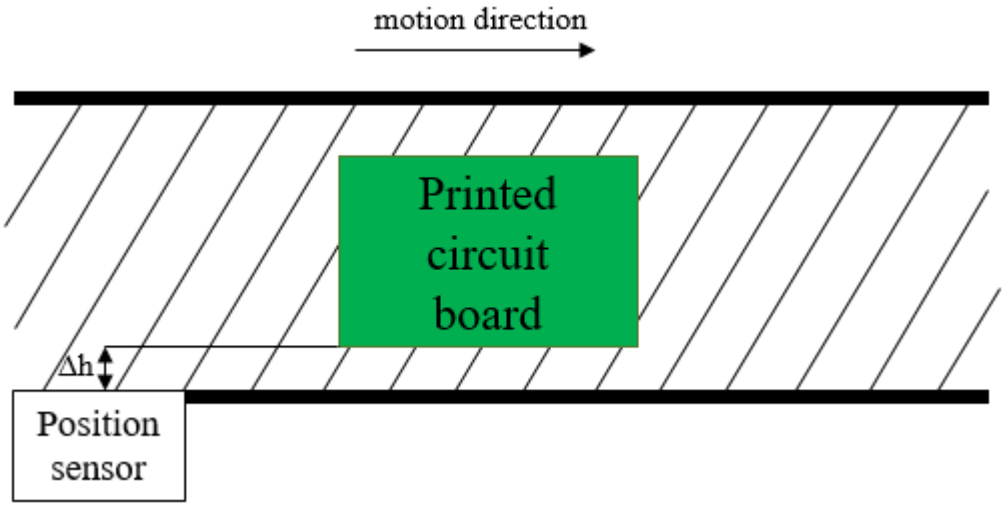

Figure 12. The principle of the position measurement.

By considering sensor's measurement principle, the horizontal orientation of the laser beam direction and the fact that the conveyor line height is much less then the average adult person's height, the possible harm of the laser irradiation for operating personnel's eyes is minimized. Moreover, the safety of the used laser is declared by the manufacturer in supplementary documentation. The compact dimensions of the sensor (4.9 x 2.5 x 1.56 mm) provide accurate installation and avoidance of disproportionality of the conveyor line after the sensor fitting. The operational temperature range (between -20 °C and 85 °C) ensures its successful usage with the PCB-mounting equipment (VL53L1X 2018).

The sensor allows to measure distance less than 4 meters precisely; as the current project intends the distances no more than tens or hundreds of mm, than this sensor is a suitable selection for the application. Moreover, the *FlightSense* technology used by the sensor provides an opportunity of measuring the distance from sensor to the object independently from its colour, brightness or other visual characteristics. It is important because of different possible colours of the PCB and reflection of its sides at different working space illumination. The colour of PCB is often green, but naturally it can be varied by the PCB's designer.

Presence of an internal microcontroller combined with three memory types – ROM, RAM and NVRAM - in structure (internal structure diagram is given in Fig. 13) allows the sensor to realize calibration, to change measurement parameters dynamically and to implement

self-tuning based on the stored data. According to the sensor documentation, the measurement accuracy on a short distance (up to 135 cm) is not drastically influenced by the inner lightness on the contrary to the medium and long distances. This is considered as another advantage of the sensor for the current project. For such distances the time needed to make the measurement is about 20 ms which is applicable for the operation of the whole conveyor line system and the control part, as the normal conveyor motion speed in the investigated circumstances is not increasing units of m/s (VL53L1X 2018).

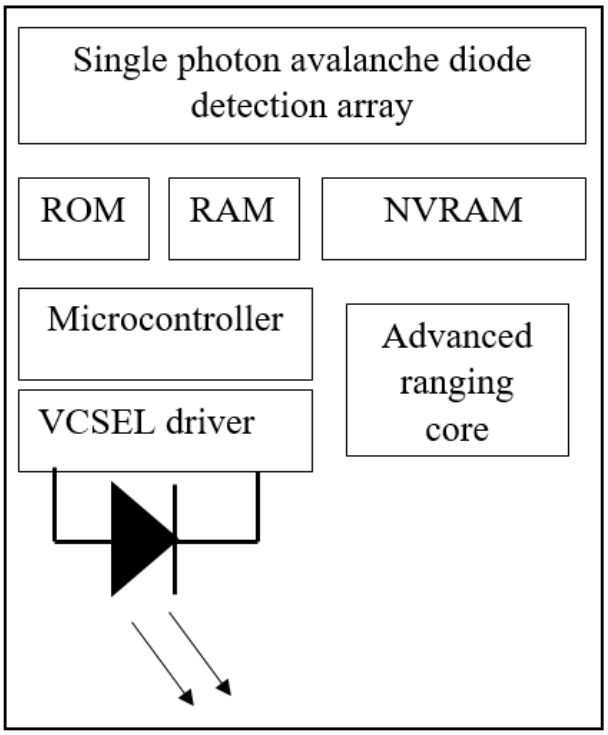

Figure 13. Internal sensor's structure (modified from (VL53L1X 2018).

4.5 Transceiver

As was mentioned before, the communication between the master and slave units is carried out through RS-485 interface. This standard is designed for setting the parameters of the receiving and transmission equipment for serial communication. One twisted pair is used for communication; depending on the project requirements, the entire system complexity and importance of delivered data, the twisted pair can be accompanied with a common wire or a braided shield. The simplicity and reliability of this standard promoted its high usage level in

different technical systems and for this reason it is a basis for many wide spread industrial protocols.

The exact data transmission is defined in RS-485 by mean of the differential signals. In case of the superiority of the inverting line voltage over the non-inverting one the operating signal is interpreted as logic unit; the opposite relation between lines' values is interpreted as logic zero. To decrease the resonant effects in the twisted pair, the terminator resistors of a resistance, equal to the cable impedance, must be installed on both sides of each communication line.

For transmission/receiving handling the *V6602 RS-485* transceiver by *Vango Technologies* was selected. This unit combines both the driver and the receiver functions; it ensures half-duplex communication (which means that at every instance the data transfer can be realized only in one direction) on the maximum speed of 2 Mbps (V660X 2016). According to the general requirements of RS-485, the manufacturer of *V6602* declares the opportunity of simultaneous usage of more than 200 transceivers on a single communication bus (V660X 2016). In the application under study, the number of such transceivers on the bus is much less than mentioned, thus this particular transceiver matches the application's needs successfully. For implementation of serial communication between operating units, the selected transceivers should be installed on each side of serial bus. In other words, four of such transceivers are required. The operational temperature ranges (from -40 °C to 85 °C) of the transceiver are also suitable for the current application (V660X 2016).

The inner structure of the transceiver is given in Fig. 14. Based on the Fig. it can be noticed that the working principle of the transceiver is simple: the delivering data are divided into two parallel branches, one for receiving - R - and one for transmission - D. Depending on the control signals from the client's side and internal polarity detection block's operation they are transmitted by one of these ways, according to required data flow direction.

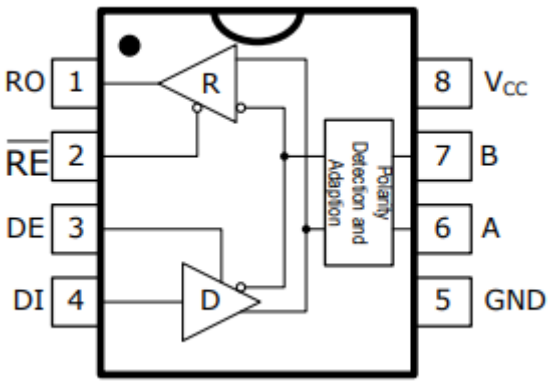

Figure 14. Transceiver's structure (modified from (V660X 2016).

The RS-485 standard does not define particular requirements for galvanic or other isolation of the communication lines, but in the application under study both the optical and galvanic isolation are implemented. The optical one is realized by use of the optocouplers, while the galvanic is introduced by use of the DC/DC converter. Such serious attitude to the isolation is dictated by necessity of ensuring the safe system operation and minimization of the harmful impact of one system part's possible failure on another system branches or utility operation. Although the transceiver has an internal over-voltage and short-circuit protection, the additional optical and galvanic isolation is required to ensure the maximum safety of the system usage.

As the transceiver operates on a 5 V power input, an intermediate stepping-down DC/DC converter between microcontroller and transceiver must be installed. To fulfill this requirement the DC/DC converter *AM1S-1205S-N* by *Aimtec* was selected. Besides of its main function – decreasing the value of DC voltage, it also provides a mentioned above possibility of additional galvanic isolation between the microcontroller and other connections on a PCB. Adding some filtering components on the output stage of the converter helps to decrease the output voltage ripple.

5 TOPOLOGY DESCRIPTION

In order to implement the overall developed control system, three PCB's types should be designed and manufactured for the system under study. The first one is the position sensor board that should include the actual sensor, the plug to receive the supply voltage and to communicate with the microcontroller and capacitors for the internal laser's supply voltage filtering. The second board type is the master's microcontroller board for which the plugs for supply voltage and communication with peripheral devices should be presented as well as RS-485 transceiver and a number of filtering and pulling-up elements in order to maintain the required level of the exploited voltages and currents. Finally, the third board type is the slave microcontroller board that is almost similar as the master's one but with removed connector for LCD-keypad shield as it is out of use in slave case.

As the slave and master topologies differ from each other only by absence or presence of the connection to the LCD-keypad shield, then it is decided not to develop a separate topology for the slave microcontrollers. In practice during the components' mounting the only difference will be that the corresponding connector will not be mounted on the slaves' boards and consequently the pattern for this element will be left free.

The connection between the stepper motor and its' driver is realized directly, as presented in Fig. 15. From microcontroller the driver receives the signals that enables the motor's operation, set the direction of the shaft rotation and the step realization order (*ENA*-, *DIR*- and *PUL*- respectively) and after their processing sends control signals to the stepper motor's phases *A* and *B*. Also, the driver receives 5 V supply voltage from the microcontroller's side for the internal microchip, and a direct voltage (in ranges between 9-40 V) from external source for supplying the stepper motor phases.

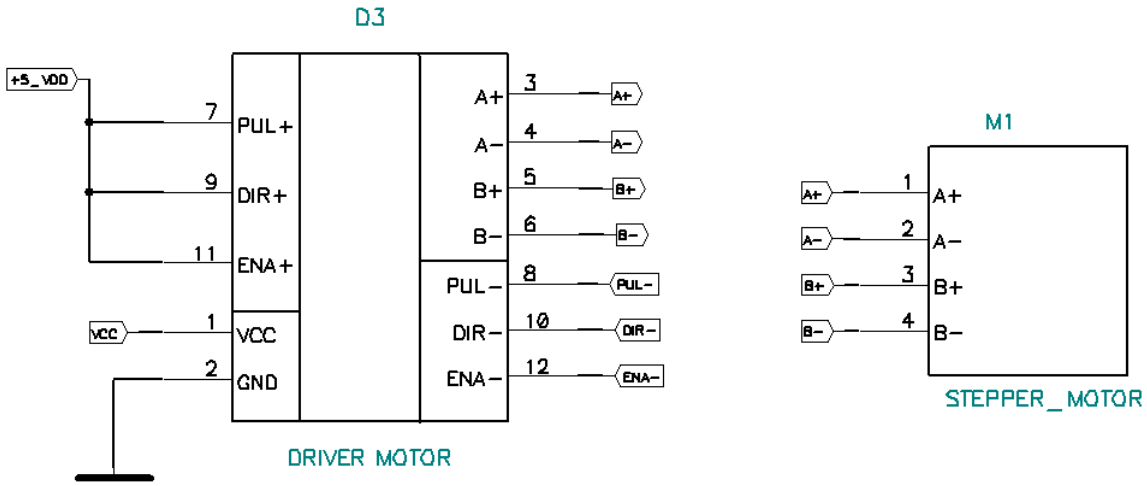

Figure 15. Stepper motor and driver connection.

In the design process, all the circuitry is done according to the recommendations and the requirements, described in the components' documentation. The topology drawings were made in P-CAD Schematic environment, while the printed circuit boards topologies were designed by the use of P-CAD PCB software.

5.1 Master microcontroller topology

The connections of the master's microcontroller are shown in Fig. 16. The supply is received by a *VDD* port and is grounded from the *VSS* with a parallel filter capacitor of 0.1 μF capacitance value and critical voltage of 16 V.

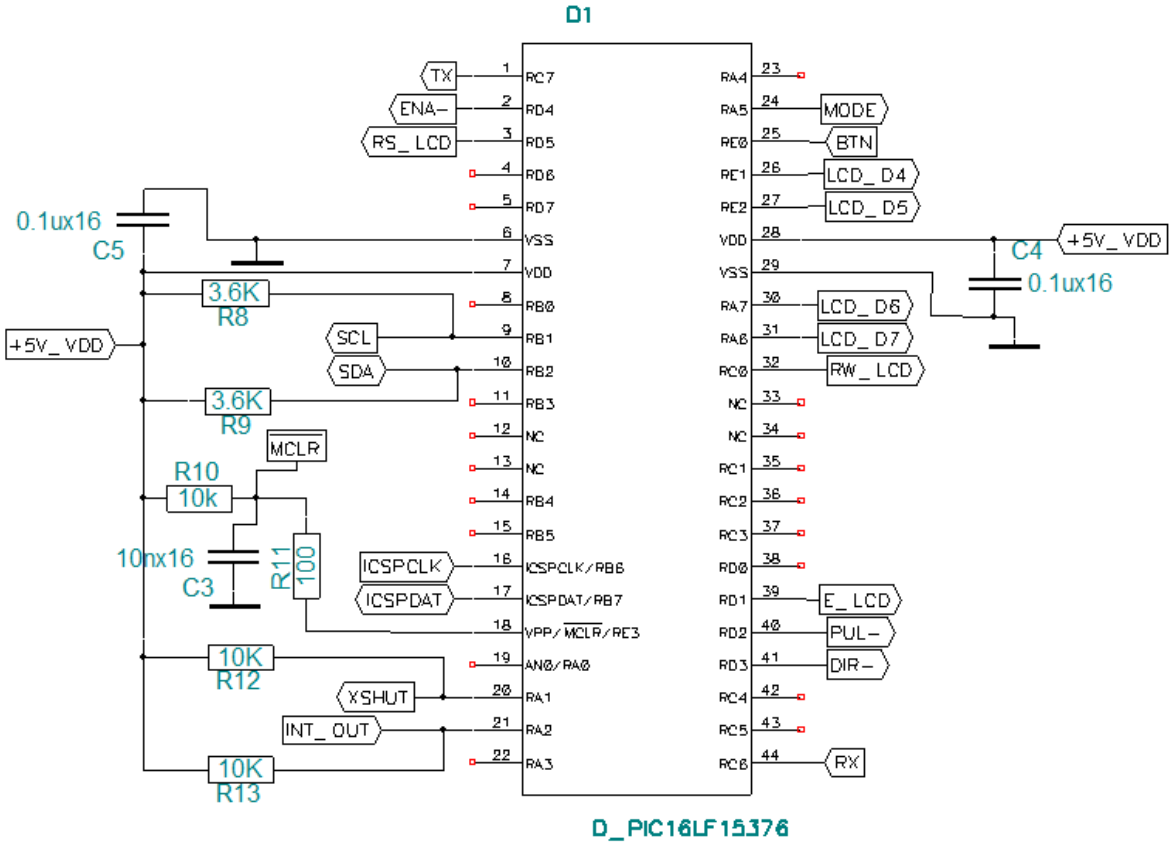

Figure 16. Master microcontroller schematic.

The programming of the microcontroller is possible by the special programmer software, designed for PIC microcontrollers. It requires six-pad pattern, which connects with the *ICSPCLK* and *ICSPDAT* ports of the controller to set the clock pulses and transmit programming data respectively; also it requires the voltage supply and connection to master-clear port ($\overline{MCLR}$).

When looking the connections given in Fig. 16, the following inputs/outputs are used for communication with the position sensor. The port *RA1* is used for setting the shutdown digital signal for the position sensor and on the contrary the *RA2* port is used for receiving the interrupt signals from the position sensor. The ports *RB1* and *RB2* are used for the serial communication with the position sensor. To be more specific, they are used for setting the clock pulses sequence and data transmission respectively. These ports are initially designed for I^2C or other serial communication, so these ports are strongly reserved for these purposes in this project.

The communication between the microcontroller and the position sensor is done by I^2C serial bus. It consists of two lines: for transmission of data and clock signals. Both lines require pull-up resistors for the supply voltage, which resistance depends from the line capacity and operational voltage. In general case, the pull-up is necessary to implement only once for each line and the corresponding resistors are mounted as closer as possible to the control device (microcontroller). Each device that is connected to the bus can receive or transmit the data by the data line (*SDA*), but only the master device can set the pulses of the clock line (*SCL*). As in traditional master-slave communication model, the transmission is initiated by the master and the communication between slaves is impossible.

When looking microcontroller schematic in Fig. 16, the communication with the LCD-keypad shield is done through the ten-pin plug, that is connected to the following microcontrollers ports (despite voltage supplying source and ground nets). The *RC0* is used for setting the reading or writing operating mode of the LCD and the *RD1* for enabling the LCD. Moreover the *RD5* is applied for sending to LCD the signal which defines the transmitted bites type: data or commands, while *RA6, RA7, RE1* and *RE2* are needed for sending the signals to switch on or off the segments of the LCD in four-bit mode. *RE0* is used as microcontroller's input that receives the buttons' press data.

The communication with the stepper motor driver is done by the *RD2-RD4* ports and they are used for sending to the stepper motor driver the step realization, movement direction and enabling signals respectively.

The microcontroller communicates with RS-485 transceiver by the following ports: the *RA5* is used for setting the direction of the communication (transmitting or receiving mode) and the *RC6* and *RC7* are used for delivering data to or from the microcontroller from/to transceiver, respectively. The transceiver circuitry is presented in Fig. 17. It has optical and galvanic isolation from the other connections on the board in order to secure the transceiver from possible critical voltage peaks on the microcontroller side and vice versa. Through the terminal block *XS2* it is connected to the communication bus. The ports *A* and *B* are connected to the inverting and non-inverting bus lines and are terminated by the 120 Ω resistor (standard value for such connection) in order to prevent reflection of the transmitted signal. The ports *DI* and *RO* represent the poles of the transmitter input and receiver output and on the contrary the *DE* and *RE* ports are designed for receiving microcontroller signals that set the transceiver into

transmitting or receiving modes. The opposite meanings of these modes lead to reduction of two necessary mode signals to one and possibility of its connection to both mode inputs. This solution is motivated also by the inverting character of the *RE* port. Thus, when the equal signal is given to both ports, the second one is interpreted as opposite to original and ensures this way the correct setting of the current operational mode of data transmission.

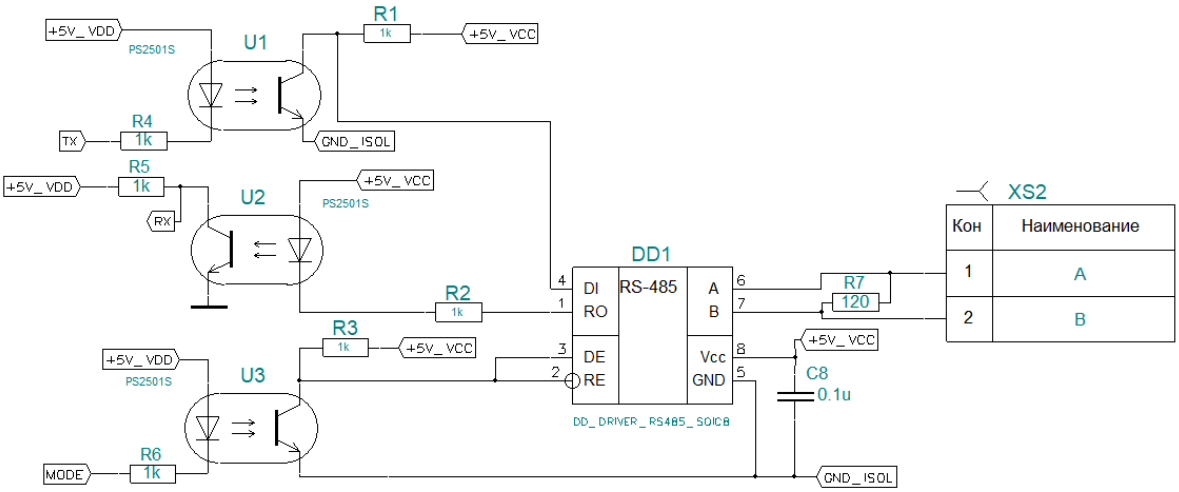

Figure 17. Transceiver's circuitry.

The microcontroller, position sensor, LCD-keypad shield and the "left" optocouplers' sides receive 5 V from the voltage regulator, which circuitry is depicted in Fig. 18. Besides the stepping-down the voltage from 12 V to 5 V, it also ensures the galvanic isolation of input and output voltages. The "right" optocouplers' sides are supplied with the same regulator surrounded by the same filtering components in order to isolate fully the transceiver from the entire circuit; it produces the *+5_VCC* output signal. The values for the filtering capacitors and inductance are selected according to the converter' supplementary documentation's recommendations but they can be increased (AM1S-N 2012).

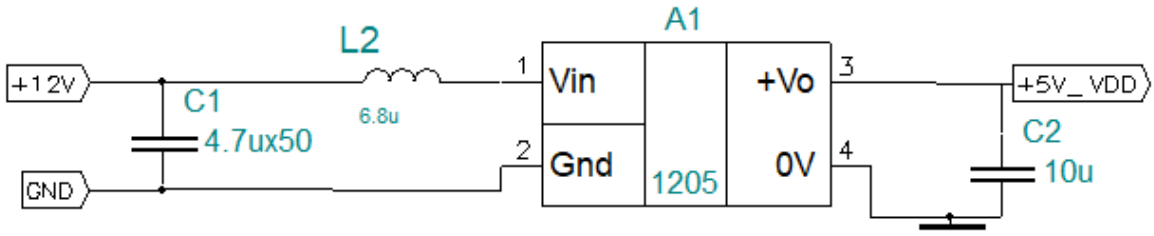

Figure 18. Voltage regulator circuitry.

Next the PCB containing the microcontroller was designed according to the circuitry presented above. Its topology is shown in Fig. 19. In order to make the topology easier for visual perception, the following indication colours were selected. The connections or their parts which are located in the upper layer of the PCB are marked with red and ones that are located in the bottom layer - by green. The vias (through holes) are marked with blue while the outlines of the components that are located on the upper layer are marked with brown; components that are placed on the bottom layer are indicated with grey. The main components are labeled in Fig. 19; the microcontroller is marked as PIC.

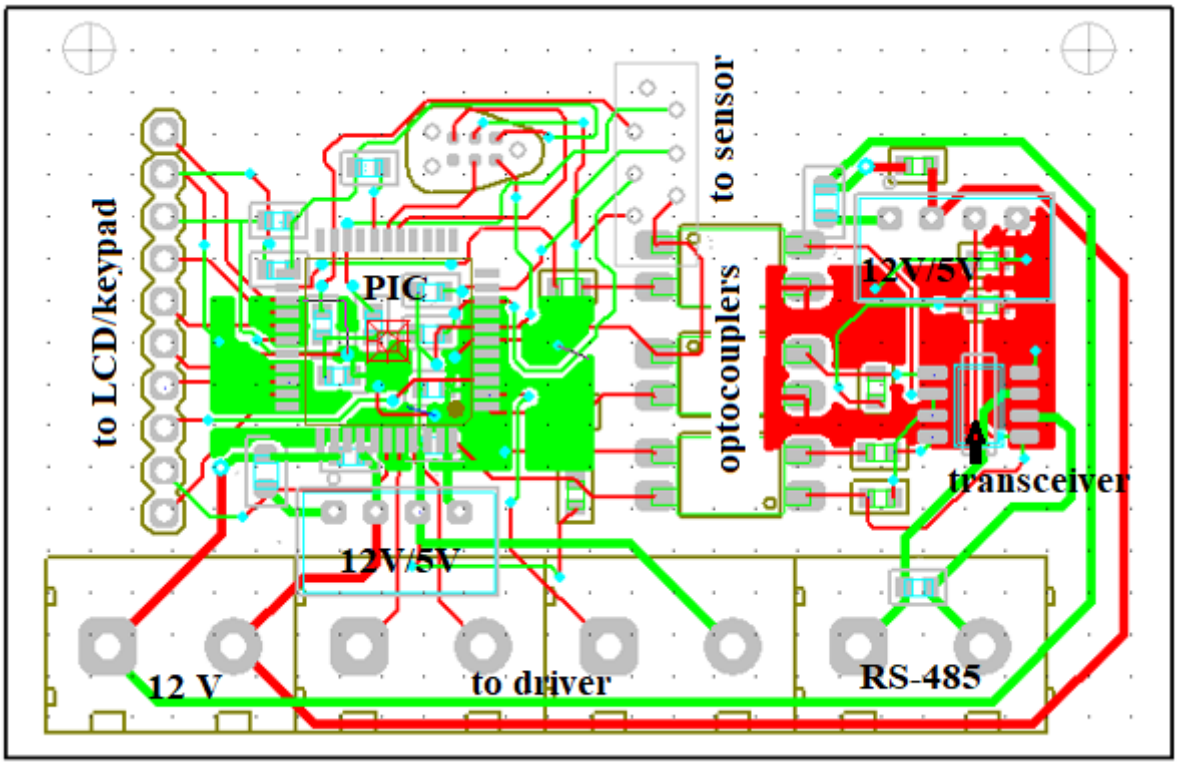

Figure 19. Microcontroller PCB's topology.

The designed PCB has 70 times 45 dimensions and has two mounting holes for further fixation by M3 screw. During the PCB's design it was decided to place the most large-dimensioned elements on the one PCB's side except the DC/DC converters and position sensor's connector in order to ensure the weight-balanced and compact components' location on the board's surface. This positioning also supports PCB dimensions' minimization.

The default conductors' width is set to 0.25 mm because of large amount of surface-mounted device (SMD) components with small pads in use. Such conductor width's selection provides reliable contact between pads and conductors. Only the connections between the

power supplies and filtering capacitors are done with 0.5 mm width to minimize the active losses in conductors as the internal resistance decreases with the cross-section area's increase. For such enlarged conductors the enlarged vias are designed (internal/external diameters relation: 0.457/1.016 mm, comparing to the default case: 0.3/0.6 mm).

Special attention was focused on the avoidance of parallel and long conductors' design in order to prevent the electromagnetic noise appearance. Such conductors were made only while it was the one possible solution, for example, to supply the RS-485's side DC/DC converter. The amount of such conductors is small so they do not have a serious impact on the designed PCB's operation.

When focusing on the PCB's area minimization, all the resistors, capacitors and inductors were presented by SMD components of 0603 typical size. That also ensures the more compact components disposition because of through holes absence. For instance, all the capacitors and resistors, which filter and pull-up the microcontroller's connections, are located on the opposite PCB's side directly behind the microcontroller providing thus the best possible quality of the supply and output voltages.

The components that are used for communication by RS-485 standard with the other microcontrollers (optocouplers, transceiver, DC/DC converter, bus connector and side elements) are grouped near each other on the right PCB's side in order to ensure the galvanic isolation from the other PCB's areas and to simplify the topology. Also the close location of these provides minimization of transmission losses which are of critical importance in transmission process.

Terminal blocks which are used for grid voltage supply, communication with the stepper motor drivers and other microcontrollers by the RS-485 bus, are grouped on the bottom of the board and connected with each other. Such connection is promoted by the terminal blocks construction: all of them have grooves and protrusions on their sides.

Connector for the position sensor's board and the programming connector are mounted on the opposite sides of the PCB in order to ensure the microcontroller's reprogramming during the system's operation without turning-off. In case if it was not done this way, the connection plume would not allow the access for the programming pattern and the sensor's connector would be extracted from the socket. The latter is not desirable as the socket is sensitive for any extraction and becomes quickly wearing out.

The ground connections were made not only directly, but also by the copper polygons usage: the first one was made in the bottom layer and is connected to the microcontroller's side filtering capacitors, while the second polygon was designed in the upper layer and is connected to the RS-485 equipment that needs grounding. This polygon is done separately in order to also ensure the isolation of the ground layer.

Based on the discussion above and the justified selections, the PCB fulfils all the requirements for its construction. All the connections meet the electrical and usage safety regulations, hence the PCB can be treated without any harm for the user and environment.

5.2 Slave microcontrollers' topology

As was mentioned above, the slave microcontrollers' topology is designed similarly as the master's, but without the connection to LCD-keypad shield according to the general idea of the implemented control system. Therefore, the number of the microcontroller connections is reduced, thus resulting in the reduction of the boards dimensions.

But in order to simplify the PCB's manufacturing and components mounting it was decided to use the same board both for the master and slave microcontrollers. During the components mounting the pattern for the corresponding connector must be left unused in case of slave boards development. The functionality and nominal values of the remained connections and components are the same as for the same ones on the master circuitry and do not require any changes.

5.3 Position sensor topology

The circuitry for the position sensor is made according to the requirements and recommendations, mentioned in the sensor's supplementary documentation. The full sensor schematic is presented in Fig. 20 where the main connections and ports can be seen.

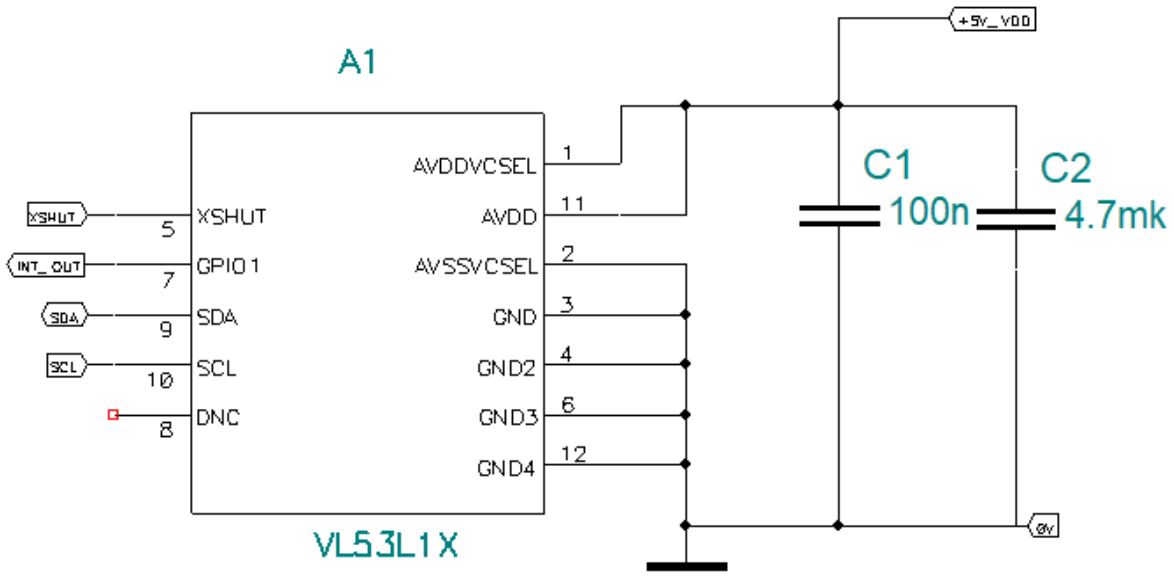

Figure 20. Position sensor schematic.

The sensor obtains the information and supplying signals from the microcontroller through the six-pin connector; the communication uses a plume as physical medium. The serial I^2C -communication between the sensor and microcontroller is done by use of four digital inputs and outputs on the sensor side seen on Fig. 20. The *XSHUT* is an input used for the shutdown of the sensor operation. Therefore, in normal working state this input is zero. The G*PIO1* is used for delivering the interrupt signals from the sensor to the controller while the *SDA* is the actual input and output port for the communication, i.e., the data bus is connected to this port. Finally, the *SCL* is an input for the clock bus.

When looking the technical requirement of the sensor, the *DNC* port is not needed and thus it is not considered in this work. The four mentioned input/output ports are equipped with pull-up resistors to ensure the correct digital values transferred to these ports. The serial communication ports *SDA* and *SCL* requires the pull-up resistors of 3.6 kΩ according to the low I^2C load capacitance. The shutdown port is supplied through the 10 kΩ resistor according to the sensor documentation's recommendation (VL53L1X 2018). It is done to prevent the possible current leakage in case of undefined microcontroller state. Due to recommendations that are given in the sensor's technical documentation, all these pull-up resistors are located on the microcontroller's side in order to avoid the signals' distortion (these are *R8, R9, R12* and *R13* in the microcontroller circuitry, seen in Fig. 16) (VL53L1X 2018).

The ports *AVDDVCSEL* and *AVDD* seen in Fig. 20 provide the supply for the laser emitter and the inner operation equipment, respectively. Therefore, they are connected to the common supply through the filtering capacitors which are selected according to the sensor documentation. These capacitors should be located as close to the mentioned ports as possible (VL53L1X 2018). All ground ports including the *AVSSVCSEL* are grounded together.

After finishing the development of sensors connection, the corresponding PCB's topology was designed in the form that is ready for sending to the PCB's manufacturing enterprise as a complete order. The following practical requirements were given for the board's development: the board should be square, its side should be as small as possible because the sensor's dimensions are low and the PCB has to be mounted on the narrow side of the conveyor line. Also the board should have mounting holes for M3 screw in order to be mounted on the conveyor's side.

The designed PCB's topology is shown in Fig. 21. It has a side length of 17.5 mm value, which fulfils the announced design requirements successfully. The components placement is done according to all the necessary requirements for the PCB's development and also in order to minimize the occupied board's area. The position sensor and connector to microcontroller ("to PIC") are labeled on Fig. 21.

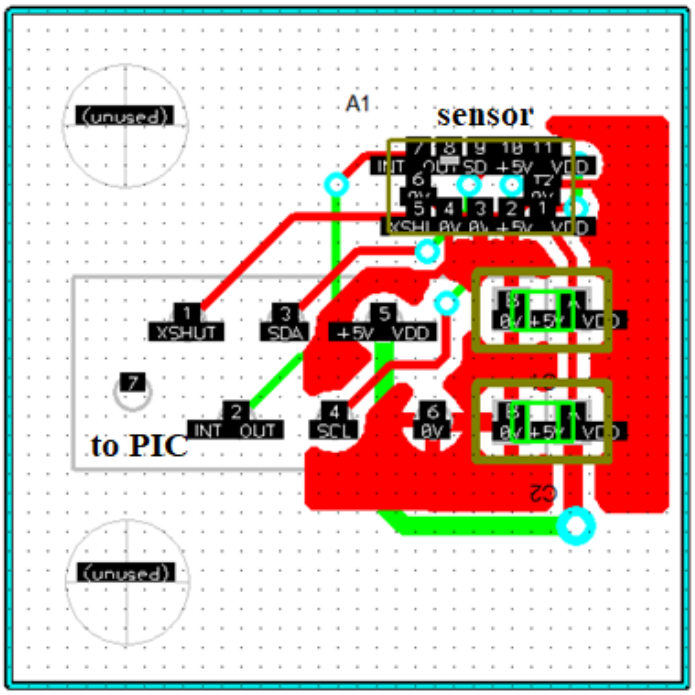

Figure 21. Position sensor PCB's topology.

For better visual representation of the topology in the current paper the marking colours were changed the following way: the top and bottom layers' connections are indicated by red and green colors; vias are marked with blue and packages of the components that are located on the top board's layer (filtering capacitors and position sensor) are marked with brown color. The connector on the bottom side is marked with grey color.

As for the microcontroller's PCB, the presence of large amount of SMD components comparing to the through-hole components number dictated the necessity of the 0.25 mm width conductors' usage. Only the conductors, which connect the supplying ports with the filtering capacitors, are done with 0.5 mm width. Thus they are accompanied with the enlarged vias (the relations are the same as in case of the microcontroller's PCB). Such increasing of the power connections' dimensions is also caused by the fact that the mentioned supplying voltage is used on the microcontroller's board by a number of components, therefore the doubts about the power losses becomes even more critical. The connections were made in a manner that promoted the avoidance of long parallel conductors. Such cases are rare on this PCB and are done only when it was the one real possibility to route the elements. As the PCB's dimensions are small, then the undesirable electromagnetic effects are not strong.

The connector and the sensor are mounted on the opposite sides of the PCB in order to prevent the overlap of the sensor by the communication plume. The filtering capacitors are mounted according to the recommendations in the supplementary sensor's documentation as close to the supply ports of the laser as possible. The copper pour is done for the ground net. The board is mounted to the inner frame through two holes designed for M3 screw.

6 MICROCONTROLLERS' CODE EXPLANATION

This chapter introduces the microcontrollers' programming principles. The developed code is necessary for successful system's operating and has to be loaded to the microcontrollers memory, both to master and slaves with some corresponding minor variations. It is worth mentioning that the entire codes are not presented here due to long code length and the fact that they are almost done based on the special functions that are contained in universal header files, which are designed for simplifying the procedures of coding the communication between microcontrollers with its peripherals. All the header files are borrowed from the open internet resources. The communication with other peripherals is based on the header files' functions and basic programming structures (loops, cases, conditions etc.), so it is enough to understand the codes' representation basing only on the verbal description.

6.1 Control signals' formation

The principles of control signals' formation differ in master and slave cases slightly. The common factor is the receiving of the position sensor measurement of the PCB's offset. The main controller's routine is interrupted by the sensor interruption signals, which indicate the appearance of a new measurement results. The data transmission is realized by the I^2C serial bus; the microcontroller forms the clock pulses sequence which accompanies the actual data transfer. The received data are compared then with the conveyor belt width in order to filter the false alarm measurements.

Then the verified data are transferred for the next function block, which builds the control law based on it. For this purpose, the microcontroller requires the reference value of the controlled variable. This reference value is interpreted as maximum allowable offset of the PCB location while it is moved by the conveyor belt with some fixed percentage deviation. The reference value is set by the buttons, included in the LCD-keypad shield, and it is transferred to the master microcontroller by the serial protocol. The microcontroller-shield interface defines the default values of the reference and its' increment/decrement step. The communication process between the shield and microcontroller is explained in more detail in section 6.4.

The received reference value is broadcasted then simultaneously to slaves by the master microcontroller. The slaves are using PID-functions to form the control signals, which are needed for keeping the offset value below the defined reference range.

The forming of the control signals is the most important process realized by the developed system. Therefore, the corresponding function of control signal derivation was done aa a reentrant function, providing thus avoidance of the share data problem. The values of the variables used by the control function are not stored inside the function and are not set to it directly, but by use of pointers to registers where the mentioned variables are located. The specified solution is necessary in order to prevent the share data problem appearance in such cases, while the main microcontroller's routine is interrupted by the position sensor measurement values or the LCD-keypad shield actions.

The master microcontroller produces the control law for one of the stepper motors directly in addition to the reference value broadcasting function. In this case, the parts of the code which realize the control are the same as for the corresponding slaves' functions. Such functional combination in master microcontroller case provides the whole system dimensions' optimization and the increase in master microcontroller exploitation's efficiency.

In the same considerations' purposes only the master microcontroller is accompanied by the LCD-keypad shield in order to reduce the number of such modules in use from the amount equal to the conveyor's number to one unit. It also ensures the synchronous control of the conveyors: the reference value is guaranteed to be delivered to all the slaves simultaneously and the line operator(s) has to set the reference only once. That also provides the minimization of the mistakes during the reference setting.

6.2 Communication between the microcontroller and the position sensor

For the implementation of the microcontroller-to-sensor communication the header file *SparkFun_VL53L1X.h* was used (SparkFun_VL53L1X.h n.d.). During the setup of communication process while using this library, the first step is to set the numbers of sensor's pins that are responsible for shutdown and interrupt signals.

The next step is to define the mode of operation depending on the measurement distance. In case of the system under study, the distance between the sensor and the moving PCB is located in ranges of tens millimeters, therefore the corresponding library function is set to the short-distance mode. The selection of this mode provides prevention from false alarms, which are typical for the long-distance metering. If required by the application, the metering mode can be simply changed by varying the corresponding function's argument. It is worth mentioning that when using the long-distance mode it is important to keep in mind that it is more effective when it is used under dark light conditions. The short-distance mode does not require such light limitations for the measurement environment as the metering results are not affected strongly by the surrounding light and light conditions in general.

The actual measurement process is realized also by use of the mentioned header file's functions. The measurement is implemented as an endless loop that contains the following stages: starting the measurement process, writing the measured value to the local integer variable *distance*, stopping the measurement and after that sending the measured value to the microcontroller.

The starting and stopping functions of the measurement are functions that operate over the shutdown signals, changing it from high level to low and vice versa in order to reach the required state of the sensor. Interrupt signal is used for informing the microcontroller about a new measurement appearance, in order to change the latter's routine to specific one that is used for processing the sensor's data.

The measured value that is sent to the microcontroller is already presented in millimeters and therefore no conversion of the measured value is needed in this work. Although such conversion can be easily implemented by introducing an additional arithmetic line in the measurement loop in order to convert millimeters into the desirable units. The data can be transmitted to the microcontroller in different units at the same time. The processing of the measurement should consider the inaccuracy of about 1 mm of the measurement implemented by this sensor.

6.3 Communication between the microcontroller and the stepper motor driver

For the stepper motor driver's successful operation, it requires only two signal types (despite the supply and enabling ones): the *PUL*- is used for realization of one motor's step and *DIR*- is used for setting the direction of the motor's shaft rotation.

As the stepper motor control is based on the utilization of more complicated signals, such as rotation speed, acceleration and deceleration, which can be represented as the combinations of two signals mentioned above, then the programming should be realized basing on the speed and acceleration/deceleration as functions of *PUL*- and *DIR*- signals. For implementing this, the header file *AccelStepper.h* is exploited as a basis for communication with the stepper motor driver in use (AccelStepper.h n.d.). For operations with the functions of this library, firstly it is necessary to set the driver ports' numbers, which are related to the *PUL*- and *DIR*- signals delivery. Thus, without any additional coding the low or high voltage on these pins will be interpreted by the program the following way: the logic unit on the *PUL*- pin defines the necessity of the motor's shaft turn for one step; logic zero means the idle state of the motor and on the contrary the logic unit on the *DIR*- pin means the setting of the motor's shaft rotation in the default direction; logic zero changes the rotation direction to the opposite.

On the next programming stage, the default values of the rotation direction, speed and acceleration should be set by using special library functions. During the control system's operation these parameters are varied dynamically depending on the control law derived by the microcontroller. While setting the default values for speed and acceleration the programmer should keep in mind the fact that their measurement units in this case are steps in second or steps in second^2 respectively, not the metric ones. Thus, the speed and acceleration are represented here as functions of sequences of the *PUL*- and *DIR*- signals of zero and unit values; the delay is interpreted as low level of the *PUL*- signal.

Then the value of the offset should be set, i.e. the amount of steps during one working cycle. In the current work such quantity is a number of steps needed for passing the whole conveyor line's length (or it can be the length of the entire conveyor's belt). The conveyor's motion can be set such way that after realizing the forward number of steps, it should reverse the motion direction and by doing the same amount of steps return to the initial position. In the standard situation where only one PCB is being moved by the conveyor, such mode can be realized. But keeping in mind the possible case of more than one PCB's quantity presence on

the same line simultaneously, such operational mode cannot be implemented. Moreover, from the efficiency point of view such technique is useless because during the reverse movement the conveyor does not transport anything, doing therefore an additional operation. Thus, the rotation direction must stay the same during the system's operation in the developed system.

Therefore, the program for the microcontroller-driver communication requires only setting the default values of the shaft rotation's direction, desirable speed, acceleration and offset during one operational cycle by the use of special library's functions. The speed and acceleration are controlled such way that at the instances of getting the PCB on the conveyor and its' releasing the conveyor's motion speed is minimal in order to minimize the offset of the PCB's location. Between these two instances, the conveyor is accelerated for the maximum allowed value in order to transport the PCB faster and at the end of the line is decelerated for ensuring rough transportation of the PCB to the receiving equipment. In typical case, while after releasing the PCB from the conveyor, the latter is set to idle mode waiting for the next board's arrival.

6.4 Communication between the master microcontroller and the LCD-keypad shield

Standard *Arduino* header file *LiquidCrystal.h* which contains functions for communication with LCD, is used in this thesis. On the initial phase of the microcontroller-shield communication's setup it is necessary to define the shield pins, which are used for transceiving the following signals: enabling LCD operation signal *E*, the signal *RS*, which defines the type of the transmitted information type: data or commands, and signals for LCD segments in four-bit mode (*DB4-DB7*).

The file *LiquidCrystal.h* is a universal library for communication with LCDs, so next we need to set the LCD's matrix dimension (16 times 2). Another features that can be defined during the setup are the greeting message that appears on the display while it is loading after power on and the initial cursor location which points the place of the first printed symbol appearance.

Then the buttons presses should be tuned. First of all, for clarity we need to provide each active button with symbolic names according to the numeration given in Fig. 22. First button is

"right", second – "up", third – "down", fourth – "left" and fifth – "select". The far right button is reserved for shield reset and can not be used in other purposes.

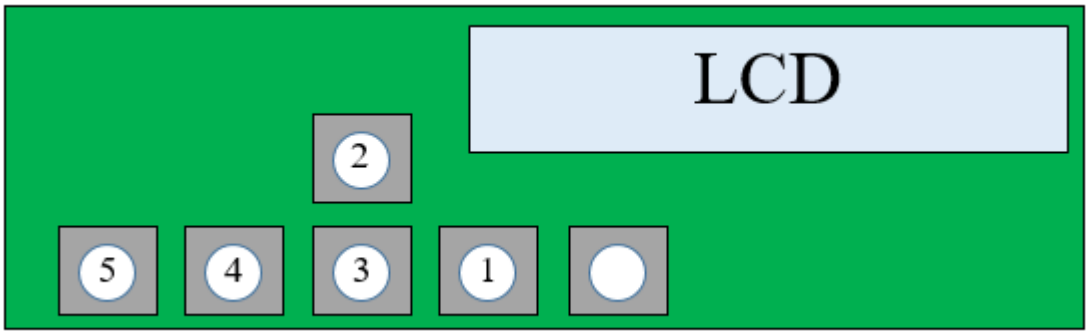

Figure 22. Buttons numeration.

Analog signal that is formed at every button press is transferred to the microcontroller through its analog input to the internal ADC. The proper value of the corresponding analog voltage signal depends on the particular button press. For further processing of the input analog signal it should be transformed by ADC into digital value that is proportional to the analog value. For identifying the proper pressed button's number, the basic code was tested with the shield. By pressing the buttons the digital ranges for corresponding buttons were defined approximately:

- pressing the "right" button matches the 0-200 range;
- pressing the "up" button matches the 200-600 range;
- pressing the "down" button matches the 600-1200 range;
- pressing the "left" button matches the 1200-1800 range;
- pressing the "select" button matches the 1800-2400 range;
- values that are out of these ranges matches the case when no button is pressed.

Based on these numeric ranges, the switch-case structure of the button identification by use of the digitalized value of the analog signal was developed. Then the following functionality of the buttons was defined:

- "up" and "down" buttons are used for switching between controlled parameters (conveyors' speed and the distance values in the current work);

- “right” and “left” buttons are used for increment and decrement of the parameter (which is selected by use of the “up” and “down” buttons) by the fixed value;
- “select” button is used for approval of the selected parameter variation.

When buttons are being pressed, the corresponding messages about the varied parameter and its’ new value appears on the LCD. After approval of the changes, the screen is cleaned after predefined delay and it goes to the idle mode. For parameters change mode activation one of the five active buttons should be pressed.

The value which is typed by the user with help of the keypad is then transferred to the master microcontroller and after its’ processing is delivered to slave microcontrollers for forming the control laws by themselves for their relative stepper motors. Also it is transformed by the master itself for direct control of one of the motors. Besides the visualization of the parameters changing process, the LCD communication code can be tuned for displaying the system’s error and warning messages.

6.5 Communication between microcontrollers

The program code that is realized by microcontrollers slightly differs depending on the microcontroller’s role inside the control system – master or slave. In both case the *Arduino* header file *SoftwareSerial.h* is used as basis for programming. The setup process for both master and slaves starts from the transceiver pins identification: it is necessary to define the numbers of ports that are responsible for receiving and transmission of the signals to or from the connected directly to transceiver microcontroller. Then the following variables are identified: *value* is introduced for writing and storing the transmitted at the current instance data and *D_RE* is used for setting the direction of the data transportation (transmission or receiving mode).

On the next stage the entire serial communication by bus is started in receiving mode by default. The checking of the presence of the data, which waits for being transmitted from the microcontroller to other/others in the chain, is always done. In case of such data appearance, the transceiver mode is changed to the transmission until the end of the receiving data for transmission from the directly connected microcontroller.

In case of receiving data from the external microcontrollers, the transceiver mode is changed to the receiving. In both cases, the transported data are copied to the *value* variable; after that action this changed variable state is transferred to the target microcontroller. The main difference between master and slave in such system is that slaves can not change their mode to the transmission one independently from the master.

7 SIMULATION OF THE DEVELOPED SYSTEM

Due to the practical difficulties during this thesis the experimental evaluation of the proposed embedded system cannot be tested. For this reason, a simplified simulation is considered in this thesis. The electrical complex, which is controlled by the developed control system, includes a set of technological operations. The modeling of them might be difficult and in some cases their behavior is impossible to predict and model with reasonable accuracy. This for instance concerns modeling the movement of the PCB along the conveyor, since this process depends on the parameters which are varied during the operation of the conveyor and the stepper motor, and also on a number of external factors (vibrations of equipment and walls of the room where the complex operates, the state of the conveyor belt itself, mass and dimensions of the printed circuit board to be moved, etc.). Based on this reason, in this work, the following assumptions were made to simplify the modeling process:

1. The controlled variable is not the PCB's offset, but the conveyor's belt motion speed. This assumption is justified by the fact that the mentioned speed is one of the parameters that can be set by the operator as desired for further usage during the control law formation. The control of the offset of the PCB position comes down to setting the maximum value of the motion speed at which the offset does not exceed the allowable range. Thus, by controlling the speed value within the specified limits, the desired value of the distance between the board and the conveyor can be guaranteed with a certain degree of accuracy.
2. The master microcontroller is simplified to a constant block which transfers the reference value to slaves.

By considering these and some other assumptions and simplifications, the modelling process becomes easier and the resulting model structure looks clearer. After the development of the mathematical models of the system's main parts, the simulation model was built in the *Simulink* environment. The implemented model does not have practical value for this study as the controlled parameters cannot be obtained from this model and real devices' communication cannot be simulated realistically. Thus, the simulation model below serves the demonstration of how the cooperation of stepper motor and conveyor is realized in general.

7.1 Stepper motor modelling

Simulink contains already made stepper motor model, so there is no necessity in developing its mathematical model. When used as part of simulation mode, it only requires setting the parameters related to the particular motor's specification in the setup window (see Fig. 23). The values needed for the block are the number of motor's phases, winding's inductance and resistance, the motor's step value, decent torque and rotor's inertia (42HS4813A n.d.). In the simulation the other parameters needed for the block are left with their default values.

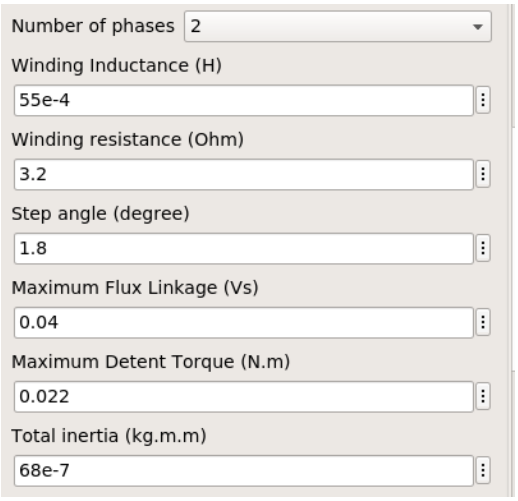

Figure 23. Setting the stepper motor's parameters.

The model of the stepper motor's driver can be also found from *Simulink* library and it can be connected to the motor's model directly. To validate and illustrate the motor and driver models' performance, first a simple open-loop control system was implemented that is seen in Fig. 24. It ensures the *DIR*- and *PUL*- pulses by the *Signal Builder* block.

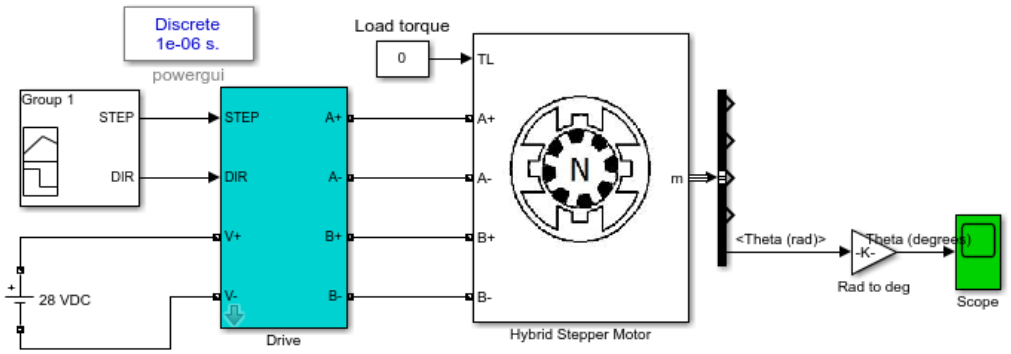

Figure 24. Open-loop stepper motor control tested in Simulink.

During the model testing the rotation angle was measured from the motor output using the Scope block. The input pulses were defined so that a continuous high *PUL*- is given for 0.1 s and then turned off and the direction pulses were in high state during these 0.1 seconds. After 0.05 s off-time the *PUL*- pulses were restarted and at the same time the direction pulse was reversed to zero value. Thus, the opposite shaft rotation was implemented.

The simulation results are presented in Fig. 25. It is obvious from this result that open-loop control can be implemented quite well, but it can be noticed that a 5-7 ms delay exists at the beginning. The pulses given to the driver and the corresponding positioning of the motor are noticeable from the picture below.

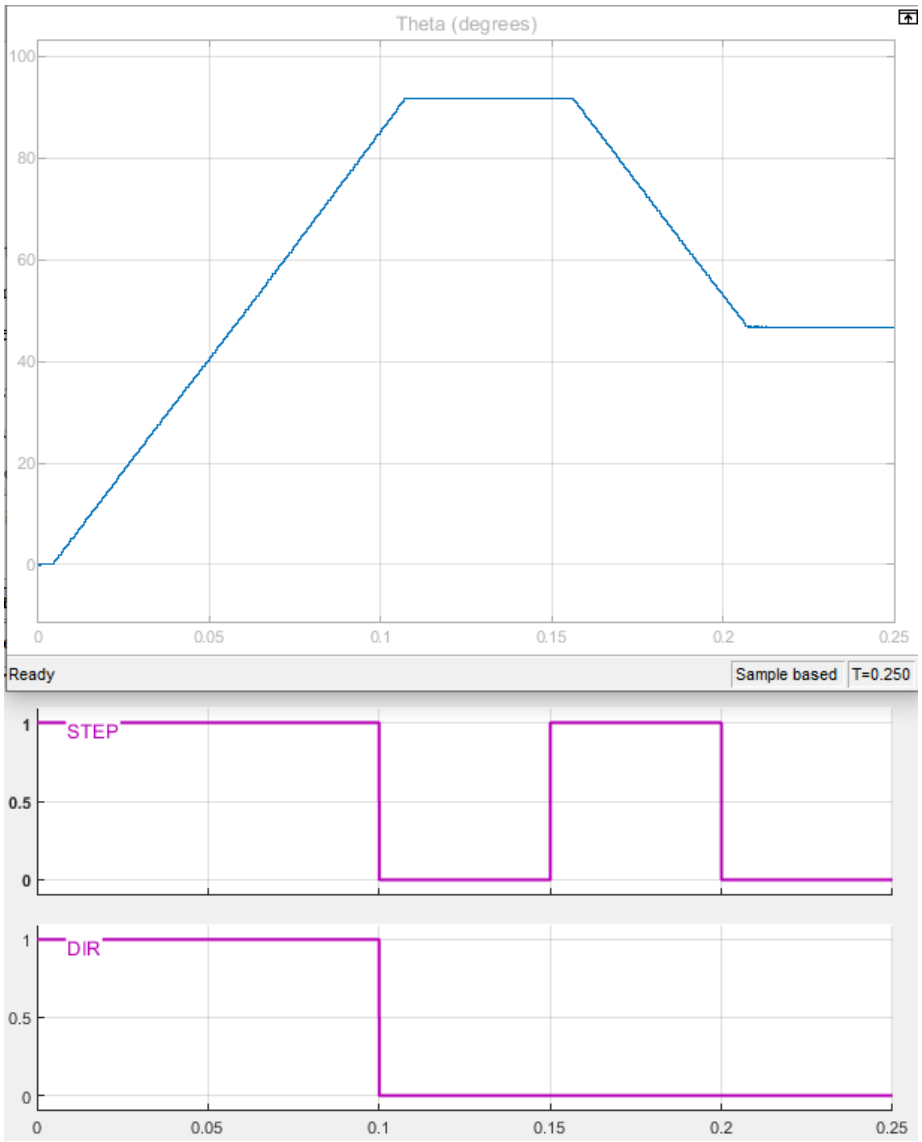

Figure 25. Results of the open-loop stepper motor control in idle mode: the plot of rotor's rotation angle (degrees) and control driver's input signals.

7.2 Modeling of conveyor

The development of mathematical model of the conveyor is difficult process that cannot be derived completely because of its complex nature. The solution used in this thesis and presented below is based on principles and strategies that are described in (J. Parkkinen et al. 2013; Ковальчук 2017; Kovalchuk 2018; Павлов 2018; Дмитриева 2008; Дмитриева 2005).

As this thesis does not focus on the conveyor's physics and the real control variable (PCB's offset) cannot be modelled due to its complex nature, then the modelling can be done in simplified manner. The conveyors' modelling is implemented according to the simplified design scheme given in Fig. 26. Here the conveyor is interpreted as a simple two-mass model representing dynamics between two pulleys.

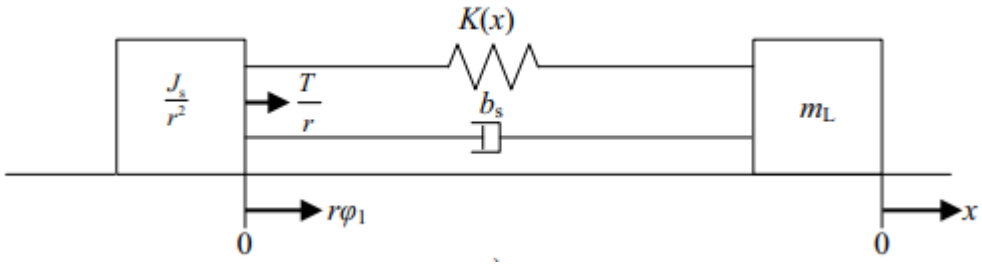

Figure 26. One-drive conveyor design scheme (modified from (Parkkinen et al. 2013).

The linearized conveyor belt model presented in Fig. 26 can be described by two following differential equations (Parkkinen et al. 2013):

$$\begin{cases} m\ddot{x} = b(r\dot{\varphi} - \dot{x}) + K(r\varphi - x) & (6.1) \\ J\ddot{\varphi} = \dfrac{T_e}{r} - b(r\dot{\varphi} - \dot{x}) - K(r\varphi - x) & (6.2)' \end{cases}$$

where m – mass of the transported PCB, kg; x – position of the transported PCB, m; b – belt damping constant; r – conveyor shaft radius, m; φ – drive shaft angular coordinate; K – equivalent spring constant, N/m; J - motor inertia, kg·m^2; T_e – drive's torque, Nm.

Using these equations, a state-space representation of the conveyor model can be obtained:

$$\begin{cases} \dot{x}(t) = \mathbf{A}x(t) + \mathbf{B}u(t) & (6.3) \\ y(t) = \mathbf{C}x(t) & (6.4)' \end{cases}$$

where the state matrix **A**, input matrix **B** and measurement matrix **C** are as follows

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{K}{m} & -\frac{b}{m} & \frac{kr}{m} & \frac{br}{m} \\ 0 & 0 & 0 & 1 \\ \frac{K}{J} & \frac{b}{J} & -\frac{kr}{J} & -\frac{br}{J} \end{bmatrix}; B = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{1}{rJ} \end{bmatrix}; C = [0\ 1\ 0\ 0].$$

The dynamics of this model is defined by the state vector $x = \begin{bmatrix} x \\ \dot{x} \\ \varphi \\ \dot{\varphi} \end{bmatrix}$. Here the drive torque T_e is the system input and the output is belt motion speed $\dot{x}$.

Considering the standard state-space matrices above, the conveyor model was built in *Simulink* environment as shown in Fig. 27. All three matrices are presented as multiplying gains.

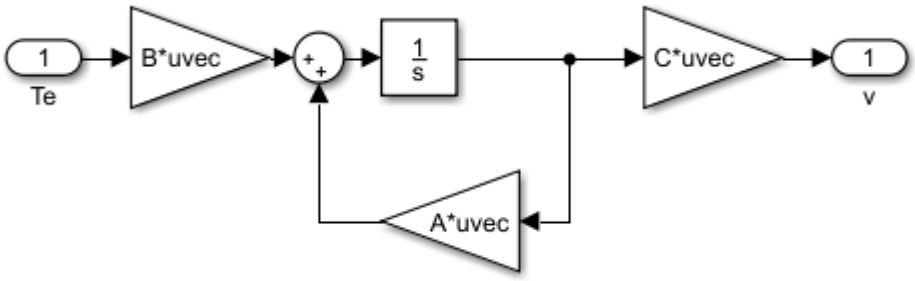

Figure 27. Simulink-model of the conveyor.

To start the model, the initial values and state-space matrices have to be defined in *MATLAB*-file. That was done according to the conveyor technical documentation data (CYA330L, n.d.). Also by using the *MATLAB* functions, the frequency response of the defined system can be plotted as is seen in Fig. 28.

```
K=5.75e5; % equivalent spring constant, N/m
b=0.01; % belt damping constant
r=0.002; % drive's reel radius, m
m=0.1; % transported PCB weight, kg
J=68e-7; % drive's inertia, kg*m2
v_ref=0.25; % belt motion speed reference, m/s

% state-space matrices
A=[0     1    0      0;
   -K/m  -b/m K*r/m  b*r/m;
   0     0    0      1;
   K/J   b/J  -K*r/J -b*r/J];
B=[0 0 0 1/(r*J)]';
C=[0 1 0 0];
sys=ss(A, B, C, 0);

bode(sys);
```

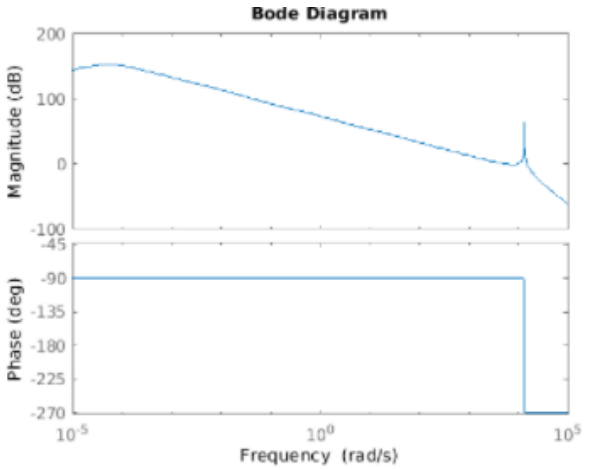

Figure 28. Initial data inserting and plotting the frequency response of the conveyor model.

7.3 Entire system modelling

As the developed system is expected to be tested on real technological equipment, then the construction of a full and deep model seems to be unnecessary complex and difficult in realization in the selected simulation environment. Moreover, the model's development is not a main purpose of this thesis, so it is presented as a side part of work in order to have a more clear understanding of the driver-motor-conveyor system behavior. As a result, a simplified model that performs conveyor motion speed control was designed for a single motor system because the real synchronous programmable control of the PCB offset is quite difficult to be implemented in *Simulink*.

This modelling stage is dedicated to the connection of the developed models of the stepper motor, its' driver and conveyor, and then adding the control circuit. In order to create clearer whole model structure, the conveyor model was hidden inside the subsystem *Conveyor*.

The conveyor model receives for its input the value of the motor torque from the corresponding motor's output. The conveyor model output *v*, which indicates the conveyor belt motion speed, is used by control system as a feedback value in order to form the control law for the stepper motor driver.

The actual conveyor belt speed is received by the block, which calculates the error value by comparing the actual value to the reference. Then the calculated error goes to the PID-controller block which produces the analog control signal for the driver. After that, the defined control signal is discretized by ADC to the digital form, in which the signal can have only two states – high or low. All three mentioned operations (calculation of an error, formation of the continuous control signal and its further discretization) are done in real system by a single microcontroller. The direction defining port gets a constant high level signal as the investigated system is not expected to be working in opposite motion directions. The structure of the developed simulation model for single-motor control system is presented in Fig. 29.

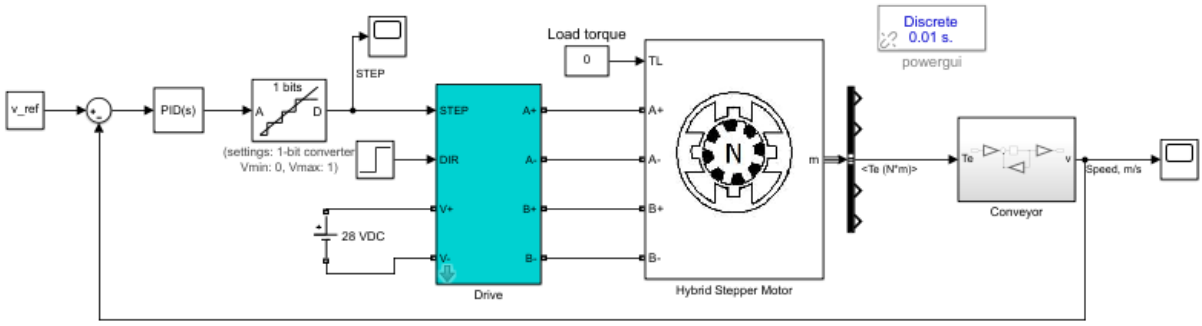

Figure 29. Simplified model of a single-motor control system.

Tuning of the PID-controller was done using block *Closed-Loop PID Autotuner*, which was temporary inserted between the PID and ADC (Fig. 30). It uses the controller output and the system output signals in order to define the most suitable gain values. Unitary constant, connected to its' third input, enables the block operation. The recommended gains are performed after the system running by the displaying element, connected directly to the tuner corresponding output. After testing the system, the values, which were calculated by the tuning block (the proportional gain K_p = 59.07, integral gain K_i = 1.07 and derivative gain K_d = 1.78) were typed as the controller gains in the parameters window of the controller block. When the definition of the appropriate control gains was finished, the autotuning block and its connections were removed from the model.

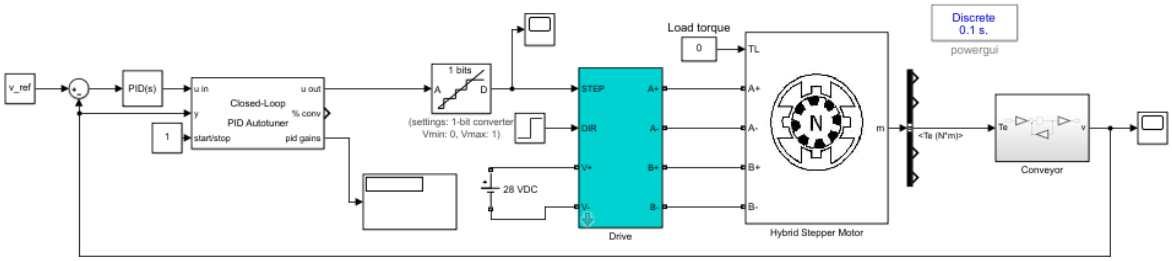

Figure 30. Autotuning of the controller gains.

The last step of the entire model simulation was its testing with selected parameters. Setting of the 0.25 m/s conveyor belt speed was the aim of the realized tests. The results of the simulation are presented in Fig. 31. The graph of the *STEP* pulses is not attached because this signal changes its value so frequently that the plot becomes unreadable.

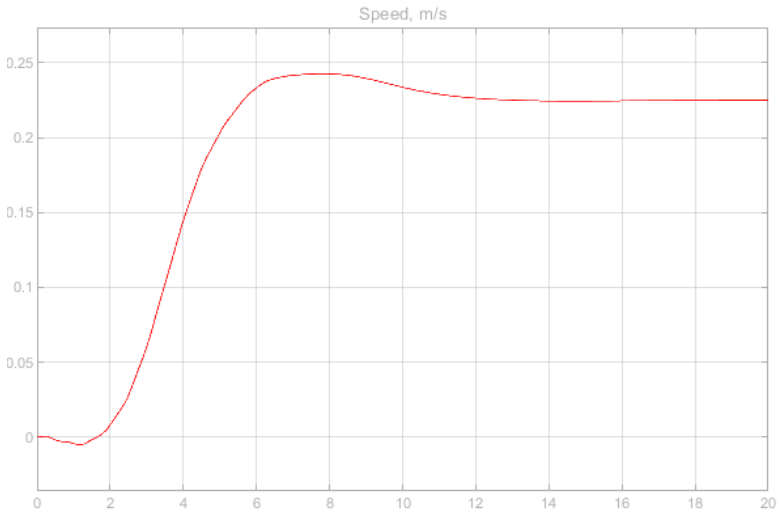

Figure 31. Results of the model simulation: setting the 0.25 m/s linear speed of the conveyor belt motion.

The presented graph shows that the control is implemented quite successfully, although the steady-state error is about 10%. It also indicates that the control block does not work correctly with the stepper driver and motor, as the control do not correct the given reference. The input of the block might be given in the wrong format. Because the presented model has a very slight similarity to the real developed system, the controlled variables are different and the modelling of a real operating system is unreasonably difficult in this study framework, then the further improvement of the control action seems to be unnecessary. So this simulation results do not have practical value for this investigation and have only demonstrative character. As the developed control system is expected to be implemented and tested on the real industrial

equipment, then the better way will be the manual tuning of the control gains. Therefore, the designed simulation model has more theoretical interest in purposes of deeper understanding of the physics of the stepper motor to conveyor communication than the practical value.

8 CONCLUSION

In this thesis, a synchronous control system for the electric drive of conveyor lines part of manufacturing process, which carries out surface mounting of electronic components on the surface of a printed circuit board, was developed. The control system is based on the interaction of microcontrollers according to the *Master-Slave* communication model. The microcontrollers generate a control action that regulates the rotation speed of the stepper motor shaft driving the conveyor line, based on the comparison of the actual position value of the printed circuit board moving along the conveyor with the required value. The measurements of the current position of the board are made using position sensors that transmit data to microcontrollers. To ensure convenient interaction of the system operator with the entire control system (first of all, to set to the master's microcontroller the required value of the offset of the printed circuit board location on the conveyor belt), a user interface has been developed. It allows selection of the required value by pressing buttons and in addition, the selected value is displayed on the LCD screen for user convenience.

For centralized control of the system, one of the microcontrollers is set as the master: processing the input required value of the position of the board on the conveyor, it converts this value into a control signal transmitted to the slave microcontrollers. The latter, on the basis of the received signal, as well as based on the measurements of the position sensor, generate a control signal transmitted to the stepper motor by means of the driver. Thus, the speed of the movement of the shaft of each motor is set synchronously to all drivers of the stepper motors of the conveyors by microcontrollers, each assigned to one specific motor.

To achieve a more compact and rational arrangement of the elements of the control system, it was suggested that it is possible to use only one microcontroller for communication with LCD-keypad shield in order to receive the reference value and then to broadcast it for other microcontrollers (realizing thus master-slave communication). Additionally, this master microcontroller also controls one of the drives.

To fulfill the assigned tasks an analysis was made about the existing solutions to the problem of synchronous control of the electric drive of conveyor lines, that guided the control system development. In the process of its creation and preparation of this thesis, a number of actions were done. First of all, a generalized description of the principles of operation of the investigated control system were given and the most important devices and components used

in the system were briefly described in order to have a more complete understanding of the features of the operation of this system. Then the connections between microcontrollers and peripheral devices were presented in the form of schematic electrical diagrams, as well as the layout of the wiring of two types of printed circuit boards: one including microcontroller of master or slave types and a printed circuit board on which a position sensor is installed and connected to the corresponding microcontroller. After that the main functional parts of the program code used in the operation of microcontrollers of both types were described. Finally, the developed control system was simulated in the *Simulink* environment in generalized and simplified manner; the purpose was to become more familiar with the communication processes between the driver, stepper motor and the conveyor belt.

Summarizing all the above, it should be concluded that the goals and objectives set in the description of the problem have been successfully solved and the developed system can be used to control a real electrical complex. Moreover, being a fairly versatile solution, this control system can be adopted to other technological complexes that include several conveyor lines. This study preparation resulted in high-efficient in use, dimensions-optimized and economically-concerned solution of the conveyors drive control issues.

REFERENCES

1. AccelStepper.h library, accessed 12 May 2021.

2. Kien Minh Le, Hung Van Hoang, and Jae Wook Jeon, “An advanced closed-loop control to improve the performance of hybrid stepper motors”. In: IEEE Transactions on power electronics, Vol. 32, No. 9, September 2017, 7244-7255

3. AM1S-N Series [datasheet], AIMTEC (n.d.)

4. CYA330L [technical passport], 1 CLICK SMT TECHNOLOGY CO (n.d.)

5. Dae Hwan Kim, Hyun Jong Kim, Sang Bong Kim, “Closed Loop Motion Synchronous Velocity Control for AC Motor Drives – A Solution for Increasing Speed of a Cross-Belt Sorting Conveyor System. AETA 2016: Recent Advances in Electrical Engineering and Related Sciences, vol. 415, 1-10

6. Ivan Virgala, Michal Kelemen, Alexander Gmiterko, and Tomáš Lipták, “Control of Stepper Motor by Microcontroller.” Journal of Automation and Control, vol. 3, no. 3, 2015, 131-134

7. J. W. Jung, Y. S. Choi, V. Q. Leu, and H. H. Choi, “Fuzzy PI-type current controllers for permanent magnet synchronous motors,” IET Electr. Power Appl., vol. 5, no. 1, pp. 143–152, 2011.

8. M. Marufuzzaman, M. B. I. Reaz, and M. A. M. Ali, “FPGA implementation of an intelligent current dq PI controller for FOC PMSM drive,” in Proc. IEEE Comput. Appl. Ind. Electron., 2010, pp. 602–605

9. Kovalchuk M., Baburin S. “Modelling and control system of multi motor conveyor” IOP Conf. Ser.: Mater. Sci. Eng. 2018, 327, 1-6

10. J. Parkkinen, M. Jokinen, M. Niemela, T. Lindh and J. Pyrhonen, “Motion Synchronization of Two Linear Tooth Belt Drives Using Cross-Coupled Controller” 2013 15th European Conference on Power Electronics and Applications, 1-7

11. PIC16(L)F15356/75/76/85/86 [datasheet], Microchip Technology Inc, 2018

12. Li Hai Xia, and Jie Hui Zeng, “Remote Control Stepper Motor Based on the GPRS.” Applied Mechanics and Materials, vol. 496–500, Trans Tech Publications, Ltd., Jan. 2014, 1603–1608.

13. Biao Yu, Hui Zhu and Chi Xue, “Research on Adaptive Fuzzy PID Synchronous Control Strategy of Double-Motor” I.J. Intelligent Systems and Applications, 2011, 28-33

14. Y. I. Son, I. H. Kim, D. S. Choi, and H. Shim, “Robust cascade control of electric motor drives using dual reduced-order PI observer,” IEEE Trans. Ind. Electron., vol. 62, no. 6, pp. 3672–3682, Jun. 2015.

15. SparkFun_VL53L1X.h library; accessed 12 May 2021

16. Daode Zhang, Jingqi Wang, Lei Qian, Jun Yi, “Stepper motor open-loop control system modeling and control strategy optimization” Archives Of Electrical Engineering Vol. 68(1), 2019, 63-75

17. Stepper motor parameters; accessed 11 May 2021

18. TB6600HG [datasheet], Toshiba, 2016

19. V660X RS-485 [datasheet], Vango Technologies Inc, 2016

20. VL53L1X [datasheet], STMicroelectronics, 2018

21. WH1602A [datasheet], Winstar, 2008

22. Дмитриева В.В. Разработка и исследование системы автоматической стабилизации погонной нагрузки магистрального конвейера. Автореферат диссертации на соискание ученой степени кандидата технических наук. Москва, 2005. 28 с.

23. Дмитриева В.В., Гершун С.В. “Разработка математической модели ленточного конвейера с двухдвигательным приводом” Материалы симпозиума «Неделя горняка-2007». 2008. С. 295-304

24. Ковальчук М.С., Поддубный Д.А. “Моделирование и разработка алгоритма управления многодвигательным электроприводом конвейерного транспорта” Современная наука и практика. 2017. № 3 (20). С. 10–15

25. Павлов В.Е. “Исследование режимов пуска электропривода ленточного конвейера методом компьютерного моделирования” Вестник Иркутского государственного технического университета. 2018. Т. 22. № 4. С. 136–147. DOI: 10.21285/1814-3520-2018-4-136-147