DEGREE PROJECT IN TECHNOLOGY,
FIRST CYCLE, 15 CREDITS
STOCKHOLM, SWEDEN 2019

Robotic Arm controlled by Arm Movements

MIKO NORE

CASPAR WESTERBERG

KTH ROYAL INSTITUTE OF TECHNOLOGY
SCHOOL OF INDUSTRIAL ENGINEERING AND MANAGEMENT

Robotic Arm controlled by Arm Movements

MIKO NORE
CASPAR WESTERBERG

Bachelor's Thesis at ITM
Supervisor: Nihad Subasic
Examiner: Nihad Subasic

TRITA-ITM-EX 2019:26

Abstract

In recent decades human workers in manufacturing and overall industry have largely been replaced with robots and automated machines, but there are still plenty of tasks where human cognition is necessary. This paper presents the development of a wireless robotic arm controlled by a human arm, allowing both for the combination of a robotic arms strength to be combined with a humans cognition, and also for a human to execute dynamic tasks without being present. An application suited for work in toxic or otherwise harmful environments. This was accomplished by using a controller in the form of an exo-skeleton attached to the operators right arm and connected to the robotic arm through a transmitter. The controller measures the movements in each joint using potentiometers and the robotic arm mimics these movements. A glove with a flex sensor on the index finger was then attached to the controller to measure the finger motions. All the information containing the angle of rotations are sent wirelessly to the robotic arm using Arduino Uno and transceiver modules. The robotic arm received the information through another set of Arduino Uno and transceiver module which made each servomotor on the robotic arm to move accordingly.

The result showed that the robotic arm could imitate the operator's arm very well and was able to grab and move different objects with different weight and surfaces. The wireless control was reliable and could control the robotic arm while being in a different room, making it possible to use this robot for harmful environments for humans.

Keywords
Mechatronics, Robotic arm, Remote Control, Wireless.

Referat

Robotarm som följer armrörelser

Under senare årtionden har mänskliga arbetare inom tillverkning och industri över lag i stor utsträckning ersatts av robotar och automatiserade maskiner, men det finns fortfarande uppgifter som kräver mänsklig tankeförmåga. Denna rapport presenterar utvecklingen av en trådlös robotarm styrd av en människas arm, vilket möjliggör både att kombinera en maskins styrka med en människas intelligens, samt för en människa att utföra dynamiska uppgifter utan att vara närvarande. En applikation lämplig för arbete i farliga miljöer. Detta uppnåddes med en styrenhet i form av ett exo-skelett fastsatt på operatörens högra arm och kopplad till robotarmen genom en sändare. Styrenheten mäter rörelserna i varje led med potentiometrar och robotarmen härmar dessa rörelser. En handske med en flexsensor på pekfingret fästes sedan på styrenheten för att mäta fingerrörelsen. All information som innehåller vinklar skickas trådlöst till robotarmen med hjälp av Arduino Uno och transceiver moduler. Robotarmen mottog informationen via en annan uppsättning Arduino Uno och transceiver modul som fick varje servomotor på robotarmen att rotera i enlighet.

Resultatet visade att robotarmen kunde imitera operatörens arm väl och kunde bära olika föremål med olika vikter och ytor. Den trådlösa styrningen var pålitlig och kunde styra robotarmen från ett annat rum, vilket gör det möjligt att använda denna robot i skadliga miljöer för människor.

Nyckelord
Mekatronik, Robotarm, Fjärrstyrning, Trådlös.

Acknowledgements

We would like to thank Nihad Subasic for guiding us through this project. We would also like to thank Seshagopalan Thorapalli Muralidharan and Sresht Iyer for providing assistance in the lab and Staffan Qvarnström and Tomas Östberg for helping us with acquiring parts and guidance on using the machines. A special thanks to Johan Widmark for helping with 3D printing. Without anyone of them this project would not have been possible.

Contents

List of Figures

Nomenclature

CAD	Computer-Aided Design
PID	Proportional-Integral-Derivative
PWM	Pulse Width Modulation
RF	Radio Frequency
SPI	Serial Peripheral Interface

Chapter 1

Introduction

This project considers the construction of a robotic arm that can be steered by a controller in the form of an exoskeleton attached to the operator's arm. The aim is to get the robotic arm to imitate the movements of the operator's arm and be able to grab things. The signals between the controller and the robotic arm should be wireless, allowing for free movement while wielding the controller. A further development of this project could be to introduce memory for the robotic arm, making it possible to easily automate motions.

In recent years many applications for robotic arms have been researched. Wireless control applications are increasing and demand to send information through long distances are more important than ever. Wireless control makes it possible to place and pick up hazardous objects, or handle objects in a hazardous environment, from afar [1]. In the future controlling a mechanic arm in this way may be possible using only the mind, but until then, using an exoskeleton controller, is the most efficient [2].

1.1 Purpose

This task was chosen as it has a concrete and clear goal that should be realizable. The degree of difficulty seems reasonable and there are plenty of conceivable applications for the product. These applications primarily consist of manual work and interaction in environments with high risk of injury, hazardous substances or high voltage lines. Examples of this could be work in radioactive areas at nuclear power plants.

These are the research questions that will be investigated in this thesis:

- How well can the robotic arm imitate the operating arm?
- How well can the robotic arm grab and move objects?
- What applications are there for the product? For example, industry or harmful environments?

If time allows, another researching topic would be to investigate how well the robotic arm can remember certain movements.

1.2 Scope

Due to the time limit, there were few limitations. The robotic arm's movements were limited in degrees of freedom in each joint; one degree of freedom in the wrist, elbow and the shoulder attachment (bending in one plane), and a hand that performs a clamping movement. The robotic arm's movement on each joint is illustrated in Figure 1.1. If time allows an extra degree of freedom might be added to the shoulder attachment, allowing the arm to rotate around a center axis.

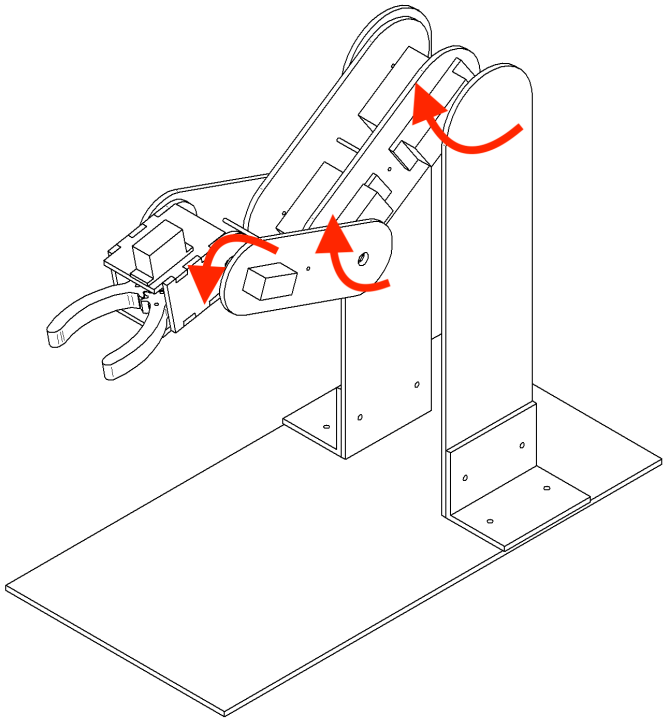


Figure 1.1: Degrees of freedom in each joint of the robotic arm (sketch made in Solid Edge ST10 [3]).

1.3 Method

The project started with researching different methods to build a robotic arm, taking inspiration from other projects. The method most suited was then used to make the first sketch of the robotic arm and the controller. The construction design was made in CAD using Solid Edge ST10 [3]. After assembling the structural pieces and motors of the robotic arm, the theory behind remote control was researched and applied to the construction. When the desired interaction between the controller and the robotic arm was achieved and if time allowed, the extra developments and research questions would be investigated.

Chapter 2

Theory

2.1 Past Research

A previous Bachelor's Thesis "Human controlled robotic arm" by Simon Ahlin Högfeldt and Daniel Söderman was used for inspiration. The goal of their robotic arm was to test the feedback system using PID controller and to implement haptic control [6]. Haptic means "Of or relating to the sense of touch". Haptic control combines vision, sound, touch, force, pressure or heat (or a combination of several) and often used to make the simulation realistic to the user [7]. The gaming industry has for example developed haptic feedback on the game controllers with force feedback and vibrations.

Though it was not the purpose of this project to test the haptic feedback of the robot, there were many similarities and lots of useful information. Mainly the controlling part, using servomotors for the robotic arm and potentiometers for the arm unit. The problems they faced was the servomotors not being able to create the torque given by the manufacturer and this caused them to shorten the length and weight of the robot arm itself rather than buying a stronger servomotor. A haptic feedback would give the user a feeling of the object improves the application of the robot, especially if the robotic arm and the user are separated.

The wireless inspiration came from the Bachelor's thesis "Communications solution for refugee settlement: Investigation of nRF24L01+ modules for use in a communications network" by Martin Engquist and Simon Bethdavid. The Radio Frequency Module, NRF24L01 is used for the wireless network combined with Arduino. Messages such as text, audio and commands were able to send and recieve but the NRF24L01 Modules was not very reliable to their project [8].

2.2 Potentiometer

A potentiometer is a variable resistance consisting of a rotating knob, a resistance and three pins, one for feeding voltage, one for ground and one connected to a sliding contact on the resistance. The knob is the moving part and is restricted to rotation

between 0-180 degrees. As you turn the knob the contact between the measuring pin and the resistance moves along the resistance, varying the measured voltage from 0 to some maximum value V depending on the feeding voltage. Reading this voltage and recalibrating it from 0 - V to 0 - 180 (or 180 - 0 for a mirrored rotation) effectively gives you the angular position of the potentiometer in degrees from 0° to 180°. This is how the position of the joint was given to the servomotors [11].

2.3 Flex sensor

A Flex sensor (also called bend sensor) is a resistor whose resistance varies as it bends. The variations in resistance caused by the bending can then be digitally recorded [12]. The measured deflection is the angle between the front and the back of the sensor and has two terminals because it measures resistance.

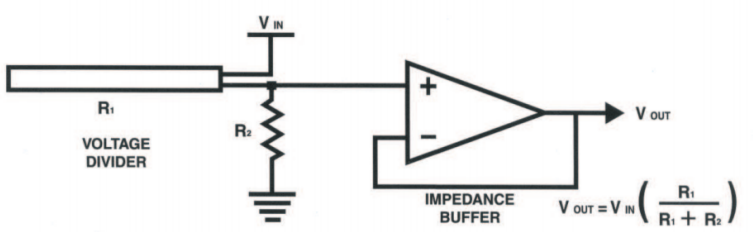


Figure 2.1: Basic Flex Sensor Circuit (see Appendix A.9).

Figure 2.1 shows how the basic flex sensor circuit works. The circuit is a voltage divider circuit and works in such a way that the sensors uses the impedance buffer as a single sided operational amplifier (op amp), because of the low current of the op amp reduced error due to source impedance.

2.4 Standard rotary servomotor

Servomotors contain both a motor and a regulator. The regulator takes command input from outside as reference and takes feedback from an internal potentiometer connected to the output shaft and uses these to regulate the motor. Standard rotary servomotors are restricted to rotation between 0-180 degrees and takes an angle as command input. The motors inside have low torque and high speed but are connected to a set of gears significantly reducing speed and increasing torque [10].

Since the feedback system is internal, the standard servomotor has a closed-loop system inside the servomotor case, but open-looped considering the microcontroller [9]. This causes the problem that you ca not confirm if the movement of the servomotor is as desired.

2.4.1 Pulse Width Modulation

Pulse Width Modulation is used to control servomotors, which is a technique to translate digital outputs to control analog circuits. The initial position of the servomotor is 90° and can move the same amount in both directions for a total of 180° movement [10].

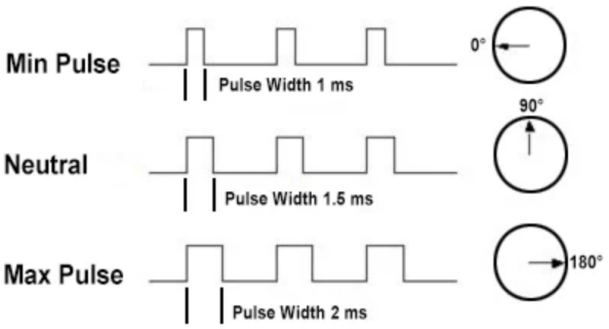


Figure 2.2: Servo position based on PWM [10].

For example, a 1.5 ms pulse will turn the servomotor to 90°, as shown in Figure 2.2. Shorter pulse will move it 90° counter clockwise, to the 0° position and longer pulse will move it clockwise to the 180° position.

The PWM will tell the motor to move in the determined position of the shaft and the duration of the pulse will make the rotor turn to that desired position. Every 20 ms a new pulse will determine how far the motor turns. After given a certain position, the motor will stay in place and resists to move even when an external force pushes it. The maximum amount of force the servo can exert is called the torque rating of the servo. But to remain in the same position, the new pulse must repeat itself otherwise it will move to a new position [10].

2.5 Radio Frequency Module

Radio frequency (RF) refers to the rate of oscillation of electromagnetic radio waves in the range of 3 kHz to 300 GHz, as well as the alternating currents carrying the radio signals. These electromagnetic waves are used to transport information over a distance and are radiated by electric charges undergoing acceleration. Two RF Modules are needed to create a communications link [13].

2.5.1 NRF24L01 Module

The NRF24L01 Module is a 2.4 GHz RF transceiver module which means that each module can both send as well as receive data. This single chip transceiver has an embedded baseband protocol engine for ultra low power wireless applications and operates at 3.3 V. The NRF24L01 Module has 125 channels and maximum data

rate of 2 MBps. One NRF24L01 Module can actively listen up to 6 other Modules. This makes it possible to build a wireless network with multiple NRF24L01 modules [14].

The NRF24L01 module uses Serial Peripheral Interface (SPI) to communicate. The SPI uses one line for data and a separate line for a "clock" that keeps the data flow synchronous. The oscillating signal from the clock tells the receiver when to sample the bits on the data line. Asynchronous serial uses a single "clock" to communicate which can be a problem when two systems with different clocks tries to communicate with each other because random amouts of data can be send at any time in either direction [15].

The pin connections are CE, CSN, SCK, MOSI, MISO and IRQ. CE activates data reception and transmission when in Receiver and Transmitter mode, respectively. CSN stands for Chip Select Not and enables pin for the SPI bus. SCK is the serial clock for the SPI bus. In SPI only one side generates the clock signal. The side that generates the clock is the "master" and the other side is the "slave". There can only be one master but there can be multiple slaves. When the master sends data to the slave, the data line is called MOSI which stands for Master Out Slave In. If the slave sends a response back to the master, MISO is used. Similar to MOSI, MISO stands for Master In Slave Out. IRQ is used for interruption and is generally not necessary [15].

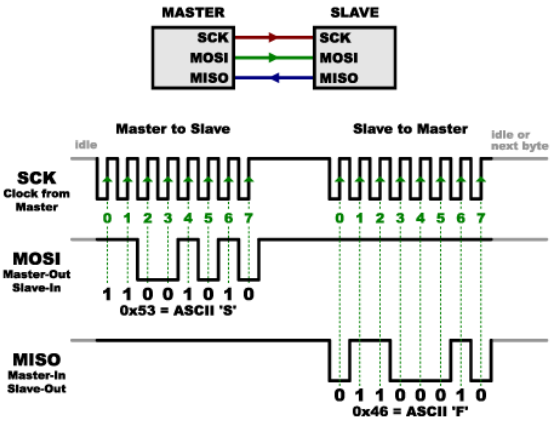


Figure 2.3: Recieving data using SPI [15].

In Figure 2.3 the SCK, MOSI and MISO signals are illustrated. The master sends the clock signal using SCK, which is prearranged. The master knows in advance when a slave returns data and how much data will be returned and thus making the data flow syncronous.

Chapter 3

Demonstrator

This chapter shows the constructions and how the prototype was made. The prototype was then used for testing in order to answer the research questions.

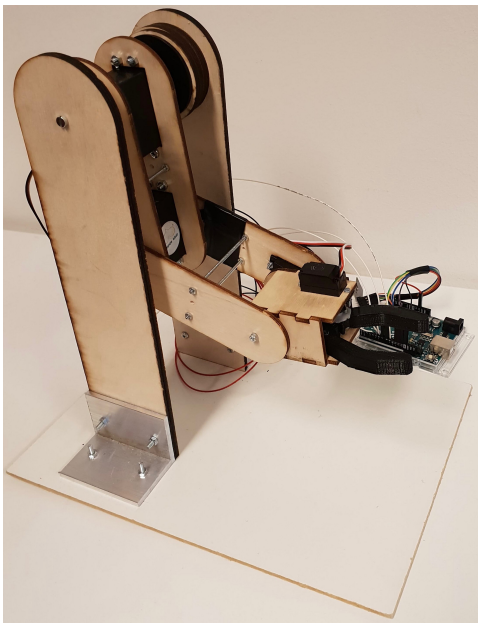
3.1 Construction

The construction of this prototype consists of two parts, the controller and the robotic arm. The controller's purpose is to allow a human arm control of the robotic arm. Once attached to the operator it will wirelessly transmit information of the operating arm's movements to the robotic arm. The robotic arm will then imitate these movements.

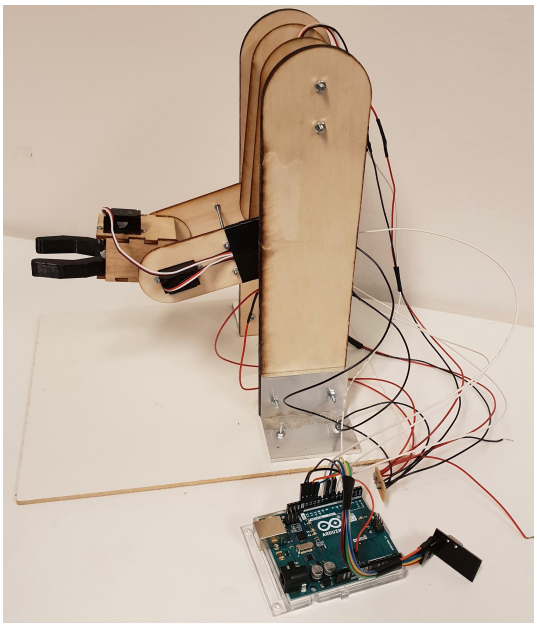
3.1.1 The robotic arm design

The robotic arm consists of the upper arm, lower arm, hand and housing. The housing was the simplest construction of the four, consisting of two parallel plates perpendicularly screwed to the footing plate using aluminum brackets. Both the upper and lower arm consist of two plates held at a fixed distance using threaded rods and nuts, with one of the two plates holding the servomotors. The hand consists of four plates glued together and holds the two claws. One claw was loosely fixed to the bottom plate with a screw allowing rotation. The driving claw is glued to a servomotor fixed to the top plate and transfers torque to the other claw through a set of cogs.

The robotic arm was first designed using Solid Edge [3]. All plates apart from the claw, the footing plate and the aluminum brackets are made from plywood and were laser cut to their appropriate shape. The claw was 3D-printed using Ultimaker 3. Plywood was chosen as the main material as it is cheap, easily replaceable and very easy to work with, making it suitable for a prototype [4]. The finished prototype can be seen on the next page, Figures 3.1 and 3.2.



(a) The robotic arm from the right.



(b) The robotic arm from the left.

Figure 3.1: The robotic arm prototype.

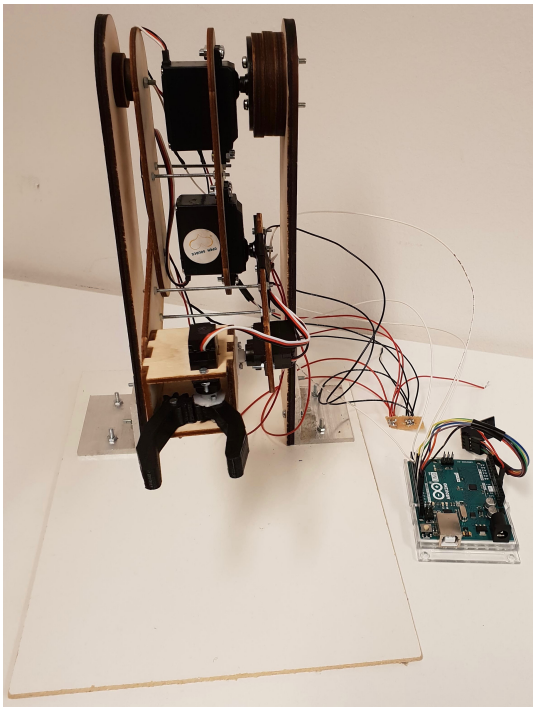


Figure 3.2: The robotic arm from the front.

3.1.2 The controller design

The controller mainly consists of 4 rigid parts made in plywood: back plate, upper arm, lower arm and hand. Each part was fixed to the corresponding bodypart of the operator with hook-and-loop fastener and was connected to the next rigid part in a joint consisting of a potentiometer. The rotating knob of the potentiometer was fixed to one of the rigid parts and the body of the potentiometer to the other. This way the potentiometer registers angular movement and sends it to the arduino screwed to the back plate.

Just like the robotic arm the controller was first designed in Solid Edge [3] and then laser cut and constructed in plywood. The finished controller attached to the operator is shown in Figure 3.3 and the controller detached is shown in Figure 3.4.

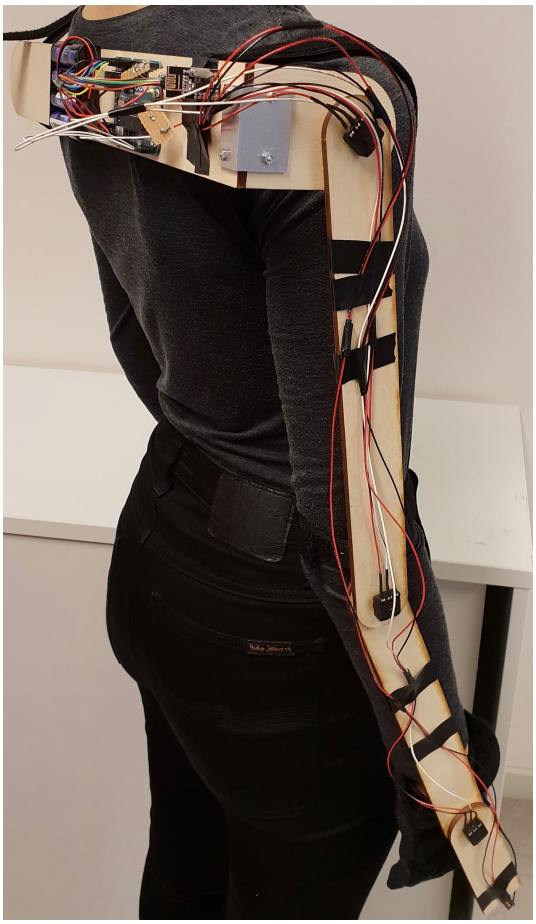


Figure 3.3: The controller attached to the operator.

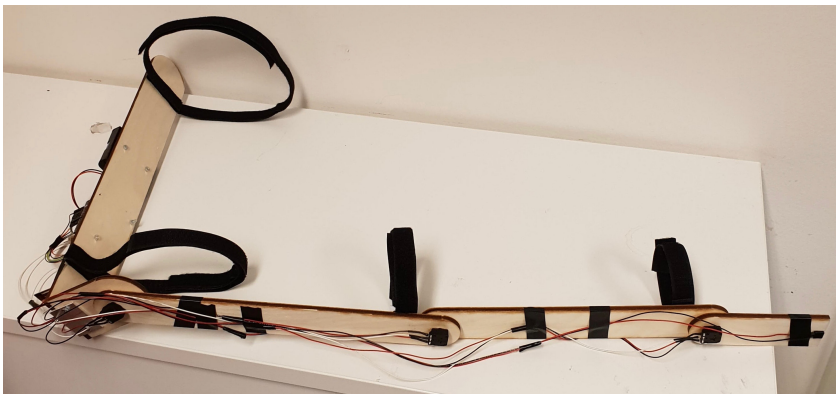


Figure 3.4: The controller.

The hands gripping motion is measured using a flex sensor sewn to a glove connected to the rigid hand part. In this prototype only one flex sensor on the index finger was used to measure the angular position and measures the degrees of bending, which will be sufficient to control the claw's gripping movement. The glove is detachable from the main controller and velcro is sewn on the glove to put it in place when attached to the main controller. The glove is shown in Figure 3.5.

Figure 3.5: The glove with the flex sensor.
sensor

3.2 Software

The software used to control the robotic arm is running on one Arduino Uno and the software used to measure the angular rotation is running on a second Arduino Uno. The software is written in C in the Arduino environment [5]. The full code for the transmitter and receiver can be found at Appendix A.1.

3.3 Electronic components

Two Arduino Uno and two NRF24L01 Modules are necessary to make the construction wireless. One for the transmitter and one for the receiver. The controller is on the transmitter side and the robotic arm on the receiver side. The transmitter sends the measured angles from the potentiometers and flex sensor to the receiver side which is used to control the rotation of the four servomotors.

3.3.1 The robotic arm components

The robotic arm uses servomotors for movement. A standard servomotors rotation is easily controlled using Arduino by simply telling it what angle to rotate to, generally limited to 180 degrees rotation. By importing the value from the potentiometers and flex sensor and remapping it to a value between 0-180 the rotation of the servomotor accurately reflects that of the potentiometer and flex sensor [16].

To make it wireless a pair of radio frequency transceivers, NRF24L01 module is connected. One transceiver on the robotic arm and one on the controller. The robotic arm works as the receiver and the electronic components used for the circuit are:

- 1 Arduino Uno
- 2 Standard servomotors
- 2 Sub-Micro servomotors
- 2 NRF24L01 Transceiver Module

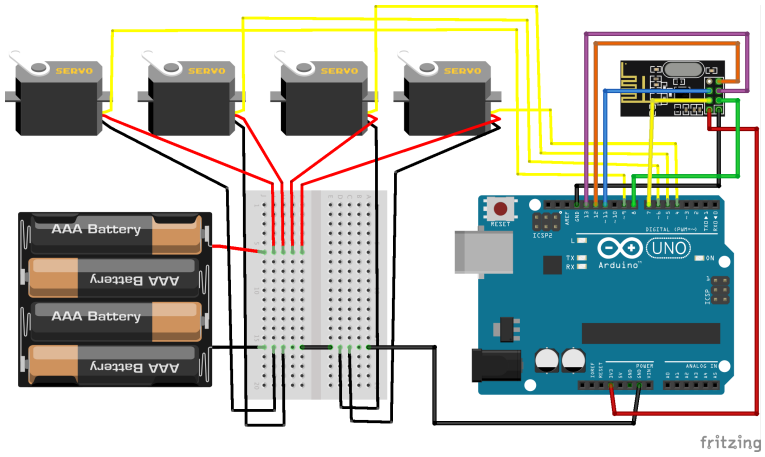


Figure 3.6: Receiver circuit made using Fritzing [17].

The servomotors in the shoulder and elbow joint of the robotic arm are Parallax Standard Servomotors (see Appendix A.6) and are larger and stronger than the the Sub-Micro servomotors (see Appendix A.7) used in the hand joint and gripping claw. An external power source is needed to power all the servomotors, without damaging the Arduino Uno.

3.3.2 The Controller components

The controller works as the transmitter and the electronic components used for the circuit are:

- 1 Arduino Uno
- 3 Potentiometers
- 1 Flex Sensor
- 1 NRF24L01 Transceiver Module
- 1 10k Resistor

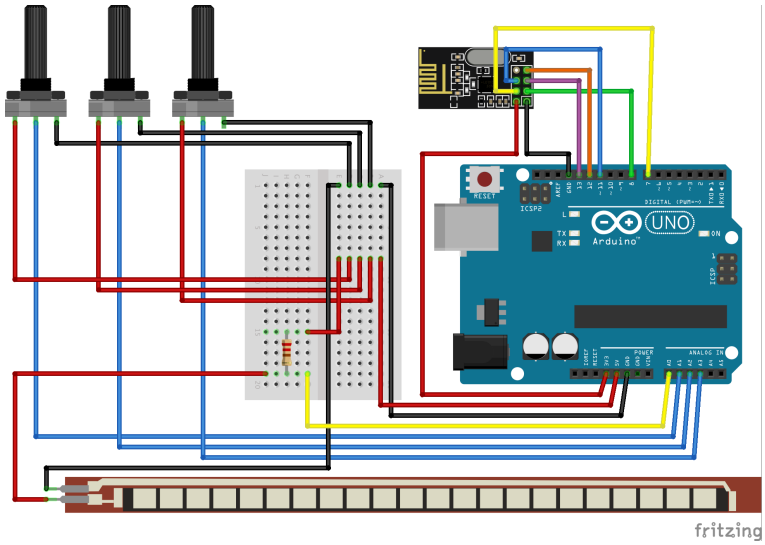


Figure 3.7: Transmitter circuit made using Fritzing [17].

To make the controller wireless, a set of four AA batteries will be connected to the DC jack of the Arduino.

3.4 Results

The result from the testing concludes that the robotic arm was able to imitate the operator's arm at each joint in real time without any noticeable delay. The rotations in the joints were accurately imitated by the servomotors unless the arm was carrying a heavy load. The error when carrying a load can likely be attributed to the weak motors.

The transmitters worked well with the operator being able to control the robotic arm standing in a different room. The largest successfully attempted distance was around 20 meters with obstacles and a wall in between the operator and the arm. According to the transmitter it could manage 100 meters without obstacles .We could not confirm this as we would have to do it outdoors and we are dependent on

an external power supply. The claw could hold objects with different shapes and surfaces and the arm could lift weights up to around 70 grams before the upper arms inner motor began to fail.

Chapter 4

Discussion and conclusion

This chapter discusses the results and the conclusions to the researched questions.

4.1 Discussion

The robotic arm was able to imitate the operating arm very well. It moved in real time as the operator. Servomotors are not very strong and because the feedback is internal they can be unpredictable and sometimes start shaking.

The proportions of the controller were not perfect. For example the potentiometer joint connecting the back plate to the upper arm should be centered over the operators shoulder joint, which it wasn't. This caused an error in the information of the operating arms movements transferred to the robotic arm. This would not be difficult to adjust but it would take some time. Overall the proportions of the controller were decent, but seeing as the fit differs from user to user they will never be perfect unless they are made adjustable.

The robotic arm could grab different objects with different weights and surfaces. However, it could only hold items up to 70 grams. Over 70 grams and it won't be able to move it higher up as it became too heavy for the shoulder servo.

The NRF24L01 Module was easy to learn and is great for small projects like this as the signal is reliable. The robotic arm could be controlled from another room, making the original idea of using the arm in environments harmful for humans feasible, even without upgrading the transmitter. The wireless control could be improved with the use of antennas. Each part made from plywood could easily be replaced with a part made from materials capable of withstanding for example high temperature and be adjusted to hold stronger more reliable motors.

4.2 Conclusion

The research questions all had positive answers, although there were some limitations. The movements worked very well, but there are areas for improvements. The controller was uncomfortable and not very practical as attaching it was dependent

on the help from another person. It could not be attached without help but it could be removed without help. It is hard to move around with the controller attached and the movements are limited. The robotic arm could also be made bigger and stronger.

A possible addition to the robotic arm could be the ability to record movements and repeat them allowing the user to easily automate repetitive tasks. This could rather easily be implemented by saving all angles in some vector or matrix every 0.1 seconds or so, then at execution going through the vector, moving the servos to the recorded angles. The only thing needed to add this functionality is some button or signal to start and stop recording.

Another possible development on the controller is to make a color recognition sensor for the arm movements instead of potentiometers. This will make it easier to put the controller on the arm and would not be as uncomfortable and unpractical. Flex sensor could also be used instead of potentiometers on each joint, which will greatly reduce the heavy and awkward controller. Although this alternative is more expensive.

Bibliography

[1] Mohd Ashiq Kamaril Yusoffa, Reza Ezuan Saminb and Babul Salam Kader Ibrahimc, "Wireless Mobile Robotic Arm" (2012). Date accessed: 2019-04-03. [Online]. Available: *https* : *//www.sciencedirect.com/science/article/pii/S*1877705812026859

[2] Salvador Dura-Bernal, George L. Chadderdon, Samuel A. Neymotin, Joseph T. Francis and William W. Lytton, "Towards a real-time interface between a biomimetic model of sensorimotor cortex and a robotic arm" (2013). Date accessed: 2019-04-03. [Online]. Available: *https* : *//www.sciencedirect.com/science/article/pii/S*0167865513002171

[3] *Solid Edge ST10* 2019, Software *https* : *//solidedge.siemens.com/en/*

[4] *Ultimaker* 2019, 3D printer *https* : *//ultimaker.com/*

[5] *Arduino* 2019, Software. *https* : *//www.arduino.cc/en/Main/Software*

[6] S. Ahlin Högfeldt and D. Söderman, "Human controlled robotic arm", KTH, Tech. Rep., 2016. Date accessed: 2019-02-14. [Online]. Available: *http* : *//kth.diva − portal.org/smash/get/diva*2 : 955299/*FULLTEXT*01.*pdf*

[7] Stone R.J. Haptic feedback: a brief history from telepresence to virtual reality (2001). Date accessed: 2019-04-03. [Online]. Available: *http* : *//citeseerx.ist.psu.edu/viewdoc/download?doi* = 10.1.1.21.8916*rep* = *rep*1*type* = *pdf*

[8] M. Engquist and S. Bethdavid, "Communications solution for refugee settlement: Investigation of nRF24L01+ modules for use in a communications network", Uppsala Univeristy, Tech. Rep., 2018. Date accessed: 2019-04-03. [Online]. Available: *http* : *//www.diva − portal.org/smash/get/diva*2 : 1222330/*FULLTEXT*01.*pdf*

[9] B. Earl, "About Servos and Feedback" (2013). Date accessed: 2019-04-02. [Online]. Available: *https* : *//learn.adafruit.com/analog − feedback − servos/about − servos − and − feedback*

[10] *Jameco Electronics*, "How Servo Motors Work" (2012). Date accessed: 2019-04-02. [Online]. Available: *https* : *//www.jameco.com/jameco/workshop/howitworks/how − servo − motors − work.html*

[11] *Elliot Sound Products*, R. Elliot, "Beginners' Guide to Potentiometers" (2002). Date accessed: 2019-02-14.

[12] A. Sreejan and Y. Sivraj Narayan "A Review on Applications of Flex Sensors", Volume 7 (2017). Date accessed: 2019-02-13.

[13] A. Raj, "RF Transmitter and Receiver Circuit" (2017). Date accessed: 2019-02-13. [Online]. Available: *https* : *//circuitdigest.com/electronic − circuits/rf − transmitter − and − receiver − circuit − diagram*

[14] *Nordic Semiconductor* (2007). nRF24L01 Single Chip 2.4GHz Transceiver Product Specification. Date accessed: 2019-03-28. [Online]. Available: *https* : *//www.electrokit.com/uploads/productfile/*41013/$DS_nRF24L01 - 2.pdf$

[15] *Motorola, Inc.* (2000). SPI Block Guide V03.06. Date accessed: 2019-03-28. [Online]. Available: *https* : *//web.archive.org/web/*20150413003534/*http* : *//www.ee.nmt.edu/ teare/ee*308*l/datasheets/S*12*SPIV*3*.pdf*

[16] S. Kobalczyk, "Robotic Arm - Hardware" (2012). *Szymon Kobalczyk's Bloh - A Developer's Notebook*. Date accessed: 2019-02-06. [Online]. Available: *http* : *//geekswithblogs.net/kobush/archive/*2012/04/09/149258*.aspx*

[17] *Fritzing* 2019, Software. *http* : *//fritzing.org/home/*

Appendix A

Appendix

A.1 Arduino code

```
//=============================================================
// Transmitter code for the controller
// Sending the measured angles from the potentiometers
// and flexsensors to the receiver (the robotic arm)
// Miko Nore and Caspar Westerberg
// KTH, 2019-05-22
//=============================================================

#include <SPI.h>
#include "RF24.h"

const int num = 4;
int Array[num];
RF24 radio(8,7);
const uint64_t pipe = 0xE8E8F0F0E1LL;
const int flexpin = 0; // flex sensor for the claw
int potpin1 = 1; // hand
int potpin2 = 2; // elbow
int potpin3 = 3; // shoulder
int val1;
int val2;
int val3;
int val4;

void setup(void){
  radio.begin();
  radio.openWritingPipe(pipe);
  }

void loop(void){
  val1 = analogRead(flexpin); // read the value from the flex sensor
  val1 = map(val1, 600, 900, 160, 120); // convert the value to an
    angle

  val2 = analogRead(potpin1); // read the value from the potentiometer
```

```
    on the hand
  val2 = map(val2, 0, 1023, 179, 0); // convert the value to an angle

  val3 = analogRead(potpin2); // read the value from the potentiometer
    on the elbow
  val3 = map(val3, 0, 1023, 0, 179); // convert the value to an angle

  val4 = analogRead(potpin3); // read the value from the potentiometer
    on the shoulder
  val4 = map(val4, 0, 1023, 0, 179); // convert the value to an angle

  Array[0] = val1; Array[1] = val2; Array[2] = val3; Array[3] = val4;
    // an array with all the angles

  radio.write(&Array, sizeof(Array)); // send the array to the robot
    arm
  }
```

```
//=============================================================
// Receiver code for the robotic arm
// Receiving the measured angles from the transmitter
// and moving the servo motors accordingly
// Miko Nore and Caspar Westerberg
// KTH, 2019-05-22
//=============================================================

#include <SPI.h>
#include <Servo.h>
#include "RF24.h"
const int num = 4;
int Array[num];
RF24 radio(8,7);
const uint64_t pipe = 0xE8E8F0F0E1LL;
Servo servo1; // claw servo
Servo servo2; // hand servo
Servo servo3; // elbow servo
Servo servo4; // shoulder servo

void setup(){
  servo1.attach(9);
  servo2.attach(6);
  servo3.attach(5);
  servo4.attach(4);
  radio.begin();
  radio.openReadingPipe(1, pipe);
  radio.startListening();
  }

void loop(){
  if (radio.available()){ // if a message is available
    bool done = false;
    while (!done){
      done = radio.read(&Array, sizeof(Array)); // receive the array in
    the message
```

```
36        servo1.write(Array[0]); // first value moves the claw
37        servo2.write(Array[1]); // second value moves the hand
38        servo3.write(Array[2]); // third value moves the elbow
39        servo4.write(Array[3]); // fourth vale moves the shoulder
40        }
41      }
42    }
```

A.2 Parallax Servo

PARALLAX

Web Site: www.parallax.com
Forums: forums.parallax.com
Sales: sales@parallax.com
Technical: support@parallax.com

Office: (916) 624-8333
Fax: (916) 624-8003
Sales: (888) 512-1024
Tech Support: (888) 997-8267

Parallax Standard Servo (#900-00005)

The Parallax Standard Servo provides 180° range of motion and position control to your project. Great for animatronics and robotics applications.

Features

- Holds any position between 0 and 180 degrees
- 38 oz-in torque at 6 VDC
- Accepts four mounting screws
- Easy to interface with any Parallax microcontroller or PWM-capable device
- Simple to control with the PULSOUT command in PBASIC
- High-precision gear made of POM (polyacetal) resin makes for smooth operation with no backlash
- Weighs only 1.55 oz (44 g)

Key Specifications

- Power requirements: 4 to 6 VDC*; Maximum current draw is 140 +/- 50 mA at 6 VDC when operating in no load conditions, 15 mA when in static state
- Communication: Pulse-width modulation, 0.75–2.25 ms high pulse, 20 ms intervals
- Dimensions approx 2.2 x 0.8 x 1.6 in (5.58 x 1.9 x 40.6 cm) excluding servo horn
- Operating temperature range: 14 to 122 °F (-10 to +50 °C)

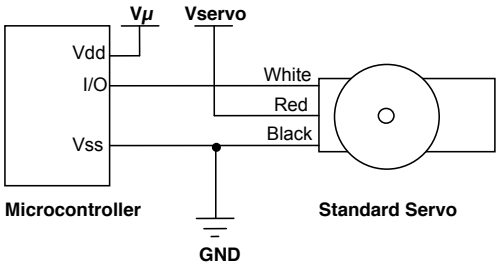
*Power Requirement Notes

The manufacturer specifies 4-6 VDC for this servo. However, we find that this servo is tolerant of a 9 V battery for very brief periods of time when there is no load, as used in some activities in the Stamps in Class series of tutorials. (Slight jittering may be observed when batteries are fresh; this does not cause damage). Do not use this servo with an unregulated wall-mount supply, or a regulated wall mount supply exceeding 6 VDC.

Servo current draw can spike while under load. Be sure that your application's power supply and voltage regulator is prepared to supply adequate current for all servos used. Do not try to power this servo directly from a BASIC Stamp module's or any microcontroller's Vdd or Vin pins; do not connect the servo's Vss line directly to the BASIC Stamp module's or any microcontroller's Vss pin.

A.3 Parallax Servo

Quick-Start Circuit



Vμ = microcontroller voltage supply

Vservo = 4 to 6 VDC, regulated or battery

I/O = PWM TTL or CMOS output signal from microcontroller: 3.3 to 5 V, not to exceed Vservo + 0.2 V

Specifications

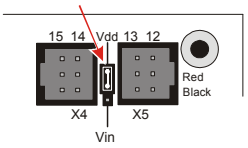
Pin	Name	Description	Minimum	Typical	Maximum	Units
1 (White)	Signal	Input; TTL or CMOS	3.3	5.0	Vservo + 0.2	V
2 (Red)	Vservo	Power Supply	4.0	5.0	6.0	V
3 (Black)	Vss	Ground		0		V

Power Precautions

- Do not use this servo with an unregulated wall-mount supply. Such power supplies may deliver variable voltage far above the stated voltage.
- Do not power this servo through the BASIC Stamp Module's Vdd pin.
- Servo current draw can spike while under peak load; be sure your application's regulator is prepared to supply adequate current for all servos used in combination.
- Some Stamps in Class tutorials, such as *What's a Microcontroller?* instruct the user to briefly power these servos with a 9 V battery when using a HomeWork Board and no load; this does not cause damage.

Board of Education Jumper Connection

When connecting the servo to the Board of Education Rev C's servo header, be sure the jumper is set to Vdd (regulated 5 VDC for this board) as shown in the figure below. Failure to place the jumper at this setting can cause damage your servo.



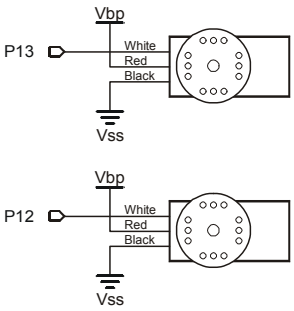
Using a Separate Power Supply on a HomeWork Board

The BASIC Stamp HomeWork Board uses a 9 V battery for a power supply. A servo can drain a fresh 9 V battery in under 20 minutes! Follow these directions to build two servo ports on the breadboard, and power them with a separate battery pack.

Hardware Required
(1) BASIC Stamp HomeWork Board (serial #555-28158 or USB #555-28188)
(1) Battery pack with tinned leads (Parallax #753-00001)
(2) Parallax standard servos (#900-00005)
(2) 3-pin male-male headers (Parallax #451-00303)
(4) Jumper wires (10-pack: Parallax #800-00016)
(4) 1.5 V AA batteries

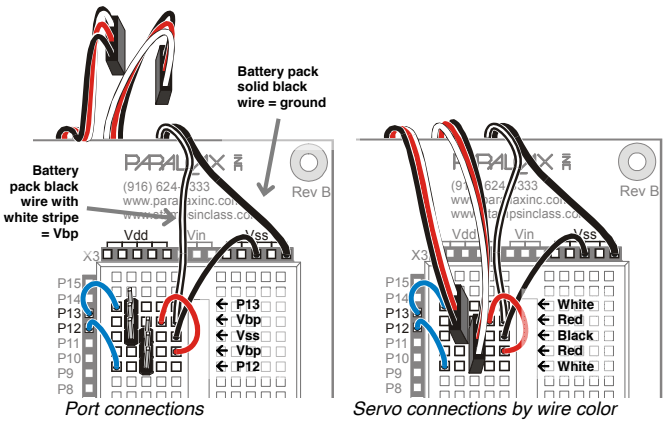
A.4 Parallax Servo

- √ Disconnect the 9 V battery from the board, and do not put the AA batteries in their holder yet.
- √ Build the servo ports shown by the schematic and wiring diagram below.
- √ Double-check to make sure the black wire with the white stripe is connected to Vbp, the solid black wire is connected to Vss, and that all the connections for P13, Vbp, Vss, Vbp (another one), and P12 all exactly match the wiring diagram.
- √ Connect the servo plugs to the male headers on the right side of the wiring diagram.
- √ Connect the 9 V battery, and insert the AA batteries into their holder.

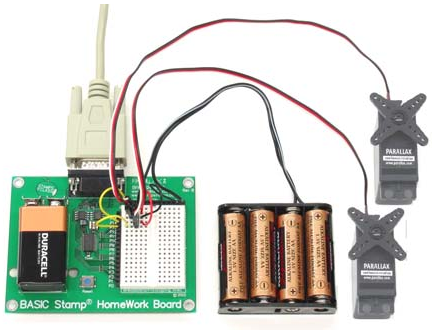


Vbp stands for Voltage battery pack.

It refers to the 6 VDC supplied by the four 1.5 V batteries. This is brought directly to the breadboard to power the servos. BASIC Stamp is still powered by the 9 V battery.



Port connections *Servo connections by wire color*

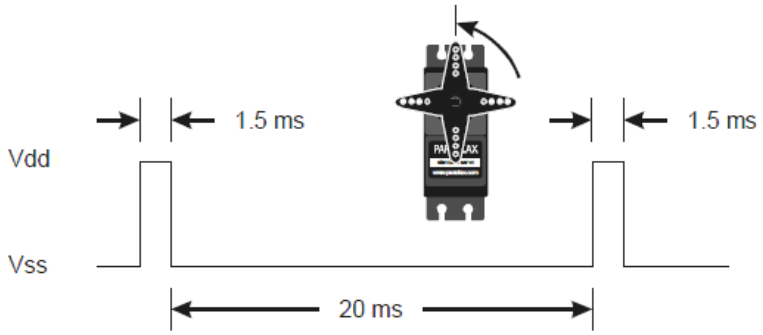


HomeWork Board with servo ports built on the breadboard, with a separate battery pack power supply for the servos.

A.5 Parallax Servo

Communication Protocol

The Parallax Standard Servo is controlled through pulse width modulation, where the position of the servo shaft is dependent on the duration of the pulse. In order to hold its position, the servo needs to receive a pulse every 20 ms. Below is a sample timing diagram for the center position of the Parallax Standard Servo.



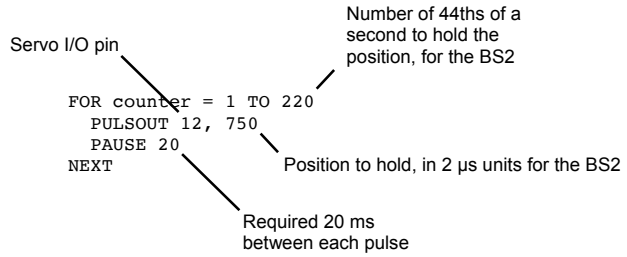
BASIC Stamp 2 Programming Example

PBASIC has a PULSOUT command that sets the I/O pin to an output and sends a pulse of a specified duration.

PULSOUT *Pin*, *Duration*

The example shown below for a BASIC Stamp 2 causes a servo connected to BASIC Stamp I/0 pin 12 to turn to and hold its center position for approximately 5 seconds.

```
' CenterStdServo.bs2
' {$STAMP BS2}
' {$PBASIC 2.5}

counter VAR Word

FOR counter = 1 TO 220

  PULSOUT 12, 750
  PAUSE 20

NEXT
```



A.6 Parallax Servo

For detailed explanations using the BASIC Stamp 2, see *What's a Microcontroller?* Chapter 4, available for free download from www.parallax.com/go/WAM.

Different BASIC Stamp modules use different units for the PULSOUT command's *Duration* argument. When adapting BS2 code to another BASIC Stamp model, you may need to make adjustments. See the article "Adapt BS2 Code to Other Models" in the PBASIC Language section of the BASIC Stamp Editor Help.

The table below lists the PULSOUT ranges for each BASIC Stamp model.

BASIC Stamp Module	0.75 ms	1.5 ms (center)	2.25 ms
BS1	75	150	225
BS2, BS2e, BS2pe	375	750	1125
BS2sx, BS2px, BS2p24/40	938	1875	2813

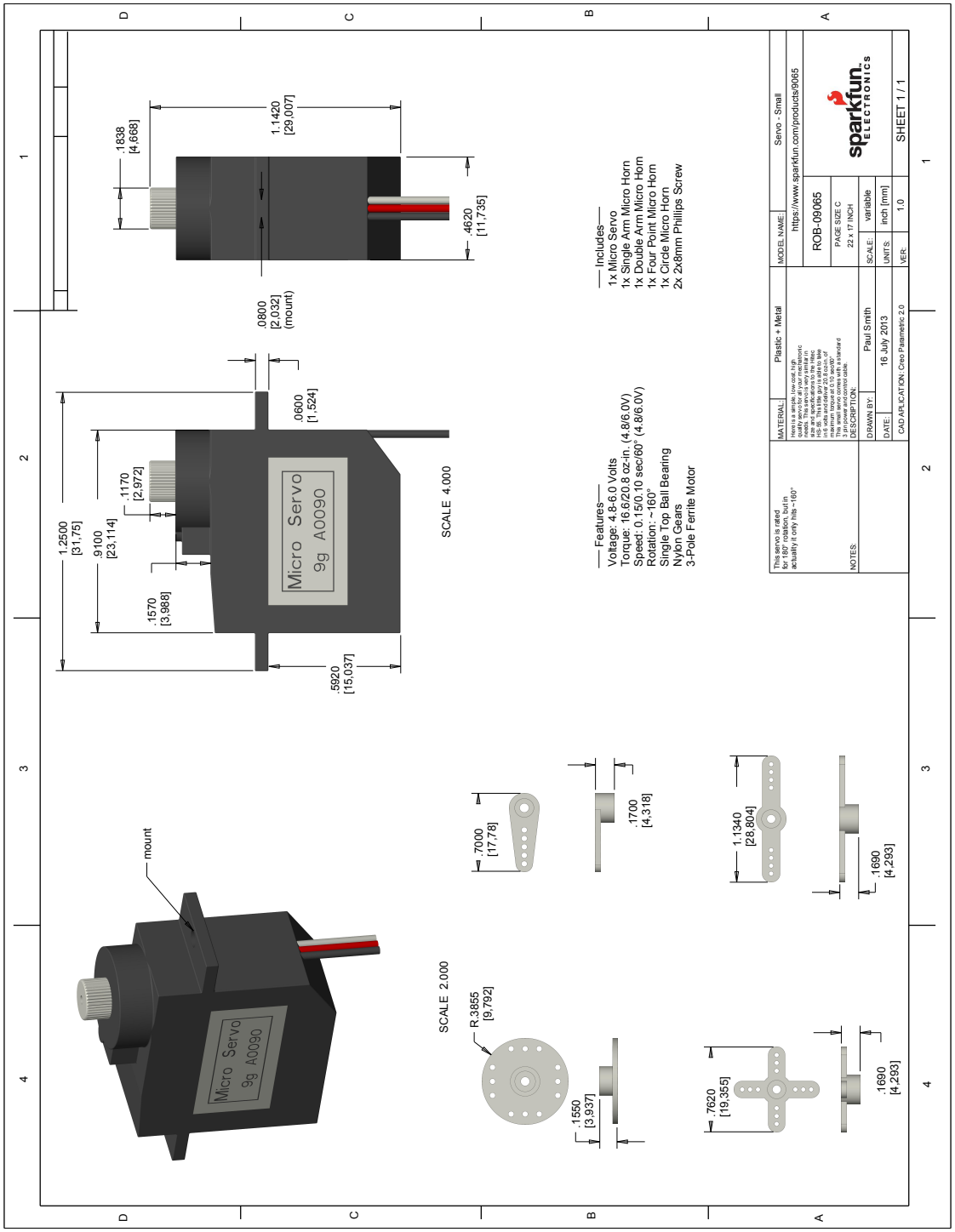
Propeller P8X32A Programming Example

Servo control with the Propeller chip is simplified by using a cog's counter modules. The code below causes a servo connected to I/O pin P0 to turn to and hold the 90° position. For more information about counter modules and PWM with the Propeller, see Chapter 7 in the Propeller Education Kit Labs: Fundamentals text, which is included as a PDF in the Propeller Tool IDE Help.

```
{{ CenterParallaxServo.spin
For centering Parallax Continuous Rotation Servo
or holding Parallax Standard Servo at 90° position.
Sends a 1.5 ms pulse approx every 20 ms }}

CON
_clkmode = xtal1 + pll16x                ' System clock → 80 MHz
_xinfreq = 5_000_000                     ' Using 5 MHz external crystal oscillator
servoPin = 0                             ' Servo signal to this I/O pin-change if needed

PUB CenterServo | tInc, tc, tHa, t

ctra[30..26] := %00100                   ' Configure Counter A to NCO
ctra[8..0]   := servoPin

frqa := 1
dira[servoPin]~~

' Set up cycle and high times
tInc := clkfreq/1_000_000
tC   := tInc * 21_500
tHa  := tInc * 1500
t    := cnt                              ' Mark counter time

repeat                                   ' Repeat PWM signal
  phsa := -tHa                           ' Set up the pulse
  t += tC                                ' Calculate next cycle repeat
  waitcnt(t)                             ' Wait for next cycle
```

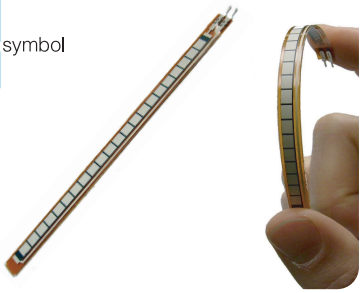
Revision History
Version 2.1: corrected specifications for torque. Updated PULSOUT range table. Added Using a Separate Power Supply on a HomeWork Board section beginning on page 2.

A.7 Micro Servo



A.8 Flex Sensor

spectrasymbol



FLEX SENSOR FS

Features

- Angle Displacement Measurement
- Bends and Flexes physically with motion device
- Possible Uses
  - Robotics
  - Gaming (Virtual Motion)
  - Medical Devices
  - Computer Peripherals
  - Musical Instruments
  - Physical Therapy
- Simple Construction
- Low Profile

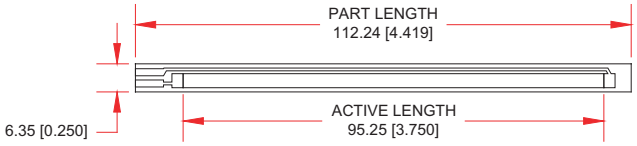
Mechanical Specifications

-Life Cycle: >1 million
-Height: ≤0.43mm (0.017")
-Temperature Range: -35°C to +80°C

Electrical Specifications

-Flat Resistance: 10K Ohms
-Resistance Tolerance: ±30%
-Bend Resistance Range: 60K to 110K Ohms
-Power Rating : 0.50 Watts continuous. 1 Watt Peak

Dimensional Diagram - Stock Flex Sensor



How to Order - Stock Flex Sensor

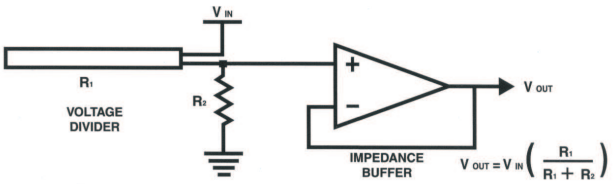
FS	—	L	—	0095	—	103	—	ST
Series		Model		Active Length		Resistance		Connectors
FS = Flex Sensor		L = Linear		0095 = 95.25mm		103 = 10 KOhms		ST = Solder Tab

How It Works

A.9 Flex Sensor

Schematics

BASIC FLEX SENSOR CIRCUIT:

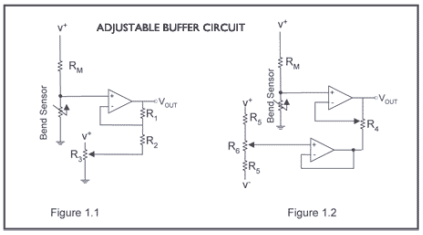


Following are notes from the ITP Flex Sensor Workshop

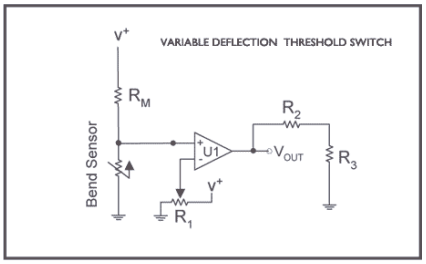
"The impedance buffer in the [Basic Flex Sensor Circuit] (above) is a single sided operational amplifier, used with these sensors because the low bias current of the op amp reduces errer due to source impedance of the flex sensor as voltage divider. Suggested op amps are the LM358 or LM324."

"You can also test your flex sensor using the simplest circut, and skip the op amp."

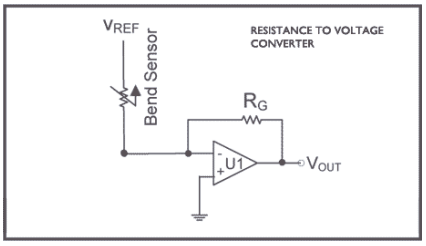
"Adjustable Buffer - a potentiometer can be added to the circuit to adjust the sensitivity range."



"Variable Deflection Threshold Switch - an op amp is used and outputs either high or low depending on the voltage of the inverting input. In this way you can use the flex sensor as a switch without going through a microcontroller."



"Resistance to Voltage Converter - use the sensor as the input of a resistance to voltage converter using a dual sided supply op-amp. A negative reference voltage will give a positive output. Should be used in situations when you want output at a low degree of bending."



TRITA ITM-EX 2019:26