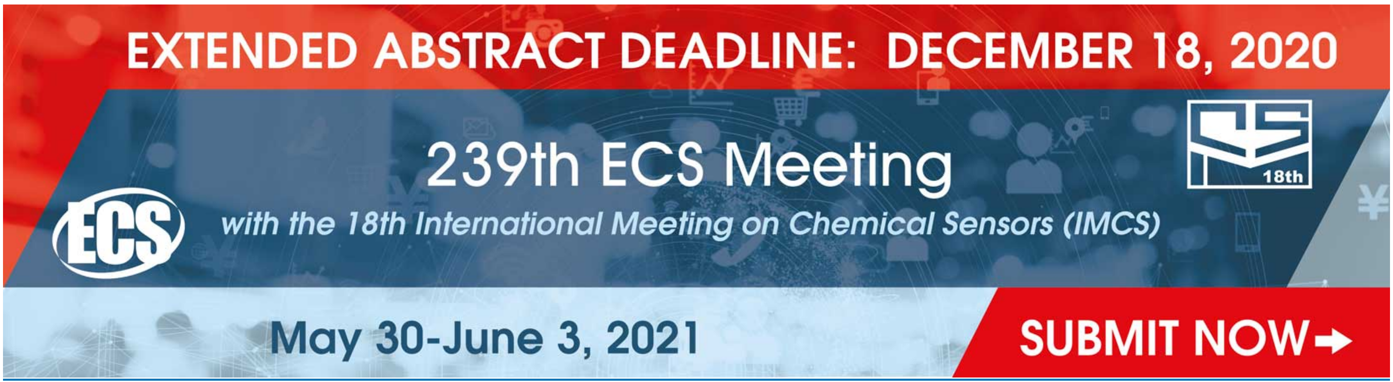
IOP Conference Series: Materials Science and Engineering

Advanced hardware control for seven DOFs robotic arm-neuro arm

To cite this article: B Cvetkovi *et al* 2018 *IOP Conf. Ser.: Mater. Sci. Eng.* **393** 012110



Advanced hardware control for seven DOFs robotic arm-neuro arm

B Cvetković[1], V Nešić[2], M Lazarević[1], P Mandić[1], P Marić[2] and M Dragović[3]

[1]University of Belgrade, Faculty of Mechanical Engineering, Kraljice Marije 16, 11000 Belgrade, Serbia
[2]IMP-Automation and control, Institute of Mihajlo Pupin, Volgina 15, Belgrade, Serbia
[3]Uno-lux processing, Poručnika Spasića i Mašere 14, 11147 Belgrade, Serbia

E-mail: boskocvetkovic@gmail.com

Abstract. In this paper we will implement new hardware control for 7 DOFs (Degrees of freedom) advanced robotic arm research platform (NeuroArm) equipped with various sensors. New hardware platform consists of Nano Pi platform which is used as a PC platform for standard PLC (programmable logic controller's) unit and will replace old system that consists of 7 16-bit ATmega processors. The goal of new PLC which is based on Linux operating system (Debian distribution) that is patched by Xenomai real time system for reducing control response time and better entire system control. Robotic arm is powered by 7 DC motors which are controlled by two PWM4 (Pulse Width Modulation) modules. Position data is acquired from 1kΩ resistive sensors using RI8 (Resistive module) module and from optical quadrature encoders using one DI16 (Digital Input) module. Also other modules PWM4, RI8 and DI16 communicate with PikoAtlas CPU module by I2C bus.

1. Introduction

It is well known that robotic systems are more and more ubiquitoused in industrial applications as well as in the field of direct interaction and in health care [4] and for helping people so-called friendly home environment. As one of these robotic systems capable of operating in such environments is NeuroArm robotic system [1]. It is an integral part of the Laboratory of Applied Mechanics, Mechanical Engineering in Belgrade, (Figure 1a) and serves the educational purposes within the course Mechanics of robots, the Master works as well as for scientific research in the field of robotics to doctoral studies.

Within NeuroArm Manipulator System - there are a rich set of options that enable scientists and engineers to configure your robot that will meet the needs [2], [3]. From the mechanical point of view NeuroArm robotic arm has 7 degrees of freedom (6 for rotating and 1 translating), including gripper [5].

Mechanism of gripper has the possibility of a parallel movement and control of position and speed. The maximum load that can raise the robot is up to 2-3 kg depending on the additional equipment. Control is based on the applying of robotic processors Atmega 128 with I2C or SPI high-speed highway (Figure 1b). Processor Atmel Atmega 128 Robot ® - integrate other coprocessors that are used to control on the executive level, surveillance sensors, robot status, while for the security of the robot uses NeuroBackdrive ™ & NeuroArm ™ Protocol. In the main joints there are sensors for measuring position, velocity, acceleration and torque. This robot has 3 Amp Maxon DC motor with planetary gearing and high-resolution encoders for each robot joint, respectively. These provide a

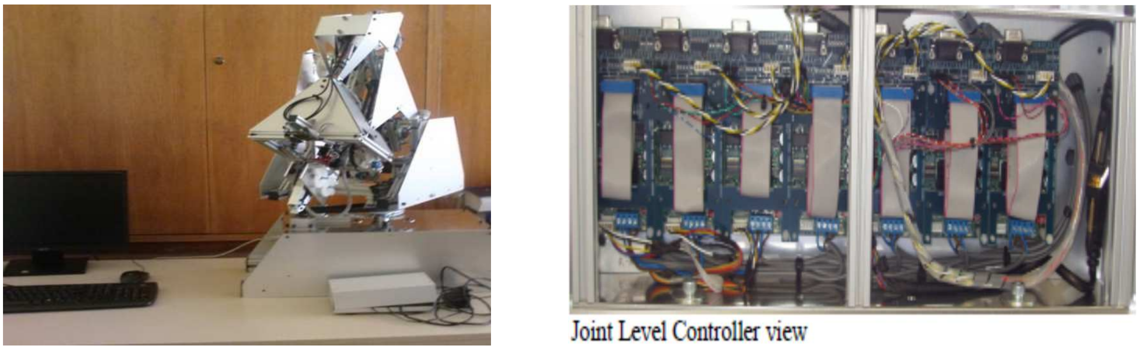


Figure 1. a) laboratory NeuroArm robotic arm and b) hardware structure of control system of given robot.

good/smooth control in closed feedback loop in respect to position, velocity, acceleration of a given profile, as well as control which is based on the torque (torque control). Besides, there are inertial sensors on the end-effector in the purpose of determining and taking into account the inertial effects caused by the inertial forces. Also, the part of the equipment is haptic joystick with force feedback. It allows us to control a robot remotely over the web, with the ability of the operator "feel" what the robot "feels".

2. Choosing advanced control platform

Due to the limitations of existing platform in a term of control and inability of implementing new control logic, it is decided that it needs to be replaced. Arduino DUE, Raspberry Pi model 3, Beaglebone Black and Nano Pi were the platforms that were considered as a replacement hardware control. After testing, Arduino DUE was discarded because of a lack of possibility to perform multiple complex operations in the same time that are needed at this moment or in the future development and adding new sensors or additional motors. Additional reason for that is that Arduino lacks higher level operation system that would allow easier implementation of additional sensors like camera. For first implementation advantage was given to Beaglebone Black instead of Raspberry Pi because it has two 46-pin headers which allows maximum of 92 possible connections including 4 timers and 25 PRU low-latency I/Os. Despite that Beaglebone Black has less powerful processor and less memory, it is enough powerful for control of 7 DC motors with complex control algorithms and additional sensors. Nano Pi with it’s additional add-on boards for controlling motors like NanoHat Motor module is still in consideration for a final use of main controller of entire NeuroArm robot platform.

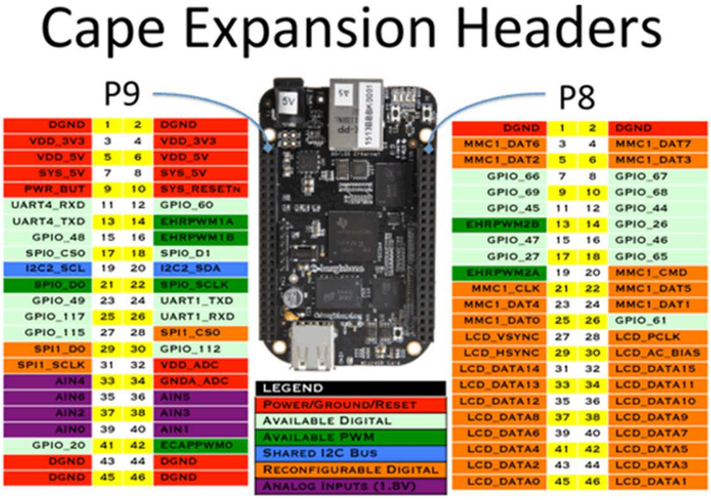
2.1. Technical specification of Beaglebone Black

Hardware specification:

- Processor: AM335x 1GHz ARM® Cortex-A8
- 512MB DDR3 RAM
- 4GB 8-bit eMMC on-board flash storage
- 3D graphics accelerator
- NEON floating-point accelerator
- 2x PRU 32-bit microcontrollers

The BeagleBone Black comes with two 46-pin headers which allow making 92 possible connections. They are specified as:

- 65 GPIO pins
- 7 Analog inputs
- 8 PWM outputs
- 2 SPI buses
- 2 I2C buses
- 4 Timers
- 4 UARTs



P9

DGND	1	2	DGND
VDD_3V3	3	4	VDD_3V3
VDD_5V	5	6	VDD_5V
SYS_5V	7	8	SYS_5V
PWR_BUT	9	10	SYS_RESETN
UART4_RXD	11	12	GPIO_60
UART4_TXD	13	14	EHRPWM1A
GPIO_48	15	16	EHRPWM1B
SPI0_CS0	17	18	SPI0_D1
I2C2_SCL	19	20	I2C2_SDA
SPI0_D0	21	22	SPI0_SCLK
GPIO_49	23	24	UART1_TXD
GPIO_117	25	26	UART1_RXD
GPIO_115	27	28	SPI1_CS0
SPI1_D0	29	30	GPIO_112
SPI1_SCLK	31	32	VDD_ADC
AIN4	33	34	GNDA_ADC
AIN6	35	36	AIN5
AIN2	37	38	AIN3
AIN0	39	40	AIN1
GPIO_20	41	42	ECAPPWM0
DGND	43	44	DGND
DGND	45	46	DGND

P8

DGND	1	2	DGND
MMC1_DAT6	3	4	MMC1_DAT7
MMC1_DAT2	5	6	MMC1_DAT3
GPIO_66	7	8	GPIO_67
GPIO_69	9	10	GPIO_68
GPIO_45	11	12	GPIO_44
EHRPWM2B	13	14	GPIO_26
GPIO_47	15	16	GPIO_46
GPIO_27	17	18	GPIO_65
EHRPWM2A	19	20	MMC1_CMD
MMC1_CLK	21	22	MMC1_DAT5
MMC1_DAT4	23	24	MMC1_DAT1
MMC1_DAT0	25	26	GPIO_61
LCD_VSYNC	27	28	LCD_PCLK
LCD_HSYNC	29	30	LCD_AC_BIAS
LCD_DATA14	31	32	LCD_DATA15
LCD_DATA13	33	34	LCD_DATA11
LCD_DATA12	35	36	LCD_DATA10
LCD_DATA8	37	38	LCD_DATA9
LCD_DATA6	39	40	LCD_DATA7
LCD_DATA4	41	42	LCD_DATA5
LCD_DATA2	43	44	LCD_DATA3
LCD_DATA0	45	46	LCD_DATA1

Figure 2. Beaglebone Black expansion headers.

- 25 PRU low-latency I/O

The BeagleBone Black supports eight types of modes which includes GPIO too.

2.2 Software platform of BeagleBone black: Xenomai kernel 3.x

For BeagleBone Black (NanoPi) we are using Debian distribution of Linux operating system patched with Xenomail kernel 3 in order to reduce latency. Xenomai 3 is the new architecture of the Xenomai real-time framework, which can run seamlessly side-by-side Linux as a co-kernel system like Xenomai 2, or natively over mainline Linux kernels. In our case, the mainline kernel is supplemented by the PREEMPT-RT patch to meet stricter response time requirements than standard kernel preemption would bring. This new architecture that Xenomai 3 brings exhibits two real-time cores, selected at build time. The dual kernel nicknamed Cobalt, is a significant rework of the Xenomai 2.x system. Cobalt implements the RTDM specification for interfacing with real-time device drivers. In comparison to RTL (Real-time Linux) Xenomai performs better in most tasks and offers far less jitter.

2.3 Technical specification of PikoAtlas PLC device

The piko Atlas®-RTL device is miniature modular remote terminal unit (RTU) for data acquisition and control, with possibility of PLC algorithm implementation. It consists of one master CPU module and several I/O slave modules, maximum 8 modules of each type. All modules are connected through

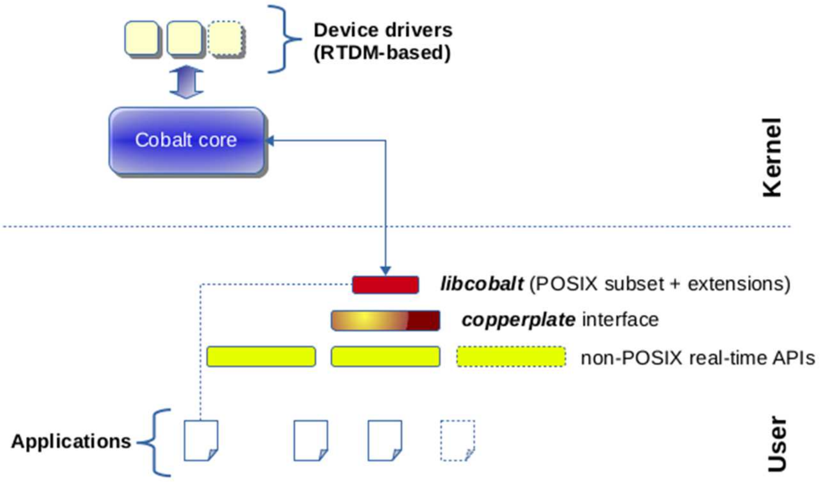


Figure 3. Dual kernel Cobalt architecture.

I2C bus. Piko Atlas®-RTL represents processor module for RTU/PLC device piko Atlas® (or pAtlas). Data concentrator and protocol converter functions for the purpose of connecting different DCS segments, as well as real-time processing of data collected from I/O modules and different slave devices using PLC algorithms created according to IEC 61131-3 standard. Reliable data acquisition from various devices is achieved using different communication directions and protocols. Collected and processed data can be visualised locally (HMI touchscreen) and/or sent to supervising DCS server.
Technical characteristics

- Processor AM3358 1GHz ARM Cortex-A8
- 4GB 8-bit eMMC on-board flash storage
- microSD card (up to 16GB)
- Simultaneous communication with 4 master centres
- 2 x RS-232/485 port (isolated)
- 2 x RS-232 (isolated)
- 1 x CAN (isolated)
- Ethernet port 10/100Mbps

2.4 Development of modules for NeuroArm robot platform

Following modules are being developed: RI8, OQE, DC Motor Driver

2.4.1. Technical specifications of new RI8 (Resistive Input) module

- 8 channels,
- 16bit resolution,
- 1000 measurements per second,
- 0-1kΩ measurement range,
- 2-wire measurement using precision constant current source,
- Communication with CPU module via I2C bus

2.4.2 Technical specification for new OQE (optical quadrature encoder) module

- Supports reading of 2 quadrature encoders with zero mark
- Returns 16 bit value for each encoder
- Communication with CPU module via I2C bus

2.4.3 Technical specification of a new DC Motor Driver module

- Full bridge driver module for driving 4 DC motors
- Maximum voltage 30V DC
- Output current 3.5A
- PWM control
- Supports CW, CCW, short brake and stop modes
- Overcurrent and thermal protection
- Communication with CPU module via I2C bus

2.5 Description of the graphic tool that is used for programming - EDICOPT

EDICOPT is a modern software tool that enables industrial controller’s configuration and programming. The application is designed to facilitate further development of both small and large distributive systems. EDICOPT is a flexible tool that enables the usage of various hardware platforms and operating systems. Automated systems can be further developed so as to fit the Users requirements.
EDICOPT software‘s key characteristics:

- fully supports both Linux and Windows OS,
- simple installation and running of the application,
- intuitive graphics,

- supports MySQL database,
- supports work in the local regime (no database),
- enables configuration, programming, and observing the work of the stations,
- supports IEC 60870-5-101, IEC 60870-5-104, and MODBUS RTU/TCP protocols,
- is compliant with both IEC 61499 and IEC 61131 standards.

EDICOPT components EDICOPT consists of two main components: FBD editor (Figure 5) and the Programming terminal.

FBD editor is a part of EDICOPT software that is used for drawing functional block diagrams. It is made by IEC standards (IEC 61131). This software supports tools that translate ladders to ST code as well as to the code understandable to the station (RTU).

EDICOPT enables a two-way data transfer between multiple sources, which in turn facilitates easier real time data exchange, regardless of the number of controllers – be it one in the entire project or several distributed via a local network.

FBD editor's tools are as follows:

- control logic editing,
- debugging,
- ST code generating,
- documenting the version and authors, diagram printing,
- database editing,
- downloading the programs to Atlas Max-RTL® controllers,
- communication pathway settings,
- online supervision and simulation.

The Programming terminal is a software that enables the User to connect to Atlas Max® and Atlas Max-RTL® workstations online and to supervise the operations. The program is developed at the Mihajlo Pupin Institute for the purpose of online operational diagnostics of Atlas® workstations. It is designed to communicate with the workstations via Atlas-IEC protocol. The User can supervise the workstation with which they communicate and monitor the change in acquired or computed values via the graphics interface. An application that allows the User to control the system resources and the input/output controllers of the Atlas family is incorporated in the Programming terminal. The application fully adheres to all standard configuration parameters: time and networking data, basic system services, as well as the coupling input/output data.

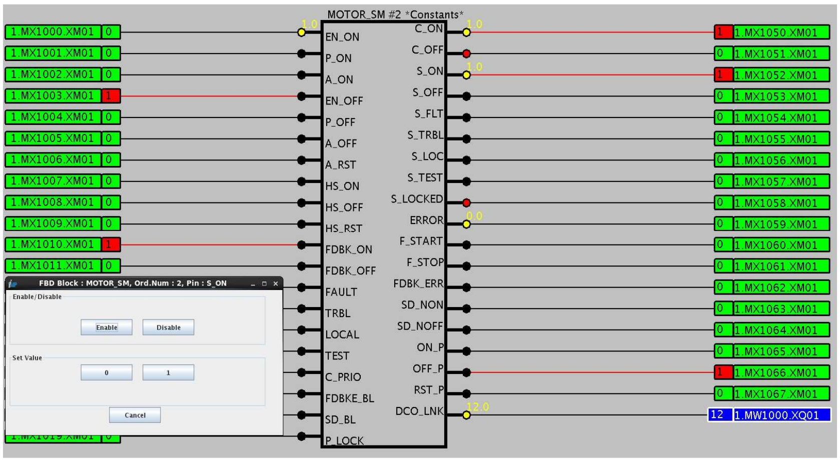


Figure 4. Function Block Diagram (FBD).

3. Implementation of new controller and programming logic

In order to make everything working we must start with one DC motor, because the same way will be implemented to all other motors. First step for connecting to one DC motor is to check available

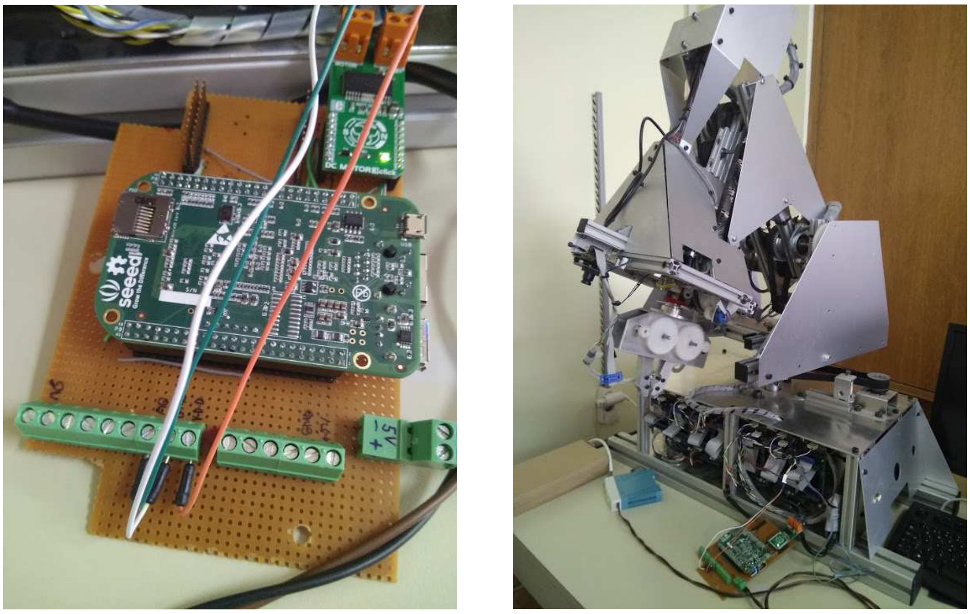


Figure 5. Connection of development protoboard to NeuroArm robot DC motor.

connections and wires from existing control system. Wires has been tested and removed from old control system and connected to new control system, (Figure 5).

Once that setup is done, we could start testing control of DC motor control. Position data is acquired from 1kΩ resistive sensors using RI8 (Resistive module) module and from optical quadrature encoders using one DI16 (Digital Input) module.

4. Software control

XBB PLC logic has been created using EDICOPT application using function block diagram (FBD) by using simple ON/OFF control logic. FBD is shown in Figure 6.

Figure 6. Simple ON/OFF control logic.

Variable 1.MW01.XQ01 is control signal and it defines rotation direction, turn off, or break depending on its value:

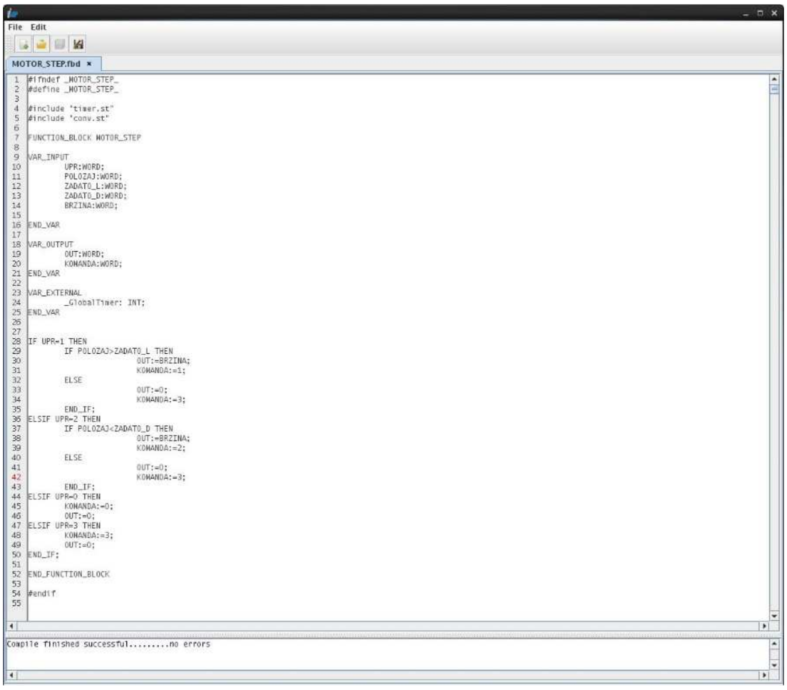
- 0 is turn off
- 1 is go to left
- 2 is go to right
- 3 is break

A01-0101 is a variable that gets value of encoder position

By variables 1.MW002.XQ01 and 1.MW003.XQ01 are defined the limits of the right and left position. Variable 1.MW005.XQ01 defines speed.

Exit of the block Q01_0101 is the value of predefined speed and control signal value is in variable 1.MW004.XQ01.

Block functions that depending on a given rotation direction and position it moves and stops motor. Block coding is written in ST language and is shown on Figure 7.



```
#ifndef _MOTOR_STEP_
#define _MOTOR_STEP_

#include "timer.st"
#include "conv.st"

FUNCTION_BLOCK MOTOR_STEP

VAR_INPUT
        UPR:WORD;
        POLOZAJ:WORD;
        ZADATO_L:WORD;
        ZADATO_D:WORD;
        BRZINA:WORD;

END_VAR

VAR_OUTPUT
        OUT:WORD;
        KOMANDA:WORD;
END_VAR

VAR_EXTERNAL
        _GlobalTimer: INT;
END_VAR


IF UPR=1 THEN
        IF POLOZAJ>ZADATO_L THEN
                        OUT:=BRZINA;
                        KOMANDA:=1;
        ELSE
                        OUT:=0;
                        KOMANDA:=3;
        END_IF;
ELSIF UPR=2 THEN
        IF POLOZAJ<ZADATO_D THEN
                        OUT:=BRZINA;
                        KOMANDA:=2;
        ELSE
                        OUT:=0;
                        KOMANDA:=3;
        END_IF;
ELSIF UPR=0 THEN
        KOMANDA:=0;
        OUT:=0;
ELSIF UPR=3 THEN
        KOMANDA:=3;
        OUT:=0;
END_IF;

END_FUNCTION_BLOCK

#endif
```

```
Compile finished successful.........no errors
```

Figure 7. ST coding for motor control.

5. Conclusion

After successfully implementing control of a motor with a simple FBD and using one DC motor, same HW is used in order to control all 7 motors of NeuroArm robot, with a following changes. In order to make new NeuroArm control platform easy expandable with possibility to add all the sensors for future needs, it is agreed to use modular design. PikoAtlas CPU modul have two variants for future implementation. One is with Beaglebone Black and another one is with NanoPi. That way, for every additional Analog or Binary I/O new module will be connected to PikoAtlas expanding NeuroArm robot platform possibilities. New software platform that is based on Debian Linux gives possibility to add any USB peripheral that is supported like external HDD for storing data, IP cameras for remote control, and many others. This way, newly created NeuroArm robot platform will be base for additional improvements of robotic arm algorithms by researchers, for example, using Matlab Simulink and making all their ideas to come true, allowing them to focus on specific tasks without losing time on preparing HW control, but just simple adding additional modules and use them by only coding.

Acknowledgment

Authors gratefully acknowledge the support of Ministry of Education, Science and Technological Development of the Republic of Serbia under the project TR 35006 (M.P.L.) as well as supported by project TR 33047 (P.D.M.)

References

[1] http://www.neurorobotics.co.uk
[2] Čović V and Lazarević M 2009 *Robot Mechanics*, Faculty of Mechanical Engineering in Belgrade
[3] Siciliano B, Sciavicco L, Villani L and Oriolo G 2009 *Robotics*, Springer-Verlag
[4] Angeles J 2007 *Fundamentals of Robotic Mechanical Systems: Theory, Methods, and Algorithms*, Springer
[5] Mandić P, Lazarević M, Stokić Z and Šekara T 2017 *Dynamic Modelling and Control Design of Seven Degrees of Freedom Robotic A*rm, 6th International Congress of Serbian Society of Mechanics, Tara, Serbia, June 19-21, C1d-1–C1d-8