МБОУ лицей №21 города Кузнецка.

Научно-практическая работа по информатике

на тему:

«Мобильный 3х-осевой токарный (роторно-фрезерный) станок, оборудованный Числовым Программным Управлением (ЧПУ)»

Выполнил:

Столяренко Даниил Владимирович

учащийся 11 «А» класса

Руководитель:

Кузнецов Алексей Александрович

учитель информатики

2020 г.

Оглавление.

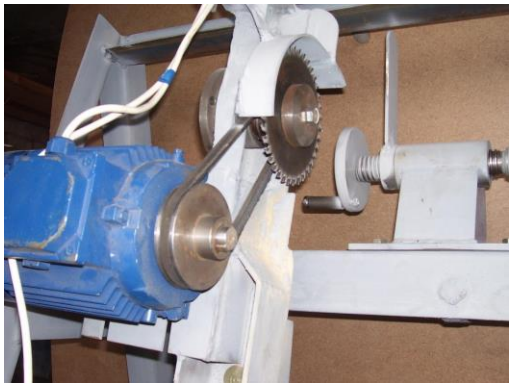
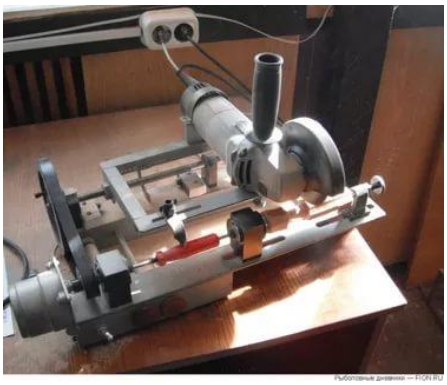
1. Введение

Актуальность и значимость моей работы заключается в том, что малогабаритный токарный (роторно-фрезерный) станок с ЧПУ является уникальным устройством, имеющим опции шуруповёрта наряду с полноценным функционалом токарных и фрезерных станков.

Станок похож на современный шуруповёрт, независимый от сети 220V, имеющий малые габариты, расширяющие возможности его применения. Данные свойства упрощают работу со станком, превращая сложный и массивный станок в мобильный и легкоуправляемый инструмент.

Сначала необходимо было выбрать тип станка. Пересмотрев большое количество вариантов самодельных токарных деревообрабатывающих станков в интернете, пришёл к выводу, что у всех есть свои недостатки: трудоёмкая в изготовлении конструкция, отсутствие ТБ для работы оператора. Современные самодельные токарные деревообрабатывающие станки используют вместо резцов, какой либо электроинструмент (ручная дрель, угловая шлифовальная машина, шуруповёрт).



Для создания нового принципа работы токарных станков, имеющим опции шуруповёрта (малый вес, аккумуляторный ИП), наряду с полноценным функционалом токарных станков, был сконструирован новый вид станка.

Изюминка конструкции моего станка заключается в его мобильной конструкции с автономным источником питания(12V), совмещающей в себе функции токарных и фрезерных станков, что делает его незаменимым для малого бизнеса в помещениях без электросети, но альтернативными источниками питания: солнечными батареями, ветро и гидро-генераторами.

Цель проекта:

Разработать и изготовить малогабаритный токарный (роторно-фрезерный) станок для обработки древесины, сочетающий в себе: ширину применения, сопоставимую с шуруповёртом, удобство в транспортировке и автономный источник питания (12 вольт). Написать программное обеспечение (ПО) для работы станка и взаимодействия оператора с ним.

Задачи проекта:

1. Сконструировать малогабаритный токарно (роторно-фрезерный) станок, питаемый автономным источником питания (ИП) (12 вольт).
2. Разработать принципиальную и соединительную электрическую схему станка.
3. Исследовать возможности малогабаритного токарного (роторно-фрезерного) станка с ЧПУ, по отношению к стандартным станкам.
4. Разработать софт (ПО) для работы станка и взаимодействия оператора с ним.

Методы:

- Сбор и анализ информации о принципах работы современных станков.
- Нахождение перспективного направления в строении станков.

Гипотеза:

Первая гипотеза исследования заключается в том, что соединение возможностей шуруповёрта и токарного станка обеспечит не только сочетание их свойств и функций, но создаст абсолютно новый принцип и вид станка.

Вторая гипотеза, которую выдвинули – возможна ли работа ЧПУ станка от аккумуляторного ИП с прежним КПД.

Третья гипотеза - возможна ли разработка собственного ПО для управления ЧПУ станком посредством G-кода и мобильного приложения (по BlueTooth).

Проблема:

Проблема данного исследования заключается в отсутствии станков подобного вида, как на рынке профессионального инструмента, так и в категории любительских самоделок. Проблема в создании источника питания (ИП) 12V с достаточной выдаваемой мощностью и КПД, его тоже пришлось делать с нуля.

В интернете много различных прошивок (открытого Программного Обеспечения) для станков с ЧПУ, но как оказалось, ни одна для выбранной конфигурации станка не подошла.

2. Основная часть

2.1. Современные самодельные станки токарные по дереву

Электрифицированный токарный станок по дереву практически не изменился с конца XIX в. На сегодняшний день существует два способа деревообработки — ручная и механизированная. Токарный станок, в отличии от ручного, позволяет значительно ускорить технологический процесс, а так же качество обработки изделия, к тому же такое оборудование можно сделать своими руками.

2.2. Этапы конструирования Мобильного 3х-осевого Токарного Роторно-Фрезерного станка, оборудованного Числовым Программным Управлением (ЧПУ).

Стратегия достижения поставленных целей.

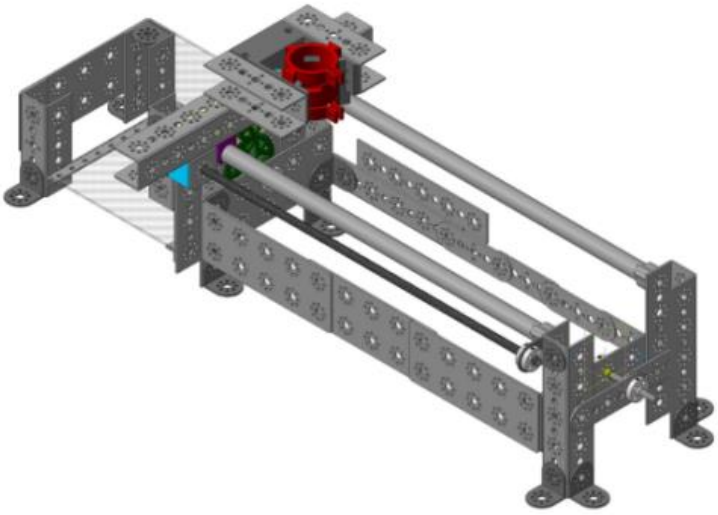
Реализация поставленных задач проходила в 6 этапов:

1. Разработка трехмерной модели станка (моделирование в Компас-3D).
2. Сборка токарного (роторно-фрезерного) станка
3. Разработка принципиальной и соединительной электрической схемы в программе fritzing.
4. Сборка/пайка электрических компонентов станка между собой
5. Написание ПО (прошивка станка, мобильное приложение)
6. Тестирование токарного (роторно-фрезерного) станка

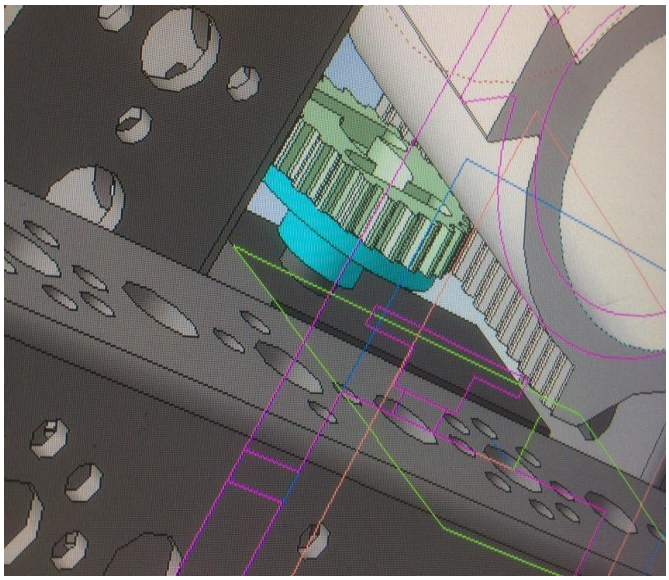
Для изготовления станка были приняты во внимание следующие факторы:

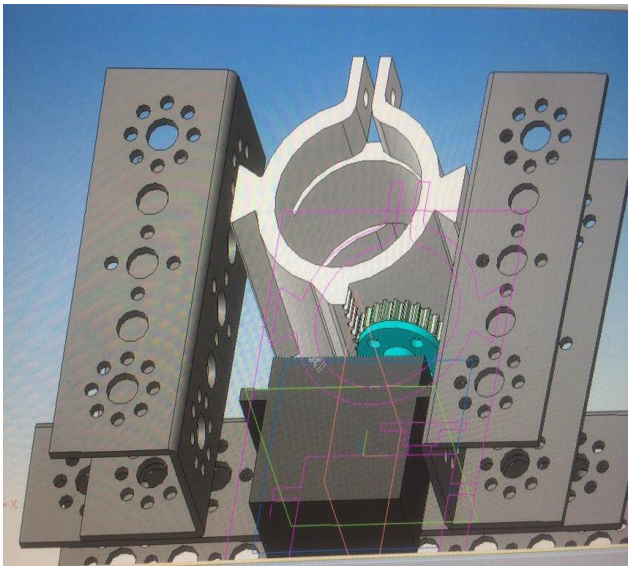
- Сложность разработанной конструкции.
- Доступность материала для выполнения изделия (36256 Конструктор TETRIX Стартовый набор).
- Наличие оборудования.
- Начало работы над проектом 2017 год.

2.3. Разработка трехмерной модели станка в КОМПАС-3D 16.1



Основой станка является его рама. Для создания рамы компактного и мобильного станка был выбран набор-конструктор TETRIX. Так как конструктор находился в школе, я создал базу данных запчастей (измерил размеры каждой детали и создал её электронную копию) из TETRIX в программе КОМПАС-3D 16.1 и начал проектировать будущий станок.



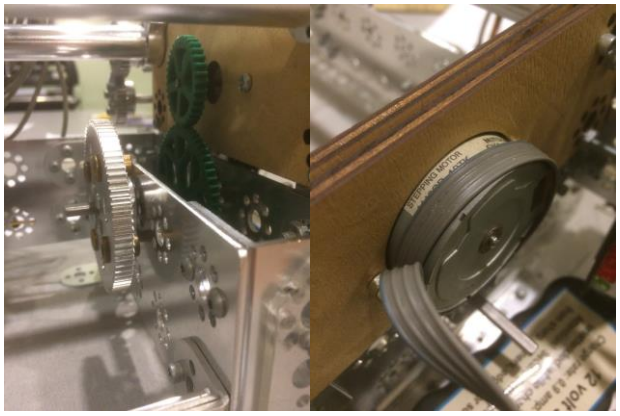


Также при проектировании возникала необходимость в создании новых запчастей – это плоские фанерные и акриловые детали, предназначенные для резки на лазерном станке, а также пластиковые запчасти сложной формы, которые необходимо было напечатать на 3D-принтере. Также на токарном станке изготовил крепление для фрезы и переходник с шагового двигателя на вал передней бабки.

2.4. Сборка токарного (роторно-фрезерного) станка

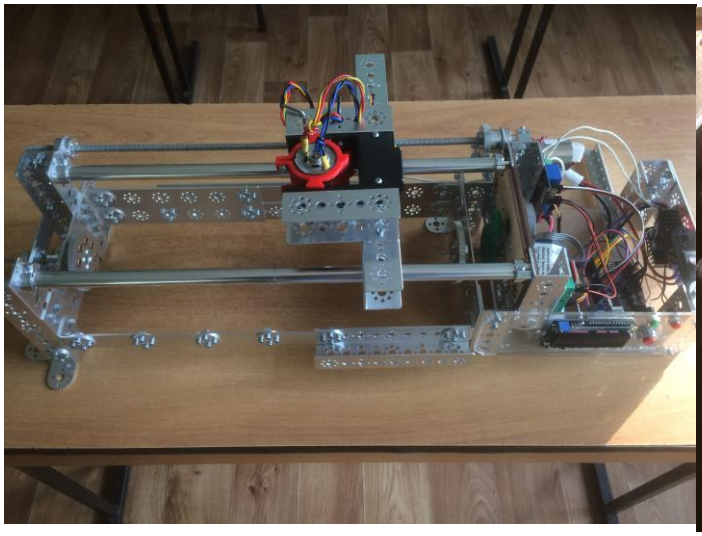
Изготовил на токарном станке свою первую деталь – крепление («патрон») для фрезы 8мм.

Держатель фрезы



Шаговый двигатель

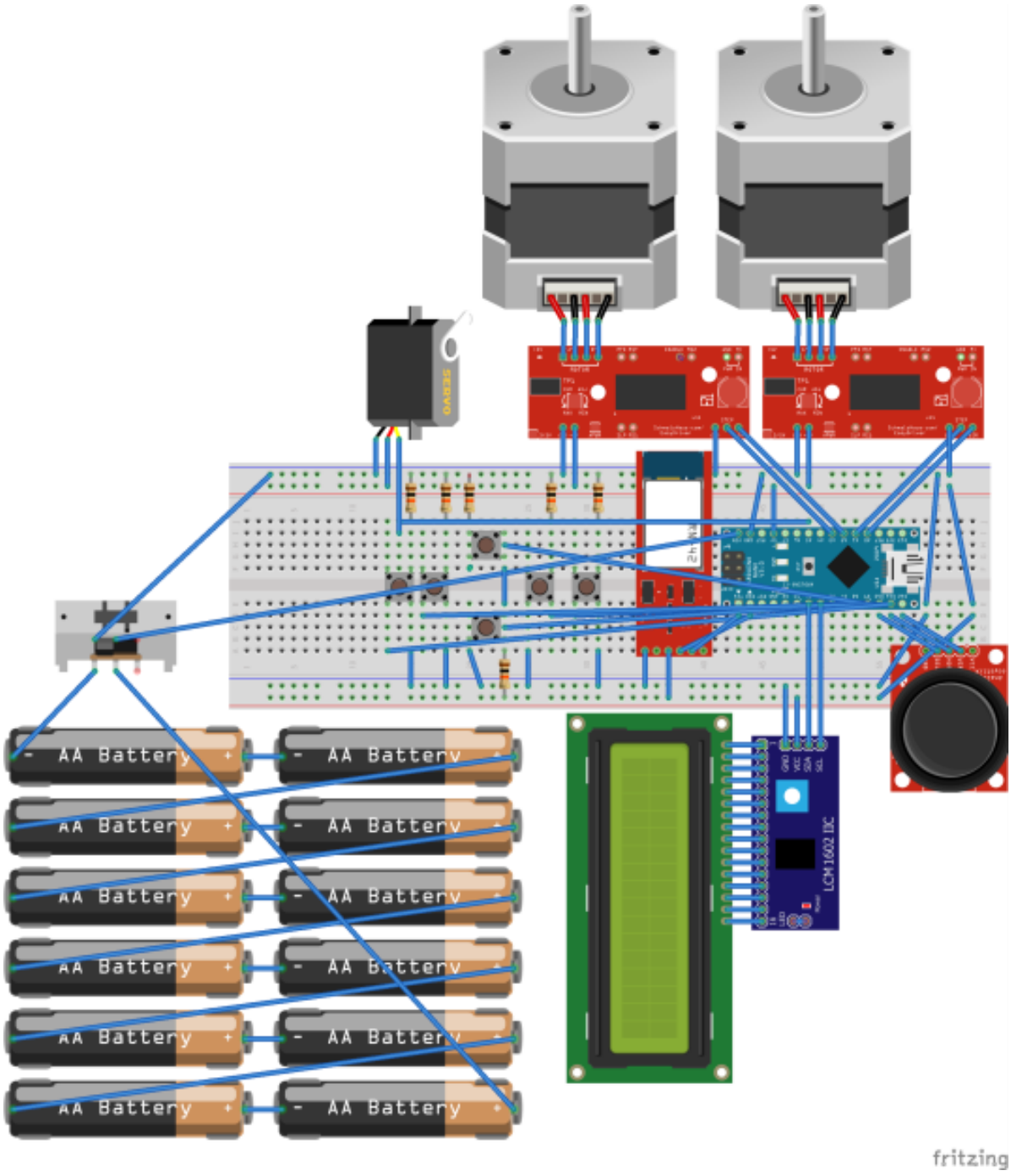
В процессе сборки установил на станок смоделированное оборудование (вращающие вал передней бабки шестерни(отношение 1 к 2) закрепил на валу шаговика; кронштейн-салазки установил на место, крепление фрезы установлено на валу фрезерующего двигателя).



Станок снабжен выключателем, позволяющим моментально обесточить станок по окончании работы или в нештатной ситуации. Станок оборудован тремя магнитными реле, которые управляют фрезерующим двигателем и направляющим ось X двигателем.

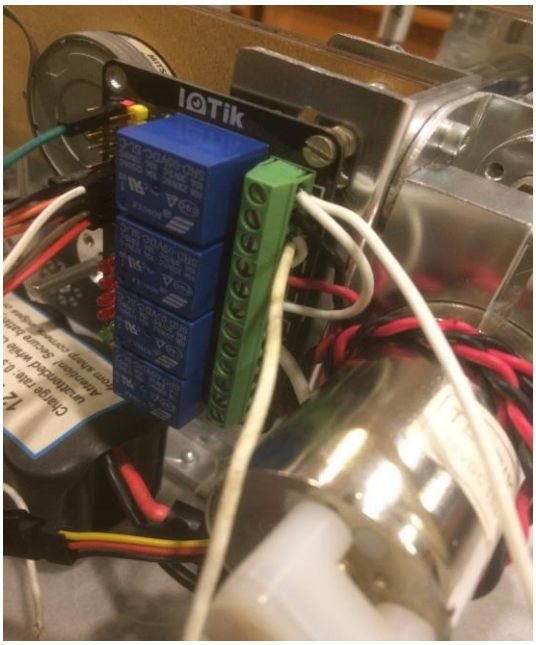
Фрезерующую головку электродвигателя под фрезу смоделировал самостоятельно в программе Компас-3D. Позже распечатал на 3д-принтере ABS пластиком. Деталь состоит из трех деталей (база с ласточкиным хвостом внутрь, рабочая часть с ласточкиным хвостом наружу и шестерня для серво-двигателя). Конструкция фрезерующей головки получилась несложной, но надежной.

2.5. Разработка принципиальной и соединительной электрической схемы в программе Fritzing.



Разработал принципиальную электрическую схему станка в программе Fritzing.

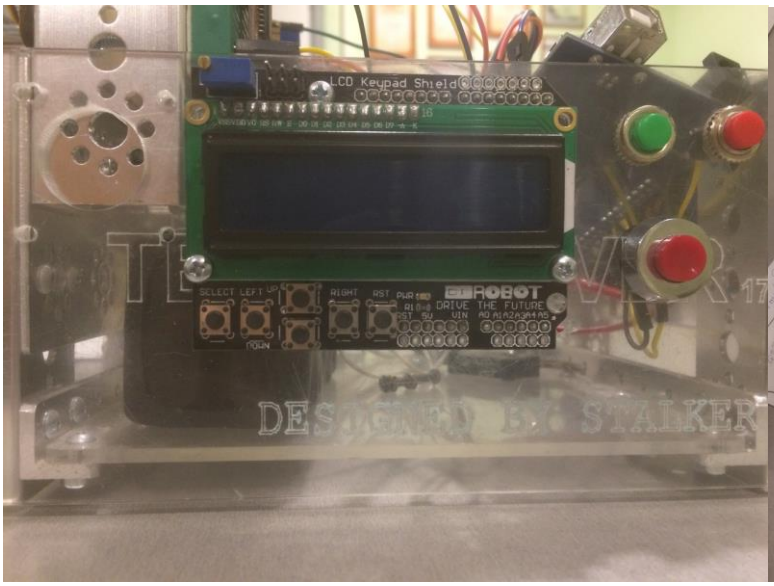
Сборка/пайка электрических компонентов станка между собой

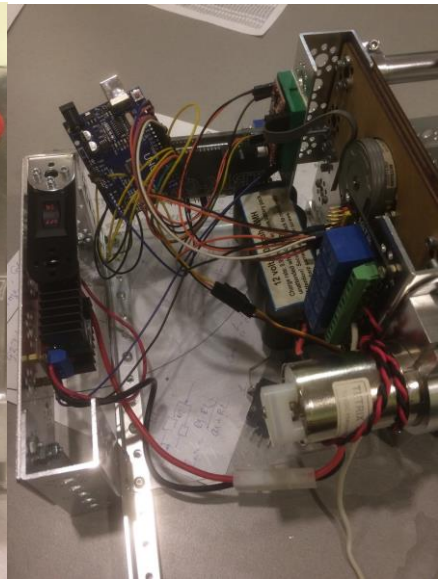


В панель управления был вмонтирован Шилд (с дисплеем 1602 и пятью кнопками для навигации), Джойстик. Была собрана электрическая цепь для автономного питания и работы станка.

Первый запуск изделия.

Станок получился даже более надежным, чем я предполагал в начале работы над проектом. Он выполнил все свои функции для пробного запуска. Но также выявился скрытый существенный недостаток в применении шагового двигателя (шаговый двигатель Mitsumi M49SP-2K LF добыл из сломанного матричного принтера HP) – он обладает малым крутящим моментом и для работы моего станка не подходит.





Испытания подтвердили надежность, устройства и безопасность станка. Не было нареканий ни к электросети, ни конструкции, ни к экологически значимым показателям.

2.6. Написание ПО (прошивка станка, мобильное приложение).

Так как актуальные прошивки для ЧПУ станков написаны исключительно под типичные конструкции станков, то они попросту не могут работать с оборудованием построенного мной станка. Понадобилось написать свою прошивку, которая смогла бы отвечать моим требованиям: наличие ручного режима (без компьютера), ЧПУ режима (с компьютером), шаблонных функций (проточка цилиндра по длине и радиусу) и настроек. За несколько месяцев интенсивной работы была получена рабочая прошивка, полностью разработанная мною.

```
98  void setup() {
99    pinMode(pinX, INPUT);
100   pinMode(pinY, INPUT);
101   pinMode(buttonPin, INPUT);
102   pinMode(concevik, INPUT);
103   pinMode(12, OUTPUT);
104   pinMode(13, OUTPUT);
105
106   myservo.attach(9);
107   myservo.write(0);
108
109   Serial.begin(9600);
110
111   lcd.init();
112   lcd.backlight();
113
114   digitalWrite(dirPin_1, HIGH);
115   while (digitalRead(concevik) == LOW) {
116     digitalWrite(stepPin_1, HIGH);
117     delayMicroseconds(steptime_1);
118     digitalWrite(stepPin_1, LOW);
119     delayMicroseconds(steptime_1);
120   }
121   delay(100);
122   digitalWrite(dirPin_1, LOW);
123   for (int i = 0; i < 200; i++) {
124     digitalWrite(stepPin_1, HIGH);
125     delayMicroseconds(steptime_1);
126     digitalWrite(stepPin_1, LOW);
127     delayMicroseconds(steptime_1);
128   }
129 }
```

Прошивка станка состоит из множества функций и подфункций. При включении станка (запуске прошивки) первым делом запускается функция void setup(), это обязательная часть прошивки. Внутри функции void setup() устанавливаются режимы работы входных и выходных контактов микроконтроллера Atmega 328p, также внутри void setup() запускается цикл while, который перемещает ось X в нулевую координату, пока не зажат концевик. Строчка myservo.attach(9); инициализирует серво-привод и также ставит его в нулевую координату

Главным телом является тело void loop() (вечный цикл, работает всегда), эта функция в первую очередь работает с переменными stroka и func, когда внутри void loop() работает if(func == 0), то на экране печатается меню (берутся элементы массива menu[stroka] и menu[stroka+1]) и прослушивается значения с

```
void loop() {

  if (func == 0) {                          // menu + sphindel 1/0

    spindle();

    if (stroka == 0) {
      lcd.setCursor(1, 0);
      lcd.print(menu_main[stroka]);
      if (engine == 1) lcd.print(" (ON)");
      else lcd.print(" (OFF)");


    } else {
      lcd.setCursor(1, 0);
      lcd.print(menu_main[stroka]);
    }

    lcd.setCursor(1, 1);
    lcd.print(menu_main[stroka + 1]);
    lcd.setCursor(0, 0);
    lcd.print(">");

    keyState = analogRead(keyPin);

    if (keyState < 2) {                         // right
    if (keyState > 98 && keyState < 102) {      // up
    if (keyState > 254 && keyState < 258) {     // down
    if (keyState > 408 && keyState < 412) {     // left
    if (keyState > 638 && keyState < 642) {     // select
  }
```

аналогового пина A0, к которому присоединена аналоговая (к каждой кнопке подсоединен резистор разного номинала), если находим нужное значение, то изменяем переменную stroka (stroka++; stroka--;) или func (func ++; func --;). Все зависит от нажатой кнопки. Подобным образом работают ручной режим, функции для настройки станка, часть шаблонных функций (выбор длинны и радиуса).

ЧПУ режим станка работает по следующему алгоритму:

1. Микроконтроллер Atmega 328p считывает 4-байтную строку buffer через COM-порт коммутатора CH340G, который тоже является частью платы Arduino.
2. Алгоритм преобразует buffer в массив char[] str.
3. Проверяет значение элемента str[0] (например, switch ((char)str[0]) {case 'X': … break;})
4. Далее будут описан алгоритм для оси X, для Y и Z осей он аналогичен
5. Сохраняет текущую координату оси (например, int X_old = X_pos;)
6. Алгоритм преобразует str[] в переменную типа int и записывает ее значение в X_pos (например, int X_pos = str.toInt( );)
7. Далее функция перемещения шагового двигателя за счет разности X_old и X_pos определяет направление движения и перемещает фрезерующую головку на указанное в строке buffer расстояние (мм) относительно нулевой координаты.

3. Заключение.

Цели научно-практической работы достигнуты. Я действительно смог разработать и изготовить малогабаритный токарный (роторно-фрезерный) станок для обработки древесины, сочетающий в себе: ширину применения, сопоставимую с шуруповёртом, удобство в транспортировке и автономный источник питания. Текущее программное обеспечение станка полностью удовлетворяет поставленным требованиям.

При исследовании конструкции станка были выявлены следующий плюсы: автономный ИП(12V) токарного (роторно-фрезерного) станка позволяет станку работать без потерь в мощности, словно от сети 220V, габариты и вес станка позволяют человеку перенести станок, в программном обеспечении станка предусмотрено ручное и ЧПУ управление.

Таким образом, считаю, что первая гипотеза моей работы доказана, так как сконструированный малогабаритный токарный роторно-фрезерный станок является представителем нового направления в станкостроении.

Вторая гипотеза также была доказана, поскольку автономный источник питания, разработанный для этого станка, обеспечивает бесперебойную работу инструмента.

Первая часть третьей гипотезы (разработка прошивки для ЧПУ станка, основанной на использовании аппаратных вычислений микроконтроллера Arduino Nano) была исполнена, но вторая часть требует дополнительных знаний для написания мобильного приложения под систему Android. На данный момент ЧПУ режим работает и без написанного для этого приложения, так как для этого достаточно любого приложения для отправки сообщений по COM-порту. Ручной режим станка полностью реализован.

Считаю, что данный станок можно использовать не только в малом бизнесе, но и в образовательных учреждениях для обучения детей на уроках труда и информатики (создание чертежей для изготовления на станке).

4. Список использованных источников и литературы

1. Бергер И.И. Токарное дело. - М.: Высш. шк.., 1990. - 314 с.

2. Брунштейн Б.Е.; Дементьев В.И. Токарное дело, М.: Высшая школа, 1987.

3. Зайцев Б.Г., Завгороднев П.И., Справочник молодого токаря, М.: Высшая школа, 1976.

4. Захаров В.А., Чистоклетов А.С., Токарь, М.: Машиностроение, 1999.

5. Оглобин А.Н. Основы токарного дела, М.: Машиностроение, 1997.

6. Лакирев С.Г. Обработка отверстий: Справочник.- М.: Машиностроение., 2004. - 208 с.

7. Тишенина Т.И.; Фёдоров Б.В. Токарные станки и работы на них. - М.: Машиностроение, 2002.

cnc_one_FINAL_cnc_concevik_2 | Arduino 1.8.3

Файл Правка Скетч Инструменты Помощь

cnc_one_FINAL_cnc_concevik_2

```
/*
  Прошивка для токарно-фрезерного чпу станка
  функции:
    чпу режим, по 4 символа на команду(X,Y,Z,E)
    ручной режим(полн.)
    возвращение домой
    изменение направление вращения шпинделя
    инфо
    возвращение домой при включении(нахождении точки нуля)

  8/11/2019 22:33 - меню завершено (660)
  10/12/2019 7:51 - логика управления шаговиками (701)
  15/12/2019 11:03 - новое инфо (710)
  15/12/2019 11:18 - возвращение домой при запуске (725)
  17/12/2019 20:16 - добавлены подменю и несколько функций

  ход SET 1,5 дюйма == 3,81 см

  X драйвер - 2(dir) 5(step)
  Y драйвер - 3(dir) 6(step)
  Z серва - 9

  экран - i2c 0x3f
  аналог. клава - A0
  джойстик - A3,A1,A2
  концевик - 11
  реле - 12,13
*/

#include <EEPROM.h>
#include <Wire.h>

#include <Servo.h>
Servo myservo;

#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x3f, 16, 2);

/* * * * * * * * * * * * * * M E N U * * * * * * * * * * * * * * * * * * */
#define keyPin A0
int keyState;

boolean engine = LOW;
int dir = EEPROM.read(0);

String menu_main[] = {"spindle", "enter CNC mode", "enter hand mode", "presets", "service", "info", " "};
int stroka = 0;
int func = 0;

String menu_hand[] = {"back", "cylinder", "spiral", " "};
int stroka_hand = 0;
String menu_service[] = {"back", "return home", "change freza", "set rotation", " "};
int stroka_service = 0;

/* * * * * * * * * * * * C O N C E V I K * * * * * * * * * * * * * * * */
#define concevik 11

/* * * * * * * * * * * * * * * * P O S * * * * * * * * * * * * * * * * * */
int X_pos = 0; // 450
int Y_pos = 0; // 200
int Z_pos = 0; // 180
int buferX, buferY, buferZ, X_old, Y_old;

char bufer[4]; // X450 Y200 Z180

/* * * * * * * * * * F R E Z A S M E N A * P R E S E T S * * * * * * * */
int fpos = 0;
int mkay = 0;
int dlina = 0;
int radius = 0;
int operation_steps;
int coef = 15; // mm/rotation

/* * * * * * * * * * * * S T E P P E R * * * * * * * * * * * * * * * * * */
#define dirPin_1 2
#define stepPin_1 5
#define steptime_1 450

#define dirPin_2 3
#define stepPin_2 6
#define steptime_2 10000

/* * * * * * * * * * * * T I M E R * * * * * * * * * * * * * * * * * * * */
#define PERIOD 1000
unsigned long timer = 0;
int seconds = 0;
int minutes = 0;

/* * * * * * * * * * * J O Y S T I C K * * * * * * * * * * * * * * * * * */
int constX, constY;

#define buttonPin A2
#define pinX A3
#define pinY A1

/* * * * * * * * * * * * * * * C O D E * * * * * * * * * * * * * * * * * */
```

```
#define buttonPin A2
#define pinX A3
#define pinY A1

/* * * * * * * * * * * * * * C O D E * * * * * * * * * * * * * * * * */

void setup() {                              // inicializing + X Z to home
void loop() {
  if (func == 0) {                          // menu + sphindel 1/0
  if ((stroka == 1) && (func == 1)) {       // position + CNC MODE
  if ((stroka == 2) && (func == 1)) {       // position + HAND MODE
  if ((stroka == 3) && (func == 1)) {       // hand presets
  if ((stroka_hand == 1) && (func == 2)) {  // cylinder
  if ((stroka_hand == 2) && (func == 2)) {  // spiral
  if ((stroka == 4) && (func == 1)) {       // service
  if ((stroka_service == 1) && (func == 2)) { //return home
  if ((stroka_service == 2) && (func == 2)) { // CHANGE FREZA
  if ((stroka_service == 3) && (func == 2)) { // sphindle direction
  if ((stroka == 5) && (func == 1)) {       // info
   //keyboardlog();
}
/* * * * * * * * * * F U N C T I O N S * * * * * * * * * * * * * * */
void timer_print(int schet_allow) {
void printpos() {                   // print position
void spindle() {                    // spindle input
void spindleoff() {                 // put spindle off
/* * * * * * * * * * * I N P U T * * * * * * * * * * * * * * * * * */
void joystick(int X_allow, int Y_allow) {                   // joystick input
void cncmove() {                    // CNC MODE + SERIAL READING
/* * * * * * * * * * S E R V I C E * * * * * * * * * * * * * * * * */
void returnhome() {                 // return home
/* * * * * * * A N Y F U N C T I O N S * * * * * * * * * * * * * * */
void keyboardlog() {                // print data from A6 pin by Serial.print() command
void info() {                       // easter egg
```

Arduino Nano, ATmega328 на COM3

```
Компиляция завершена
Linking everything together...
"C:\Program Files (x86)\Arduino\hardware\tools\avr/bin/avr-gcc"  -Os -g -flto -fuse-linker-plugin -Wl,--gc-sections -mmcu=atmega328p  -o "C:\Users\MRRO
"C:\Program Files (x86)\Arduino\hardware\tools\avr/bin/avr-objcopy" -O ihex -j .eeprom --set-section-flags=.eeprom=alloc,load --no-change-warnings --ch
"C:\Program Files (x86)\Arduino\hardware\tools\avr/bin/avr-objcopy" -O ihex -R .eeprom  "C:\Users\MRROBO~1.2)\AppData\Local\Temp\arduino_build_721458/c
Используем библиотеку EEPROM версии 2.0 из папки: C:\Program Files (x86)\Arduino\hardware\arduino\avr\libraries\EEPROM
Используем библиотеку Wire версии 1.0 из папки: C:\Program Files (x86)\Arduino\hardware\arduino\avr\libraries\Wire
Используем библиотеку Servo версии 1.1.2 из папки: C:\Program Files (x86)\Arduino\libraries\Servo
Используем библиотеку LiquidCrystal_I2C-master версии 1.1.4 из папки: C:\Users\mr.Robot(vol.2)\Documents\Arduino\libraries\LiquidCrystal_I2C-master
Скетч использует 17668 байт (57%) памяти устройства. Всего доступно 30720 байт.
Глобальные переменные используют 1253 байт (61%) динамической памяти, оставляя 795 байт для локальных переменных. Максимум: 2048 байт.
```

Arduino Nano, ATmega328 на COM3

Общий объем прошивки 1557 строчек, она использует всего 4 библиотеки и занимает 17668 байт (57%) памяти микроконтроллера Atmega328p. Глобальные переменные используют 1253 байт (61%) динамической памяти.