UNIVERSITY “POLITEHNICA” OF BUCHAREST
Faculty of Electronics, Telecommunications and Information Technology

MES Project

Developing the drive system of a syringe infusion pump for drug delivery *and blood sampling*

Authors: Palade Bianca, Master program: ACES

2020

Contents

1. Project theme

Continuous and slow administration of medications such as analgesics, cytostatics, thrombolytics, anesthetics, as well as parietal fluids or nutrients is essential for patients with complex and/or chronic diseases because any change in infusion rate, by only a few ml per hour, can have dramatic effects for an emergent patient. Therefore, it is necessary to use an automatic drug infusion system that allows the possibility of adjusting the specific parameters of the infusion rate and the administration of medical fluids continuously, for a long time, with high accuracy.

In this paper, was designed a syringe pump for drug delivery and sampling/drainage of intracavitary fluids. This module allows the possibility of use injector as an infusion pump or as a suction pump for collecting various biological samples such as blood, plasma, lymph or intracavitary fluids. This would be an advantage in terms of cost, time of preparation, and use of the device in the hospital or outpatient.

The infusion and suction system was implemented using a nut screw mechanism driven by a 42shdc4043z-22b bipolar stepper motor, controlled by an Arduino board.

Keywords - ***syringe pump, Arduino, bipolar stepper motor, biological sampling, intracavitary fluid drainage, infusion pump, suction pump***

2. Introduction

A syringe pump or syringe driver is a small pump used to deliver small amounts of fluid in a specific environment. Syringe pumps have a wide range of applications from electrospinning, microdialysis, microfluidics to organ perfusion, in fields from health care to pharmaceutical industries. There are two broad types of pumps: research syringe pump and infusion pump.

An infusion system is a portable infusion pump for delivering small to high volume of medical fluids with high precision. It is the most commonly electro-medical device use to infuse drugs in solution or other parenteral fluids to the patient for therapeutic purposes. [1] [2] [3] It is a very efficient, accurate and rapid method [2] to administrate high concentrations of medications for long stretches of uninterrupted time, therefore the infusion pump became an indispensable device in anesthesia and intensive therapy, cardiac and diabetes units of hospitals.[4] The main routes of parenteral administration provided by the infusion pump are intravenous, subcutaneous, epidural and enteral administration. [1] [3]

There are four main types of infusion pumps: gravity infusion devices [5], volumetric pumps[6], patient-controlled analgesia pumps [7], and syringe pumps.[8]

The gravity infusion pump uses gravity to infuse fluid. It was the first infusion pump used. It has been in use since the late 1960s. The system has two main advantages: the simplicity of mechanism and low production cost. However, these infusion devices are difficult to use precisely because the infusion rate is depending on the viscosity of the fluid. [5]

Volumetric pumps are used to infuse large amounts of medication. Typically, these pumps are more advanced, with many smart modules such as sensitive alert systems. However, this method is not always accessible, because the medical personnel needs specialized training. [6]

Patient-controlled analgesia pumps are infusion systems that allow patients to control their dosage as necessary. These pumps are designed for specific drugs such as analgesics (painkillers) or antiemetics in long-term, consistent treatment. Typically, they are more safety systems put in place for preventing overdosing. [7]

Syringe pumps deliver small doses of high concentration medications for a long period. The flow control may be volumetric or non-volumetric. [8]

Volumetric flow control sets the volume infused per time (ml/h), independent on the type of liquid. [1] [2] In a non-volumetric control, the pump controls the number of drops per time (drops/min). In this case, velocity depends on temperature, viscosity, and density of the fluid. [2][8] Similar to volumetric pumps, these are also often difficult to use and require specialized training to be implemented. [1][8]

Nowadays syringe pumps become an indispensable medical device, more than 90 percent of surgical patients and one-third of non-surgical patients receive some form of intravenous therapy while in the hospital.[9]

Other necessary medical devices in hospitals are drainage suction pumps used for evacuation of post-operative accumulated liquids or in the drainage of intracavitary fluids in acute or chronic diseases, such as refractory ascites (in hepatic failure syndrome). [10]

The system proposed in this paper allows the movement of syringe plunger in both directions of the motion axis. Thus, the system can be used for infusing drugs, aspirating intracavitary fluids or taking biological samples.

In this work, it is desired to design an infusion system using: an Arduino board, a bipolar stepper motor 42shdc4043z-22b, an Arduino LCD keypad shield 1602, an end-stop limit switch, a buzzer, a green LED, a red LED.

3. State-of-the-art

Researchers originally developed infusion pumps for controlled drug delivery. The first infusion pump was invented by Christopher Wren and Robert Boyle in 1658.[11] Throughout their experiments, Wren, Boyle, and other investigators primarily used bladder based pump type syringe and unit gravity feeding of liquid through a quill into blood vessels, veins in generally. Further developments, it was continued only in the 20th century.

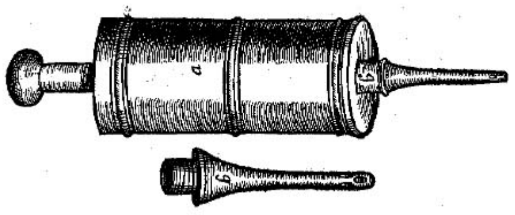

Figure III.1. - First infusion pump [11]

In 1967, J. M. Schwartz developed a new mechanism for a syringe pump based on a rack-pinion mechanism. [12] This mechanism will be present in the following figures.

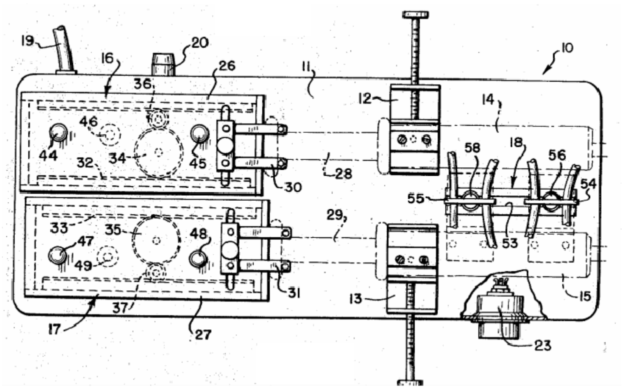

Figure III.2. – Frontal view of syringe pump [12]

The driver mechanism (16) has two separately drive carriages (26) (27), . The plunger (28), (29) of syringe (14),(15) are secured by adjustable plunger clamps (30),(31). The drive carriages (26) comprises a box-like structure, having a longitudinally extending rack (32) which are engaged by drive pinions (34). The idler gears (36) are disposed on the other side of each drive carriage (26,27). As shown in the figure, the drive pinion (35) of drive carriage (27) is driven through a reduction gear train (39) by a reversible, variable-speed motor (41). [12] When the motor is moving, the drive carriage with the racks drive in the direction of fluid infusion and generate the moving of plunger in the direction of fluid infusion. [12]

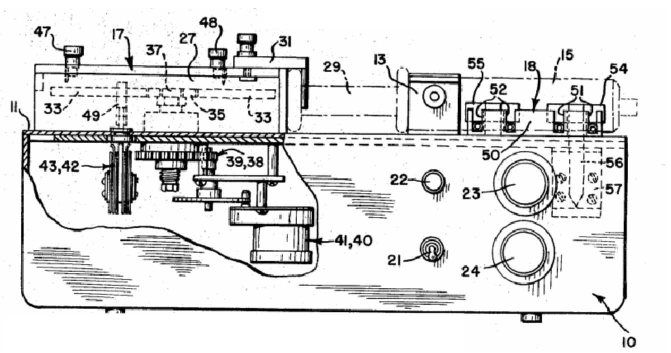

Figure III.3. – Longitudinal section view of syringe pump [12]

In 1989, Alexandre Tsoukalis designed a syringe infusion pump that includes a lead screw mechanism for changing the rotational movement of a drive motor into a linear movement. The carriage is guided parallel to the axis of the lead screw and generates the moving of the plunger of the syringe. The carriage is disengageable from the lead screw for permitting the exchange of a syringe. [13] Syringe infusion pump mechanism will be present in the following figures.

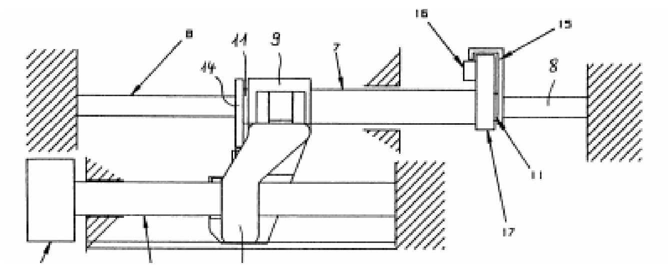

Figure III.4. – Kinematic diagram of a syringe pump [12]

The mechanism comprising a lead screw mechanism for translating rotational movement of a drive motor (18) into linear movement of a carriage (9) guided in parallel to the axis of the lead screw (1) The external thread of the lead screw (1) is driven by the drive motor (18), and an actuating mechanism (11, 14) controlling a brake (3; 3a, 3b) for normally arresting rotation of the worm wheel (2; 2a, 2b) in both directions for transmission of linear movement, and for 20 releasing the worm wheel (2; 2a, 2b) for free rotation in both directions and disengaging the transmission of linear movement. [13]

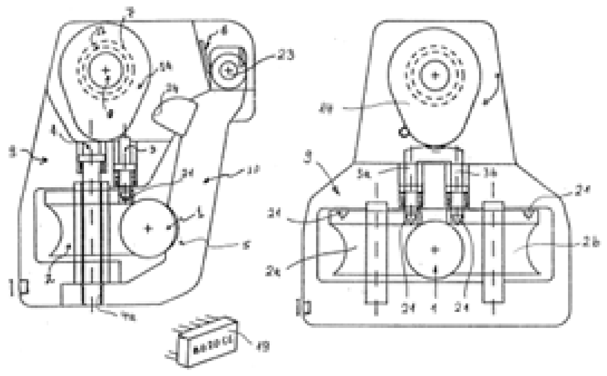

Figure III.5. – Kinematic diagram of a syringe pump, lateral view [12]

The mechanism and the operating principle of the Alexandre Tsoukalis injection have been preserved over the years. The only changes that have been made are at the sensors and alarm systems level. These changes were implemented because industry reports indicate an estimated 400,000 drug-related injuries occur in hospitals annually generating S3.5 billion in extra medical costs. [15] In the five years between 2005 and 2010 the MHRA investigated 1,085 incidents involving infusion pumps alone in the UK. In 68% of the reports no cause was established, 21% were attributed to user error, including maintenance, damage and contamination problems and 11% of incidents were due to device-related issues such as performance problems, degradation, inadequate quality assurance and design and labelling. [1]

Safety software, including drug error reduction systems (DERS), has been developed to reduce medication errors, enhance quality care, and improve work flow. [9]

Drug error reduction systems (DERS) typically use a drug library editor (DLE) to develop drug libraries including protocols, rule sets, and/or pump configuration settings, which are then loaded onto an infusion pump.[9] The drug library at the infusion pump provides medication pick lists for the caregiver to select the desired therapy, medication profiles including hard and Soft limits, and pump configuration settings including but not limited to distal pressure occlusion limits, air-in-line limits, callback settings, back light display settings, etc.[12]

4. Project description

In this project, it is desired to design a syringe pump for drug delivery and sampling/drainage of intracavitary fluids. The mechanism is based on a lead screw mechanism driven by a bipolar stepper motor controlled by an Arduino board. The control panel consists on a keypad, to set the infusion parameters. The data output is visualized through an alphanumeric display (Liquid Crystal Display – LCD). This display shows the information on the infusion in course, the total volume to be infused, the flow rate (ml/min).

The 3D model of the system purge was made in AutoCAD, using the dimensions of the standard syringes. AutoCAD is a computer-aided graphics and design environment, produced and marketed by the American company Autodesk. It is a CAD program successfully used in fields such as architecture, medicine, and technology.

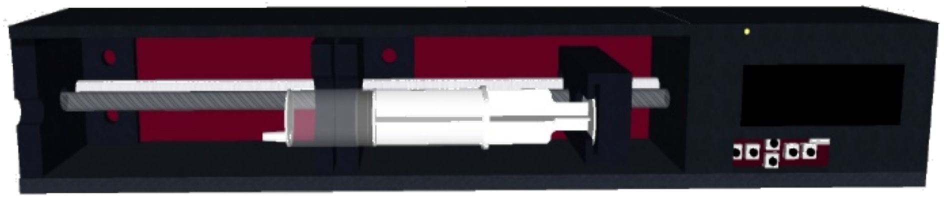

Figure IV.1. – 3D model of the system purge

A 3D FFF printer, was used to print the mechanical components of the syringe pump: rod mount syringe, rod mount plunger, rod mount barrel.

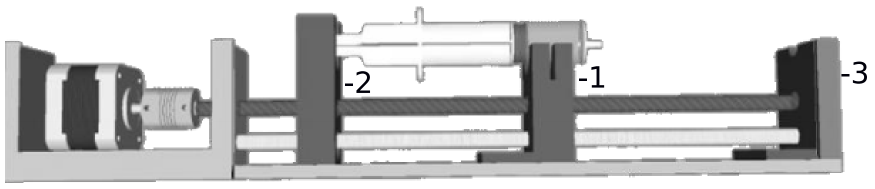

Figure 4.2. – 3D model of the system purge mechanism (1) rod mount plunger, (2) rod mount plunger, (3) rod mount barrel

5. Hardware description

In this project, a lead screw mechanism was used for changing the rotational movement of a drive motor into a linear movement. The mechanism has a T8 nut and a threaded rod with a trapezoidal profile, diameter 8mm, step 2mm and length 400 mm. It is driven by a bipolar stepper motor 42shdc4043z-22b. (Figure 5.1.-A)

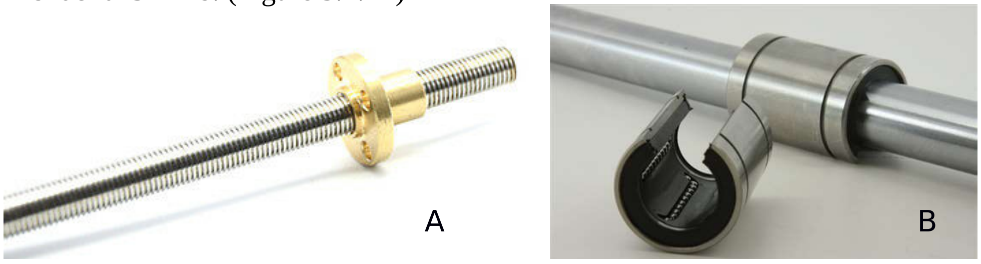

Figure 5.1. – (A) Threaded rod with a T8 nut (B) Linear bearing

The shaft of the motor is coupled by the threaded rod through an aluminum elastic coupling with an internal diameter of 8 mm. The nut was mounted in the first bore of the 3D printed support. To prevent the rotation of the support element, it uses an aluminum rod with an outer diameter of 8mm. To reduce friction between aluminum rod and support, it was inserted a linear bearing.(Figure 5.1. - B) To engage the mechanism was used a two-phase bipolar motor with 200 steps per revolution, powered by 12V, 1A power supply(Figure 5.2.A). The motor is controlled by an A4988 driver(Figure 5.2 - B).

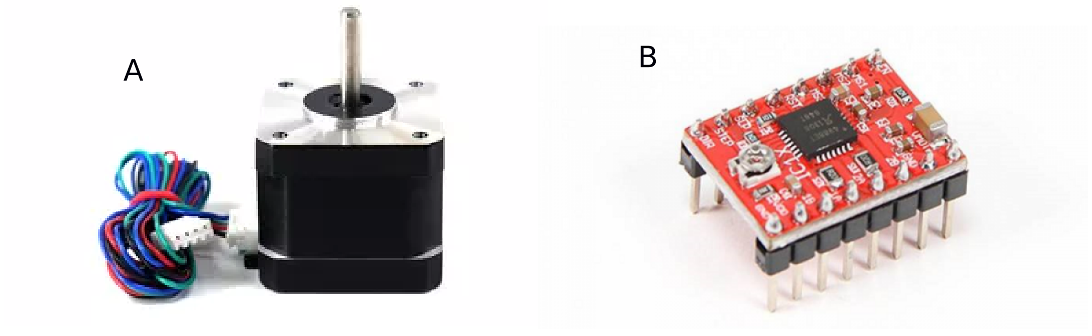

Figure 5.2. – (A) Bipolar stepper motor 42shdc4043z-22b *(B) Stepper motor driver A4988*

The working parameters of the syringe pump are set using a 1602 LCD Keypad Shield for Arduino. The parameters which can be adjusted are syringe-type (10-100 ml), flow rate (0.5-100 ml/min), flow direction (-1/1 for suction/infusion mode) and heat mode (0/1 no heat/heat at 35*C).

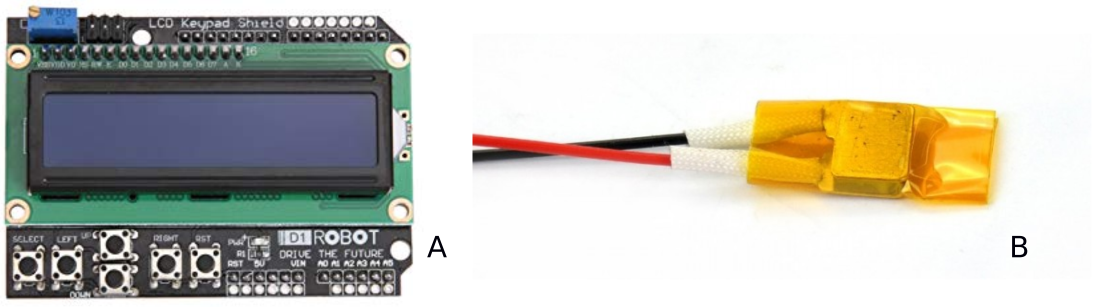

Figure 5.3. – (A) 1602 LCD Keypad Shield *(B) Mini PTC Heating Element 5V / 70 ℃*

Three LEDs were used: two LEDs to indicate how to use the pump (infusion, suction), and a third LED to indicate the successful completion of the command.

The syringe pump cannot recognize the type of syringe. Therefore it is necessary a system calibration for each device use. The stepper motor will rotate anticlockwise and generate a linear motion of the rod mount in the suction direction until the limit switch was touched. Then, it will calculate the number of steps until the rod mount plunger will be in a position that can permit the mount of the syringe. Afterward, it will wait for the confirmation of the operator and next will initiate the infusion or suction mode.

The housing will be contain a light intensity control system using infrared sensor. When the infrared sensor detects an object on its proximity (a hand waving in front of it), the light will turn on and off on the following rule: short wave:the led will turn on or off, according to the current state, if the hand wave is >1s, the LED starts to decrease its intensity gradually, until a minimum value. The intensity can be increased in only one way: short hand wave which will turn off the light, followed by a short wave hand, which will turn on the light again, at its maximum intensity.

The following figure shows the diagram of the electric circuit. To make the injection system was used a 1602 LCD Keypad Arduino Shield, 42shdc4043z-22b stepper motor connected to an Arduino MEGA 2560 board through an A4988 driver.

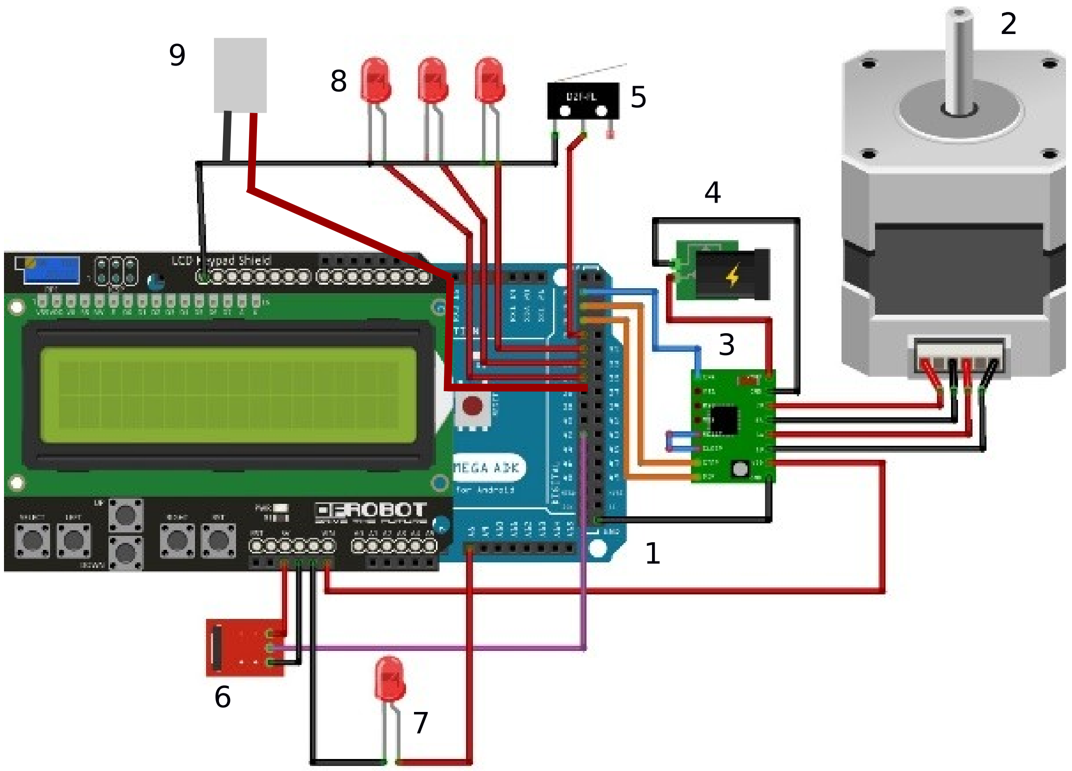

Figure 5.4. – (1) Arduino Mega AT 2560(2) Bipolar stepper motor 42shdc4043z-22b, (3) Bipolar Driver Stepper Motor A4988, (4) Adapter 12V, 1A, (5) Limit Switch (6) IR Sensor, (7-8) LEDs, (9) Mini PTC Heating Element

6. Software description

6.1. LCD Menu

The program of LCD Menu use <Wire.h> and <LiquidCrystal.h> Arduino Library. The LCD shield has 6 buttons, only 4 of which are used în program: up, down, left and right.

The program has four main functions: mainMenuDraw(), drawCursor(), operateMainMenu() and subMenu(). MainMenuDraw() will generate the two menu items that can fit on the screen. The items will change with the scroll through the menu. Up and down arrows will indicate the current menu position.

The function drawCursor() will erase the current cursor and redraw it based on the cursorPosition() and menuPage() variables. The menu is set up to be progressive: menuPage 0 = Item 1 and Item 2, menuPage 1 = Item 2 and Item 3, menuPage 2 = Item 3 and Item 4, etc. To determine the position of the current cursor it is necessary to decide if the number of pages is odd or even menu and if the current cursor is odd or even. Thus, there are four possibilities: if the menu page is even and the cursor position is even or menu page is odd and the cursor position is odd that means the cursor should be on line 1 if the menu page is even and the cursor position is odd or the menu page is odd and the cursor position is even that means the cursor should be on line 2.

In function operateMainMenu() it is evaluated the current state of buttons. If it presses the forward button, the function corresponding to the current item will be called. If it presses up or down button it will change the cursor position and will redraw the cursor and change the menu page number. The functions of each menu item are: menuItem1(corresponding to menu title), menuItem2 (set syringe type), menuItem3 (set the flow rate), menuItem4 (set the flow direction), menuItem5 (set the heating on or off), menuItem6 (call the function that start the infusion/suction). Each menuItem function, except menuItem6, will call the submenu function, which allows displaying, to set the desired parameters with constraints of each menuItemfunction and to return to the main menu.

The function menuItem6() will allow setting the stare of the light system during the infusion/suction. After setting the desired stare of the light system it will be called the motor() function.

6.2. Control Stepper Motor

The function that control the motor speed and direction use <AccelStepper.h> Arduino library. First, the function will calculate the rotational speed of the motor in steps/sec. The syringe pump cannot recognize the type of syringe. Therefore it is necessary a motor calibration for each device use. The stepper motor will rotate anticlockwise and generate a linear motion of the rod mount in the suction direction until the limit switch was touched. Then, it will calculate the number of steps until the rod mount plunger will be in a position that can permit the mount of the syringe. Afterward, it will wait for the confirmation of the operator and next will initiate the infusion or suction mode.

6.3. Control Light System

Light intensity control system use an infrared sensor. When the infrared sensor detects an object on its proximity (a hand waving in front of it), the light will turn on and off on the following rule: short wave:the led will turn on or off, according to the current state, if the hand wave is >1s, the LED starts to decrease its intensity gradually, until a minimum value. The intensity can be increased in only one way: short hand wave which will turn off the light, followed by a short wave hand, which will turn on the light again, at its maximum intensity.

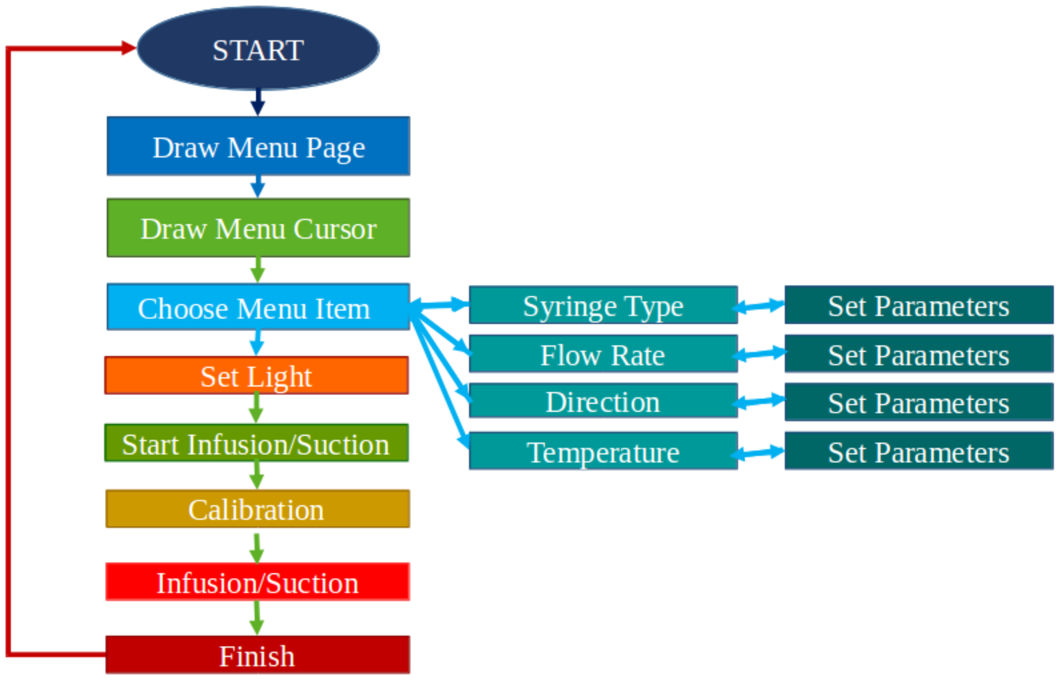

Figure 6.1. – Software diagram

7. Calculation Report

From the main menu, the user can select the infusion or aspiration flow rate (ml/min). The speed of rotation of the motor corresponding to the desired flow rate is calculated according to the formula:

$$\omega = \frac{Q \cdot 360°}{\pi \cdot R^2 \cdot p \cdot \partial}$$

where Q = flow rate, R = radius of cross- sectional surface, p = motor step, ∂ = step angle. Suppose the flow rate defined by the flow of the volume of fluid V through a surface per unit time T:

$$Q = \frac{V}{T}$$

The volume of liquid pushed to a step is:

$$V = S \cdot p = \pi \cdot R^2 \cdot p$$

where S = cross-sectional vector surface

Then engine speed is:

$$n = \frac{1}{T} = \frac{Q}{V} = \frac{Q}{\pi \cdot R^2 \cdot p}$$

Suppose the engine speed defined in steps per second:

$$\omega = \frac{Q \cdot 360°}{\pi \cdot R^2 \cdot p \cdot \partial}$$

The radius of the cross-sectional surface depends on the syringe volume. The characteristic dimensions for a 10 – 60 ml syringe are illustrated in the table below.

Syringe volume	Syringe radius [mm]	Plunger length [mm]
10 ml	7.25	60
20 ml	9.565	70
30 ml	10.85	75
40 ml	10.85	80
50 ml	13.35	87
60 ml	13.35	102

8. Conclusions

The developed system could be used as an infusion pump or suction pump. The device allows the setting of the basic parameters of the syringe pump.

During the experiments, it was found that the function of the stepper motor produces too much vibration in the infused or aspirated liquid so that the biological samples such as blood or lymph can not be taken. The stepper motor cannot perform small enough steps so that the piston speed to be approximately constant at very low speeds, therefore it is necessary to add a 1: 1000 gearbox. Also, due to vibration, it could not be implemented an alert system for occlusion or bubbles detection.

9. Source code

```
#include <LiquidCrystal.h>
#include <AccelStepper.h>
#include <Wire.h>
#include "Menu_Characters.h"

//----- Bipolar Stepper Motor 42SHDDC4043Z-22B -------//
const int EN_MOT = 26;
const int STEPS  = 24;
const int DIR    = 22;
const int motorInterfaceType = 1;
AccelStepper stepper = AccelStepper(motorInterfaceType, STEPS, DIR);
//----------------------------------------------------//


//------- Sensor FC-123 --------//
const int IR     = 28;
const int enable = 29;
//----------------------------//


// ------Switch Limit---------//
const int IR0    = 42;
//--------------------------//


//--------- LEDs -------------//
const int LED_INF  = 48;
const int LED_SU   = 36;
const int LED_Stop = 37;
const int LED      = 38;
//Delay
const int delay_LED = 10;
const int  delay_IR = 50;
//
float count   = 0;
//Current state of LED
int         a = 1;
//LED max value
int led   = 126;
//---------------------------//


//-----LCD pin to Arduino------//
const int pin_RS = 8;
const int pin_EN = 9;
const int pin_d4 = 4;
const int pin_d5 = 5;
const int pin_d6 = 6;
const int pin_d7 = 7;
const int pin_BL = 10;
//---------------------------//
```

```
//------ Button value --------//
int button;
int cursorPosition = 0;
int read_key;
//---------------------------//


//-----Control parameters-----//
unsigned int syringe   = 0;
       float rate      = 0;
         int dir       = 1;
unsigned int occlusion = 0;
//---------------------------//


//---- Function prototypes-----//
void menuItem1();
void menuItem2();
void menuItem3();
void menuItem4();
void menuItem5();
void menuItem6();
//-----------------------------//


//----------------- Array of pointers to function -------------------------------------//
void (*menuItem[])() = {menuItem1, menuItem2, menuItem3, menuItem4, menuItem5, menuItem6};
String menuItems[] = {"SETTINGS","SYRINGE TYPE", "FLOW RATE", "FLOW DIRECT.", "OCCLUSION DET",
"START"};
//-------------------------------------------------------------------------------------//


//----- Navigation button variables -----//
int readKey;
int savedDistance = 0;
//--------------------------------------//


//------ Menu control variables --------//
int menuPage = 0;
int maxMenuPages = round(((sizeof(menuItems) / sizeof(String)) / 2) + .5);
//--------------------------------------//


LiquidCrystal lcd( pin_RS,  pin_EN,  pin_d4,  pin_d5,  pin_d6,  pin_d7);

void setup() {

  // Initializes serial communication
  Serial.begin(9600);
  // Set driver stepper motor pins
  pinMode(EN_MOT, OUTPUT);
  pinMode(   DIR, OUTPUT);
  pinMode( STEPS, OUTPUT);
  //
  digitalWrite(EN_MOT,HIGH);
  digitalWrite(   DIR,HIGH);
  // Set max stepper motor speed
  stepper.setMaxSpeed(1000);

  // Initializes and clears the LCD screen
  lcd.begin(16, 2);
  lcd.clear();

  lcd.setCursor(1,0);
  lcd.print("Syringe Driver");
  lcd.setCursor(2,1);
  lcd.print("I==[");
  lcd.createChar(4, menuSyringe);
  lcd.createChar(5, menuFill);
  lcd.setCursor(6,1);
  lcd.write(byte(4));
  lcd.setCursor(7,1);
  lcd.write(byte(4));
  lcd.setCursor(8,1);
  lcd.write(byte(5));
  lcd.setCursor(9,1);
  lcd.write(byte(5));
  lcd.setCursor(10,1);
  lcd.write(byte(5));
  lcd.setCursor(11,1);
  lcd.print("]--");

  delay(3000);
  lcd.clear();

  // Creates the byte for the 3 custom characters
  lcd.createChar(0, menuCursor);
  lcd.createChar(1, upArrow);
  lcd.createChar(2, downArrow);
}


void loop() {
  mainMenuDraw();
  drawCursor();
  operateMainMenu();
}

void mainMenuDraw() {
  Serial.print(menuPage);
  lcd.clear();
  lcd.setCursor(1, 0);
  lcd.print(menuItems[menuPage]);
  lcd.setCursor(1, 1);
  lcd.print(menuItems[menuPage + 1]);
  if (menuPage == 0) {
    lcd.setCursor(15, 1);
    lcd.write(byte(2));
  } else if (menuPage > 0 and menuPage < maxMenuPages) {
    lcd.setCursor(15, 1);
    lcd.write(byte(2));
    lcd.setCursor(15, 0);
    lcd.write(byte(1));
  } else if (menuPage == maxMenuPages) {
    lcd.setCursor(15, 0);
    lcd.write(byte(1));
  }
}

// Erase the current cursor and redraw it based on the cursorPosition and menuPage variables.
void drawCursor() {
  for (int x = 0; x < 2; x++) {     // Erases current cursor
    lcd.setCursor(0, x);
    lcd.print(" ");
  }

  // The menu is set up to be progressive: menuPage 0 = Item 1 & Item 2, menuPage 1 = Item 2 &
Item 3, menuPage 2 = Item 3 & Item 4,


  if (menuPage % 2 == 0) {
     if (cursorPosition % 2 == 0) {  // menu -> even && cursor position -> even => cursor ->
line 1
      lcd.setCursor(0, 0);
      lcd.write(byte(0));
    }
     if (cursorPosition % 2 != 0) {  // menu -> even && cursor position -> odd => cursor ->
line 2
      lcd.setCursor(0, 1);
      lcd.write(byte(0));
    }
  }
  if (menuPage % 2 != 0) {
     if (cursorPosition % 2 == 0) {  // menu -> odd && cursor position -> even => cursor ->
line 2
      lcd.setCursor(0, 1);
      lcd.write(byte(0));
    }
    if (cursorPosition % 2 != 0) {  // menu -> odd && cursor position -> odd => cursor -> line
1
      lcd.setCursor(0, 0);
      lcd.write(byte(0));
    }
  }
}

void operateMainMenu() {
  while(1)
      {
        read_key = analogRead(0);
        delay(200);
        if(read_key < 790) break;
      }

  button = key_press(read_key);

  switch(button){
  case 0:
        button = 0;
        switch (cursorPosition) {
          case 0:
            menuItem1();
            break;
          case 1:
            menuItem2();
            break;
          case 2:
            menuItem3();
            break;
          case 3:
            menuItem4();
            break;
          case 4:
            menuItem5();
            break;
          case 5:
            menuItem6();
        }
        mainMenuDraw();
        drawCursor();
        break;
  case 1:
        button = 0;
        if (menuPage == 0) {
          cursorPosition = cursorPosition - 1;
           cursorPosition = constrain(cursorPosition, 0, ((sizeof(menuItems) / sizeof(String))
- 1));
        }
        if (menuPage % 2 == 0 and cursorPosition % 2 == 0) {
          menuPage = menuPage - 1;
          menuPage = constrain(menuPage, 0, maxMenuPages);
        }

        if (menuPage % 2 != 0 and cursorPosition % 2 != 0) {
          menuPage = menuPage - 1;
          menuPage = constrain(menuPage, 0, maxMenuPages);
        }

        cursorPosition = cursorPosition - 1;
         cursorPosition = constrain(cursorPosition, 0, ((sizeof(menuItems) / sizeof(String)) -
1));

        mainMenuDraw();
        drawCursor();
        break;
  case 2:
        button = 0;
        if (menuPage % 2 == 0 and cursorPosition % 2 != 0) {
          menuPage = menuPage + 1;
          menuPage = constrain(menuPage, 0, maxMenuPages);
        }

        if (menuPage % 2 != 0 and cursorPosition % 2 == 0) {
          menuPage = menuPage + 1;
          menuPage = constrain(menuPage, 0, maxMenuPages);
        }

        cursorPosition = cursorPosition + 1;
         cursorPosition = constrain(cursorPosition, 0, ((sizeof(menuItems) / sizeof(String)) -
1));
        mainMenuDraw();
        drawCursor();
        break;
  }
}

int key_press(int key_val){

    return (key_val < 60 )? 0 :      // RIGHT
           (key_val < 200)? 1 :      // UP
           (key_val < 400)? 2 :      // DOWN
           (key_val < 600)? 3 : 4;   // LEFT : SELECT
}

//----------- SETTINGS----------------//
void menuItem1() { // case 0 from main menu
  }
//------------------------------------//

//-------- SYRINGE TTYPE ------------//
void menuItem2() { // case 1 from main menu


    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("> SYRINGE:   ");
    lcd.setCursor(13,0);
    lcd.print(syringe,DEC);
    lcd.setCursor(0,1);
    lcd.print("              OK");
    syringe = submenu(syringe, 10, 3, 0, 0, 100, 2);
}
//------------------------------------//

//----------- FLOW RATE -------------//
void menuItem3() { // case 2 from main menu
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("> FLOW RATE:   ");
    lcd.setCursor(13,0);
    lcd.print(rate,DEC);
    lcd.setCursor(0,1);
    lcd.print("  ml/min      OK");
    rate = submenu(rate, 0.5, 6, 3, 0, 100, 3);
  }
//------------------------------------//

//--------- FLOW DIRECTION ----------//
void menuItem4() { // case 3 from main menu
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("> FLOW DIR.:   ");
    lcd.setCursor(13,0);
    lcd.print(dir,DEC);
    lcd.setCursor(0,1);
    lcd.print("              OK");
    dir = submenu(dir, 2, 2, 0, -1, 1, 4);
  }
//-----------------------------------//

//----------- TEMPERATURE -----------//
void menuItem5() { // case 4 from main menu
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("> OCC. DET.:   ");
    lcd.setCursor(13,0);
    lcd.print(occlusion,DEC);
    lcd.setCursor(0,1);
    lcd.print("              OK");
    occlusion = submenu(occlusion, 5, 3, 0, 0, 30, 5);
  }
//-----------------------------------//


//--------------START----------------//
void menuItem6() { // case 5 from main menu
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("Calibration....");
    light();
    int i = motor(rate,syringe,dir);
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("End");
    delay(2000);
  }
//-------------------------------------//


//-------------Light System--------------//
void light()
{
while(1)
  {
  read_key = analogRead(0);
  if(read_key < 790) break;

 // Read data from sensor
  int ir_read = digitalRead(IR);

  // Print data to the serial port in monitor
 // Serial.println(ir_read);
  delay(delay_IR);

  while(!ir_read)
  {
    delay(50);
    ir_read = digitalRead(IR);
    Serial.println(ir_read);
  }

  while(ir_read && (count <= 1))
  {
    delay(delay_IR);
    count = count + 0.05;
    Serial.println(ir_read);
    Serial.println(count);
    ir_read = digitalRead(IR);
  }

  if(count < 1)
  {
    if(a)
    {
       analogWrite(LED,126);
       count = 0;
    }
    else
    {
       analogWrite(LED,0);
       count = 0;
    }
  }
  else
  {
    if(a)
    {
      led = 0;
      for( /*no initialization */; led < 126; led++)
        {
          analogWrite(LED, led);
          delay(delay_LED);
          count = 0;
        }
      }
     else
     {
      for( /*no initialization */; led >= 0; led--)
        {
          analogWrite(LED, led);
          delay(delay_LED);
          count = 0;
        }
      }
  }
  while(ir_read)
  {
    delay(50);
    ir_read = digitalRead(IR);
    Serial.println(ir_read);
    }
  a = !a;
//  Serial.println("a = ");
//  Serial.println(a);
}
}


//----------------STEPPER MOTOR-------------------//
int motor(int rate, int syringe, int dir)
{
  int number_of_steps;
  float radius =  (syringe == 10)? 7.25  :
                  (syringe == 20)? 9.565 :
                  (syringe == 30)? 10.85 :
                  (syringe == 40)? 10.85 : 13.35;

  int speed_mot = 1000 * (rate/(1.884*radius*radius));

  int plunger   = (syringe == 20)? 70 :
                  (syringe == 30)? 75 :
                  (syringe == 40)? 80 :
                  (syringe == 50)? 87 :
                  (syringe == 60)? 102:
                  (syringe == 70)? 102:
                  (syringe == 80)? 102:
                  (syringe == 90)? 102:
                  (syringe == 100)? 102: 0;

  digitalWrite(EN_MOT, LOW);
  if(dir > 0)
  {
    number_of_steps = (113 - plunger)/2.0 * 200;
  }
  else
  {
    number_of_steps = 113/2.0 * 200;
  }
  while (digitalRead(IR0)) {  // Do this until the switch is activated
    stepper.setSpeed(-400);
    stepper.runSpeed();
  }
  Serial.println(digitalRead(IR0));
  delay(2000);

  stepper.setCurrentPosition(0);

  while(stepper.currentPosition() != number_of_steps)
  {
    stepper.setSpeed(400);
    stepper.runSpeed();
  }
  delay(1000);

  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("If it's ready to");
  lcd.setCursor(0,1);
  lcd.print("use press SELECT");

  while(1)
      {
        read_key = analogRead(0);
        delay(200);
        if(read_key < 790) break;
      }
    lcd.clear();
    lcd.setCursor(0,1);
    lcd.print("rate: ");
    lcd.setCursor(5,1);
    lcd.print(speed_mot,DEC);
    lcd.setCursor(9,1);
    lcd.print(" srn: ");
    lcd.setCursor(14,1);
    lcd.print(plunger,DEC);

    if(dir == 1)
    {
      digitalWrite(LED_INF, HIGH);
      lcd.setCursor(0,0);
      lcd.print("> Infusion pump  ");
      number_of_steps = plunger/2.0 * 200;
      stepper.setCurrentPosition(0);
      while(stepper.currentPosition() != number_of_steps)
      {
      stepper.setSpeed(speed_mot);
      stepper.runSpeed();
      }}
    else if(dir == -1)
    {
      digitalWrite(LED_SU, HIGH);
      lcd.setCursor(0,0);
      lcd.print("> Suction pump  ");
      number_of_steps = plunger/2.0 * 200;
      stepper.setCurrentPosition(0);
      while(stepper.currentPosition() != -number_of_steps){
      stepper.setSpeed(-speed_mot);
      stepper.runSpeed();
      }}
    digitalWrite(LED_INF, LOW);
    digitalWrite(LED_SU,  LOW);
    digitalWrite(EN_MOT,  HIGH);
  return 0;
}
//------------------------------------------------//


//-------------------SubMenu----------------------//
float submenu(float x, float y, int pos, int precision, int min_constrain, int max_constrain,
int i)
```

```
{
    int a = 1;
    char str[7];
    while(a)
      {
        read_key = analogRead(0);
        delay(200);
        if(read_key < 790)
        {
        button = key_press(read_key);
        switch(button)
        {
        case 0:
            x = x + y;
            x = constrain(x, min_constrain, max_constrain);
            dtostrf(x, pos, precision, str);
            lcd.setCursor(12,0);
            lcd.print("      ");
            lcd.setCursor(12,0);
            lcd.print(str);
            break;
        case 3:
            x = x - y;
            x = constrain(x, min_constrain, max_constrain);
            dtostrf(x, pos, precision, str);
            lcd.setCursor(12,0);
            lcd.print("      ");
            lcd.setCursor(12,0);
            lcd.print(str);
            break;
        case 2:
            lcd.setCursor(0, 1);
            lcd.write(byte(0));
            lcd.setCursor(0, 0);
            lcd.print(" ");
            while(1){
              read_key = analogRead(0);
              delay(200);
              if(read_key < 790) break;
            }
            button = key_press(read_key);
            switch(button)
            {
              case 1:
                  lcd.clear();
                  menuItem[i-1]();
                  a = 0;
                  break;
              case 0:
                  //lcd.clear();
                  //mainMenuDraw();
                  //drawCursor();
                  //operateMainMenu();
                  a = 0;
                  break;}
            break;}}
        if(a == 0) break;}
      return x;}
```

10.References

[1] Medicines and Healthcare products Regulatory Agency, "Infusion Systems," December 2013.

[2] R. Assunção and et al., "Developing the control system of a syringe infusion pump," in *11th International Conference on Remote Engineering and Virtual Instrumentation*, Porto, Portugal, 2014.

[3] B Braun, Infusomat Space - Service Manual, B Braun Sharing Expertise, 2011.

[4] A.Andrew Silva and et al., "Advanced Control System for Syringe and Infusion Pump Using IoT," *International Journal of Innovative Research in Advanced Engineering ,* vol. VI, no. 3, pp. 221 - 224, 2019.

[5] J. Huang, "Disposable multifunctional silica gel pump and gravity infusion device". China Patent WO 2017/012493, 26 January 2016.

[6] M. Becker, "Infusion Pump". Germany Patent DE102018104229B3, 16 May 2019.

[7] Timothy W. Vandereen, "Opioid Management System". United States of America Patent 2018/0365386 A1, 20 December 2018.

[8] Fish et al., "Medical Infusion Pump for Sequentially Injecting Solutions from Multiple Syringes". United States of America Patent 9,981,082 B2, 29 May 2018.

[9] James D. Jacobson et al., "Infusion Pump System and Method with Multiple Drug Library Editor Source Capability". United States of America Patent 2016/0350513 A1, 1 December 2016.

[10] Guido Stirnimann, et al., "Automated low-flow ascites pump for the treatment of cirrhotic patients with refractory ascites," *Therapeutic Advances in Gastroenterology,* vol. X, no. 2, pp. 283-292, 2017.

[11] "Evolution of Medical Application of Syringe," *Indian J Physiol Pharmacol,* vol. 50, no. 3, pp. 199-204, 2006.

[12] J.M. Schwartz et al., "Syringe Pump". United States of America Patent 3,554,673, 12 January 1971.

[13] Alexandre Tsoukalis et al., "Syringe Pump". Germany Patent 402,553 B1, 21 October 1989.

[14] Naoki Kondo and Juhwan Oh, "Suicide and karoshi (death from overwork) during the recent economic crises in Japan: the impacts, mechanisms and political responses, " *Journal Epidemiol Community Health,*, vol. 64, no. 8, pp. 649-650, 2010.