DEVELOPMENT OF BIPED ROBOT
(SENSOR AND ACTUATOR CONTROL)

YEUN TEONG JIM

A project report submitted in partial fulfilment of the requirements for the award of Bachelor of Engineering (Hons.) Mechatronics Engineering

Faculty of Engineering and Science
Universiti Tunku Abdul Rahman

April 2011

DECLARATION

I hereby declare that this project report is based on my original work except for citations and quotations which have been duly acknowledged. I also declare that it has not been previously and concurrently submitted for any other degree or award at UTAR or other institutions.

Signature : ____________________________

Name : ____________________________

ID No. : ____________________________

Date : ____________________________

APPROVAL FOR SUBMISSION

I certify that this project report entitled **"DEVELOPMENT OF BIPED ROBOT (SENSOR AND ACTUATOR CONTROL)"** was prepared by Yeun Teong Jim has met the required standard for submission in partial fulfilment of the requirements for the award of Bachelor of Engineering (Hons.) Mechatronics Engineering at Universiti Tunku Abdul Rahman.

Approved by,

Signature : _________________________

Supervisor : Mr Chong Yu Zheng

Date : _________________________

Specially dedicated to
my beloved family,

ACKNOWLEDGEMENTS

I would like to thank everyone who had contributed to the successful completion of this project. I would like to express my gratitude to my research supervisor, Mr Chong Yu Zheng for his invaluable advice, guidance and his enormous patience throughout the development of the research. I’m also very grateful to my teammates which provide me with full support and cooperation in completing this project namely, Chin Kon Sin, Low Wai Loong and The Wey Yew.

In addition, I would also like to express my gratitude to my loving parent and friends who had helped and given me encouragement, especially my mother who has always take care of my health and my sister who has always look after me.

DEVELOPMENT OF BIPED ROBOT
(SENSOR AND ACTUATOR CONTROL)

ABSTRACT

With the new development of pneumatic air muscles, many robotic applications which are usually actuated by electric motor can now also be actuated through pneumatic system which is controlled by solenoid valves. Many researchers have research methods which are inexpensive and efficient for controlling the pneumatic air muscles. One of the methods is controlling the air muscles using fast switching valves which are controlled by PWM signals. In this report, a biped robot which is actuated using pneumatic air muscle would be developed. Researches which are related to biped robot are examined and discussed. The focus of this report is to select possible sensors that can be implemented onto the biped robot, and also to develop suitable actuating methods to control the actuators. Firstly, numerous sensors that are possibly required by a biped robot are discussed. Secondly, biped robot actuating methods done by other researchers are examined. Lastly, suitable sensors, valves, actuator setups and also actuator controlling methods are developed. Based on the result, it is concluded that it is possible to use pneumatic actuating system to control the movement of the biped robot.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS / ABBREVIATIONS

PID	proportional integral derivative
PWM	Pulse width modulation
ms	millisecond
mm	milimeter
V	Voltage
DCV	Directional Control Valve
rpm	Revolution per Minute
SSOP	Small Pb-free package
ADC	Analog to Digital Converter
SPI	Serial Peripheral Interface Bus
IC	Integrated circuit
I/O	Inputs and outputs
DIP	Dual in-line package
RC	Resistor–capacitor
F	Farad
Hz	Hertz
LED	Light-emitting Diode
PC	Personal Computer
UART	Universal Asynchronous Receiver Transmitter
Kp	Proportional gain

CHAPTER 1

INTRODUCTION

1.1 Background

Scientists and engineers have developed many different robots to aid and relieve the work of humans in the community. These include robots that aid in the manufacturing process, transportations, explorations, and also robots that help in the medical field.(Chevallereau, Bessonnet, Abba, & Aoustin, 2009) Locomotion of a robot describes how a robot moves through its environment. There are various methods for a robot to achieve movement, for example, robots could move on the ground through the use of wheels, tracks and even legs.

Wheels are by far the most popular robotic locomotion. This is because wheels are easily controlled and implemented through the use of electrical motors and stability of the robot is easily achieved. The control algorithm for an electrical motor is also well developed and precision control of an electrical motor is possible. The only drawback of the wheels is that it is not well adapted to uneven terrain and areas with low friction. This problem can be overcome by designing tracked robot, as it has the ability to mow through all sorts of obstacles on an uneven terrain. Since tracks have large contract area with the ground, it will increase traction and also has the ability to distribute the weight of the robot over a larger area of the ground. Therefore the pressure created between the tracked robot and the ground is lesser compared to wheel robots which enable it to move even on soft grounds like mud or snow. Despite the advantages, tracked robot also has its own limitations. Compared to wheeled robot, tracked robot is tends to have lower top speed, and the mechanical structure will be more complex. Due to the larger contact area with the ground, friction between the grounds will be high. As a result, steering a tracked robot would be more difficult and would consume more power while turning.

There are many different types of legged robots, for example a biped, quadruped and a hexapod which used 2, 4 and 6 legs respectively. With more legs, it is easier to achieve stability. Despite that, the robot with more legs are generally larger in size and required more space to move around, therefore it might be well suited for outdoor activities compared to wheeled or tracks locomotion, but for indoor activities, a biped robot would be more suitable compared to multi legged robots. This is because the size of a biped robot is similar to a human which would be smaller and lighter compared to multi legged robot, and it also uses two legs to achieve movements which closely resembles how human walk. Therefore, biped robots will adapt to the environment that is usually designed for humans better, for example inside houses or factories. They can also ascend or descend stairs easily compared to other locomotion (Figliolini & Ceccarelli, 1999). To create a robot to service and help humans, being able to move freely in the environment where humans live is one of the most important requirements.

Sensors are essential components to any robot. It is the only way the robot can collect information about the internal state as well as external environment of the robot. Information collected by the sensors would be directed to a control unit to determine the current state of the robot.

Once the designed and control of a biped robot is well developed, the biped robot can be further integrated with a robotic upper extremities such as robotic arms and head to form a humanoid robot which can probably access to about anywhere that is accessible to humans. With that, the robot can then be applied as a service robot to help with daily tasks, housework, or even servicing work at hospitals. Other than that, the robot can also be used to work in places which are hazardous to humans such as firefighting, a radioactive zone, and landmine fields and so on. There are also some who use a humanoid robot as a surveillance robot.(Chevallereau, Bessonnet, Abba, & Aoustin, 2009)

1.2 Aims and Objectives

As this project consists of 4 members, task for creating the biped robot will be distributed. The main group objective is to create a pneumatic actuated biped robot that can walk on even terrain, squat and stand up without falling.

The objectives and aims of this project will be as listed below:

1. To select appropriate sensors that can produce feedbacks which is required for the control algorithm of a pneumatic powered biped robot for movement control.

2. To develop control methods to control the pneumatic actuators

3. To develop controller boards that is able to gather sensor data and control the pneumatic actuators.

CHAPTER 2

LITERATURE REVIEW

2.1 General Overview

Many different kinds of biped robots have been developed by engineers and scientists. All of the biped robots being developed are aimed to achieve a locomotion which closely resembles the human locomotion. Despite having the same aim, the components used to build up a biped robot are all different. For example, the biped robot could be powered by different actuators such as an electrical motors or pneumatics cylinders, different sensors located at different areas could be implemented, and lastly, the control method and algorithm used to balance and direct the biped robot could also be different. Table 2.1 gives a summary of the different actuators, sensors, and control methods being implemented on various bipedal prototypes being developed.

Based on Table 2.1, the biped robots are generally separated into 3 parts, actuators used, sensors used and control methods implemented. Further discussions on the sensors and actuators used would be stated in the following sections. Control methods being implemented will not be emphasised as it is not the main focus of this report.

Table 2.1: List of Biped Robot Prototypes

Source	Name of Biped prototype	Actuators used	Sensors used	Control method
(Figliolini & Ceccarelli, 1999)	EP-WAR	Pneumatic linear cylinders controlled with five way/two-position valve, pneumatic rotary cylinder, and suction cups below robotic foot	Reed switches for linear cylinder, and electric switches for rotary cylinder.	Controlled with PLC in On/Off environment
(Azevedo, Andreff, & Arias, 2004)	BIP	Brushless DC motors	Synchro-resolvers, potentiometers at joints, limit switches as joint limit and three force sensor at each foot.	Statically stable waking.
(Takuma, Hosada, & Asada, 2005)	Que-Kaku	Antagonistic pairs of pneumatic actuators (McKibben artificial muscles)	Potentiometers for joint angle, and ON / OFF sensor on the foot	Focuses on walking cycle of biped which is controlled by PI controller.
(Verrelst, 2005)	Lucy	Antagonistic pairs of pneumatic actuators (Pleated Pneumatic Artificial Muscle)	HEDS-6540 Optical incremental encoder, pressure sensors	Joint Trajectory Tracking Controller
(Wisse & Richard, 2007)	Baps	Agonist–antagonist couple using McKibben Muscles controlled with 3 way valve.	Gyroscope at hip joint	Passive walking aided with actuators located at hip joint
(Manoonpong, Geng, Kulvicius, Porr, & Wo¨rgo¨tter, 2007)	RunBot	RC servo motor	Built in potentiometer of RC servo motor for joint angle, switch sensor to detect ground contact, an accelerometer and IR sensor for detecting modelled ramp for experiments.	Mechanical stopper at knee joint. Controlled with neural network.
(Hosoda, Takuma, Nakomato, & Hayashi, 2008)	Two-Dimensional Biped Robot	Agonist–antagonist couple using McKibben Muscles controlled with 5 / 3 way valve.	Touch sensor below foot	Controlled in a feed forward manner according to a fixed sequence of valve operation
(Corpuz, Lafoteza, Broas, & Ramos, 2009)	YICAL Leg 2 Biped	Geared DC motor	Potentiometers at motor joint. Gyroscope and accelerometer at hip part with Kalman filer	Uses closed-loop system to achieve static balancing.

2.2 Sensors

Based on Table 2.1 various sensors are used to create the biped robots. Sensors are used to provide feedback to the robot about the state of the robot and also the environment. In robotics, sensors can be classified into proprioceptive or exteroceptive. Proprioceptive sensor are sensors that measures properties that are internal to the robot, for example, the angle of the robotic joint, speed of the actuator, or battery voltage. Exteroceptive are sensors that acquire information that are external to the robot, for example, distance from objects, light intensity, or sounds (Siegwart & Nourbakhsh, 2004).

When selecting a suitable sensor to be used, there are a few criteria to look into which can determine the sensors performance. A brief explanation of the criteria would be listed below: (Bolton, 2003)

- Range and span – Range of the sensor defines the limits between which the input can vary. Span on the other hand is the maximum input value minus the minimum input value of the sensor.

- Error – The difference between the measured value of a sensor and the actual value being measured is known as error.

- Sensitivity – The sensor's sensitivity is defined as the change in output per input.

- Resolution – Resolution of a sensor is the smallest increment of input that can be detected by the sensor.

- Repeatability – Repeatability of a sensor is the sensor's ability to reproduce identical output for the same input. A sensor with high repeatability is said to be precise.

- Accuracy – The sensors accuracy is inversely proportional to the error. It is the extent to which the value measured by the sensor might be wrong. A sensor with high accuracy would produce less error.

- Deadband – Deadband of a sensor is the range of the input for which it produces no output.

After the sensors are selected, there are also some issues to look into. Often the signal produced by the sensors requires post processing before it can be used by the controller. For example, the signals might need to be amplified, filtered, demodulated, or isolated. Analog-to-digital converter might also be needed to convert the signals to digital signals that can be analysed by a digital controller. This is known as signal conditioning. Besides that, calibration of the sensors might also be required to maintain the accuracy of the sensors used. (Bishop, 2006)

To construct a biped robot, there are a few types of sensors used. From Table 2.1, the various categories of sensor can be categorised into joint sensors and tactile sensors. A brief introduction of the sensors would be described in the following section.

2.2.1 Joint Sensor

Joint sensor gives feedback of the robots joint angle to the controller. This information is important to determine the orientation of the robots leg and the whole structure of the robot. The feedback signals are also important for the controller determine the signals needed to actuate the actuators. Joint sensors are considered as proprioceptive sensors.

From Table 2.1, potentiometers are the most commonly used sensor to detect the joint angle of the biped robot. Potentiometer operates by using the concept of voltage divider. Based on Figure 2.1, terminal 2 would be controlled by a mechanically coupled wiper that can be moved externally, in this case, the wiper would be moved through the rotation of the joint of the robot. This will change the position of the wiper on across the resistance element and produce a potential difference. By measuring the potential difference, the joints orientation can be determined. The output voltage is given by the equation below: (Everett, 1995)

$$V_o = V_{ref} \frac{r}{R} \tag{2.1}$$

Where:

V_o = output voltage from wiper

V_{ref} = reference voltage across potentiometer

r = wiper-to-ground resistance

R = total potentiometer resistance

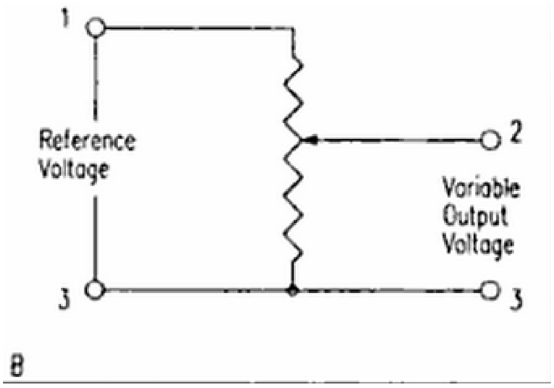


Figure 2.1: The circuit of a potentiometer

(Everett, 1995)

Optical encoder can also be used to sense the angular position of the robot joint. Optical encoder consists of a photodetector and a phototransmitter. The light generated by the phototransmitter would be directly aimed at the photodetector. The beam of light would then be periodically interrupted by a coded transparent patter on a rotating intermediate disk attached on the rotating shaft / joint. This would generate a digital output that can be used to calculate the position of the shaft / joint. There are two types if optical encoder, incremental and absolute. Incremental version measures instantaneous angular position of a shaft relative to a datum point, but are unable to indicate the absolute position of the shaft. Absolute version on the other hand is able to measure the shaft position at any time.

2.2.2 Tactile Sensor

Tactile sensors are categorized as exteroceptive sensors. Tactile sensors are typically used for collision detection. The tactile sensors used for a biped robot is normally place below the foot of the robot to detect the collision of the foot with the ground. It can be use to determine the walking phase of the biped robot. In some journals, the activation timing of the robots actuators are solely dependent on the signals from the tactile sensors(Takuma, Hosada, & Asada, 2005)(Hosoda, Takuma, Nakomato, & Hayashi, 2008). The tactile sensor used in the biped prototype Que-Kaku is a single pole, single throw limit switch as shown in Figure 2.2.

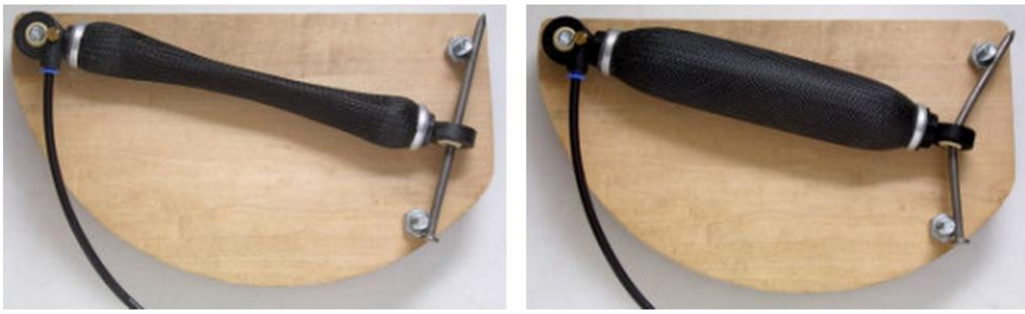
Figure 2.2: SPST Switch Used for Biped Prototype Que-Kaku
(Takuma, Hosada, & Asada, 2005)

2.3 Actuators (McKibben Air Muscle)

From Table 2.1, there a two main actuators that are being used, electrical motor, and pneumatic actuators. Actuators using electrical motors are preferred, there are also many successful humanoids developed using electrical motors such as Honda humanoid robot ASIMO (Sakagami, Watanabe, Aoyama, Matsunaga, Higaki, & Fujimura, Oct 2002) the Sony humanoid QRIO(Nagasaka, Kuroki, Suzuki, Itoh, & Yamaguchi, 2004). Electrical motors are widely used because the characteristic and control of an electrical motor are well-known and high precision control of these actuators can be achieved. Despite that, electrical motors also have its limitations. Electrical motors have to run in a nominal speed with low torque, therefore in order to achieve a speed and torque that is suitable to be used at the joints of the biped robot, a gearing mechanism is required. This would increase the weight and complexity or the biped robot joints design and induce high reflected inertia. (Verrelst, 2005). Other than that, the joints that are driven by electrical motor do not have back-drivability again an external force or torque due to the gearing mechanism used. (Hosoda, Takuma, Nakomato, & Hayashi, 2008)

In this project, the biped robot that will be developed will be using pneumatic actuators which are the McKibben type pneumatic actuator. The reason for choosing a pneumatic actuator over the electrical motor is the compliance characteristic and high force-to-weight ratio of the pneumatic actuators which allows it to be directly coupled to the joints. Compliance is due to the compressibility of air in the pneumatic actuators, which can be adjusted by controlling the pressure inside the actuator. This can provide a damping effect or stiffness to the system that is controlled using the actuators, unlike an electrical motor which is very rigid. (Daerden & Lefeber, 2000). The air muscle are also said to be similar to biological muscles because forces can only generated through contraction of the air muscles and in a force or position control mode, such actuator is highly nonlinear. Air muscles also have other advantages. For example it is safer to operate compared to electrical motor even when the actuators fail. Other than that, it provides little contamination to the environment as it is powered by air and can be cheaply built. (Repperger, Phillips, Neidhard-Doll, Reynolds, & Berlin, 2006)

To control the pneumatic actuator, the basic operation of the actuator must be understood. McKibben air muscle consists of an inner rubber tube wound by braided wires. It only has one inlet valve and contracts in the longitudinal direction on inflation and expands in the radial direction. This produce a force at both ends of the air muscles where it is connected. Figure 2.3 shows a McKibben air muscle produce by Shadow Robot Company.



(a) (b)

Figure 2.3: Shadow Air Muscle in (a) Relaxed and (b) Inflated Condition

(The Shadow Robot Company: Shadow Air Muscle, 30mm)

The degree of contraction depends on the pressure in the muscle and also the external load applied on the actuator. Other than that, the state of inflation of the muscle also affects the contraction ratio of the actuator. The volume inside the air muscle would change in a nonlinear manner even thou the pressure increase linearly in the air muscle. The contractile force that is generated at both ends of the air muscle is proportional to the net change of the cross-section surface area affected via the inflation as follows:

$$\Delta \text{ Force} = \text{Pressure X } \Delta \text{ Area} \qquad (2.2)$$

Where Pressure refers to gauge pressure (air pressure inside the bladder above the atmosphere or external environment) and Δ Area refers to the change in the cross section area of the air muscle during inflation. (Repperger, Phillips, Neidhard-Doll, Reynolds, & Berlin, 2006). Since the volume change in a nonlinear manner, the cross section area of the air muscle in the equation above would also change in a nonlinear manner.

Figure 2.4 shows the force-length relation of the McKibben air muscle tested with 3 different constant pressures level. The figure shows that the force-length relationship of the air muscle is approximately linear when the elongation is below 20% and becomes strongly non-linear after 20%. From the figure, it can be shown that the maximum elongation of the air muscle being tested is roughly 30% of its original length. Operating the muscle in the non-linear region is undesirable, but some biped researchers make use of this property and applied it as a joint angle limit for the biped (Wisse & Richard, 2007)

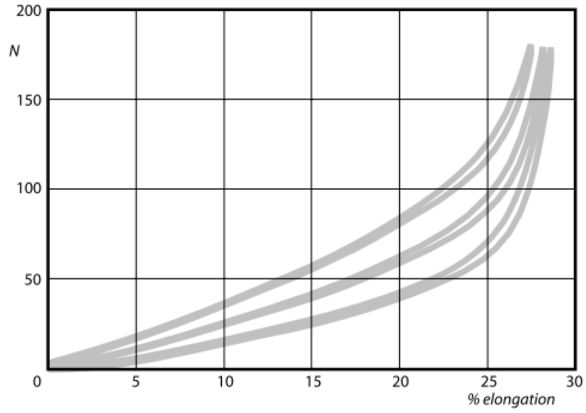
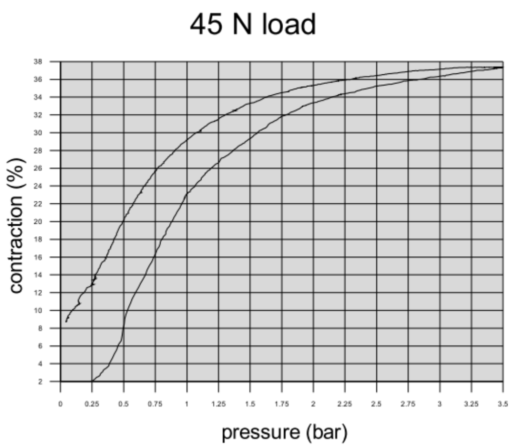


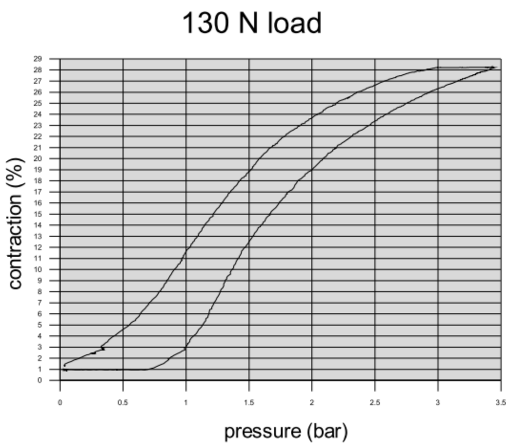
Figure 2.4: Measured muscle force-length relation at three different pressures.

(Wisse & Richard, 2007)

Shadow Robot Company is one of the suppliers for readymade McKibben type air muscles know as shadow air muscles. From the website, a technical specification sheet for a 30mm (diameter of air muscle when pressurize to 3 bar) shadow air muscle is provided (The Shadow Robot Company: Shadow Air Muscle, 30mm). Figure 2.5 shows the dynamic characteristics of the air muscle.



(a)



(b)

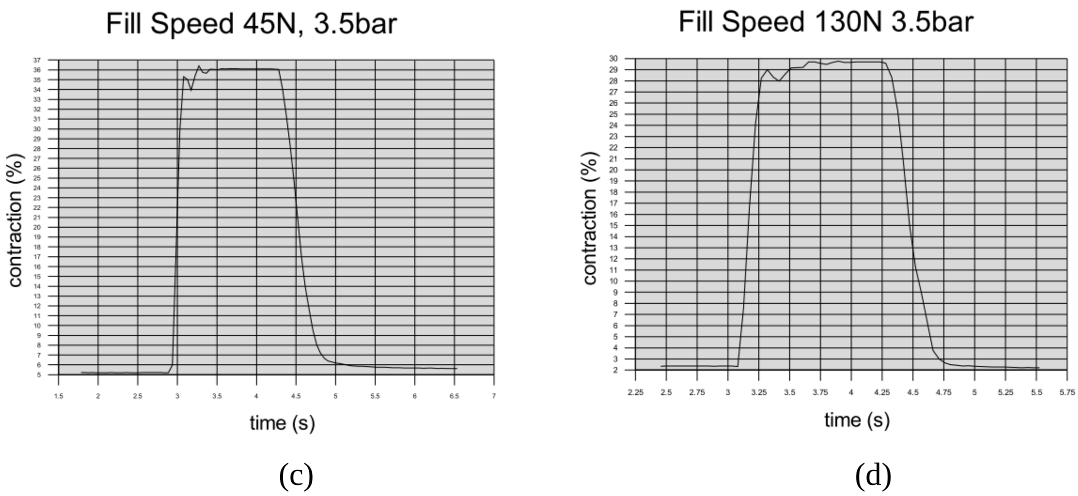


Figure 2.5: Dynamic Characteristic of 30mm Shadow Air Muscle

(The Shadow Robot Company: Shadow Air Muscle, 30mm)

From Figure 2.5 (a) and (b), the graphs show the contraction of the muscle as the pressure is increased to 3.5 bar (lower line), then decreased back to 0 bar (upper line), under several static loads. Figure 2.5 (c) and (d) shows the fill speed of the muscles. (The Shadow Robot Company: Shadow Air Muscle, 30mm) From these four figures, it is concluded that the McKibben muscles would experience hysteresis in the percentage of contraction when the pressure is increase and then decreased again regardless of the load applied. The percentage of contraction would also be nonlinear as pressure increase. It is also concluded that the external load applied on the air muscle would affect the fill speed of the muscle, more load takes longer time to fill the air muscles.

2.4 Agonist-antagonist Setup Using McKibben Air Muscle

One of the most common control setup of McKibben air muscle is the agonist–antagonist control. Refer to Figure 2.6. This setup is biologically inspired by the working principle of the muscles in living beings for example the arm muscles triceps and biceps. Usually the rotational motion setup in Figure 2.6 would be used to create the joint for the biped robot. In this setup, force can only be produced when one of the muscles is contracted (agonist) and the other being relaxed (antagonist), with this, a bidirectional motion can be created.

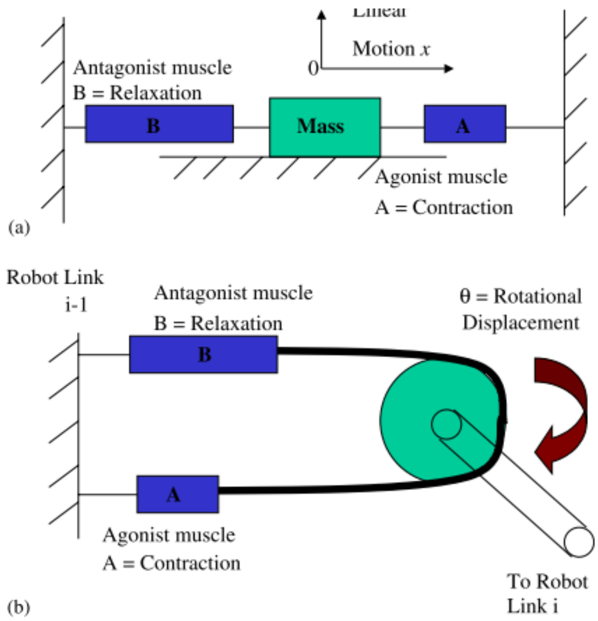


Figure 2.6: Agonist–antagonist control: (a) linear motion, (b) rotational motion

(Repperger, Phillips, Neidhard-Doll, Reynolds, & Berlin, 2006)

Agonist-antagonist setup is considered to be the key for realizing more than one locomotion mode (walking, jumping, and running) for a biped robot. So far most of the biped robot developed only focus on one locomotion mode at a time. This is because the compliance of the joints of the biped is different during walking and running. During jumping or running phase, compliance is needed to reduce impact and also for storing and releasing the impact energy. Therefore, compliance is naturally larger for running compared to walking robots. Air muscles connected in agonist-antagonist setup is able to change its compliance easily therefore to create a biped robot that is able to adapt to more than one locomotion mode is possible using this setup. (Hosoda, Takuma, Nakomato, & Hayashi, 2008). Compliance depends on the pressure inside the air muscles. Higher pressure would produce a less compliance or stiff joint and vice versa. A stiff joint in this setup means that the joint can hold its position and would be less influence by external disturbance forces.

A few biped robots using agonist–antagonist joints controlled with air muscle actuators would be review in the following sub-section.

2.4.1 Two-Dimensional Biped Robot

According to (Hosoda, Takuma, Nakomato, & Hayashi, 2008), the biped robot was not given a prototype name. Therefore the robot would be referred as two-dimensional biped robot throughout this report. The two-dimensional biped robot developed has a total of 4 legs to restrict its motion in the sagittal plane. It has a total of 14 McKibben air muscles, 4 for each ankle, 2 for each knee, and 2 for the hip (Figure 2.7). Each of these air muscles are controlled by a 5 /3 way solenoid valve with a closed centre position which is a compact on/off valve VQZ1000 produced by SMC Co., Ltd., with a maximum flow rate of 313.2 (l/min). Only two signals are need to control the valve, one signal is used to supply air to the air muscle and the other is used to expel air from the air muscle. When no signal is applied, there will be no in or outflow of air in the air muscle. The setup of the air muscles are shown in Figure 2.8.

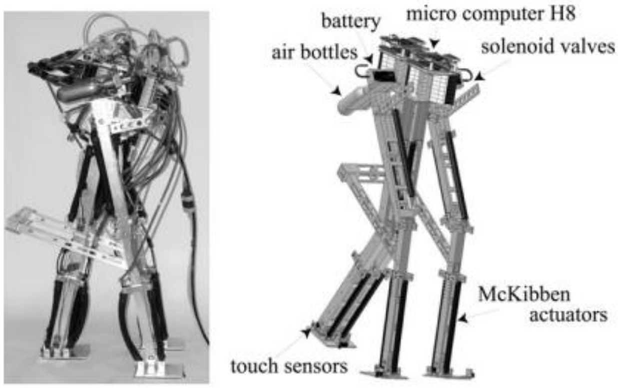


Figure 2.7: Two-dimensional Biped Robot

(Hosoda, Takuma, Nakomato, & Hayashi, 2008)

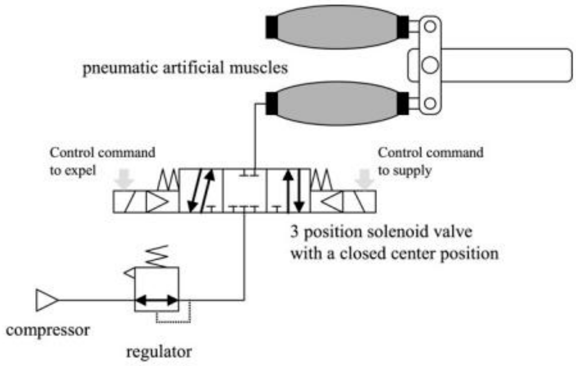


Figure 2.8: Air muscle connected to a 3-way solenoid valve
(Hosoda, Takuma, Nakomato, & Hayashi, 2008)

The main objective of creating the two-dimensional biped robot is to determine the contribution of joint compliance to multimodal dynamic locomotion (walking, jumping, and running). The actuators in two-dimensional biped robot are basically controlled in a feed forward manner according to a fixed sequence of valve operation. Every valve would operate only in on / off condition. PWM control of the solenoid is said to be able to modulate the pressure in the air muscles for achieving more precise control of the joint motion, but for the sake of simplicity, it would not be implemented in this journal. There would be a touch sensor below the foot of the robot to monitor the state of the robot. These touch sensors would also be used to trigger the activation of the air muscles. The activation of the muscles would follow a chart that is predefined for the purpose of walking as shown in Figure 2.9.

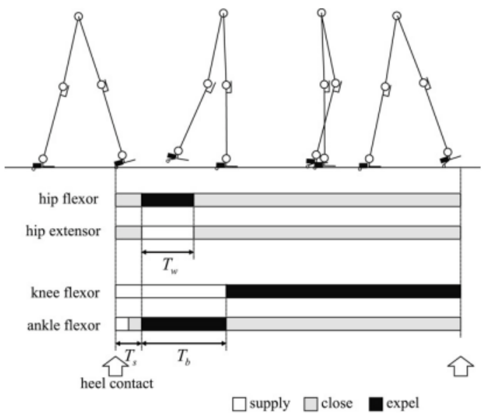


Figure 2.9: Proposed valve operation scheme for dynamic walking of Two-dimensional Biped Robot (Hosoda, Takuma, Nakomato, & Hayashi, 2008)

The effect of joint compliances on the walking cycle of the robot is investigated. The compliance of the ankles is changed by regulating the duration to supply air to both pneumatic actuators of each ankle joint. The longer the duration is, the less compliant the ankle joint becomes. The results recorded from the journal are shown in Figure 2.10. It shows that the compliance of the joint would affect the walking cycle of the robot. It can be concluded form the result that the walking is most efficient when the duration of the supply air to the ankle joint is around 300 (ms). Other than walking, jumping and running experiment was also conducted to test the effect of compliance joint on the locomotion mode. In the end of this journal, it is concluded that the compliance of the robot should be changed to suite different locomotion modes.

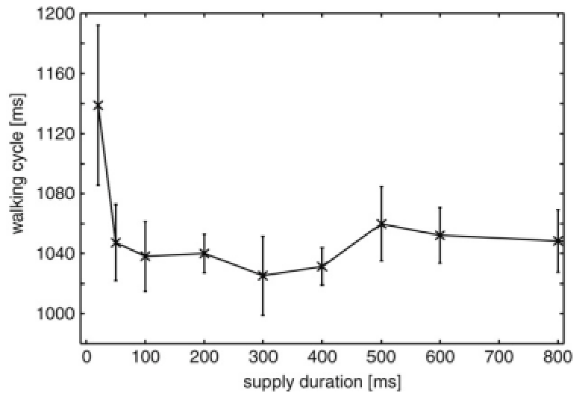


Figure 2.10: The relationship between the walking cycle and supply duration to muscles of the ankle (Hosoda, Takuma, Nakomato, & Hayashi, 2008)

2.4.2 Biped Robot: Baps

Based on (Wisse & Richard, 2007), it is believe that by using passive dynamic control combined with ballistic control actuation using McKibben muscles at the hip joint, an active dynamic walking robot with energy efficient walking that was comparable to that of human can be achieved.

Baps is modified based on the control theory of passive dynamic. There are a total of 6 McKibben muscles being used in biped robot Baps, 3 muscles per leg. Each leg has one muscle for leg elongation of a linear joint in the leg, and a pair of antagonistic muscles around the rotational hip joint. All the muscles will be operating at a nominal pressure level to provide nominal stiffness at the joints. The reason for using Mckibben muscles in Baps is because of its compliance. Due to this compliance the muscles is said to be particularly successful in application that do not require a high bandwidth or high position accuracy such as walking.

For biped robot Baps, self made McKibben muscles are used. They found out that the combination of polyester braiding and latex tubing resulted in the highest efficiency. A piston type pressure control unit was also designed and used to control and regulate the pressure of the muscles. The agonist–antagonist couple muscle would be controlled by a three-way valve. Once triggered, the pressure in one of the muscle would increase from the nominal pressure. When the activation time has elapsed, the pressure supplied would be reduced to the nominal pressure again. The other muscle would be kept at the nominal pressure. The triggering signal would be provided by a gyroscope attached at the hip joint.

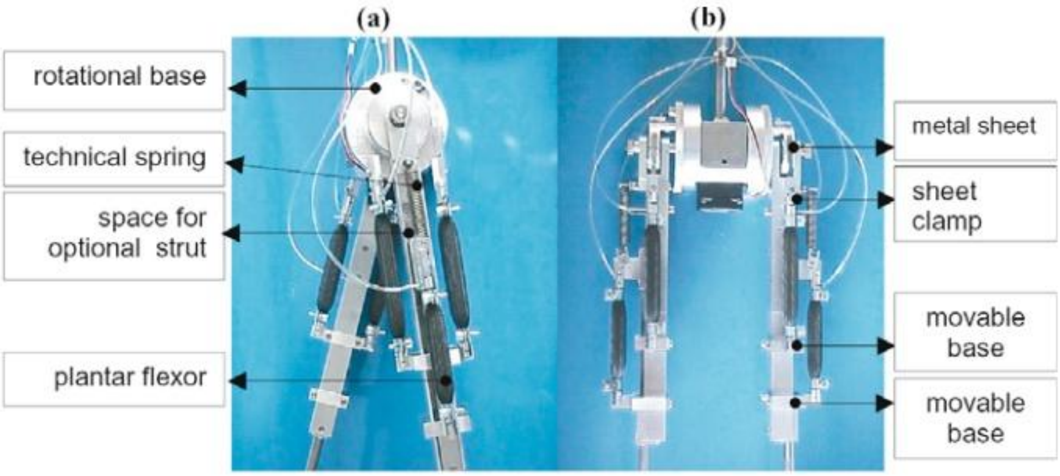


Figure 2.11: A sagittal view (a) and a frontal view (b) of the biped robot Baps

(Wisse & Richard, 2007)

2.5 Control Method for Industrial Pneumatic System

Pneumatic cylinders are very similar to pneumatic air muscles in a sense that they are both naturally compliance. The difference is that pneumatic air muscles have non-linear response, hysteresis and small stroke compared to pneumatic cylinders. Pneumatic cylinders on the other hand have internal friction forces between the piston and the cylinder which result in high stiction, and produce losses and makes small piston movements difficult to attain. It is also stated that a pneumatic air muscle would have an equilibrium length for each pair of pressure and load which is the absolute contrast to that of a pneumatic cylinder. This is because a pneumatic cylinder develops force which depends only on the pressure and the piston surface area. Therefore, a constant pressure will always produce a constant force regardless of the displacement. (Daerden & Lefeber, 2000)

Despite the problems faced by pneumatic cylinders, a position control of up to an accuracy of ±0.10 mm is still attainable. The applications of these high precision controls of pneumatic cylinders are mainly designed for industrial usage such as the robotic arm in the assembly line which requires high positioning accuracy. This section will investigate some of the methods used for position control of the pneumatic cylinders in hope that the methods used could also be applied to pneumatic air muscles.

There are 3 main valves that could be used to control a pneumatic cylinder, servo valve, proportional valve, and on/off solenoid valve. In the journal written by Varseveld and Bone (Varseveld & Bone, 1997), an on /off solenoid valve was used for position control. In the journal, Varseveld and Bone justified that on/off solenoid valve are better compared to the servo valve and proportional valve because solenoid valves are compact and cheaper compared to the other valves. By using a novel pulse width modulation (PWM) valve pulsing algorithm it is shown that the on/off solenoid valves can be used in place of the costly servo valve. Figure 2.12 shows the setup of the pneumatic cylinder being tested. In this setup, the valve used is a 3/2 way solenoid valve with a respond time of 5 ms. Manual flow controls were added before the cylinder inlets to filter out any disturbance caused by the pulsing of the solenoid valves. A linear potentiometer is used to provide position feedback.

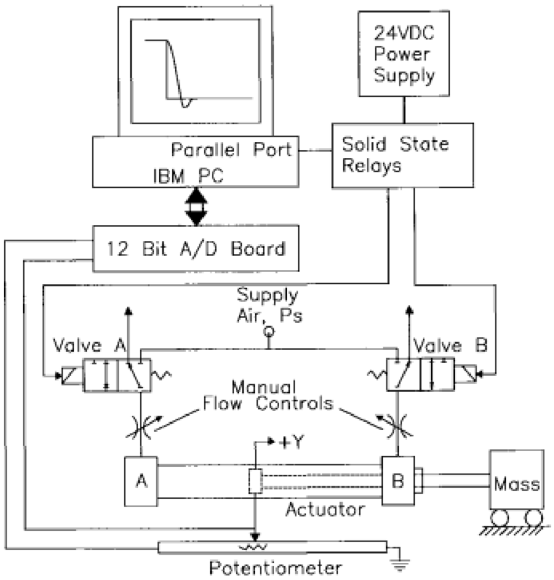


Figure 2.12: Pneumatic Cylinder setup for testing (Varseveld & Bone, 1997)

In this experiment, 4 different pulsing scheme of PWM was tested on the system. The results are show in Figure 2.13 and Figure 2.14. PWM period of 16 ms was used in all of the tests and each valves are controlled independently. Scheme 1 and 2 uses traditional linear PWM and scheme 3 and 4 uses novel PWM. From the results, a 35% deadband can be observed in the velocity profile of the cylinder in scheme 1. This is because in this range, the duty cycle produced was too low. Therefore the valves were not able to respond to the PWM signal as the minimum response time of the valve used is 5 ms. The novel PWM used in scheme 4 produced the best result and the velocity profile is quite linear during this scheme. In scheme 4, the duty cycle of the valves is not allowed to fall below the minimum possible duty cycle where the valve is able to respond. Once one of the valves is set at the minimum duty cycle, the duty cycle of the other valve would increase at twice the rate to maintain a linear output/input relationship at the velocity profile. In the end of the experiment, a PID controller with added friction compensation and position feedforward is successfully implemented using result from scheme 4.

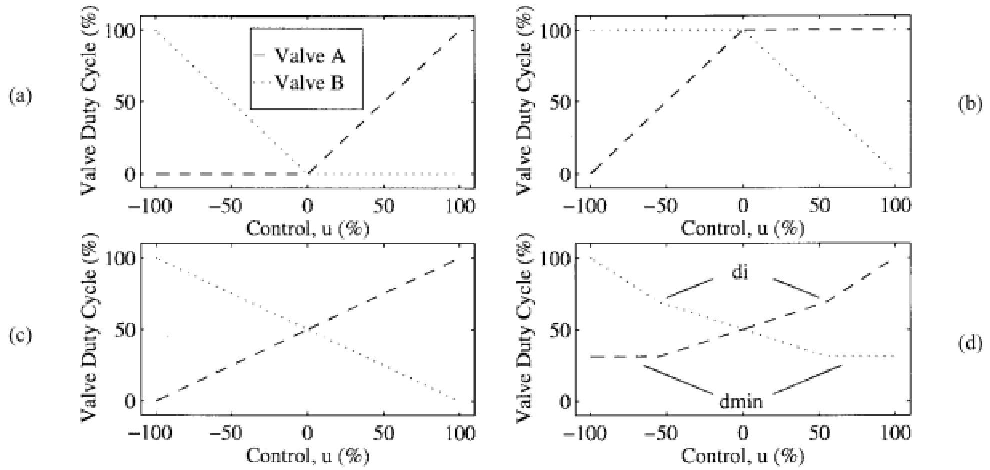


Figure 2.13: PWM valve pulsing schemes. (a) Scheme 1. (b) Scheme 2. (c) Scheme 3. (d) Scheme 4. (Varseveld & Bone, 1997)

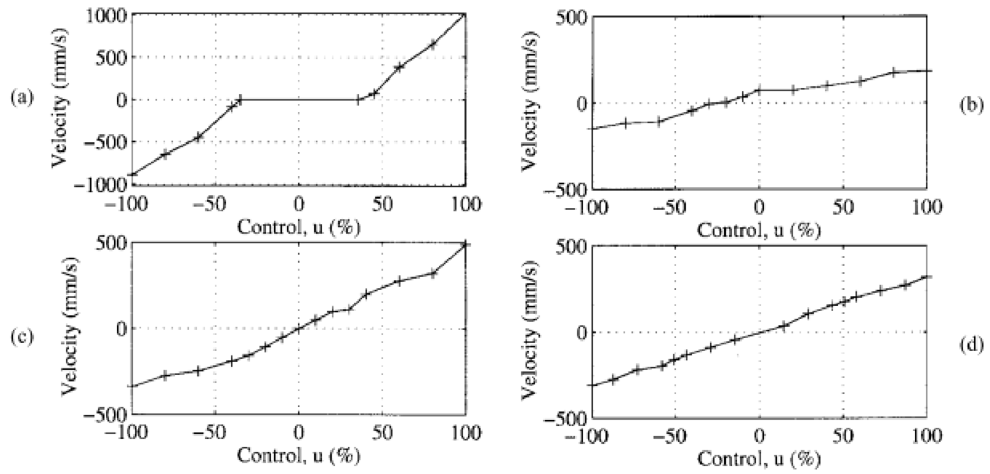


Figure 2.14: Measured actuator velocity versus controller output. (a) Scheme 1. (b) Scheme 2. (c) Scheme 3. (d) Scheme 4. (Varseveld & Bone, 1997)

CHAPTER 3

METHODOLOGY

3.1 General Overview

To achieve the objective stated in Chapter 1, research have to be made based on the sensors available that can be used to provide feedback to the system. Other than that, the characteristic of the pneumatic air muscles must also be understand, before a proper design of the pneumatic air muscles and controls can be provided. In this chapter, there are two main parts which describes topics which are related to the sensors and the actuator. In the sensor part, a comparison of a few possible types of sensor to be used in this project is made, and the characteristic and the implementation of the sensors being selected would be explained as well as sensor data acquisition methods. In the actuator part possible setup for the actuator, valve and control methods for controlling the actuators would be presented. Before the controlling method can precede, the sensor data acquisition system has to be finished, because designing the control methods are based largely on the reaction of the actuator used, to know the reaction of the actuator, the sensor has to be used to gather information such as joint angle which is manipulated by the actuator. The actuator being selected would be a self fabricated McKibben type air muscle (fabrication process of the air muscle would not be discussed).

3.2 Sensors

3.2.1 Sensor Selection

To control and balance a biped robot, information regarding the robots orientation in the environment has to be known. The only way for the robot to gather this information is through the use of sensors. To create a biped robot, sensors such as joint sensors, tactile sensors, or attitude sensors might be needed to sense the overall balancing status of the robot. The need for these sensors would depend on the control algorithm that is implemented to control the biped robot. Of all the sensors, the basic sensors needed would be the joint sensors. The parameters required for selecting the joint sensor of a biped robot would be listed below.

- Power supply – DC voltage preferred.

- Motion type – One dimension rotary sensor.

- Measurement type – Absolute measurement would be preferred over incremental measurements. (Incremental measurements requires sensors to be reinitialized to its home position every time the system is restarted)

- Range – Less than 180º

- Accuracy – Sensors that can produce moderate accuracy would be sufficient. The accuracy requirement of the sensor needed for the biped robot would not be as critical as an industrial robot such as a pick and place robot. Despite that, the accuracy requirement is also influenced by the control method that is implemented to balance the biped robot. The linearity, repeatability and resolution of the sensors output would also affect the sensors accuracy.

- Resolution – Resolution is the smallest step input the sensor can measure. High resolution means the sensor is able to sense small angles differences. For this project the sensor resolution of 1 degree is more than sufficient.

- Output – Digital signals would be preferred as it is less prone to electrical noise and it can also be readily feed into the microcontroller without the use of an analog-to-digital converter.

- Size and Weight – Small and light weight sensors would be preferred. The weight of the sensor chosen should not be too heavy as is might affect the biped robots walking cycle.

- Cost – The price within the range of RM50 is preferred as the joint sensors are required for each rotational joint of the biped robot (around 6 rotating joints) and the available budget for this project is limited.

There are various sensors that can be used as joint sensors, the most commonly used joint sensor is the potentiometer, other than that, optical encoder and rotary hall effect sensors would also be used as the robots joint sensor. For the sensor selection, the potentiometer would be a basic potentiometer from any electrical shop, optical encoders are supplied by citron, and the Hall Effect sensors are AS5040 supplied by Austriamicrosystem. Based on the parameters for joint sensors discussed above, a few important parameters for joint sensors are tabulated in Table 3.1 for comparing the three proposed sensors. The comparison between the advantages and disadvantages of these sensors are also tabulated in Table 3.2.

Table 3.1: Selection Criteria for Sensors

Type of Sensors	Potentiometer	Optical Encoder	Hall Effect
Power Supply	DC	DC	DC
Range	~270 degree	360 degree	360 degree
Resolution	Based on ADC	22.5 degree	0.35 degree
Output	Analog	Digital	Digital
Size	Small	Small	Small
Cost	< RM5	RM35	$ 5.40 ~ RM16

Table 3.2: Advantages and Disadvantages of Joint Sensors

Joint Sensors	Advantages	Disadvantages
Potentiometer	• Ease of interface • Measures absolute position • Widely available • Cheap	• May impart frictional loading to the rotating joint • Subjected to wiper wear • Requires analog to digital converter • Electrical noise may be

		introduced into the analog output signal. • Prone to vibration disturbances
Optical Encoder	• Digital Output • Adjustable resolution (based on number of slit on plate) • Does not require mechanical contact with the rotating joint. • More flexible mounting position	• Affected by external light source. • Requires more input ports from the microcontroller to process the signals • Expensive
Rotary Hall Effect Sensor	• Digital Output • Has various choice of output signals, eg, PWM, SPI, absolute, incremental) • High resolution (10 bit) • Does not require mechanical contact with the rotating joint. • More flexible mounting position • Small package • Requires only 3 inputs as the sensors can be connected using “daisy chain” concept	• Harder to interface because needs programming • Breakout board for SSOP package is widely available for sale • Has to be shipped from overseas • Require suitable magnets

Based on the two tables above, the rotary Hall Effect Sensor AS5040 proves to be more superior to the other sensors. Potentiometers are cheap and readily usable without any extra programming, 270 degree resolution is more than enough for our application, but since its signals are in analog, an ADC converter is required. The reason for not choosing the potentiometer is that it requires mechanical contact with the joint itself. Since the joint is constantly moving, the sensor might be prone to wear and tear.

Optical Encoder on the other hand are expensive, and the resolution is too low, other than that, is it also prone to external noise such as light exposure to the sensor, therefore, proper concealment of the sensor around the joint is required if this sensor were to be used.

Hall Effect sensor has the most advantages among the three sensors. The main reason for choosing this sensor is because of its ability to transfer data serially. With this method, the sensor also has a special mode called the “Daisy Chain Mode” where multiple sensors can be linked together. With this mode, only 3 inputs from the main controller are required to analyze the data which is send through the SPI Bus of the main controller. This proves to be useful because, for this project, the robot has a total of 6 joints. Therefore a total of 6 sensors are required. If each sensor require 1 input, then at least 6 inputs from the microcontroller is required. But with the Daisy Chain Mode, the inputs required are reduced to 3. With reduced inputs, the microcontroller can use its remaining I/O for other applications such as controlling the actuator. The disadvantages are that it requires more programming to implement. Other than that, the sensor also comes in small SSOP IC package. Therefore additional breakout board is required to solder the IC before it can be used. Suitable magnets are also hard to find, but since Austriamicrosystems also supplies the magnets, this is not an issue.

In conclusion, the Hall Effect Sensors AS5040 provide by Austriamicrosystems are used. A total of 8 free samples along with magnets are requested from Austriamicrosystems therefore all the sensors used for the robot are free of charge. Despite the sensors being free, the breakout board for the sensor which converts the SSOP package to DIP package have to be sourced from Singapore. Each board cost SGD 4.95 which is around RM12 each. The reason for converting to DIP is because DIP can be directly plucked onto a breadboard for testing purpose.

3.2.1 Sensor Characteristic

The rotary Hall Effect sensor AS5040 used is a 10 bit 360° programmable magnetic rotary encoder which is provided by Austriamicrosystems. As the Hall Effect sensor runs on magnetic field, it does not require mechanical contact with the joint being measured. To measure the angle of the joint, only a simple two-pole magnet needs to be attached onto the centre of rotation of the joint. For the sensor, a Diametric Magnet NdFeB, Grade N35, D6x2.5mm was used. This magnet is also supplied by Austriamicrosystems. Below is an image of the magnet used.

Figure 3.1: Diametric Magnet NdFeB, Grade N35, D6x2.5mm
(Austriamicrosystems, 2011)

The sensor would be placed over the magnet to sense the rotation angle of the joint. From the datasheet of the sensor, it is stated that the AS5040 is a system-on-chip, which combines integrated Hall elements, analog front end and digital signal processing into a single device. The sensor can measure absolute and incremental angle of the joint with a resolution of 0.35° which is equal to 1024 positions per revolution. It also has the choice to output the joint angle data in PWM signal or as a serial bi stream of digital data, or even as a programmable incremental output (Quadrature A/B and Index output signal, Step / Direction and Index output signal, and 3-phase commutation for brushless DC motors).

The sensor also has an internal voltage regulator which allows it to operate at either 3.3 V or 5 V supplies. The zero / index position of the sensor are also programmable, therefore eliminating the need for mechanical alignment of the sensors. The sensor can measure rotational speeds up to 30,000 rpm with is more than enough for our application. There are also failure detection mode for magnet placement monitoring and loss of power supply build-in in the sensor. One of the important features is that AS5040 can connect multiple sensors together through serial read-out of the data using a mode called Daisy Chain mode, with this mode all the sensor can be linked together requiring only three I/O pins from the microcontroller the read the angle information of all the joints. Lastly the sensor comes in a 16 pin SSOP package which has a measurement of 5.3 mm X 6.2 mm making it small and lightweight which can be easily mounted onto the robot joint.

3.2.2 Sensor Implementation

When connecting multiple sensors together in Daisy Chain mode, all the sensors have to be connected into a string of sensor, this means the last sensor would send the sensor date to the sensor with is before it and the signals would propagate until the first sensor which is connected to the microcontroller. Since our robot has two legs, it is unpractical to connect all 6 sensors into one long Daisy Chain as the wires would be extremely long when connecting from one leg of the robot to another and then back to the top plane of the hip where the main board for the microcontroller would be placed. Besides that, connecting sensors together with long wired Daisy chain might introduce unexpected noise over the transmission line. To overcome this problem, a low pass RC filter is placed to filter out the noise signals. Using R= 100 ohm and C = 1 nF, a max frequency of 1 MHz can be transmitted over the whole chain. (Austriamicrosystems, 2011) Other than that, the Daisy Chains in this project are spitted into two lines, one for each leg which consists of 3 sensors each. Combined with a multiplexer “MN4019B” on the main board to switch between the two Daisy Chain lines, only a total of 5 I/O ports are required to read all the data from 6 sensors. The hardware configuration of Daisy Chain Mode is shown in the Figure 3.2 and Figure 3.3.

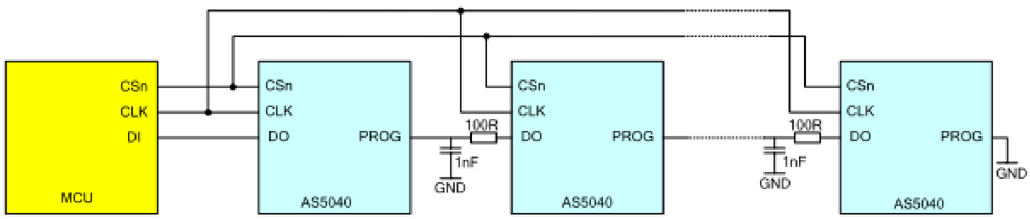


Figure 3.2: Daisy Chain Hardware Configuration

(Austriamicrosystems, 2011)

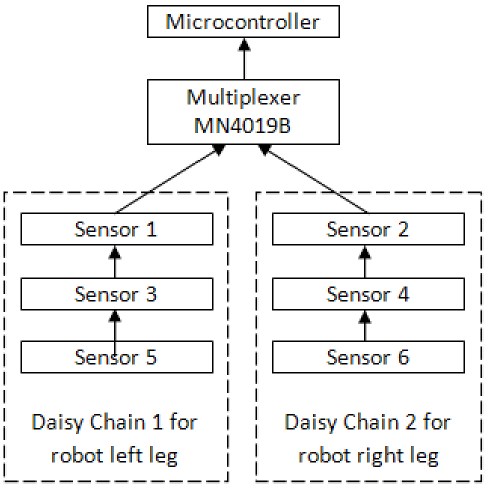


Figure 3.3: Daisy Chain Configuration (With Multiplexer)

Before the sensor can be used, it has to be soldered onto a breakout board. The breakout boards are supplied by Singapore Robotic. Below are figures showing the breakout board before and after the AS5040 has been soldered onto the board.

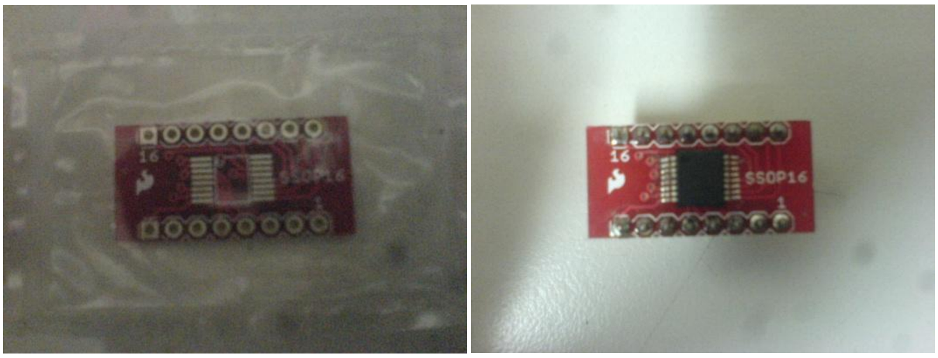


Figure 3.4: Breakout Board (Before and After Soldering)

After the IC has been soldered onto the breakout board, another circuit board has to be designed so that it can be mounted onto the robot joint fitting which is developed by the mechanical team. Figure 3.5 is the schematic and the actual board which is developed using strip board and Figure 3.6 shows the sensor being mounted onto the biped robot joint.

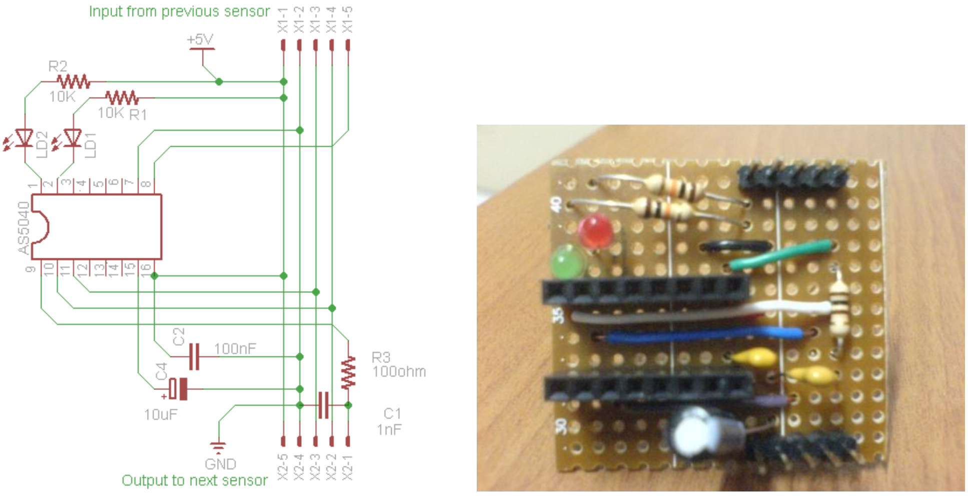


Figure 3.5: Sensor Board (Left: Schematic, Right: Fabricated Board)

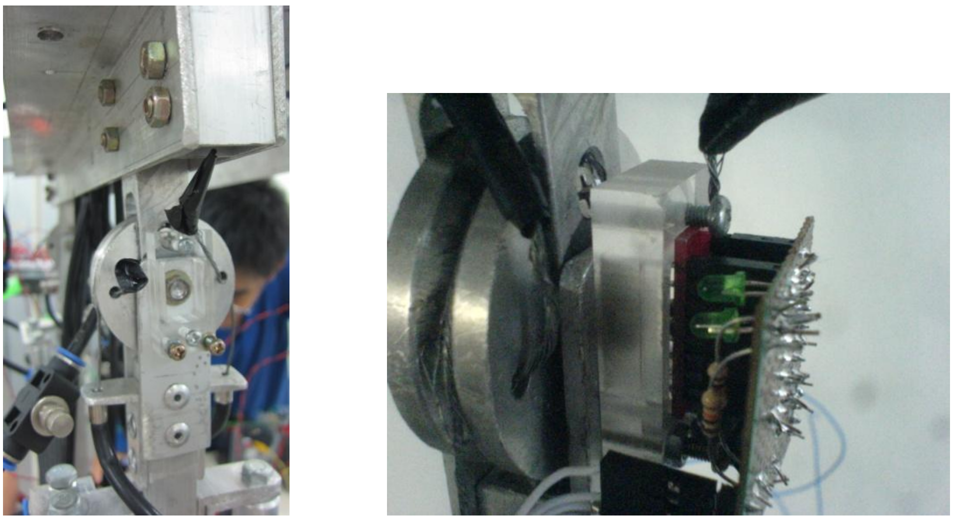


Figure 3.6: Sensor Mounting (Left: Before Mounting, Right: After Mounting)

3.2.3 Sensor Data Acquisition

The data which need to be received from the Hall Effect Sensor AS5040 is in 16 bit serial data form. Figure 3.7 shows the timing diagram of the sensor's serial output. The parameters in the diagram are shown in Figure 3.8

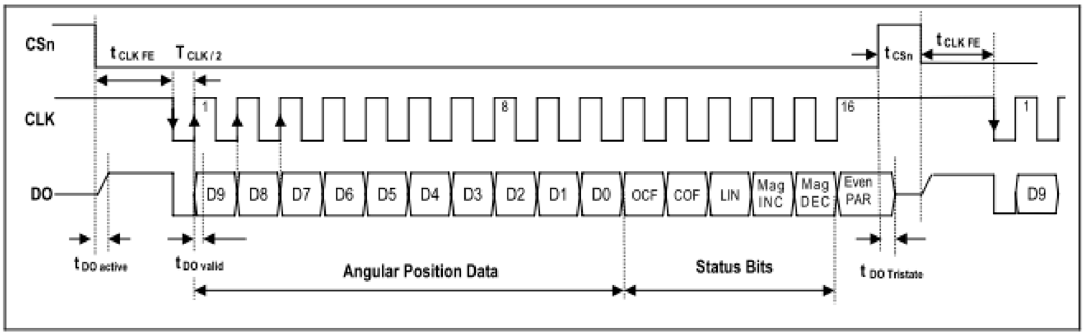


Figure 3.7: Timing Diagram of Sensor's Serial Output

(Austriamicrosystems, 2011)

Parameter	Symbol	Min	Typ	Max	Unit	Note
Data output activated (logic high)	$t_{DO\ active}$			100	ns	Time between falling edge of CSn and data output activated
First data shifted to output register	$t_{CLK\ FE}$	500			ns	Time between falling edge of CSn and first falling edge of CLK
Start of data output	$T_{CLK\ /\ 2}$	500			ns	Rising edge of CLK shifts out one bit at a time
Data output valid	$t_{DO\ valid}$	357		413	ns	Time between rising edge of CLK and data output valid
Data output tristate	$t_{DO\ tristate}$			100	ns	After the last bit DO changes back to "tristate"
Pulse width of CSn	t_{CSn}	500			ns	CSn = high; To initiate read-out of next angular position

Figure 3.8: Parameters for Timing Diagram (Austriamicrosystems, 2011)

The first 10 bits are absolute angle position data of sensor. The following 6 bits are status bits containing the sensor's system information about the validity of the angle data which are OCF, COF, LIN, Parity and Magnetic Field increase status and Magnetic Field decrease status. The sensor data is only valid when, OCF = 1, COF = 0, Lin = 0 and both Magnetic Field cannot be = 1. The Magnetic Field information can also acquired from pin 1 and 2 of the sensor. Therefore, the easiest way to determine whether the sensor data is valid is by inspecting the Magnetic Field status and making sure that both of them are not = 1.

When connected in Daisy Chain Mode, the timing diagram is slightly different. The numbers of bits required to read all the sensors connected in Daisy Chain is given in the formula:

$$n * (16+1) \text{ bits:} \tag{3.1}$$

where,

n = numbers of sensor connected in Daisy Chain Mode.

Therefore, the number of bits required increase by 1 for each sensor connected in the Daisy Chain. Figure 3.9 shows the timing diagram for Daisy Chain Mode.

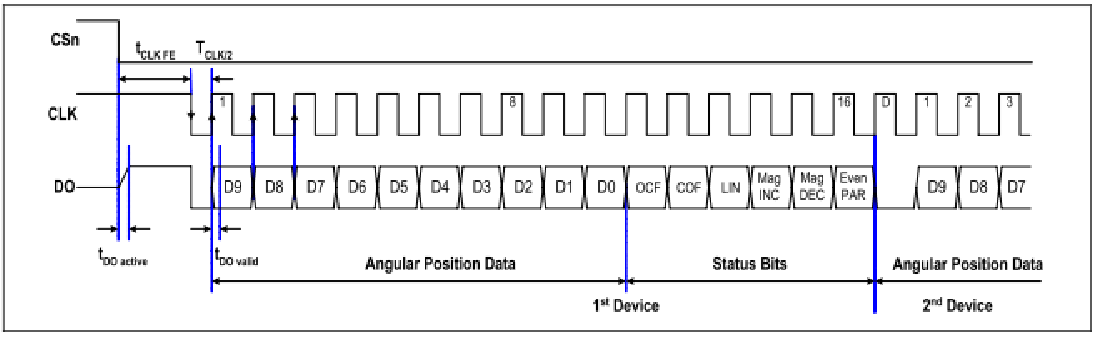


Figure 3.9: Timing Diagram for Daisy Chain Mode

(Austriamicrosystems, 2011)

The microcontroller used to receive this data is PIC18F4520. The coding would be attached in the appendix of this report. There are two method used for displaying the data. The first method is by displaying the data through LEDs connected to the microcontroller. This method is a fast and simple way of displaying the sensor data. The other method is by displaying the data to the PC through RS232 port. For displaying and transmitting the data, program such as HyperTerminal has to be used. In this project, Realterm is used to display the data on the computer. Realterm is a terminal program which is specially designed for capturing, controlling and debugging binary and other data streams. The reason for using Realterm is because it provides more options on displaying the data received rather than only displaying it through ASCII code. For example the data can be displayed in hexadecimal form, integer form and even binary form.

This second method is better than the first as the data are displayed on the PC which can be stored for further analysis. For this purpose Cytron's USB to UART converter UC00A was bought. This module can be directly plug and play into the USB port of the computer without any external power supply or circuitry. Conventional communication methods for microcontroller with computer are done through serial port DB9. However the serial port on laptop computers has already been phase out. With the USB port, the microcontroller can easily communicate with Laptop or Desktop computer. Below is a figure showing UC00A which is used.

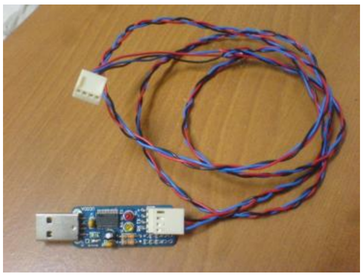


Figure 3.10: Cytron USB to UART Converter UC00A

3.3 Actuator

3.3.1 Actuator Setup Selection

There are a few possible setups for using the McKibben air muscle. A sketch of the possible setup for the air muscles at the knee joint would be shown in Figure 3.11.

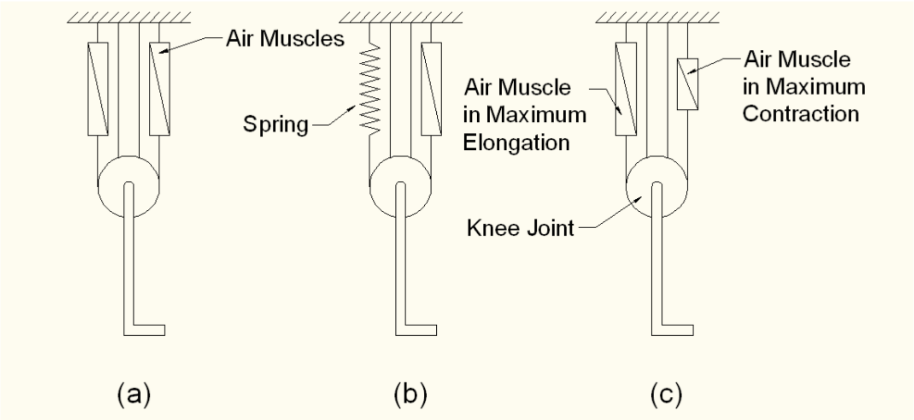


Figure 3.11: McKibben Air Muscle Setup for Knee Joint

The air muscle setup from Figure 3.11 (a) is the typical agonist-antagonist setup. With this setup, the knee joint of the robot would have 1 degree of freedom movement. Detailed descriptions of this setup are already outlined in section 2.4.

Figure 3.11 (b) is the modification of the agonist-antagonist setup. It replaces one of the air muscles with a spring. This will reduce the total numbers of actuator needed and will also simplify the control of the actuator. The spring used will act like an air muscle with constant air pressure being supplied. Therefore, when the air muscle in this configuration is in the relaxed state, the knee joint would be bended by the spring force. The knee joint would be straightened once a proper pressure is supplied to the air muscle.

In Figure 3.11 (c), the setup is exactly similar to Figure 3.11 (a). The only difference is the air muscles in Figure 3.11 (c) are used as knee joint limit to prevent hyperextension of the knee joint. From the Figure, the air muscle which controls the extension motion of the knee joint is in the state of maximum contraction when the knee is straightened. The air muscle controlling the flexion of the knee can also be setup in such a way that the maximum elongation of the muscle occurs when the knee joint is straightened. Either one of these air muscle setup will effectively limit the angle of the knee joint from further increasing. Based on (Wisse & Richard, 2007), other than acting as a knee joint limit, due to the non-linearity of the air muscle, the resistance of the muscle would increase once the muscle is close to its maximum elongation. This behaviour would add a damping effect on the knee joint and helps to slowdown the movement of the joint when it is near its limit.

Other than using the configuration shown in Figure 3.11 (c), the knee joint limit can also be implemented by mechanically or electronically. For example, the prototype RunBot uses a mechanical stopper at each knee joint to prevent hyperextension (Manoonpong, Geng, Kulvicius, Porr, & Wo¨rgo¨tter, 2007), and prototype BIP uses a limit switch to indicate joint limits so that appropriate control can be issued to the actuator.(Azevedo, Andreff, & Arias, 2004)

For the prototype of this report, a combination of air muscle setup in Figure 3.11 (a) and Figure 3.11 (c) are used. The prototype of this project has a total of 6 degree of freedom, the air muscle in Figure 3.11 (c) is suitable for the both the knee joints of the biped robot as most of the time the knee joints would be straightened and only needs to move in one direction. Configuration in Figure 3.11 (a) would be more suitable for the ankle and the hip joint of the biped robot as the joints needs to move back and forth constantly in two directions and it is not so useful in preventing the joints from hyperextension. Although configuration in Figure 3.11 (b) requires one air muscle less, is it not so suitable for our purpose because when the joints are coupled with springs, the compliance of the joint itself cannot be controlled as the spring constant is fixed.

3.3.2 Valve Setup Selection

The control method used for the McKibben air muscle is dependent on the type of valve being used. While selecting the type of valves to be used, there are a few criteria to look into, such as the air consumption of the valve, the flexibility in controlling the valve setup, number of control signals needed, cost, and weight of the valves and so on. Three types of valves and setups are being proposed to control the knee joint of the robot connected in agonist-antagonist setup as in section 3.12. The three valves are 5 / 3 close centre DCV, 3 /2 DCV, and 2 /2 DCV. Illustrations and explanations of the advantages and disadvantages of the three types of valve setup will be presented in the following paragraph and the type of valve setup being selected will be concluded at the end of this section.

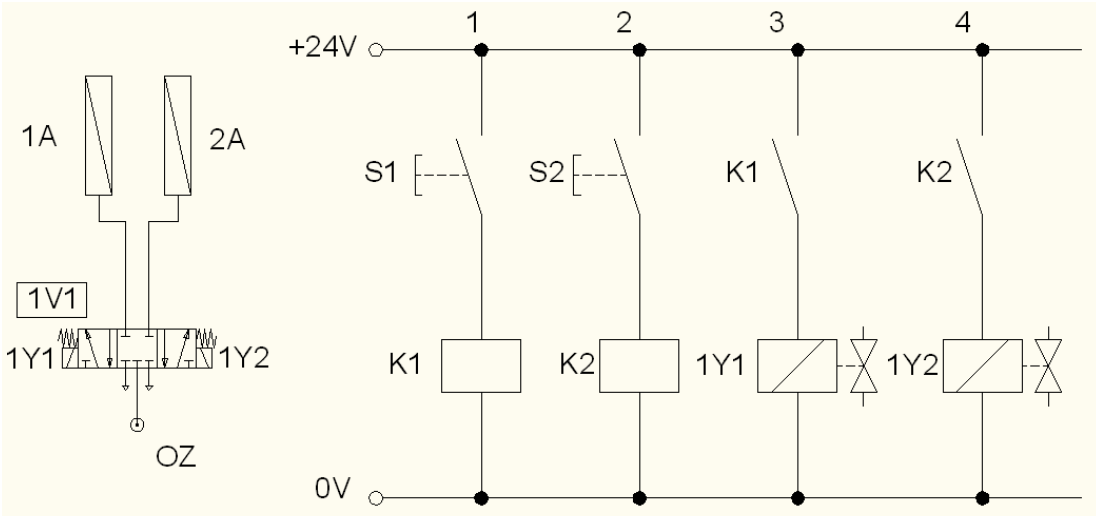


Figure 3.12: 5 / 3 Close Center DCV setup. (Left : Pneumatic Diagram, Right : Electro-pneuamtic diagram)

In Figure 3.12, a 5 / 3 close centre DCV is used to control air muscles connected in agonist-antagonist setup. Air muscle 2A would control the extension of the knee joint and 1A would control the flexion of the joint. In the electro-pneumatic diagram above, S1 and S2 are push-button with normally open contacts which are manually actuated by pushing. In practice, these two switches would be replaced with relays that can be controlled by input signals form a microcontroller.

Figure 3.13 is a demonstration of the movement of the knee joint when the valve is activated. When there is no signal provided to the solenoid, the knee joint will remains still as in Figure 3.13 (a). When S1 is activated, solenoid 1Y1 would be activated. Air from the supply OZ will start to fill into air muscle 1A which causes the air muscle to contract. This causes the knee joint to be flexed backwards as in Figure 3.13 (b). In the meantime, the air in 2A would be exhausted to the atmosphere. On the other hand, when S2 is activated, solenoid 1Y2 would be activated. Air would start to fill 2A and exhaust from 1A. The knee joint would be straightened as shown in Figure 3.13 (c).

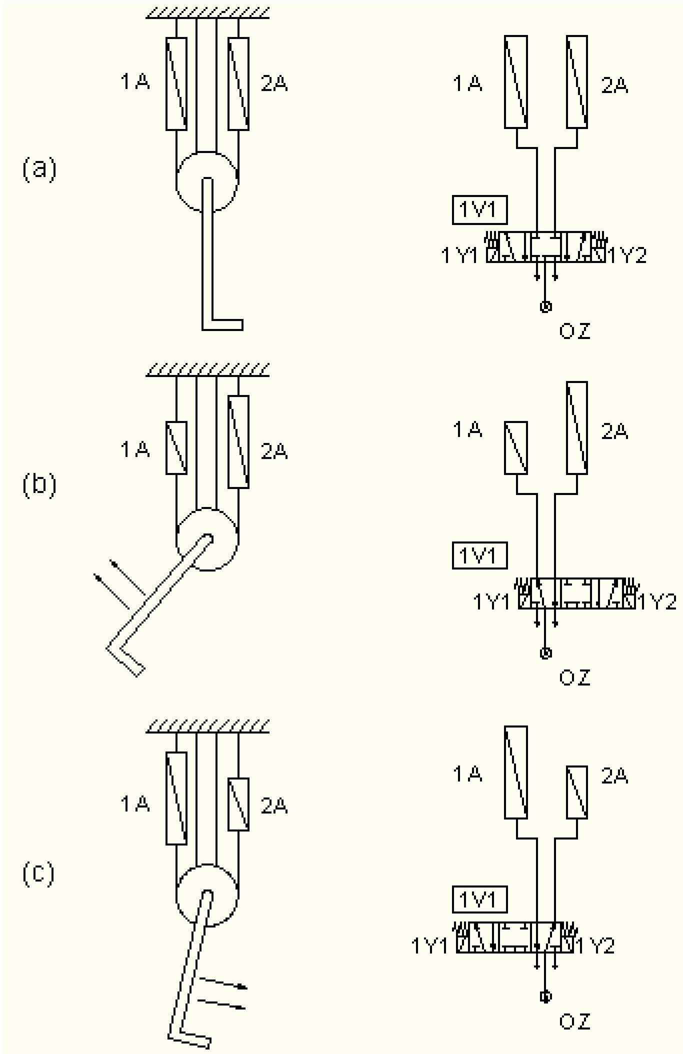


Figure 3.13: Demonstration of the Knee Joint movement when 5 / 3 Close Center DCV is activated. ((a) : Initial State, (b) : 1Y1 activated, (c) : 1Y2 activated)

The advantage of using 5 / 3 close centre DCV is that it allows joints connected to the air muscle to hold its position and cut the air flow form going in and out of the air muscle. This would be helpful when one of the biped robot's legs is in stance phase and needs to hold in that position for a period of time. It will also conserve air as no air is wasted to regulate the leg in the stance position and reduce the total air consumption of the biped robot. The conservation of air is important if the robot is designed to be self contained. Despite the advantages, using this valve causes both the air muscles to be linked together. For example, when air is supplied to 1A, the air inside of 2A would be exhausted vice versa.

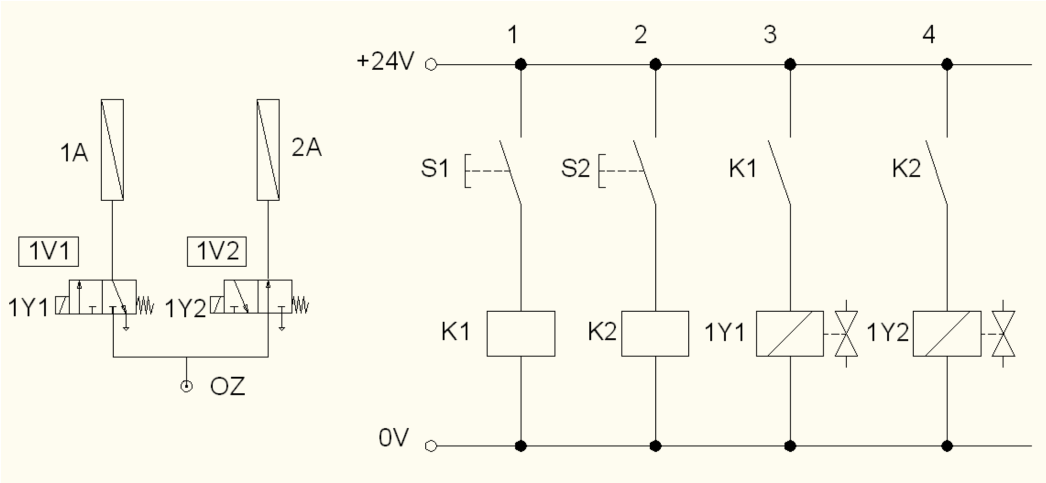


Figure 3.14: 3 / 2 DCV setup. (Left : Pneumatic Diagram, Right : Electro-pneuamtic diagram)

Figure 3.14 uses a 3 / 2 DCV to control each air muscle. Using one 3 / 2 DCV to control each muscle allows more flexible control over the air muscles as each air muscles can be controlled individually. This allows more flexible control over the compliance of the joint as the pressure of each air muscle can be controlled individually (Refer to section 2.4 and section 2.4.1 for further explanation on the importance of compliance of the joints). The drawback of using 3 / 2 DCV is it cannot trap air inside of the air muscle as air would be constantly supplied or exhausted from the air muscle. With this valve, holding the position of the joints would be difficult. Despite that, this problem might be overcome if the 3 / 2 DCV is being used for the air muscle setup in Figure 3.11 (c). This is because the knee joint is limited by the elongation of the air muscles in Figure 3.11 (c). Therefore constantly supplying air into the air muscle would not affect the joint angle of the knee and the knee joint would remain straight.

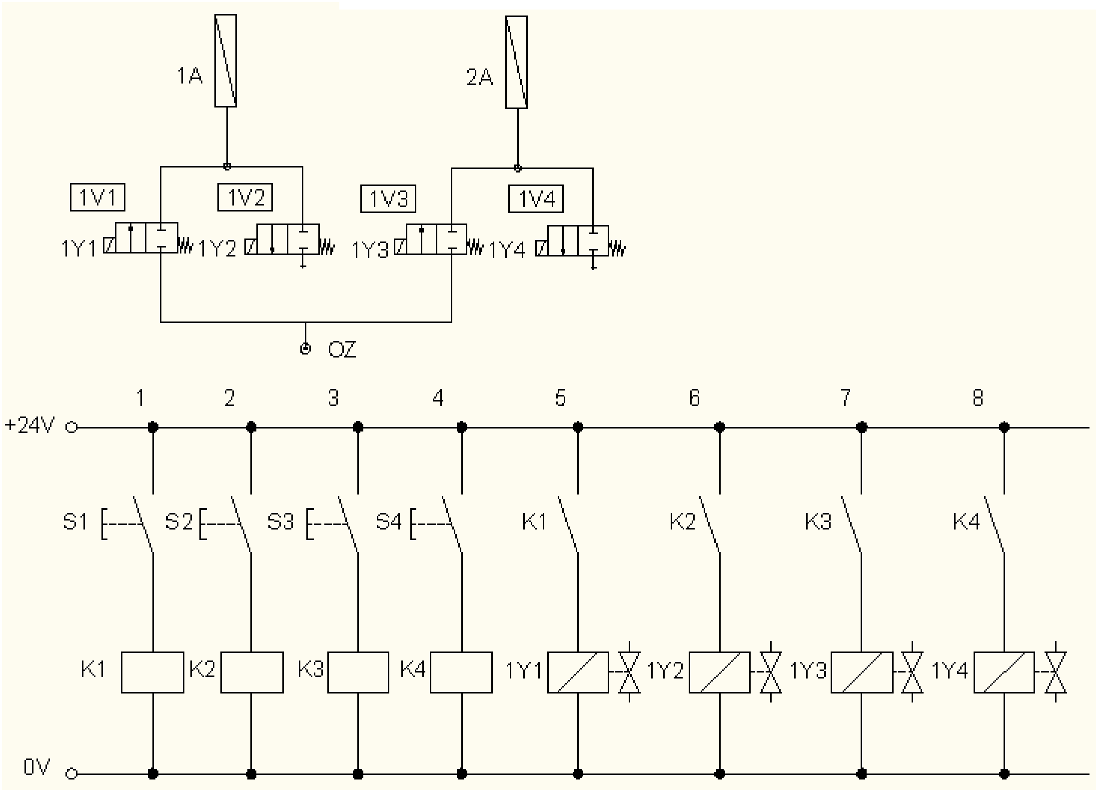


Figure 3.15: 2 / 2 DCV setup. (Top : Pneumatic Diagram, Bottom : Electro-pneuamtic diagram)

In Figure 3.15, each air muscles are controlled by two 2 / 2 DCV. This means that, for an agonist-antagonist setup, a total of 4 solenoid valves are required. For each air muscle, one 2 / 2 DCV is used for the air inlet, and the other is used for exhaust. This setup would actually overcome the problems faced by 5 / 3 DCV and 3 / 2 DCV as each air muscle could be controlled individually and the air inside of the air muscle could be sealed. Despite that, the number of valve being used and the number of control signals required to control to valves are increased. By comparing Figure 3.15 and 3.14, the number of valve and switches required in Figure 3.15 is doubled that of Figure 3.14. The total weight of the robot might also increase due to the increasing number of valves. Total cost for all these valves might also be higher compared to other valve setups.

Based on the three types of valve setup discussed above, a summary of the advantages and disadvantages of each type of valve setup is presented in the table below.

Table 3.3: Selection Criteria for Valve Setup

Type of Valve	5/3 DCV	3/2 DCV	2/2 DCV
Air Consumption	Low	High	Low
Flexibility in Control	No	Yes	Yes
No. of Valve Needed	6	12	24
No. of Signals Needed	12	12	24
Total Weight	Low	Medium	High
Cost	Low	Medium	High

Based on Table 3.2, the valve setup using 3 / 2 DCV is selected for this project. The reason for choosing 3 / 2 DCV is that it has high flexibility in control, requires the fewest amounts of control signals, and needs moderate number of valve, weight and price. The only downside of the 3 / 2 DCV is that it has high air consumption. The air consumption is important if the prototype needs to be made self-contain, but for the prototype, an external air compressor can be used to supply the air to the air muscles, therefore the air consumption would not be an issue, the air consumption of the 3 / 2 DCV can also be minimised if the response time of the 3 / 2 DCV is fast enough.

Besides that, more weight is placed on the flexibility in control of the valve setups. This is also the main reason for choosing 3 / 2 DCV over 5 / 3 DCV despite the 5 / 3 DCV having superior specifications in other aspects. The joint stiffness or compliance could be manipulated by controlling the pressure inside both air muscles. Higher pressure in both air muscles would increase the joint stiffness and reduce its compliance. Therefore, if both the air muscles are linked together as in the 5 / 3 DCV valves setup, it would reduce the flexibility to control the joint stiffness of individual joints. Most of the journal also uses valve that can control each muscle individually. (Hosoda, Takuma, Nakomato, & Hayashi, 2008)(Yamaguchi, i INOUE, Nishino, & Takanishi, 1998) (Verrelst, 2005). Further explanation of the importance of this topic is state in Section 2.4

For the 2 / 2 DCV, the main reason for not selecting this setup is its price and weight of the total valve system is very high. For the price, justification was made based on the price list given by one of the valve supplier. The price of a 5 / 3 DCV is RM 80, 3 / 2 DCV and 2 / 2 DCV has the same price which is RM 45. Therefore, multiplying with the amount of valve needed, 5 / 3 DCV would cost a total of RM 480, 3 / 2 DCV cost RM 540, and 2 / 2 DCV cost a total of RM1080. Since the budget for this whole project including the mechanical structures and sensors is only RM2000, it would be unreasonable to select 2 / 2 DCV setup as the price needed is already more than half of the budget. Furthermore, as the number of valve required increase, the weight of the total valve also increases. Since the valves would be placed on the plane on top of the hip of the robot, it would be in the best interest to reduce the total weight of the valve as the total weight of the valve on the top plane might contribute to the balancing effort needed by the robot. Higher weight also increases the air muscles strain therefore increasing the difficulty in producing suitable air muscles.

In general, by weighting each criterion in Table 3.2 it is found that the 3 / 2 DCV setup is the best among the 3 valve setups.

3.3.3 Valve Selection

For the actuator control method, PWM signals are going to be implemented to control the valves to actuate the air muscles. Since PWM signals consist of short pulses of signals, the valve selected must be able to cope with these signals. In other words, the response time of the valve selected must be quick. In most of the journals which implement PWM signals to control the solenoid valves, the responses time of the valves are in the range of 5 ms. Therefore it would be best to choose valves with respond time close to 5 ms.

Initially, two GP 3 / 2 DCV with a respond time of 50 ms was brought for testing. The data sheet of the valve would be attached in the appendix at the end of this report. The price of each valve is RM 45. The test results are tabulated in

Chapter 4. From the test result, it clearly shows that the response time of the valve is not quick enough to handle the PWM signals. This is because the minimum pulse which the valve would be able to respond is 50 ms, the valve would not activate with any signal quicker than 50 ms. With the minimum signal and adjusted duty cycle of the PWM signal, the valves response does not show promising result. This is because the whole structure of the joint is oscillating and vibrating. The vibration is due to the air being expelled into the air muscle too quickly and the valve is not quick enough to regulate the air flowing in and out of the air muscles. With the vibrations, holding the position of the joint in a certain angle seems to be impossible with the valve being used. Therefore other valves with quicker respond time must be found.

Festo’s quick respond valve MH1 and MH2 was examined. Both of them have a respond time of 4 ms and 2 ms respectively. The valve MH1 and MH2 also comes in a smaller package and also weight lighter compared to the GP valve. Other than that, the Festo MH1 valve also has the option to choose to operate in 5 volt which can be control easily by the microcontroller itself. Therefore the specification of either MH1 or MH2 fits perfectly with our application. Despite that, the price of MH1 is RM 123 and for MH2 is RM 313. A total of 12 valves are needed therefore the option of choosing these valves are dropped as the valve cannot be afforded.

Since the fast respond valve cannot be afforded, the inexpensive GP valve was re-examined to determine whether there is a solution in solving the problem encountered for the valve. It is know that the vibration occurred due to the flow rate of air being too fast for the valve to control. Therefore to rectify this problem, a throttle valve was added in front of each air muscles to reduce the speed and amount of air flowing into the air muscles. Tested result shows that the responses of the valves are rather promising. Vibration is reduced, the air muscle strength is also not affected by the throttle valve, and the only thing being affected would be the respond rate of the joint movement. This means that the overall movements of the robot would be slower and the respond of the robot structure would be slower. High response rate is needed if the robot needs to handle unexpected external forces to balance. But since the speed of the robot movement is the only parameter affected and there is no any other option, it is justifiable to use this valve coupled with the throttle valve.

In general, twelve 3 / 2 DCV supplied by GP Pneumatics, model 4V210-08, coupled with throttle valves was used for this project. Below is a figure showing the throttle valve and the solenoid valve used.

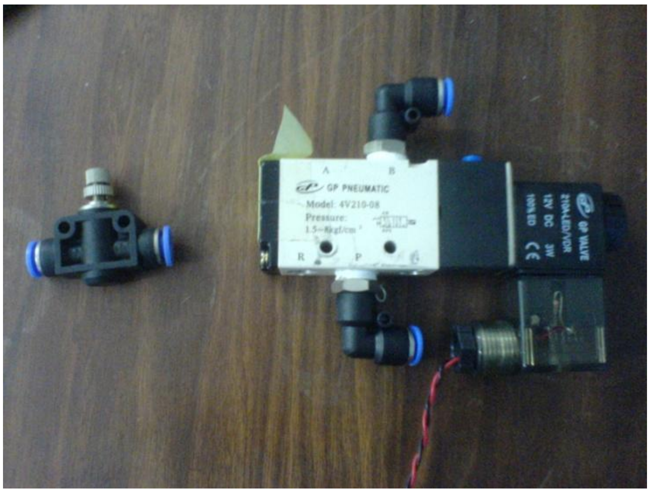


Figure 3.16: Left : Throttle Valve, Right : Solenoid Valve

3.3.4 Actuator Control Methods

There are various methods to control the McKibben air muscles. A few possible methods for controlling the actuators are derived in this section. The methods being discussed in this section would be based on the agonist-antagonist setup of the air muscles for the knee joint of the biped robot.

3.3.5 Method One

The simplest method to control the position of the knee joint is by implementing a time based on / off of the valves. Experiments would be carried out to determine the relationship between the activation time of the solenoid valve and the increment in joint angle of the robot. A table would then be tabulated. Base on the table, the time required to activate the solenoid to achieve the desired joint angle can then be determined. This method is an open loop method as no feedback is required. This means that the angle data from the sensors are not required. Therefore the system would be prone to error as the system is subjected to external loads such as gravity force, and mass of the robots body structure.

3.3.6 Method Two

To improve the previous method, a feedback signal from the position sensor can be used to control the activation of the solenoid valve. The solenoid would be activated until the desired joint angle position is achieved. This method of control might be subjected to overshoot due to the delay in respond time of the solenoid valve. Therefore, to compensate for the delay in response time, it is suggested to deactivate the solenoid valve earlier before the desired angle is achieved.

3.3.7 Method Three

Another method would be using PWM signals to control the solenoid valves, where the duty cycle of the PWM signals would be determined by the PID feedback from the position sensor. This method would be base on the literature review in section 2.5. Based on Figure 2.12 in section 2.5, the cylinder rob would be modelled as the knee joint, and the two air chambers in the cylinder would be modelled as the air muscles in agonist-antagonist setup. The method discussed in section 2.5 is suitable to be used for the valve setup in Figure 3.4 and 3.5 as the air muscles in the agonist-antagonist setup are controlled individually. For the 5 / 3 close centre DCV, only PWM scheme 2.13 (a) is possible to be adapted as both the air muscles are linked together to the same valve. Since both the air muscles are linked together, only one muscle can be effectively controlled at a time. The other air muscle would be fixed at 0 duty cycle as no air would actually be supplied into it (The air inside the air muscle would be either sealed or exhausted to the atmosphere).

3.3.8 Method Four

After carrying out experiment 3, it is found that, method three is not practical to be implemented to control the actuator of this project. The reasons for being unsuitable are that the PWM duty cycle signals cannot be zero when the error is zero or close to zero as being suggested in section 3.3.4.1 Method Three. Unlike the pneumatic cylinder, the air muscles would retract back to its original position if the duty cycle is zero rather than staying in its current position. Therefore, when error signals are zero, the duty cycle must remain the same to hold the position of the joint in that angle. The following paragraph explains the method to implement this concept.

This method also works based on the PWM control signals where the duty cycle of the PWM signals would be determined by the proportional feedback of the position sensor. The different between this method and method three is the way the PWM duty cycle was increased and handled. For this method, the PWM signals are increased based on the proportional gain of the error difference signal. Error difference is equal to the desired angle minus the current angle form the sensor data. This error is multiplied with a proportional gain, Kp which is tuned through experiment. The reason for using manual tuning is because it is the simplest method to tune the proportional controller despite being the most troublesome method. As the mathematical model of system such as the variables which controls the contraction of the air muscles are not defined, manual tuning seems to be the only method for obtaining the proportional gain of the system. Below is the flowchart showing the process flow of this scheme.

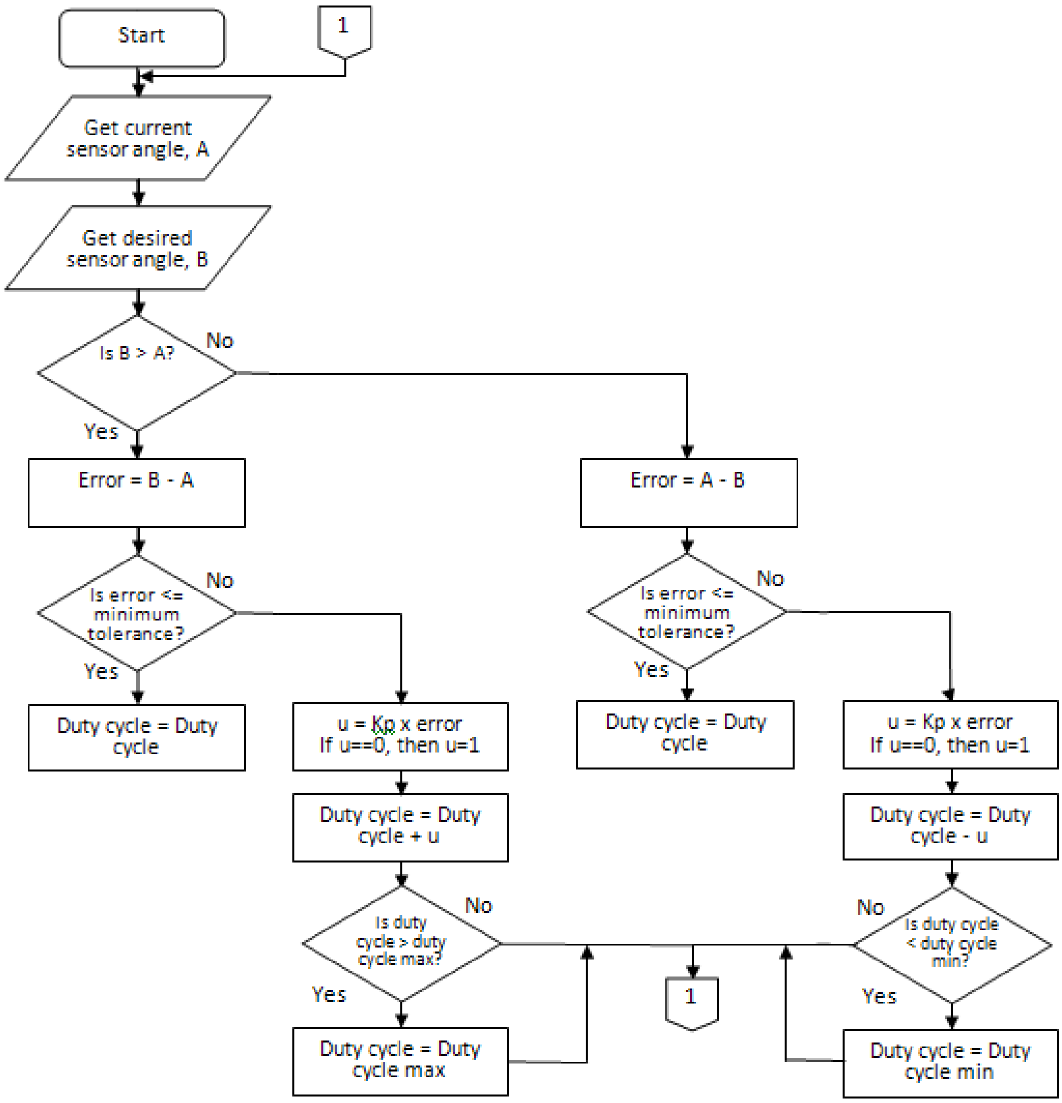


Figure 3.17: Flowchart for Method Four

From the flow chart, the process variable is the actual sensor angle, A, set point is the desired sensor angle, B, and manipulated variable is the PWM duty cycle of the valve. The comparison of the desired and actual sensor angle in the flow chart is made to determine which one has the larger value, so that the result of the minus of this two term are always positive. A minimum tolerance angle is acceptable, when the joint is within this range the duty cycle would remain the same. This is used to avoid the system from oscillating around the set point value. If the angle is not within the range, control input percentage, u would be equal to error multiplied with Kp, and the duty cycle would change accordingly. If the duty cycle after modification is less than the minimum duty cycle which is the minimum duty cycle where the valve would respond, then duty cycle would remain as duty cycle min. After that the whole process would repeat.

For this system to work, there are a few variable that has to be adjusted. First, the Kp value cannot be too large, if not the whole system would overshoot and become unstable. Second is the sampling time to determine the next duty cycle output must be appropriate. This means that the sampling time cannot be too short between two successive outputs. If it were too fast, the PWM signal would accumulate to 100% very quite even before it have reached its desired angle. This is not desired as the system might become unstable just like the previous scenario mentioned.

CHAPTER 4

RESULTS AND DISCUSSIONS

4.1 Results

There are a total of four experiments carried out to verity the sensor and actuators response which are discussed in chapter 3. Experiment 1 would verity the accuracy of the Hall Effect sensor AS5040, experiment 2 would verify the respond time of the valve being used, experiment 3 would examine on the responds of the air muscle towards PWM signals and experiment 4 would examine on the efficiency of the actuator control method being proposed in chapter 3. Discussion of analysis of the experimental result would be done in section 4.2 Discussion.

4.1.1 Experiment 1

This experiment is done to verify that the sensor is functioning properly and suitable to be used as joint angle sensor. The sensor data are gathered and displayed on the PC through UC00A using Realterm HyperTerminal program. The sensor data taken are measure in absolute position, which increase when the joint turns counter clock-wise. Measurements are taken from the actual robot joint movement. The data being displayed on the PC are in binary 10 bits form. Therefore to convert the 10 bit data into joint angle data, the binary numbers have to be multiplied by 0.3515625, which is the resolution of 1 bit of data. From the initial experiment, it was found that the most significant bit was lost while transferring the data through SPI bus. This is due to the error in writing the SPI coding for the microcontroller. With the error being discovered, proper adjustments are made and the results are tabulated in Table 4.1.

Initial angle measured = 1101010110_2 x 0.3515625=854_{10} x 0.3515625=300.2 degree

Table 4.1: Results for Experiment 1

Angle moved, degree	Current angle measured, binary	Current angle measured, degree	Measured angle = |Initial – Current angle measured| , degree
0	1101010110	300.2	0
45	1011011101	257.0	43.2
90	1001001111	207.9	92.2
180	0101001111	118.0	182.0

Figure 4.1 is a print screen of the sensor data displayed in ASCII form using Realterm. The first two letters are separators which starts with a "-" and an ASCII number which increase at the number of data sensor read increase (from 0-9). The following four letters are the angle value from the 10 bit sensor which is displayed in base 10. The angle in degree is obtained by multiplying angle value with 0.3515625.

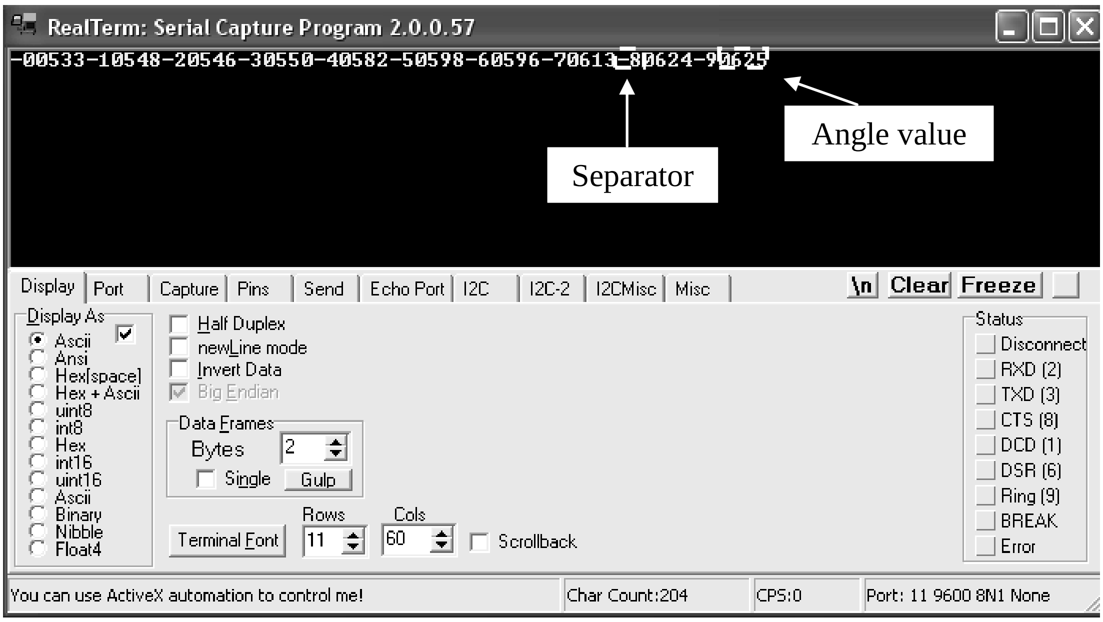


Figure 4.1: Sensor Data Displayed in ASCII

4.1.2 Experiment 2

The aim of this experiment is to determine the minimum respond time of the solenoid valve. To test the respond time of the valve, different pulse with different pulse widths were fed to the solenoid valve. For example, a pulse with lasted 0.05s, 0.04s, and 0.03s until 0.01s was sent. Assuming the valve is activated when a “click” sound can be heard from the valve. With a voltage of 12V, it is found experimentally that the valve would only respond when the pulse width is more than 0.03s. This would be the minimum time needed for the solenoid pulse to respond to the signal. In other words, dead band occurs the valves are controlled with signals shorter than 0.03s. The results can be improved if higher voltage is supplied to the valve. This is because higher voltage would increase the magnetic strength of the solenoid which makes it easier to activate the valve.

4.1.3 Experiment 3

This experiment is done to find out the reaction of the air muscle towards the varying duty cycle. Before we can develop the actuating control method, the reaction of the air muscle towards its manipulating variable has to be determined. Therefore, this information is useful for designing the actuator control method. The throttle valve also affects the reading of the data. For result in Table 4.2, the throttle valves are fully closed. Figure 4.2 is a plot of the result in Table 4.2. In Table 4.3, the duty cycle is set to 50% with the throttle valves condition being varied. For all these experiments, the PWM frequency is set to 10 Hz and the pressure supplied to the air muscle are being set to 2 bars. The PWM scheme used was developed by other members of this project.

Initial angle measured = 0011100001_2 x 0.3515625 = 79.1 degree

Table 4.2: Results for Experiment 3

PWM duty cycle,%	Measured angle, binary	Measured angle, degree	Angle moved = |Initial – measured angle| , degree	Accumulated angle, degree
0	0011100001	79.1	0	0
10	0011011100	77.3	1.8	1.8
20	0011010100	74.5	2.8	4.6
30	0011001111	72.8	1.7	6.3
40	0011001100	71.7	1.1	7.4
60	0010111101	66.5	5.2	12.6
80	0001001010	26.0	40.5	53.1
100	0000111000	19.7	6.3	59.4

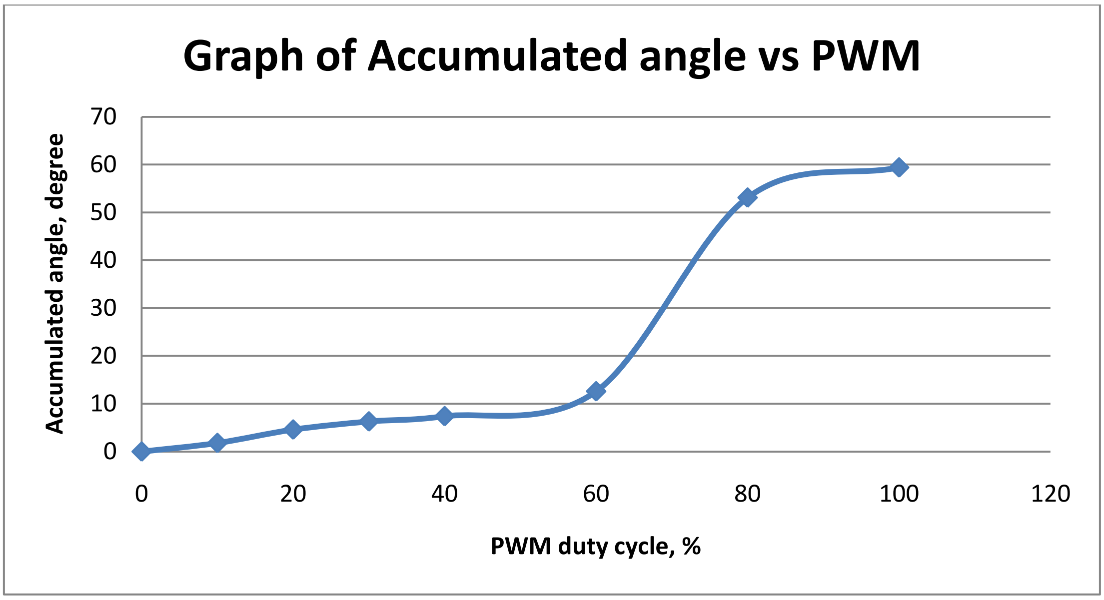


Figure 4.2: Graph of Accumulated angle vs PWM

Table 4.3: Throttle valve condition

Throttle valve condition	Observation
Fully Closed	Moves very slowly and steadily until it reach a stable angle
Partially Closed	Moves faster than fully closed but a bit of vibration is visible
Fully Opened	The whole joint is vibrating violently, stable position can't be reached as the system is oscillating

4.1.4 Experiment 4

The methodology discussed on method four was carried out. While implementing the method, 10 bit of sensor data was not fully utilized. Instead, the 8 most significant bits of the sensor data was used. Since the microcontroller can only store 8 bit of data in a memory slot, special treatment of the 10 bit data is required to perform calculation in 10 bits. If 8 most significant bits was used, only 1 byte of data needs to be handled for calculation rather than 2 bytes, therefore, the programming can be simplified significantly. With 8 bit sensor data used, the resolution of the sensor is reduced from 0.3515625 degree per step to 1.40625 degree per step.

Initial testing of the method shows unacceptable results because the system being controlled was unstable. The joint angle would not settle down on a stable position and keeps oscillating around the desired angle. Reasonable data cannot be gathered as there is difficulty in plotting the 8 bit binary data. Only visual inspection can be made. After the failure, further tuning of the variables mainly the throttle valve and the sampling time to update PWM duty cycle, and also the Kp value was made. Constrains of the system was also set properly. The constrains included are, the increment and decrement of PWM duty cycle must be within the range of 0-100%, and the minimum increment or decrement of the PWM duty cycle must be at least one as long as error is not within range .

After the tuning and adjustments, the system become stable, but the respond of the system was very slow. Kp was set to 0.1, the throttle valve was set to partially opened, the desired angle to travel is 38 degree from its initial position, and the only variable left is the sampling time to update the PWM duty cycle. The variable is set to 0.5 second and 1 second. The results are explained in the following paragraphs.

With the sampling time set to 0.5 second, the system moved very slowly and settles at the desired angle in 38 seconds. In the first 27 seconds, the angle increase rate is very low. After 27 seconds, the angle increase rate increase very fast until 32 second, this happens when the duty cycle of the PWM is around 50%, the system overshoots the desired angle and settles back down at the desired angle at 38 seconds.

With the sampling time set to 1 second, the system moved even slower and settles at the desired angle in around 1 minute. The angle increase rate increases only after 50 seconds and settles with no overshoot at 1 minute. Further discussion of this experiment would be listed in section 4.2 Discussion.

4.2 Discussion

In all of the experiment, the microcontroller PIC18F4520 was used to gather data as well as manipulating the experiment variables. This is also the microcontroller used by the main controller of the biped robot to control the overall movement of the robot. For the sensor used, only 5 pins are required from the microcontroller. UART module for displaying data to PC would require an additional of 2 pins. Since the microcontroller has 40 I/O pins, the sensor data acquisition and transmitting can be directly integrated into the main controller, reducing the need to implement two microcontrollers.

In experiment 1, it is concluded from the result that the sensor is functioning properly. The sensor is mounted directly onto the rotating joint of the biped robot. Data in experiment 1 was displayed on the PC. Initially it was though that the sensor is defected as the sensor data being displayed does not confirm to the actual movement of the angle. Through further checking, it was found out that the most significant bit was not displayed at all, as this bit remain off throughout the whole data acquisition process. With the problem being discovered, rectification can be made to the code. After the rectification, the sensors were able to display the data correctly. Further improvement on the data being displayed was made, rather than displaying the data in binary which requires conversion form binary to decimal and then to angle data, the data was displayed in ASCII form. In ASCII form, the data can be displayed directly on the monitor and conversion from the ASCII code to angle data was made easier. With this, it is possible for the data to be gathered and recorded by another program written using PC programming language. The slight difference between the actual and the measure might be largely due to the human error while aligning the joint to the desired angle. This is because the alignment processes were done only by using a protractor and ruler.

In experiment 2, the minimum respond time was found to be 0.03s at 12V power supply. This term is important while developing the actuator control system because, if the actuator is providing outputs to control the valve where the valve cannot function in this region, it would produce a situation called dead band where the valve would remain idle even thou the controlling signals are applied. This would cause the control function to be non-linearity.

In experiment 3, the reaction of the air muscle towards varying PWM duty cycle was examined. It is important to know the characteristic of the actuator that is being controlled toward the varying its input before a proper scheme to control the actuators can be developed. From the result gather by the Hall Effect sensor, the results clearly show the non-linearity of the actuator towards its controlled inputs. This relation is shown clearly in Figure 4.2. Initially, the angle movement does not seem to react much to the increasing duty cycle. The reaction of the joint angle increases when duty cycle of 60% was applied. The angle moved by the actuator achieves its peak when duty cycle of 80% was applied. After that, the amount of angle moved declined. This result was also verified by the characteristic of McKibben Air Muscle developed by “The Shadow Robotic Company” which is discussed in Chapter 2 of Section 2.3. Similarities could be found between Figure 4.2 and Figure 2.5 (a) and Figure 2.5 (b) which shows the non-linearity of the air muscle.

After this experiment, it was also concluded that the muscle would only hold at certain angle when the PWM duty cycle is not changed (Assuming no external load). If the duty cycle was reduced to zero, the joint would move back to its initial position. This is a very important property as it distinguishes the control methods which can be used by the pneumatic cylinder and control methods which can be used by the pneumatic air muscles. With this determined, the scheme which are used and discussed in Chapter 2 of Section 2.5 to control the pneumatic cylinder cannot be directly implemented onto the pneumatic air muscles despite both of them has very similar property. This is also the main reason why method 3 cannot be used and method 4 have to be derived in the Methodology Section 3.3.4.

Besides that, testing different air muscles also produce different results, as there are various variables involved. Besides that, the valve are self-fabricated, therefore, there would be many property which cannot be determined. Other controlling variable includes the throttle valves which are placed in front of each air muscle.

In experiment 4, all the previous experiments are combined into one. This includes the SPI module sensor, UART to display data onto the PC, and also PWM which is developed by other teammates of this project to actuated the pneumatic air muscle.

The initial results are not as expected, as the system is oscillating around the desired angle. Many factors can contribute to the failure of this experiment as there are a lot of variables involved in this experiment. The variables includes, the throttle valve, the air muscle fabricated, the Kp value used in the scheme, the sampling time of the sensor data and sampling time to change the duty cycle.

Air muscles are self fabricated, this means that every muscle characteristic might not be the same, and suitable adjustment have to be made for each different muscle used.

Sampling time is important because if the sampling time to change the duty cycle is too fast, the duty cycle would accumulate too fast and overshoot the desired angle. This is the main reason the initial experiment failed. Other than that, in the initial experiment, the constrains discussed in experiment 4 was not set, which cause the duty cycle to increase without bound and cause the system to overshoot.

If the sampling time is too slow, the movement of the air muscle would not be fluid and it would be very slow before it reached the desired angle as in part two of experiment 4. With low sampling time, the overshoot is lesser and the system is able to achieve steady state, but at the cost of slower respond rate.

The throttle valve and the Kp of the proportional controller must also be experimentally determined and adjusted as the mathematical model cannot be determined. Both these variables affect the respond rate of the air muscle, therefore suitable value have to be set in order for the system to be stable. In general, the results shows that control method discussed in section 3.3.8 can be use to control the pneumatic actuator and are acceptable as stability can be achieved. Despite that, the system respond is still very slow for any practical used of this method and more time have to be spent on fixing the variables one by one to determine the optimal solution to be applied to the controller. System respond might be improved if two muscles are controlled simultaneously. Oscillation, stability and system respond time can also be improved if better controller such as PID control is used.

CHAPTER 5

CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

In conclusion, the objective of creating a biped robot which is actuated by pneumatic air muscle is achieved. Besides that, proper sensor which can be implemented on the biped robot was selected and feedback from the sensor can also be gathered efficiently. Proper control schemes to control the actuators are examined. This includes the valve type and valve connection to control the pneumatic actuator, in the end of the project, one of the schemes to control the actuator were implemented, but the efficiency of the control method has yet to be improved. Proper controller board and sensor boards are also developed during this project. In generally, most of the objectives stated are achieved with reasonable results.

5.2 Recommendation and Future Improvement

The sensors used in this experiment prove to be sufficient, but on the actuator system, there are various aspects to be improved. One of the improvements would be done by changing the valves to a fast-switching valve which has a respond time of 5ms or less. With this the higher frequency PWM signals could be fed to the solenoid valve, reducing the need for using the throttle valve before each air muscle. Besides reducing weight, with fast-switching valve, the pressure and air flow in and out of the air muscles can be regulated more efficiently. Other than that, pressure sensors can be implemented into each air muscle to determine the force exerted on the joint. With the introduction of force sensors, new schemes which are used to control the joint stiffness can be developed. For example, in one of the research, both agonist-antagonist actuator pairs are fully utilized to control the joint stiffness of the robot as well as accurately positioning the desired angle position. One of the actuator is used to control the joint stiffness while the other controls the positioning of the joint (Yamaguchi, i INOUE, Nishino, & Takanishi, 1998). If possible, PID control to control two air muscles simultaneously can also be implemented rather the current proportional control with only controls one muscle. If walking was to be implemented, an adaptive controller might also be necessary to cope with different situations for example, when the leg is in stance phase or swing phase the load and control parameters required would be totally different.

REFERENCES

Austriamicrosystems. (2011). Retrieved Jan 2011, from http://www.austriamicrosystems.com

Austriamicrosystems. (2011). Retrieved Jan 2011, from AS5040 Data Sheet: http://www.austriamicrosystems.com/eng/Products/Magnetic-Encoders/Rotary-Encoders

Azevedo, C., Andreff, N., & Arias, S. (2004). BIPedal walking: from gait design to experimental analysis. *Mechatronics , 14*, 639-665.

Bishop, R. (2006). *Mechatronics: An Introduction.* CRC Press.

Bolton, W. (2003). *Mechatronics: Electronic Control System In Mechanical adn Electrical Engineering* (3rd Edition ed.). Pearspm Education Limited.

Chevallereau, C., Bessonnet, G., Abba, G., & Aoustin, Y. (2009). *Bipedal Robots: Modeling, Design and Walking Synthesis.* John Wiley & Sons, Inc.

Corpuz, F., Lafoteza, B., Broas, R., & Ramos, M. (2009). Design and Implementation of a Closed-Loop Static Balance System for the YICAL Leg 2 Biped. *TENCON 2009* (pp. 1-6). 2009 IEEE Region 10 Conference.

Cytron Technologies. (n.d.). Retrieved from Encoder With Plate: http://www.cytron.com.my

Daerden, F., & Lefeber, D. (2000). Pneumatic Artificial Muscles: actuators for robotic and automation. *European journal of Mechanical and Environmental Engineering , 47*, 10-21.

Everett, H. (1995). *Sensors for Mobile Robots : Theory and application.* K peters, Ltd.

Figliolini, G., & Ceccarelli, M. (1999). Walking Programming for an electropneumatic biped robot. *Mechatronics* , 941-964.

Hosoda, K., Takuma, T., Nakomato, A., & Hayashi, S. (2008). Biped robot design powered by antagonistic pneumatic actuators for multi-modla locomotion. *Robotics and Autonomous Systems , 56*, 46-53.

Manoonpong, P., Geng, T., Kulvicius, T., Porr, B., & Wo¨rgo¨tter, F. (2007). Adaptive, Fast Walking in a Biped Robot under Neuronal Control and Learning. *PLoS Comput Biol , 3* (7).

Nagasaka, K., Kuroki, Y., Suzuki, S., Itoh, Y., & Yamaguchi, J. (2004). Integrated Motion Control for Walking, Jumping and Running on a Small Bipedal Entertainment Robot. New Orleans: IEEE international conference on robotics and automation.

Repperger, D., Phillips, C., Neidhard-Doll, A., Reynolds, D., & Berlin, J. (2006). Actuator design using biomemicry methods and a pneumatic muscle system. *Control Engineering Practice , 14*, 999-1009.

Sakagami, Y., Watanabe, R., Aoyama, C., Matsunaga, S., Higaki, N., & Fujimura, K. (Oct 2002). The intelligent ASIMO: System overview and integration. *IEE Intl. Conference on Intelligent Rotots and System.* Lausanne, Switzerland.

Siegwart, R., & Nourbakhsh, I. (2004). *Introduction to autonomous mobile robots.* MIT Press.

Takuma, T., Hosada, K., & Asada, M. (2005). Walking Stabilization of Biped with Pneumatic Actuator against Terrain Changes. *Intelligent Robot and Systems, 2005* (pp. 4095-4100). 2005 IEEE/RSJ International Conference.

The Shadow Robot Company: Shadow Air Muscle, 30mm. (n.d.). Retrieved August 20, 2010, from Shadow Robot Company Ltd.: http://www.shadowrobot.com/downloads/datasheet_30mm_sam.pdf

Varseveld, R., & Bone, G. (1997). Accurate Position Control of a Pneumatic Actuator Using On/Off Solenoid Valves. *IEEE/ASME TRANSACTIONS ON MECHATRONICS , 2* (3).

Verrelst, B. (2005). *A dynamic walking biped actuated by pleated pneumatic artificial muscles: Basic concepts and control issues.* PhD Thesis, Vrije Universiteit Brussels.

Wisse, M., & Richard, Q. (2007). *Delft Pneumatic Bipeds.* Germany: Springer-Verlag Berlin Heidelberg.

Yamaguchi, J., i INOUE, S., Nishino, D., & Takanishi, A. (1998). Development of a Bipedal Humanoid Robot. *Intl. Conference on Intelligent Robots and Systems* .

APPENDICES

APPENDIX A: C++ programming code

Code for sensor data acquisition

```
#include <p18f4221.h>
#include <delays.h>
#include <math.h>
#include <stdio.h>

#pragma config WDT = OFF, OSC = HS, LVP = OFF

unsigned char SPI(unsigned char ByteSend);

void main()
{
        unsigned char data[2];
        unsigned char i,temp,temp2[3],angle[4],angleL=0,angleH=0;
        unsigned char counter=0x30;

        TRISBbits.TRISB5=1;
        TRISBbits.TRISB0=0;
        TRISBbits.TRISB1=0;
        TRISD = 0;
        TRISAbits.TRISA0 = 0;

        TXSTA=0x20;                     //setup UART for serial data transferring to PC
        SPBRG=0x20;
        TXSTAbits.TXEN=1;
        RCSTAbits.SPEN=1;
```

```
SSPSTAT = 0;                //SMP read at middle, CKE send on active edge
SSPCON1 = 0x32;             //master SPI enable, CKP idle high clock, Fosc / 64
TRISCbits.TRISC4 = 1;       //SDI
TRISCbits.TRISC3 = 0;       //SCLK
TRISCbits.TRISC2 = 0;       //CE
TRISCbits.TRISC5 = 0;       //SDO

while (1)
{
        PORTCbits.RC2 = 1;              //CE
        Delay10KTCYx(250);              //Delay 0.5s
        Delay10KTCYx(250);              //Delay 0.5s
        PORTCbits.RC2 = 0;              //CE
        Delay10TCYx(1);                 //Delay 2us at 20MHz

        for (i=0;i<2;i++)
        {
                data[i] = SPI(0x00);
        }

        data[0] = data[0]<<1|data[1]>>7;
        data[1]=data[1]>>5 & 0x03;
        PORTD = data[0];
        angleL=data[0]<<2|data[1];
        angleH=data[0]>>6;

        temp    = angleL / 10;          //convert 8 bit to BCD form
        angle[3]= angleL % 10;
        angle[2]= temp % 10;
        angle[1]= temp / 10;
        angle[0]= 0;

        if (angleH == 0)
        {
                temp2[0]=0x00;
                temp2[1]=0x00;
                temp2[2]=0x00;
        }
        if (angleH == 1)
        {
```

```
        temp2[0]=0x02;
        temp2[1]=0x05;
        temp2[2]=0x06;
}
if (angleH == 2)
{
        temp2[0]=0x05;
        temp2[1]=0x01;
        temp2[2]=0x02;
}
if (angleH == 3)
{
        temp2[0]=0x07;
        temp2[1]=0x06;
        temp2[2]=0x08;
}
for(i=3;i>0;i--)                    //adding MSB to calculation to form 10 bit data
{
        angle[i]=angle[i]+temp2[i-1];
        if (angle[i]>9)
        {
                angle[i]=(angle[i]+0x06)|0x20;
                angle[i-1]++;
        }
}
while(PIR1bits.TXIF==0);
TXREG= 0x2D;
while(PIR1bits.TXIF==0);
TXREG= counter;

for(i=0;i<4;i++)                    //display in 4 character ASCII I in the range of
{                                   // 0000-1023
        angle[i]=angle[i]|0x30;
        while(PIR1bits.TXIF==0);
        TXREG=angle[i];
}
PORTCbits.RC2 = 1;                          //CE
counter++;
Delay10KTCYx(250);                          //Delay 0.5ms
Delay10KTCYx(250);                          //Delay 0.5ms
```

```
        }
}
unsigned char SPI(unsigned char ByteSend)
{
        SSPBUF = ByteSend;              //transfer byte
        while(!SSPSTATbits.BF);         //receive byte
        return SSPBUF;
}
```

Code for sensor Daisy Chain Mode (2 sensor)

```
#include <p18f4520.h>
#include <delays.h>
#include <math.h>
#include <stdio.h>
#pragma config WDT = OFF, OSC = HS, LVP = OFF

unsigned char SPI(unsigned char ByteSend);
void main()
{
        unsigned char data1[2];
        unsigned char data2[2];
        unsigned char i;

        SSPSTAT = 0;                    //SMP read at middle, CKE send on active edge
        SSPCON1 = 0x32;                 //master SPI enable, CKP idle high clock, Fosc / 64
        TRISCbits.TRISC4 = 1;           //SDI
        TRISCbits.TRISC3 = 0;           //SCLK
        TRISCbits.TRISC2 = 0;           //CE
        TRISCbits.TRISC5 = 0;           //SDO
        TRISBbits.TRISB0 = 1;           //Other Inputs

        TRISAbits.TRISA0 = 0;
        TRISAbits.TRISA1 = 0;
        TRISD = 0;
        PORTBbits.RB0 = 0;

        while (1)
        {
                PORTAbits.RA0 = 1;                      //checking bit for functionality
```

```
                PORTCbits.RC2 = 1;                  //CE
                Delay10KTCYx(250);                  //Delay 0.5s
                PORTCbits.RC2 = 0;                  //CE
                Delay10TCYx(1);                     //Delay 2us at 20MHz
                for (i=0;i<2;i++)
                {
                        data1[i] = SPI(0x00);
                        data2[i] = SPI(0x00);
                        PORTCbits.RC3 = 0;
                        Nop();
                        Nop();
                        PORTCbits.RC3 = 1;
                        Nop();
                        Nop();
                }
                if (PORTBbits.RB0 == 0)         //byte select
                {
                        PORTD = data1[0]<<1;        //sensor 1 (only the 8 MSB)
                        PORTAbits.RA1 = 0;          //for checking
                }
                else
                {
                        PORTD = data1[1]<<1;        //sensor 2
                        PORTAbits.RA1 = 1;
                }
                PORTCbits.RC2 = 1;                  //CE disable SPI
                Delay10KTCYx(250);                  //Delay 0.5ms
                Delay10KTCYx(250);                  //Delay 0.5ms
        }
}
unsigned char SPI(unsigned char ByteSend)
{
        SSPBUF = ByteSend;                  //transfer byte
        while(!SSPSTATbits.BF);         //receive byte
        return SSPBUF;
}
```

Code for sensor Daisy Chain Mode (6 sensor)

```
#include <p18f4221.h>
#include <delays.h>
#include <math.h>
#include <stdio.h>

#pragma config WDT = OFF, OSC = HS, LVP = OFF

unsigned char SPI(unsigned char ByteSend);
unsigned char DASCII (unsigned char data0, unsigned char data1);

void main()
{
        unsigned char data1[6],data2[6];
        unsigned char i,j=0;
        unsigned char counter=0x30;

        TRISBbits.TRISB5=1;
        TRISBbits.TRISB0=0;
        TRISBbits.TRISB1=0;
        TRISD = 0;
        TRISAbits.TRISA0 = 0;
        TRISBbits.TRISB6=0;
        TRISBbits.TRISB7=0;

        TXSTA=0x20;                 //setup UART for serial data transferring to PC
        SPBRG=0x20;
        TXSTAbits.TXEN=1;
        RCSTAbits.SPEN=1;

        SSPSTAT = 0;                //SMP read at middle, CKE send on active edge
        SSPCON1 = 0x32;             //master SPI enable, CKP idle high clock, Fosc / 64
        TRISCbits.TRISC4 = 1;       //SDI
        TRISCbits.TRISC3 = 0;       //SCLK
        TRISCbits.TRISC2 = 0;       //CE
        TRISCbits.TRISC5 = 0;       //SDO

        T0CON=0x08;                 //no prescaler
        TMR0H=0xEC;                 //set time register for high byte
        TMR0L=0x77;                 //set time register for low byte
```

```
                                //0.2e-6 x (FFFF-EC77)=(65535-5000) x 0.2e-6 =  1ms
INTCONbits.TMR0IF=0;            //set interrupt flag to 0
INTCONbits.TMR0IE=1;            // Enable timer interrupt
T0CONbits.TMR0ON=1;             //Enable timer
INTCONbits.PEIE=1;              // Enable peripheral interrupt
INTCONbits.GIE=1;               //Enable global interrupt

while (1)
{
        PORTCbits.RC2 = 1;                      //CE
        Delay10KTCYx(250);                      //Delay 0.5ms
        Delay10KTCYx(250);                      //Delay 0.5ms

        while(PIR1bits.TXIF==0);                //display separator
        TXREG= 0x2D;
        while(PIR1bits.TXIF==0);
        TXREG= counter;

        PORTCbits.RC2 = 0;                      //CE
        Delay10TCYx(1);                         //Delay 2us at 20MHz
        j=0;
        for (i=0;i<6;i++)               //get 6 sensor data through SPI (16 bits)
        {                               //data for sensor 1 is data1[0] and data2[0]
                if (j <= 2)             //sensor 2 is data1[1] and data2[1]
                {                       //sensor 3 is data1[2] and data2[2] and so on
                        PORTBbits.RB6 = 1;      //multiplexer selector for 2 chain
                        PORTBbits.RB7 = 0;
                }
                if (j >= 3)
                {
                        PORTBbits.RB6 = 0;
                        PORTBbits.RB7 = 1;
                }
                data1[i] = SPI(0x00);
                data2[i] = SPI(0x00);
                SSPCON1 = 0x12;         //SPI disable, CKP idle high clock, Fosc / 64
                PORTCbits.RC3 = 0;
                Nop();
                Nop();
                Nop();
```

```
                PORTCbits.RC3 = 1;
                Nop();
                Nop();
                Nop();
                SSPCON1 = 0x32;         //SPI enable, CKP idle high clock, Fosc / 64
                j++;
            }
            PORTCbits.RC2 = 1;          //CE

            for (i=0;i<6;i++)           //display all 6 sensor data
            {
                data1[i] = data1[i]<<1|data2[i]>>7;
                data2[i]=data2[i]>>5 & 0x03;
                DASCII (data1[i],data2[i]);
            }
            PORTD = data1[0];
            counter++;

            Delay10KTCYx(250);          //Delay 0.5ms
            Delay10KTCYx(250);          //Delay 0.5ms
        }
}
unsigned char SPI(unsigned char ByteSend)
{
        SSPBUF = ByteSend;              //transfer byte
        while(!SSPSTATbits.BF);         //receive byte
        return SSPBUF;
}
unsigned char DASCII (unsigned char data0, unsigned char data1)
{
            unsigned char i,temp,temp2[3],angle[4],angleL=0,angleH=0;

            angleL=data0<<2|data1;    //for display sensor data in ASCII through UART
            angleH=data0>>6;
            temp    = angleL / 10;
            angle[3]= angleL % 10;
            angle[2]= temp % 10;
            angle[1]= temp / 10;
            angle[0]= 0;
            if (angleH == 0)
```

```
        {
            temp2[0]=0x00;
            temp2[1]=0x00;
            temp2[2]=0x00;
        }
        if (angleH == 1)
        {
            temp2[0]=0x02;
            temp2[1]=0x05;
            temp2[2]=0x06;
        }
        if (angleH == 2)
        {
            temp2[0]=0x05;
            temp2[1]=0x01;
            temp2[2]=0x02;
        }
        if (angleH == 3)
        {
            temp2[0]=0x07;
            temp2[1]=0x06;
            temp2[2]=0x08;
        }
        for(i=3;i>0;i--)    //adding MSB to LSB to form 4 ASCII character
        {
            angle[i]=angle[i]+temp2[i-1];
            if (angle[i]>9)
            {
                angle[i]=(angle[i]+0x06)|0x20;
                angle[i-1]++;
            }
        }
        for(i=0;i<4;i++)   //Display 4 ASCII char to PC in the range of 0000-1023
        {
            angle[i]=angle[i]|0x30;
            while(PIR1bits.TXIF==0);
            TXREG=angle[i];
        }
}
```

Implement PWM, SPI and UART for Experiment 3

```
#include <p18f4221.h>
#include <delays.h>
#include <math.h>
#include <stdio.h>

#pragma config WDT = OFF, OSC = HS, LVP = OFF

#define re1 LATBbits.LATB1
#define rf2 LATBbits.LATB0

unsigned int count=0;
unsigned char pwm3=0, pwm9=0;

//define sub-function in program
void T0_ISR(void);
unsigned char SPI(unsigned char ByteSend);

//interrupt flag checking function
#pragma interrupt chk_isr
void chk_isr (void)
{
        if (INTCONbits.TMR0IF==1)
        {
                T0_ISR();
        }
}

#pragma code high_vector=0x0008

//high priority for interrupt
void My_Hiprio_Int(void)
{
        _asm
                GOTO chk_isr
        _endasm
}

//PWM control module
void T0_ISR(void)
```

```
{
        if(count<=pwm3)
        {
                rf2=1;

        }
        else
        {
                rf2=0;
        }
        if(count<=pwm9)
        {
                re1=1;
        }
        else
        {
                re1=0;
        }
if(count==100)
{
        count=0;
}
else
{
        count =  count+1;
}
        TMR0H=0xEC;
        TMR0L=0x77;
        INTCONbits.TMR0IF=0;            //clear interrupt flag
}
#pragma code

void main()
{
        unsigned char u=0,y=0;
        unsigned int config = 0b00000011;
        unsigned char data[2];
        unsigned char i,temp,temp2[3],angle[4],angleL=0,angleH=0;
        unsigned char counter=0x30;
```

```
TRISBbits.TRISB5=1;
TRISBbits.TRISB0=0;
TRISBbits.TRISB1=0;
TRISD = 0;
TRISAbits.TRISA0 = 0;

TXSTA=0x20;                 //setup UART for serial data transferring to PC
SPBRG=0x20;
TXSTAbits.TXEN=1;
RCSTAbits.SPEN=1;

SSPSTAT = 0;                //SMP read at middle, CKE send on active edge
SSPCON1 = 0x32;             //master SPI enable, CKP idle high clock, Fosc / 64
TRISCbits.TRISC4 = 1;       //SDI
TRISCbits.TRISC3 = 0;       //SCLK
TRISCbits.TRISC2 = 0;       //CE
TRISCbits.TRISC5 = 0;       //SDO



T0CON=0x08;                 //no prescaler
TMR0H=0xEC;                 //set time register for high byte
TMR0L=0x77;                 //set time register for low byte
                            //0.2e-6 x (FFFF-EC77)=(65535-5000) x 0.2e-6 =  1ms
INTCONbits.TMR0IF=0;        //set interrupt flag to 0
INTCONbits.TMR0IE=1;        // Enable timer interrupt
T0CONbits.TMR0ON=1;         //Enable timer
INTCONbits.PEIE=1;          // Enable peripheral interrupt
INTCONbits.GIE=1;           //Enable global interrupt

while (1)
{
        pwm3=20;
        pwm9=30;

        PORTCbits.RC2 = 1;              //CE
        Delay10KTCYx(250);              //Delay 0.5ms
        Delay10KTCYx(250);              //Delay 0.5ms

        PORTCbits.RC2 = 0;              //CE
        Delay10TCYx(1);                 //Delay 2us at 20MHz
```

```
for (i=0;i<2;i++)                    //get sensor data through SPI
{
        data[i] = SPI(0x00);
}

data[0] = data[0]<<1|data[1]>>7;
data[1]=data[1]>>5 & 0x03;
PORTD = data[0];

angleL=data[0]<<2|data[1];           //for display sensor data in ASCII
angleH=data[0]>>6;                   //through UART

temp    = angleL / 10;
angle[3]= angleL % 10;
angle[2]= temp % 10;
angle[1]= temp / 10;
angle[0]= 0;
if (angleH == 0)
{
        temp2[0]=0x00;
        temp2[1]=0x00;
        temp2[2]=0x00;
}
if (angleH == 1)
{
        temp2[0]=0x02;
        temp2[1]=0x05;
        temp2[2]=0x06;
}
if (angleH == 2)
{
        temp2[0]=0x05;
        temp2[1]=0x01;
        temp2[2]=0x02;
}
if (angleH == 3)
{
        temp2[0]=0x07;
        temp2[1]=0x06;
```

```
				temp2[2]=0x08;
			}
			for(i=3;i>0;i--)		//adding MSB to LSB to form 4 ASCII character
			{
				angle[i]=angle[i]+temp2[i-1];
				if (angle[i]>9)
				{
					angle[i]=(angle[i]+0x06)|0x20;
					angle[i-1]++;
				}
			}

			while(PIR1bits.TXIF==0);
			TXREG= 0x2D;
			while(PIR1bits.TXIF==0);
			TXREG= counter;

			for(i=0;i<4;i++)		//Display 4 ASCII char to PC in the range of 0000-1023
			{
				angle[i]=angle[i]|0x30;
				while(PIR1bits.TXIF==0);
				TXREG=angle[i];
			}

			PORTCbits.RC2 = 1;			//CE

			counter++;
			Delay10KTCYx(250);			//Delay 0.5ms
			Delay10KTCYx(250);			//Delay 0.5ms
		}
}
unsigned char SPI(unsigned char ByteSend)
{
		SSPBUF = ByteSend;				//transfer byte
		while(!SSPSTATbits.BF);			//receive byte
		return SSPBUF;
}
```

Code for Experiment 4

```
#include <p18f4221.h>
#include <delays.h>
#include <math.h>
#include <stdio.h>

#pragma config WDT = OFF, OSC = HS, LVP = OFF

#define re1 LATBbits.LATB1
#define rf2 LATBbits.LATB0

unsigned int count=0;
unsigned char pwm3=0, pwm9=0;

//define sub-function in program
void T0_ISR(void);
unsigned char SPI(unsigned char ByteSend);

//interrupt flag checking function
#pragma interrupt chk_isr
void chk_isr (void)
{
	if (INTCONbits.TMR0IF==1)
	{
		T0_ISR();
	}
}

#pragma code high_vector=0x0008

//high priority for interrupt
void My_Hiprio_Int(void)
{
	_asm
		GOTO chk_isr
	_endasm
}

//PWM control module
void T0_ISR(void)
```

```
{
        if(count<=pwm3)
        {
                rf2=1;

        }
        else
        {
                rf2=0;
        }
        if(count<=pwm9)
        {
                re1=1;
        }
        else
        {
                re1=0;
        }
if(count==100)
{
        count=0;
}
else
{
        count =  count+1;
}
        TMR0H=0xEC;
        TMR0L=0x77;
        INTCONbits.TMR0IF=0;            //clear interrupt flag
}
#pragma code

void main()
{
        unsigned char u=0,y=0;
        unsigned int config = 0b00000011;
        unsigned char data[2];
        unsigned char i,temp,temp2[3],angle[4],angleL=0,angleH=0,z=1,initial,desired;
        unsigned char counter=0x30,error=0,x;
```

```
TRISBbits.TRISB5=1;
TRISBbits.TRISB0=0;
TRISBbits.TRISB1=0;
TRISD = 0;
TRISAbits.TRISA0 = 0;

TXSTA=0x20;                    //setup UART for serial data transferring to PC
SPBRG=0x20;
TXSTAbits.TXEN=1;
RCSTAbits.SPEN=1;
SSPSTAT = 0;                   //SMP read at middle, CKE send on active edge
SSPCON1 = 0x32;                //master SPI enable, CKP idle high clock, Fosc / 64
TRISCbits.TRISC4 = 1;          //SDI
TRISCbits.TRISC3 = 0;          //SCLK
TRISCbits.TRISC2 = 0;          //CE
TRISCbits.TRISC5 = 0;          //SDO

T0CON=0x08;                              //no prescaler
TMR0H=0xEC;                              //set time register for high byte
TMR0L=0x77;                              //set time register for low byte
//0.2e-6 x (FFFF-EC77)=(65535-5000) x 0.2e-6 =  1ms
INTCONbits.TMR0IF=0;  //set interrupt flag to 0
INTCONbits.TMR0IE=1; // Enable timer interrupt
T0CONbits.TMR0ON=1;            //Enable timer
INTCONbits.PEIE=1;             // Enable peripheral interrupt
INTCONbits.GIE=1;              //Enable global interrupt

Delay10KTCYx(250);                       //Delay 0.5ms
Delay10KTCYx(250);                       //Delay 0.5ms

while (1)
{
        pwm9=80;

        PORTCbits.RC2 = 1;                       //CE
        Delay10KTCYx(250);                       //Delay 0.5s
        Delay10KTCYx(250);                       //0.5
        PORTCbits.RC2 = 0;                       //CE
        Delay10TCYx(1);                          //Delay 2us at 20MHz
```

```
for (i=0;i<2;i++)                              //get sensor data through SPI
{
        data[i] = SPI(0x00);
}
data[0] = data[0]<<1|data[1]>>7;
data[1]=data[1]>>5 & 0x03;
PORTD = data[0];

if (z==1)                          //defined desired angle with initial angle
{
        initial = data[0];
        desired = initial - 0x1B; // around 38 degree
        z=0;
        while(PIR1bits.TXIF==0);
        TXREG=desired;
}
if (desired > data[0])
{
        error = desired - data[0];
        if (error>0 && error<3) //~2.8125
                pwm3=pwm3;
        else
        {
        x=error/10;                    //Kp = 0.1
                if (x==0)
                        x=1;
                pwm3=pwm3-x;
                if      (pwm3>100)
                        pwm3=100;
        }
}
if (desired < data[0])
{
        error = data[0] - desired;
        if (error>0 && error<3) //~2.8125
                pwm3=pwm3;
        else
        {
        x=error/10;
                if (x==0)
```

```
                x=1;
        pwm3=pwm3+x;
        if      (pwm3<0)
                pwm3=0;
    }
}
angleL=data[0]<<2|data[1];    //for display sensor data in ASCII through UART
angleH=data[0]>>6;
temp    = angleL / 10;
angle[3]= angleL % 10;
angle[2]= temp % 10;
angle[1]= temp / 10;
angle[0]= 0;

if (angleH == 0)
{
    temp2[0]=0x00;
    temp2[1]=0x00;
    temp2[2]=0x00;
}
if (angleH == 1)
{
    temp2[0]=0x02;
    temp2[1]=0x05;
    temp2[2]=0x06;
}
if (angleH == 2)
{
    temp2[0]=0x05;
    temp2[1]=0x01;
    temp2[2]=0x02;
}
if (angleH == 3)
{
    temp2[0]=0x07;
    temp2[1]=0x06;
    temp2[2]=0x08;
}

for(i=3;i>0;i--)
```

```
		{
			angle[i]=angle[i]+temp2[i-1];
			if (angle[i]>9)
			{
				angle[i]=(angle[i]+0x06)|0x20;
				angle[i-1]++;
			}
		}

		while(PIR1bits.TXIF==0);
		TXREG= 0x2D;
		while(PIR1bits.TXIF==0);
		TXREG= counter;

		for(i=0;i<4;i++)
		{
			angle[i]=angle[i]|0x30;
			while(PIR1bits.TXIF==0);
			TXREG=angle[i];
		}

		while(PIR1bits.TXIF==0);		//display current PWM duty cycle
		TXREG=pwm3;

		PORTCbits.RC2 = 1;			//CE
		counter++;
	}
}
unsigned char SPI(unsigned char ByteSend)
{
	SSPBUF = ByteSend;			//transfer byte
	while(!SSPSTATbits.BF);		//receive byte
	return SSPBUF;
}
```