**Разработване на 3D принтирана роботизирана ръка за жестове**

Радослав Илиев, Стефани Минчева

"Професионална гимназия по компютърни технологии и системи" - гр. Правец

Научни ръководители: доц. Иван Чавдаров, инж. Венцислав Начев

***Резюме:*** *Представя се една разработка на хуманоидна ръка предназначена за изпълняване на жестове. Конструкцията е оригинална, като се прилага подход за отпечатване на „директно сглобени пръсти” на ръката, което възможно само ако се използва технологията за 3D принтиране. За управлението и генерирането на набор от жестове се ползва „сензорна ръкавица”. Разработката намира приложение като помощно средство при комуникация с хора с увреждания. Роботизираната ръка може да се ползва например, в съдебни или учебни зали и други.*

***Abstract:*** *A design of humanoid robotics arm for gesture realization is presented. This is an original construction where directly assembled units are used. The realization and application of these units is only possible by using 3D printing technologies. A sensor glove is used for control and gesture recognition. This are could be used in communication with people with difficulties in their development. Another application could be in courts or for educational purposes.*

**Увод**

Комуникацията между Човек и робот е обект на голям интерес от изследователската общност. На сегашния етап от развитие на техниката взаимодействието между робот и човек се осъществява основно чрез дисплей и клавиатура по рядко с речево разпознаване. Хората взаимодействат по между си като ползват комплекс от средства за комуникация които включват реч, писменост, жестове, мимики и други. Тази разработка е насочена към повишаване на възможностите за взаимодействие между робот и човек.

Известни са множество разработки на хуманоидни ръце които са предназначени за създаване на протези. Други проекти се занимават със създаването на хващащи устройства за роботи с геометрия и кинематика близка до човешката ръка. Известен е и проект на студенти от университета в Антверпен, Белгия, наречен ASLAN (Antwerp’s Sign Language Actuating Node) те се надяват той да помогне на хора с увреден слух. Проектът се финансира от Европейския институт по отоларингология**.**

Първият прототип на ASLAN се състои от 25 напечатани на 3D принтер детайли, 16 сервомотора, 3 контролера и Arduino-съвместима платка. Конструирането на робота е отнело само 10 часа, докато разработката е продължила цели три години, отбелязва SlashGear.com. Бързото сглобяване и достъпността на компонентите на роботизираната ръка са особено важни за успеха на ASLAN. Работата по ASLAN все още е далеч от завършване. На следващ етап студентите ще добавят втора ръка с цел повишаване на скоростта на общуване и планират да внедрят възможности за обучаване на робота с нови жестове посредством уеб камера. <https://novotopoznanie.com/robotizirana-raka-shte-prevrashta-govor-v-zhestove/>

Целта на настоящия проект е разработването на роботизирана ръка за изпълняване на жестове. Първоначалната идея е да се разработи механична ръка (аналог на човешката), включваща длан с пет пръста, която да може да изпълнява голям брой от най-често ползваните жестове.

1. **Изисквания и проблеми при създаването на хуманоидна ръка за жестове.**

След обстойно проучване на литературни източници за конструкции и приложения на роботизирана хуманоидна ръка предназначена за изпълняване на жестове, резултатите могат да се обобщят:

*Проблеми, свързани с механичната конструкция.*

Един от основните проблеми при проектирането на хуманоидна длан с пръсти е големия брой степени на свобода (24), концентрирани в малък обем (около 500 кубични сантиметра) и изискване за малка маса (до около 500 грама). Развитието на технологиите в задвижващите механизми през последните години позволява изграждането на миниатюрни компоненти, но все още не е възможно разполагането на такъв голям брой адекватни механизми в ограничения обем на ръката. Освен това не е лесно и синхронното управление на голямото количество електродвигатели. Очевидно е необходимо да се направи разумен компромис с броя на задвижващите устройства. Необходимо е адекватна елементна база, преди всичко от материали, двигателни устройства, средства за предаване на движение на разстояние и комплексни сензори за възприятие на различните функционални дейности на ръката. Търсят се енергийни източници с малки размери, дълготрайност и лесно възстановяване.

Съществува голямо разнообразие на действията, които човек извършва чрез ръката, което е сложно да се реализира от механична конструкция.

*Проблеми, свързани с управлението на ръката.*

Голяма част от жестовете са свързани със сложни координирани движения на пръстите. В някои случай е необходимо да се опишат траектории с пръстите или комплицирано управление по скорост. Необходимо е ползването на прецизни серводвигатели и специализирани алгоритми за управлението им.

*Проблеми, свързани с управлението на сервомоторите*

Трябваше да се подбере подходящ микроконтролер, който предлага паралелно управление на поне 6 сервомотори, има удобен интерфейс за комуникация с компютър, и е достатъчно надежден за безпроблемна работа. Важен фактор беше и компактността на управляващата моторите печатна платка, като той трябваше да е може да се монтира на или близо до роботизираната ръка. Наложи се да се използва и външно захранване, поради голямата консумация на енергия от сервомоторите и недостатъчния ток от управляващият USB интерфейс.

*Проблеми, свързани с надеждността на елементите.*

Конструкциите са сложни и изградени от голям брой елементи, с цел да могат да се постигната желаните пространствени движения, което е предпоставка за понижаване на надеждността. За предаване на движенията на по-голямо разстояние често се ползват нишки и еластични елементи, които имат склонност да променят качествата си с течение на времето.

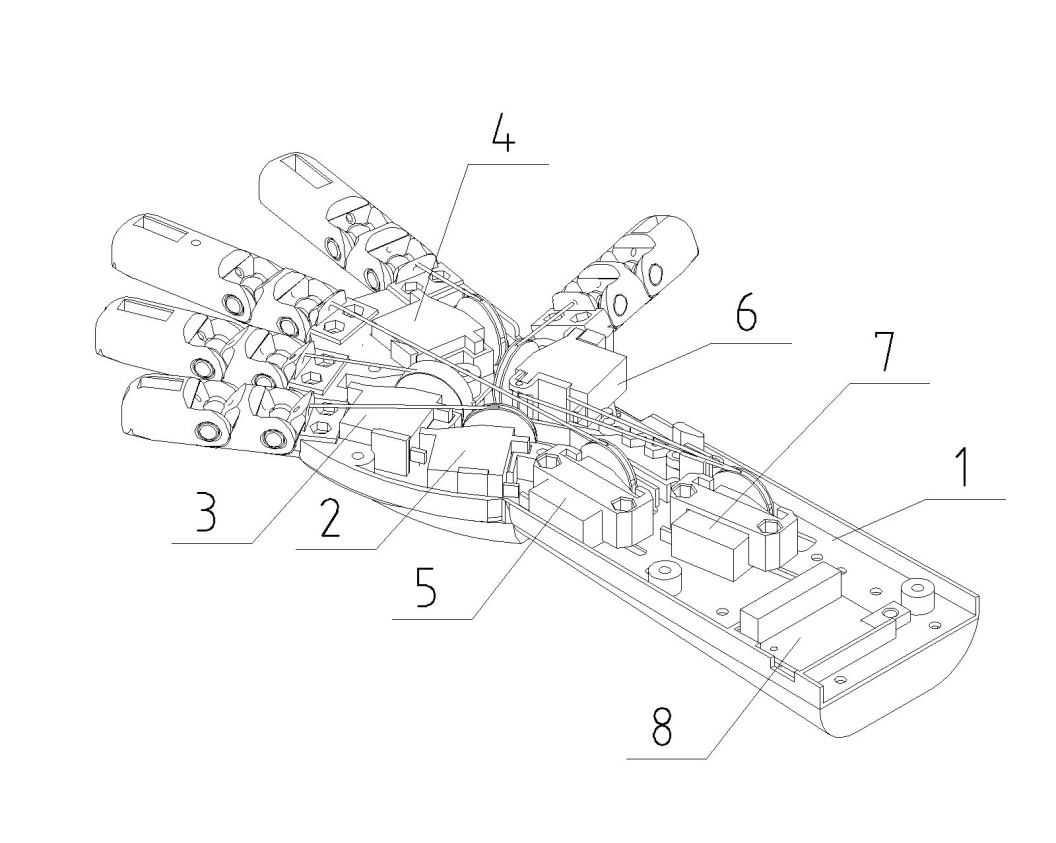
*Безопасно ползване и подходящ дизайн за лесно възприемане.*

Предназначението на модела обуславя необходимостта от безопасно боравене с него. Очевидно освен изискванията свързани с функционалните възможности на ръката за възпроизвеждане на голям брой жестове, е важно и естетическото възприемане на модела. Добре е пръстите да са леко разтворени, за да се виждат ясно от по-голямо разстояние.

1. **3D моделиране на роботизираната ръка.**

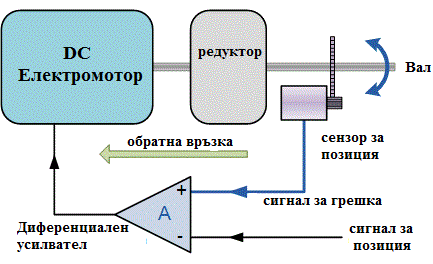
Разработва се модел на ръка, включваща дланта с пръстите. Създадени са 3D CAD модели на различни варианти за пръстите. Дискутират се механизми за предаване на движението от сервомоторите до пръстите.

Създаден е прототип при който всеки пръст да може да се задвижва от независим сервоуправляем двигател и се търси просто конструктивно решение Фиг.2. Прототипа е базиран на предишна разработка с автор един от консултантите като са извършени някои подобрения в механичната конструкция и е създаден изцяло нов хардуер и софтуер за управлението му.



Фиг.2. 3D модел на хуманоидна ръка с независимо задвижване на всичките пет пръста.

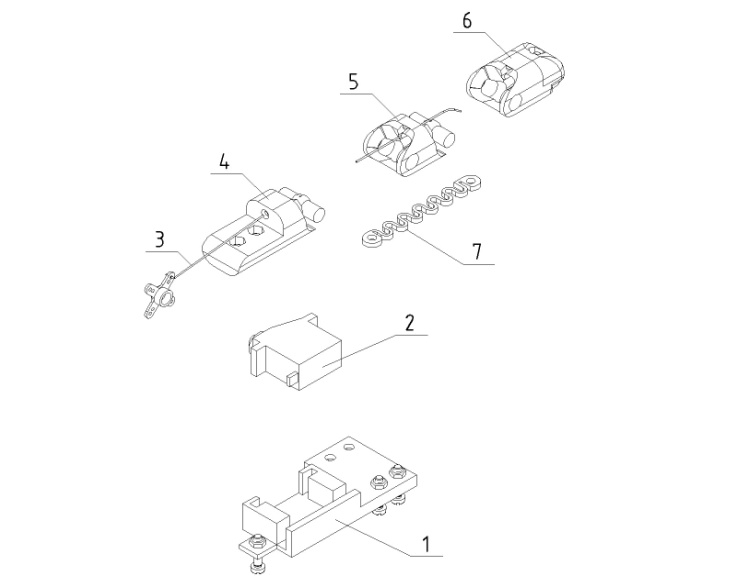
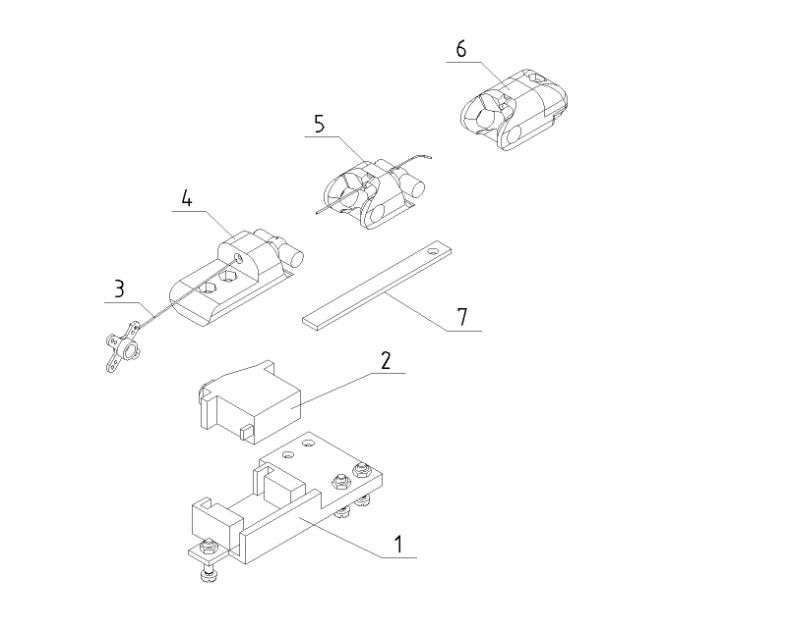
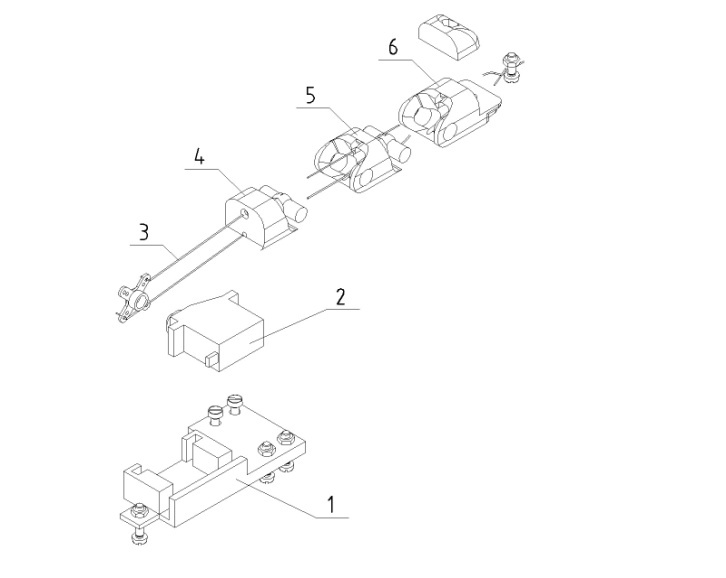
Върху основата 1 на ръката се разполагат сервомоторите (2-7), които имат вградени редуктори (MIRCO SERVO, Model No. HD-1581HB, Reduction ratio 1/522). Сервомоторът е вид електродвигател, чиято позиция на въртне на оста може да бъде контролирана с голяма прецизност. Обикновено сервомоторите са с малки габарити и мощност. Важни характеристики на сервомотора са също масата, динамиката на двигателя, равномерността на движение и ефективността. Сервомоторите се използват широко в промишлеността, например, в металургията, в автомобилостроенето, робототехниката, металообработващите машини, космическата и авиационна промишлености т.н. Думата „Серво” произлиза от латинското *servus*, което се превежда като слуга, помощник. В този смисъл серводвигателят е изпълнителен механизъм.



Фиг.3 блокова схема на сервомотор

Двигателите 2-5 задвижват посредством система от нишки по един пръст от ръката, а двигатели 6 и 7 задвижват палеца. Модула за управление 8 е разположен в основата 1. Така предложената система има по-добри функционални качества и може да реализира множество жестове.

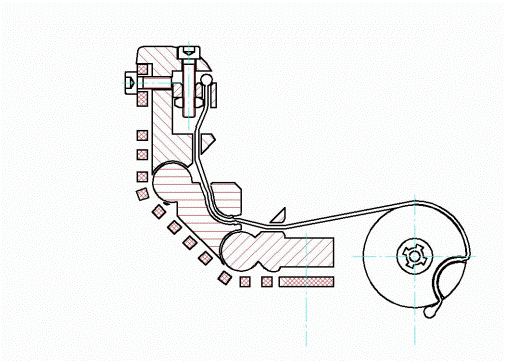
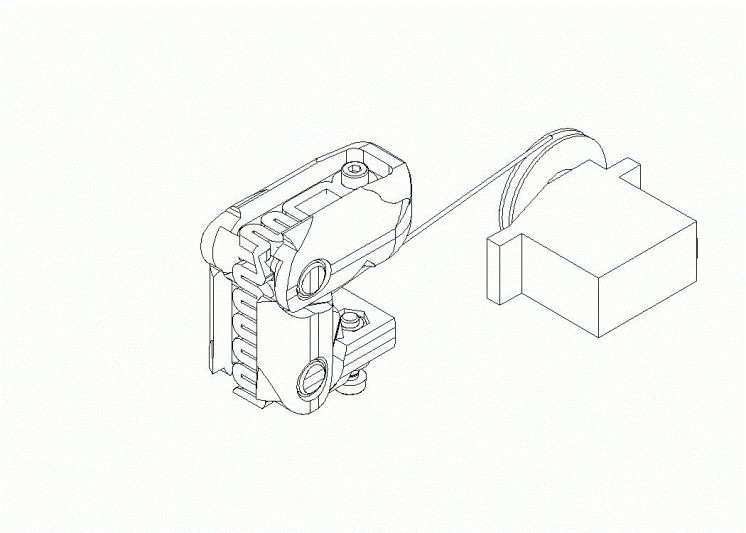
Реализирането на тази идея не е проста задача. За целта ползваме различни нови материали и технологии. Основния проблем е свързан с разположението на миниатюрната система от задвижващи компоненти, които да реализират надеждно задвижване и прецизно управление в един изключително малък обем. За целта експериментираме с различни концепции за задвижване на пръстите. Всички нестандартни елементи се отпечатват с помощта на 3D принтер. За отпечатване на еластичните елементи се ползва специализиран материал Filaflex (<https://www.youtube.com/watch?v=Vmb9iwFpaOs>). Реализирани са няколко варианта на пръсти и се извършиха експерименти за адекватността на конструкциите (Фиг.5). Част от звената на пръста се отпечатват направо сглобени, което улеснява монтажните дейности и опростява реализацията на конструкцията. Върху стенда (основа) 1 се монтира сервомотор, които задвижва пръст. Двигателя 2 задвижва посредством нишка 3 звената 5 и 6 на пръстите. При вариант а) задвижването в едната посока, както и връщането на пръста, става с помощта на две нишки (поз.3). При втория вариант b) връщането в изходна позиция на пръста се осъществява с помощта на еластична пластина 7 отпечатана с Filaflex, а при вариант c) този елемент е модифициран в пружина 7, която може да се деформира в пространството.



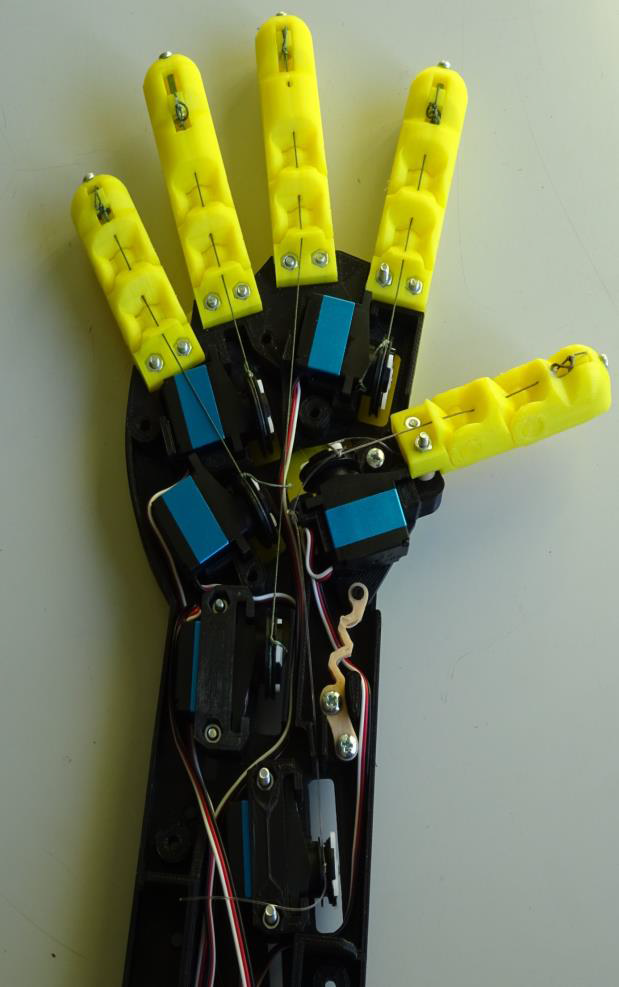
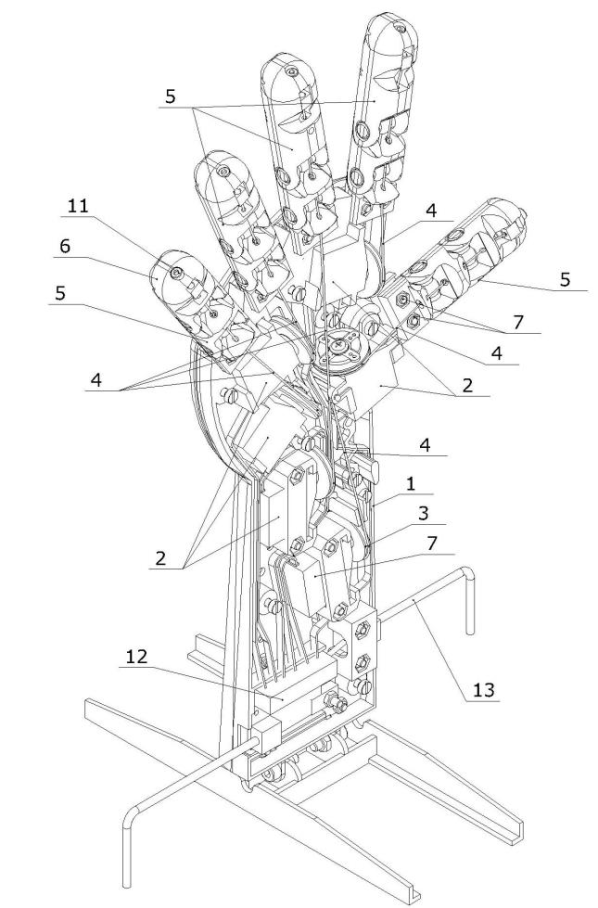
1. b) c)

Фиг.4. Различни варианти за задвижване на пръстите на роботизираната ръка.

Най добри резултати се получават при ползване на вариант c). В първия случай -а) е много трудно да се осъществи постоянно и еднакво опъване в двата клона на нишката, което от своя страна довежда до нежелан свободен ход при смяна на посоките на движение. Във втория случай се оказа, че Filaflex няма достатъчна еластичност за правилното функциониране на пръста. След отпечатване на определена форма от Filaflex с помощта на 3D принтер този материал се стреми да запази тази форма независимо от прилаганите му деформации. Това качество на материала използваме успешно в последния вариант за пръст на роботизираната ръка.

** **

Фиг.5. Схема на задвижване за всеки един от пръстите на ръката

** **

Фиг.6. Общ вид на 3D принтираната ръка за жестове

1. **Управление на роботизираната ръка**

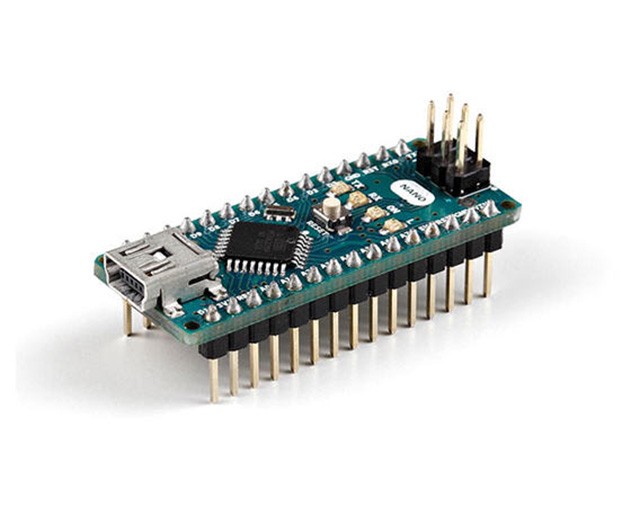
**Управлението се извършва чрез Arduino развойна платка.**

Ардуино е проект с отворен код и едноименна компания, произлязла от него, както и свързаното с него потребителско общество. Основната цел е проектиране и производство на електронна платформа с лесен за ползване свободен хардуер и софтуер. Проектът е основан на семейство платки с микроконтролери, произвеждани главно от SmartProjects, Италия, както и редица други доставчици, които използват различни 8-битови (AVR) микроконтролери или 32-битови (ARM) процесори Atmel. Осигурени са групи от цифрови и аналогови пинове за вход-изход (I/O), които позволяват свързване с други платки и вериги. Платките включват сериен комуникационен интерфейс, а при някои модели и USB, за зареждане на програми. С оглед програмирането на микроконтролерите Ардуино платформата предоставя интегрирана среда за разработка (IDE – Integrated Development Enviornment), основана на проекта Processing, който поддържа програмните езици C и C++. Първата платка Ардуино е представена през 2005г. Екипът на проекта иска да предостави на любители, ученици, студенти и професионалисти евтин и лесен начин да създават устройства, способни да взаимодействат с околната среда чрез сензори и изпълнителни устройства.

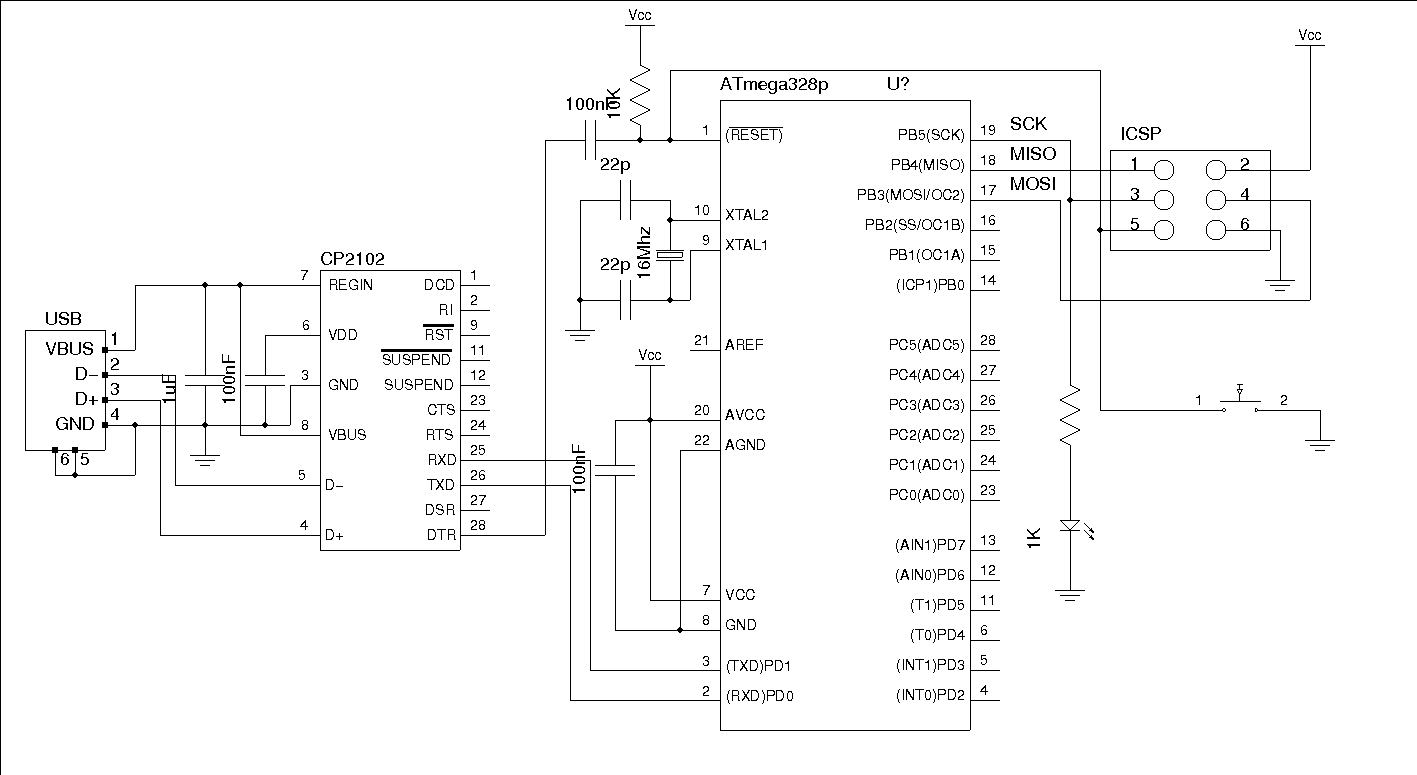
Съществуват няколко различни модела на Ардуино, като те се различават основно по изчислителна мощност, брой входно-изходни пинове, големина на платката, цена. На линка <https://www.arduino.cc/en/Main/Products> можете да откриете всички Arduino платки.

**За управлението на роботизираната ръка е използван Arduino Nano (Фиг.6, Фиг.7) развойна платка с процесор Atmel AtMega 328P, който има следните характеристики:**

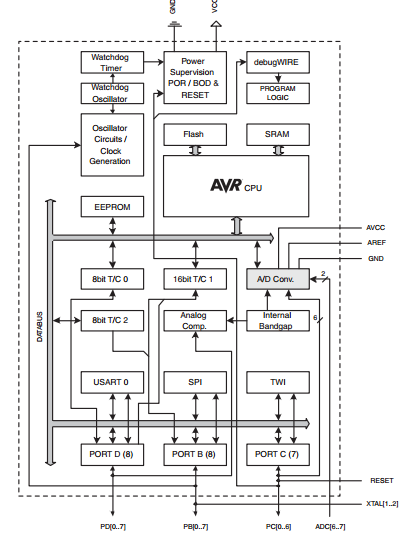
* Разредност: 8 bits;
* Tактова честота: 0-20 MHz (16МHz);
* Производителност: до 20MIPS (~16MIPS);
* Размер на оперативната памет, RAM: 2kB;
* Размер на програмната памет, Flash: 32kB;
* Размер на постоянната памет, EEPROM: 1kB;
* Входно-изходни портове:  20 броя, от които:
* Цифрови портове: до 20 бр.
* Аналогови входове: до 8 бр.
* PWM (ШИМ): до 6 бр.;
* Комуникационни интерфейси: UART, SPI, I2C (TWI);
* Захранващо напрежение: 4.5-5.5V (5V);



Фиг. 6 Arduino Nano



Фиг.7 Електрическа схема на Arduino Nano

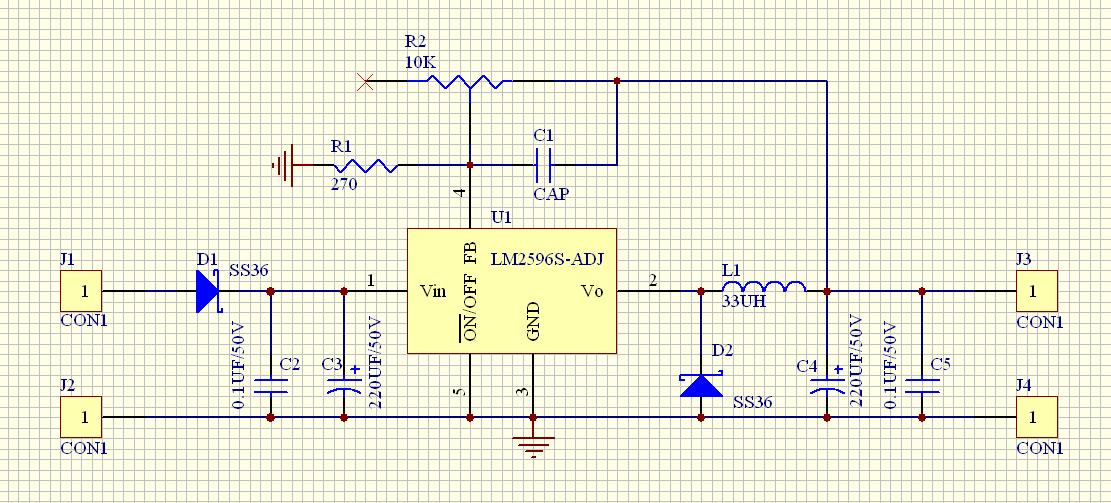


Фиг.8 Блокова схема на AtMega 328P

* **AVR CPU** – **C**ental **P**rocessing **U**nit – мозъкът на микроконтролера, който извършва аритметични и логически операции. Архитектурата е RISC – Reduced Instruction Set Computing.
* **SRAM** – **S**tatic **R**andom **A**ccess **M**emory – енергозависима памет, която служи за съхраняване на временна информация в процеса на работа на програмата.
* **Flash** – постоянна(енергонезависима) памет, в която се съхранява програмата, която изпълнява микроконтролера.
* **EEPROM** – **E**lectrically **E**rasable **P**rogrammable **R**ead **O**nly **M**emory - постоянна(енергонезависима) програмно достъпна памет.
* **Т/C** – **T**imer/**C**ounter – служи за измерване на времеви интервали / броене на електрически импулси;
* **PORT B/C/D** - програмно достъпни регистри, които управляват входно-изходните пинове с общо предназначение (GPIO).
* **Oscillator** – Осцилатор/Генератор – служи за генериране на тактови импулси, които синхронизират работата на всички подсистеми в микроконтролера (“диригента на орекестъра”);
* **Watchdog** – стражеви таймер - предпазва програмата от забиване, като прави Reset(пускане на програмта отначало).
* **Analog Comp** – аналогов компаратор – сравнява две напрежение.
* **Power Module** – грижи се за подаване на токозахранване за всички модули в микроконтролера.
* **debugWire** – специален комуникационен протокол на Atmel AVR, за on-chip дебъгване (процес на проследяване изпълнението на програмата, с цел отстраняване на грешки - “бъгове”).
* **USART** – **U**niversal **S**ynchronous/**A**synchronous **R**eceiver **T**ransmitter – Универсален Синхронен/Асинхронен Приемо-Предавател – сериен интерфейс за комуникация на микроконтролера с компютър или други други микроконтролери и елементи;
* **SPI** – Serial Peripheral Interface – комуникационен интерфейс предназначен за комуникация с други интегрални схеми и устройства. Колкото повече устройсва свързваме, толкова повче на брой връзки са необходими;
* **TWI** – **T**wo **W**ire **I**nterface (I2C) – синхронен комуникационен интерфейс само с 2 връзки(жици) предназначен за комуникация с други интегрални схеми, устройства и системи;

**Захранващ блок:**

Управлението на захранващото напрежение на платката е реализирано чрез Step-down конвертор(Фиг.9, Фиг.10), който позволява входното напрежение да е в границите от 6 до 32V, като изходното му напрежение не се променя и е оптималното за нормалното функциониране на микроконтролера и сервомоторите (5V).



Фиг. 9 и Фиг. 10 Step-down converter и ел. схема

**Описание на сензорната ръкавица и софтуер към нея**

Ръкавицата за данни DG5 VHand 3.0 (Фиг.11) е иновативен датчик за откриване на движение на ръката. Благодарение на петте си вградени сензора за свиване, е възможно точното измерване на движенията на пръстите, докато вградения Invensense ™ 9 оси (3-осов акселерометър, 3-осов byroscope и 3-осов магнитометър) сензор за движение позволява да измерите движенията на ръката, както и нейната ориентация (наклон).

Ръкавицата комуникира с външни устройства чрез USB кабел или WiFi връзка. Ако се използва USB връзка, трябва да бъде инсталиран правилният FTDI драйвер (http://www.ftdichip.com/FTDrivers.htm), докато WiFi комуникацията изисква WiFi маршрутизатор и компютър с WiFi поддръжка.

WiFi комуникацията използва стандартни TCP IP сокети, стандартният порт е 2000. Използването на WiFi комуникацията, вместо Blueetooth, позволява използването на няколко ръкавици за данни в една и съща мрежа, без да се намалява скоростта и ефективността . Ръкавицата може да се захранва директно от USB или да се използва вътрешната и батерия. Вътрешната 3.7V 300mAh батерия гарантира около 4 часа работа, като тя може да се зареди с предоставения USB кабел.

Ръкавицата за данни може да се използва в различни приложения: роботика, виртуална реалност, иновативни игри, рехабилитация, а също и като новаторска помощ за хората с увреждания. Иновативният дизайн позволява на потребителите да премахнат напълно всички сензори и електронната платка, така че ръкавицата да бъде лесно почистена или променена. Използва се вътрешен алгоритъм за управление на 9-те сензора, който осигурява стабилна и точна информация за позицията на ръката.



Фиг.11 Ръкавицата, използвана за управление на роботизираната ръка

Техническите данни на ръкавицата са:

* Заряден ток: максимум 220mA, 70mA в стандартен режим на работа;
* Работна температура: от 0 до 50 ° C
* Скорост на взимане на данни: 10 - 100Hz (персонализирано)
* Брой сензори за пръсти: 5 (по един на пръст);
* Вграден цифров акселерометър с три оси;
* Изходен порт: micro USB;
* Квалифициран 2.4-GHz IEEE 802.11b / g предавател;
* Поддържана скорост на данни 1 Mbps с TCP / IP и WPA2;
* FCC / CE / ICS сертифицирани и RoHS съвместими;

1. **Алгоритъм за реализиране на жестове**

Алгоритъмът записва в текстови файл символи по ASCII таблица от 0 до 100, които се прочитат от Processing програмата и се препращат по сериен интерфейс до роботизирана ръка, която от своя страна записва стойностите на символите като позиция на сервомоторите.

1. **Заключение**

Усъвършенствана е конструкцията на 3D принтирана хуманоидна ръка. Тествани са различни варианти за задвижване на пръстите и като за целта е създаден опитен стенд за изпитания. Създадена е изцяло нова система за управление на ръката реализирано чрез използването на сензорна ръкавица.

Бъдещото развитие на проекта включва подобрение на механичната и управляващата система на ръката, както и намиране на нови практически приложения.

**6. Използвана Литература**

[1] Kun Qian, Jie Niu and Hong Yang, Developing a Gesture Based Remote Human-Robot Interaction System Using Kinect, International Journal of Smart Home Vol. 7, No. 4, July, 2013, 203-208

[2] Oliver Brand, Max Lungarella and Davide Scaramuzza, Kinect-aided Robust Gesture Recognition for Human-Robot Interaction with Application to Quadrocopter Control, http://www.merlin.uzh.ch/publication/show/9149

[3] Biao MA, Wensheng X ,Songlin WAN, A Robot Control System Based on Gesture Recognition Using Kinect, TELKOMNIKA, Vol. 11, No. 5, May 2013, pp. 2605 ~ 2611, e-ISSN: 2087-278X, 2605-2611

[4] Z. Ren, J. Yuan, J. Meng, and Z. Zhang, “Robust part-based hand gesture recognition using kinect sensor,” IEEE Transactions on Multimedia, vol. 15, no. 5, pp. 1110–1120, 2013.

[5] Чавдаров И., Павлов В., Вацкичев А., Николов В., „Ръководство за проектиране на роботи” , издателство ТУ-София, Факултет Автоматика, катедра АЕЗ, секция Роботика, София, 2009г.

**Приложение 1. Ардуино сорс код**

#include <Servo.h>

const byte numChars = 20;

byte receivedChars[6];

Servo palec,pokazalec,sreden,bezimen, kutre;

boolean newData = false;

boolean recvInProgress = true;

void setup() {

Serial.begin(9600);

Serial.println("<Arduino is ready>");

palec.attach(7);

pokazalec.attach(6);

sreden.attach(5);

bezimen.attach(4);

kutre.attach(3);

}

void loop() {

//recvWithStartEndMarkers();

Serial.readBytes(receivedChars,5);

delay(10);

Serial.print(receivedChars[0]);

pokazalec.write(receivedChars[1]);

Serial.print(int(receivedChars[1]));

sreden.write(receivedChars[2]);

Serial.print(int(receivedChars[2]));

bezimen.write(180-receivedChars[3]);

Serial.print(int(receivedChars[3]));

kutre.write(receivedChars[4]);

Serial.println(int(receivedChars[4]));

recvInProgress = true;

}

void recvWithStartEndMarkers() {

static boolean recvInProgress = false;

static byte ndx = 0;

char startMarker = 'n';

char endMarker = 'k';

char rc;

// if (Serial.available() > 0) {

while (Serial.available() > 0 && newData == false) {

rc = Serial.read();

if (recvInProgress == true) {

if (rc != endMarker) {

receivedChars[ndx] = rc;

ndx++;

if (ndx >= numChars) {

ndx = numChars - 1;

}

}

else {

receivedChars[ndx] = '\0'; // terminate the string

recvInProgress = false;

ndx = 0;

newData = true;

}

}

else if (rc == startMarker) {

recvInProgress = true;

}

}

}

**ПРИЛОЖЕНИЕ 2 - Processing код**

import processing.serial.\*;

Serial myPort;

void setup(){

size(600,600);

myPort=new Serial(this,Serial.list()[0],9600);

}

void draw(){

background(0,200,0);

String values[]=loadStrings("myfile.txt");

text(values[values.length-1],200,200);

myPort.write(values[values.length-1]);

}

**ПРИЛОЖЕНИЕ 3 – API код за ръкавицата**

#include "stdafx.h"

#include <stdio.h>

#include <string>

#include <Windows.h>

//include dataglove class

#include "VHand30.h"

FILE \* pFile;

int \_tmain(int argc, \_TCHAR\* argv[])

{

pFile = fopen ("C:\\Users\\x\\Desktop\\Other\\vhandddd\\CD\_VHand\_3\_0\_SDK\\CD VHand 3.0 SDK\\VirtualHand02\\VirtualHand02\\ruka\\myfile.txt" , "w");

//create a new dataglove instance

VHand30 \*dataglove = new VHand30;

//set dataglove connection parameter (COM port, IP address is not important)

dataglove->SetConnectionParameters(7,"127.0.0.1");

//connect with the USB connection, created comport = 1

int ret = dataglove->Connect(CONN\_USB,STREAM\_FINGERS\_QUATERNION);

fprintf(stderr,"CONNECT RET: %d\n",ret);

//wait for connection established

long start = ::GetTickCount();

while ((dataglove->Connected == NOT\_CONNECTED))

{

if ((::GetTickCount()-start)>5000){

//timeout, error with the connection

fprintf(stderr,"Cannot connect to the dataglove\n");

dataglove->Disconnect();

return 0;

}

}

//show dataglove informations

char label[VHAND\_STRLEN];

int ID=0;

dataglove->GetID(label,&ID);

fprintf(stderr,"LABEL:%s ID:%d\n",label,ID);

//firmware version

int fw1,fw2,fw3;

dataglove->GetFWVersion(&fw1,&fw2,&fw3);

fprintf(stderr,"FIRMWARE:%d.%d.%d\n",fw1,fw2,fw3);

//access point configuration

char ssid[VHAND\_STRLEN], pwd[VHAND\_STRLEN];

dataglove->GetAPNSettings(ssid,pwd);

fprintf(stderr,"SSID: %s PWD:%s\n",ssid,pwd);

//show network information

char ip[VHAND\_STRLEN], gw[VHAND\_STRLEN], nm[VHAND\_STRLEN];

int DHCP;

dataglove->GetWiFiSettings(ip,nm,gw,&DHCP);

fprintf(stderr,"IP: %s NM:%s GW:%s DHCP:%d\n",ip,nm,gw,DHCP);

while (true){

Sleep(100);

int connstatus = dataglove->GetConnectionStatus();

//is the dataglove conected with the USB port?

if (connstatus == USB\_CONNECTED){

double fing[5];

double roll,pitch,yaw;

dataglove->GetFingers(fing);

dataglove->GetAttitude(&roll,&pitch,&yaw);

unsigned int time = dataglove->GetLastPackageTime();

fprintf(stderr,"TIME: %04d\n",time);

for(int i=0;i<sizeof fing/sizeof fing[0];i++)

{if(fing[i]>99)fing[i]=99;}

if(fing[2]>89){

fprintf(pFile,"1\n");}

else{

fprintf(pFile,"2\n");

}

fprintf(stderr,"ROLL:%.1f PITCH:%.1f YAW:%.1f\n",roll,pitch,yaw);

}

}

//disconnect socket

dataglove->Disconnect();

Sleep(1000);

//turn off module

dataglove->TurnOFF(CONN\_WIFI);

Sleep(1000);

//exit

return 0;

}