Worcester Polytechnic Institute

Major Qualifying Project Progress Report

MIRA: Modular Interchangeable Robotic Arm

Submitted on

March 3, 2018

Submitted by

Chris O'Shea, RBE
Alex Taglieri, RBE/CS
Ben Titus, RBE/ECE

Contents

1 Introduction **1**

2 Background **1**

2.1 Examples of Robot Arms 1

2.1.1 ABB Robotics 2

2.1.2 Universal Robots 2

2.1.3 KUKA AG 3

2.1.4 Small Industrial Robotic Arm Comparison 3

2.2 Modular Arms 3

2.2.1 igus Robolink 3

2.2.2 Reconfigurable Modular Manipulator 4

2.2.3 Modular Robotic Arm 4

2.3 Our Robotic Arm System 5

2.4 Control board 5

2.4.1 Joint Position Detection 5

2.4.2 Current Sensing 6

2.4.3 Off-Board Communication 8

3 Description of Work **9**

4 Methodology **10**

4.1 Kit Components 10

4.2 Connectors 10

4.2.1 Connector Position on Joints 10

4.2.2 Securing the Connection 11

4.2.3 Keying the Connector 11
4.2.4 Passing Signals 12
4.3 Sticks 13
4.4 Motor Selection 14
4.5 Joint Control Board Part selection 15
4.5.1 Joint Angle Sensor 15
4.5.2 Motor Current Sensor 16
4.5.3 Off-board Communication 16

5 Joint Control Board 17

5.1 Motor Driver 17
5.2 DC Motor 18
5.3 Demultiplexer 18
5.4 INA332 18
5.5 TM4C123GXL Launchpad 19
5.6 Hall Effect Encoder 19
5.7 CAN Bus 19
5.8 Arm Structure 19
5.9 Fourth Joint 20
5.9.1 Fourth Joint Design 20
5.9.2 Fourth Joint Prototyping 20
5.9.3 3-D Printing 21
5.10 Arm Base 21
5.11 End-of-Arm Tooling 21
5.12 Maven 21

6 Capstone Design Requirements **21**

6.1 Computer Science 22

6.2 Electrical and Computer Engineering 22

7 Constraints **23**

8 Acceptance Criteria **23**

8.1 Joint Board 23

8.2 Modify RBE3001 Arm 23

8.3 End Effector 23

8.4 Base 24

8.5 Software Application 24

8.6 Code Library 24

9 Testing **25**

9.1 Motor Driver 25

9.2 DC Motor 25

9.3 Demultiplexer 25

9.4 INA332 25

9.5 TM4C123 25

9.6 Hall Effect Encoder 26

9.7 CAN Bus 26

Appendix A Kit Components **28**

Appendix B Gantt Chart **29**

List of Figures

1 ABB IRB 120 arm [1] 2

1 Introduction

The goal of this project was to create an adaptable control system whose purpose is to drive a variety of robotic arms. To prove the usefulness of our system, we retrofitted an existing robot arm from the RBE 3001 course with our control system. In addition to this, we modified the existing arm to include an interchangeable joint on the end of our arm, controlled by our system. To interface with our system, we created a GUI for easy configuration and basic control of the arm. The base communicates with a computer running control code through the software application.

The existing RBE 3001 arm already had its own central control system. We removed this and replaced it with our own distributed one. To accomplish this, we designed a motor control board that communicates with a base controller. We also designed and implement a new joint on the arm. This new joint is interchangeable to prove that our software and control system will work with more than one configuration. The joint implements our control board.

We created a software application to interface with a constructed arm. The user inputs how they have configured their arm into this application and are then able to do some simple control. Another feature of this application is the ability to record a series of poses for the arm to perform. In addition to this software, we also created some programming libraries to allow users to control the arm with their own programs.

In this document, we will outline some existing robot arms and highlight the differences between these arms and our arm kit. We then discuss what work there is to be done on this project. After discussing the work to be done, we state how this work will satisfy the capstone design requirements for each of the three disciplines represented by our group members. Then, we state the constraints we expect going forward with this project. Next, the acceptance criteria for any deliverables at the end of this project will be outlined. Finally, we will state an estimated time line for this project.

2 Background

This section begins with an overview of some existing robotic arms that are currently in industry use. Next, we discuss some existing modular robotic arms. Finally, we highlight how our project is different from the previously discussed examples and why this is important.

2.1 Examples of Robot Arms

There are a few different types of robot arms available on the market today. Industrial robotic arms, the most common type of arm currently in use are defined as robotic systems

used for manufacturing by means of an end effector. Since our arm is relatively small and handles smaller payloads compared to most industrial arms, we will begin with an overview of existing "desktop" industrial arms. Arms that fit this description have a reach of less than 1000mm. Industrial robot arms typically cost between $50,000 and $80,000 new and $25,000 and $40,000 used [2]. Some manufacturers of these industrial arms include ABB Robotics, Universal Robots, and KUKA Robotics. It's important to note that none of these arms are modular - in fact, they can't be changed at all!

2.1.1 ABB Robotics

ABB Robotics makes many small industrial arms. The ABB IRB 120 boasts a 580mm reach, 3kg payload, and 25kg weight. It has 6 degrees of freedom and can be mounted at any angle. The ABB IRB 1200 comes in two varieties, one with a reach of 703mm reach and 7kg payload, and one with a 901mm reach and 5kg payload. Both of these arms have 6 degrees of freedom. The weights are similar at 52kg and 54kg respectively [2].

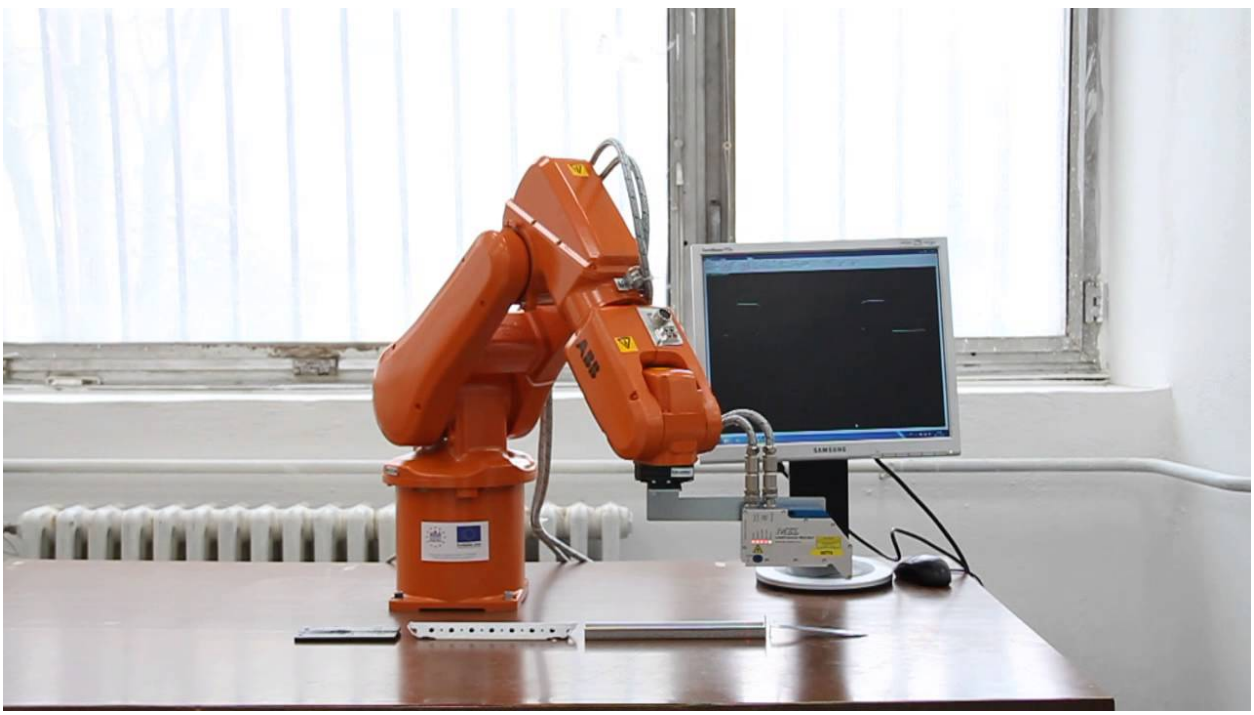

Figure 1: ABB IRB 120 arm [1]

2.1.2 Universal Robots

Universal Robots makes two robot arms in this size range. The UR3 is the smaller of the two with a reach of 500mm, payload of 3kg, and 11kg weight. A step up is the UR5 which has a 850mm reach, 5kg payload, and 18.1kg weight. Both of these arms have 6 degrees of

freedom. Universal boasts that these arms are easy to implement and re-implement due to compact and lightweight construction, and simple programming interface [2].

2.1.3 KUKA AG

KUKA makes two robot arms in this size range. The KR3 R540 has a reach of 541mm, payload of 3kg, and weight of 26kg. It can be mounted on the floor, wall, or ceiling for added utility. The K5 sixx R650 is larger with a reach of 650mm, payload of 5kg, and weight of 127kg. It can only be mounted on the floor or ceiling. Both of these arms have 6 degrees of freedom [2].

2.1.4 Small Industrial Robotic Arm Comparison

Table 1 shows a comparison of all the robotic arms discussed in this section.

Name	**Reach (mm)**	**Payload (kg)**	**Weight (kg)**	**Axes**
IRB120	580	3	25	6
IRB1200-7/0.7	703	7	52	6
IRB1200-5/0.9	901	5	54	6
UR3	500	3	11	6
UR5	850	5	18.1	6
KR3 R540	541	3	26	6
K5 sixx R650	650	5	127	6

Table 1: Comparison of <1000mm reach industrial robot arms

2.2 Modular Arms

While there are many industrial arms in production, there are very few modular robotic arms. There is one commercially available robotic arm, the Robolink, made by igus. A few modular robot arms have been developed, including the reconfigurable modular manipulator (RMM), made by TRACLabs [3], and a single joint for the Modular Robotic Arm project MQP at WPI [4].

2.2.1 igus Robolink

Robolink is a modular robotic arm kit produced by the plastics manufacturing company igus. The kit contains parts to make an arm that is up to 6 Degrees of Freedom (DOF),

with belt driven linkages powered by stepper motors that reside in the base of the robot. Robolink offers 7 individual links, ranging from 1-2 DOF and differing based upon their kind of motion (pivoting, rotating, swiveling). Each link is made of a lightweight and strong plastic or carbon fiber with cables inlaid in them, resulting in a low cost and weight arm. The cables used to control these links are made of a high strength synthetic fiber with has a tensile strength of 4,000N. Separating the actuation of each link from the joint allows the arms to be easily maneuverable with its lightweight and strong joints.

Purchasers of the kit are able to combine the links in different ways, allowing for a flexible, modular solution to robotic arms. Igus also offers their Robolink software for programming articulated arms that facilitates the programming of individual arms through the use of a simple, intuitive control software. The total cost of a kit to make a 6 DOF arm is $6000, and buying individual links will cost anywhere from $370 to $750 per link. While this price may be low cost compared to other arms such as the ABB robotic arm which can cost up to $200,000 in total, it is still not low enough for either hobbyists or people interested in learning about robotic arms who are prevented from doing so by the high entry cost. In addition to this, the belt system actuating each link requires the user to thread belts attached to the actuators to each link in order to set up the robot. The long assembly time and intricacy also detracts from the idea of modularity because the time involved in switching configurations can inhibit users from really exploring the different workspaces and combinations this kit can create [5].

2.2.2 Reconfigurable Modular Manipulator

The reconfigurable modular manipulator developed by TRACLabs for NASA is a fully modular 7-DOF robot arm. Each joint and end effector have the same connector that provides power and control lines throughout the arm. Internal power and control circuitry take in these lines and convert them into movement. Joints can be swapped out by hand in a matter of seconds. Joints accept position or velocity data from the central communication lines and store physical characteristics about the joints in memory. This robot arm is not commercially available [3].

2.2.3 Modular Robotic Arm

This project aimed to close the market gap between inexpensive toy robot arms and expensive professional grade industrial arms. The group aimed to do this by designing a single joint that could be used to assemble a robot arm. Ultimately, a single DOF joint that was heavy, difficult to manufacture, and expensive to produce was designed and constructed. In their future recommendations section, the group stated that the goal of designing a modular robot arm was possible but their design was not the solution [4].

2.3 Our Robotic Arm System

Our modular robotic arm kit aims to offer a completely different experience compared to existing products and projects. The system maintains a low cost while providing a versatile platform for beginner engineers or rapid prototyping professionals. This is achieved by avoiding expensive proprietary software and subtractive manufacturing; favoring off-the-shelf parts, 3D-printed structures, and freely available software. Providing custom-built software for controlling the arm creates a plug-and-play environment suitable for most any skill level.

2.4 Control board

The control board is meant to be implemented as an independent module that interfaces with a main controller module. Its tasks are to send and receive data from the main controller and control the position of a single motor. As such, the main factors that must be taken into account when designing the control board are methods of measuring joint position and motor torque, as well as communicate with an off-board controller. Motor torque is proportional to motor current. Therefore, the motor torque will be calculated from the measured current through the motor.

2.4.1 Joint Position Detection

Angular position sensing must be used to determine the joint angle of the motor. There are several commonly used methods of determining angular position, including potentiometers, optical encoders, and hall effect sensors [6–8]. A comparison of the different angular sensors can be seen in Table 2.

Potentiometers are very commonly used to measure angular position due to their simple implementation and low cost. In addition to being low cost, potentiometers provide high linearity and accuracy [8]. Although generall robust, these sensors do not lend themselves well to many, rapid adjustments or mechanical vibrations. Both of these significantly reduce the lifespan of the sensor [6,8]. The situations potentiometers excel in are those that require an easily adjustble voltage at low to medium adjustment frequencies, such as settings nobs on control panels or analog reference voltages as trim potentiometers [6].

Hall Effect sensors are less commonly used, and consist of a bipolar magnet rotating above a hall effect sensor with the axis of rotation perpendicular to the plane of the sensor. Since there is no contact between the rotation and the sensor, these types of sensors have very long lifespans [6]. Unfortunately, these sensors do not provide high resolution since they are susceptible to electromagnetic interference and tempterature, and also have some hysteresis [8].

Optical encoders are another method of measuring angular position. These sensors consist of a beam of light that shines on a slotted disk so that as the disk rotates, the slots break the light beam. These sensors can have very high resolutions and are resistant to shock and vibrations [7]. Like magnetic sensors, these sensors have very long lifespans since there is no mechanical connection on the sensor [6]. Unfortunately, these sensors are susceptible to foreign particles blocking the light beam from sensing the slots and causing incorrect readings. The most common kind of optical encoder, the Quadrature encoder, does not sense absolute position; it can only read relative position, meaning that a quadrature encoder would need to be combined with some other sensor in order for the robot to be able to sense its joint angles correctly. Other encoders called Absolute Encoders do not have trouble reading absolute position, but they are prohibitively expensive. [8].

Sensor	Cost	Linearity	Accuracy	Lifespan	Notes
Potentiometer	$	Depends on ADC	Moderate	Short	Repeated motion at the same angle can lead to failure
Encoder	$$$	Very High	Very High	Long	Cheap ones can't sense absolute position
Hall Effect Sensor	$$	High	High	Very Long	Requires special attention to surrounding magnetic fields when mounting

Table 2: Comparison of different angular position sensors

2.4.2 Current Sensing

Current sensing can be done in many ways. The most common way is by using a shunt resistor and an amplifier. A variant of this method is to use the resistance inherent in the wires or traces as a shunt resistor. Another common method of current sensing is to use a hall effect sensor [9].

Shunt resistors are used in either high side or low side configuration. They are simple to integrate, low cost, and capable of measuring both AC and DC currents. The downsides to this method are relatively large insertion loss that increase exponentially with current, large thermal drift that must be compensated for, as well as large system noise from am-

plification. There are two main implementations of shunt resistors, high side and low side [9].

Low side current sensing means that the shunt resistor is placed in the return current path. This method is simpler to implement since the voltage on the shunt resistor is with respect to ground, so it can simply be amplified. Some problems exist with this, however, since the resistor seperates the current path from ground. In this configuration, the circuitry used to measure the voltage on the shunt resistor will not report a fault if the sytem experiences a short circuit [9].

High side current sensing means that the shunt resistor is placed on the forward current path. This configuration is able to detect short circuit faults, an advantage to using this configuration over low side current sensing. An additional advantage is that the return current path is directly connected to ground. The downside to high side current sensing is that it requires a differential amplifier since the voltage across the shunt resistor is very close to supply voltage. [9].

Trace resistance sensing is very similar to using a shunt resistor, but there are some slight differences. Since there isn't a way to control the resistance of a copper trace, the system must be calibrated after being assembled. Another key difference is the amount of amplification needed. Copper traces have very low inherent resistance, so a very large amplification must be used. This large gain imposes a limitation on the maximum measurable bandwidth set by the gain bandwidth product of the amplifier [9].

Hall effect sensors are commonly used to measure current as well. These sensors can measure current intrusively or non-intrusively, as well as in open loop or closed loop configurations. Non-intrusive devices measure current by wrapping wire around a toroid that focuses the magnetic field on a sensor in a break in the ring of the toroid, or placing the hall effect sensor on top of the current to be measured. These work fairly well, but are very susceptible to noise from magnetic fields upwards of 10cm away. Methods of shielding these sensors exist, but are complicated and expensive to implement. Intrusive sensors route current through the device and measure the generated magnetic field with a hall effect device near the current path. Open loop applications take the voltage generated on the hall effect sensor and condition it to whatever output is needed. Closed loop sensors reroute the sensed current to a secondary coil that is used to generate a proportional current to the measured current. This proportional current is then used as feedback to reduce error [9].

Insertion loss caused by these sensors is very small. Since these sensors measure current by induction, they can only measure current in a specific frequency band, and high currents at high frequencies can cause these devices to overheat. Most of these frequencies are DC to some upper limit determined by the physical characteristics of the sensor, usually around 100kHz. These sensors cannot be used on their own, since they have an inherent voltage offset, called misalignment voltage, and suffer from high thermal drift. Integrated ICs that compensate for these factors are fairly widespread, allowing for very easy integration [9].

2.4.3 Off-Board Communication

There are many types of communication protocols that could be used to communicate with the main controller. Common protocols include SPI, I^2C, RS232, RS485, and CAN. Of these, SPI and I^2C are meant mostly for chip to chip communication while RS232, RS485, and CAN are all meant for module to module communication [10]. A comparison of these protocols can be seen in Table 3.

SPI is a full duplex, synchronous serial link consisting of 3 lines, SCLK, MOSI, MISO, and an additional line for every peripheral, CS. Data rates of up to 10MHz or more are possible due to the elimination of addressing with the CS lines and dedicated clock line [10]. Using SPI for controller-to-controller communication presents a problem, however. Since the data transfer rate is controller by the master, the slave could fall behind on processing data. This can be avoided by only transmitting data one direction at a time. Typically, SPI is limited to onboard communications since its signal degrades fairly quickly over distance [11].

I^2C is a half duplex, synchronous, multi-master bus consisting of a clock and data line. Data rates of up to 3.4MHz can be reached, and each device has a unique address or multiple addresses to avoid overlap. An interesting aspect of I^2C is clock stretching. Clock stretching is when a slave pulls the clock low to stall the master until it has enough time to process information. Typically, I^2C is limited to onboard communication since its signal degrades fairly quickly over distance [10].

RS232 is a common full duplex interface that consists of two transmitter/receiver pairs. The protocol limits communication to 1 sender and 1 receiver per line. Data rates of up to 115.2KHz are possible at a range of up to 200ft. Data is typically sent in 8N1 format with 8 data bits, no parity bit, and 1 stop bit or 7E1 format with 7 data bits, even parity bit, and 1 stop bit [10].

RS485 is a full duplex multi-master protocol that consists of up to 32 transceivers on the bus. Data transmission rates of up to 10Mbps and distances of up to 4000ft are possible. Transmission can be reduced to half duplex by removing one transceiver at each node. Data is sent much the same as in RS232 with either 8N1 or 7E1 being common formats [10].

CAN is a half duplex multi-master bus protocol that allows for many nodes to connect and send data on the two transmission lines. Messages are sent with unique addresses that also act as arbitration for bus priority. Packets are fully defined with 11 or 29 bit addresses, 0-8 bytes of data, and some additional control and verification bits [12, 13]. Data rates of up to 1MHz and distances of up to 3000ft are possible. Multiple error checks are implemented at the hardware level since packets are predefined, allowing the controller to load a transmit buffer and let the transceiver send a message or wait until a receive buffer is full before reading the message [11].
HID (Human Interface Device) is a communications protocol that defines two entities: the host and the device. It works by having devices define a data packet and a HID descriptor

for the specific device. The host can then receive interrupts from the device during which the pre-defined data packet is transmitted from device to host.

Protocol	Max Distance	Max Speed	Wires needed	Notes
SPI	Within circuit board	10MHz	SCLK, MOSI, MISO, + 1 CS for each node	No addresses needed
I^2C	Within circuit board	3.4MHz	2	Address included in message
RS232	200 feet	115.2KHz	4	Can include parity bit
RS485	4000 feet	10Mbps	4	Can transmit fast or far but not at same time
CAN	3000 feet	1MHz	2	Resilient signal

Table 3: Comparison of off-board communication protocol performance

3 Description of Work

There are many different ways to accomplish the goals we set. We decided to discretize common robot arm configurations into smart joints and their connectors, a base, and end of arm tooling. Each joint will have a motor/servo actuating it and its own control board with various sensors used to control the actuation. We also plan to control the end of arm tooling using another joint control board.

Each joint control board allows joints to connect to the main communications bus running through the system and control the motor/servo on the joint. This board will have all of the necessary components for controlling one motor and communicating to the main controller. The main controller will act as an interface between the computer that is performing all of the complex calculations and the Joints that are controlling their positions.

To exemplify the modularity of our controls system and the interchangeability of our arm we are going to be adding a link onto the existing 3001 arm. By doing this, we will show how our control system can adapt to a variable arm configurations. By taking an already existing arm and modifying it, then applying our control system, we prove that our system can be used to modify most arms that conform to the specs of our system.

We will develop a software application for controlling the arm with a GUI. This software will have a simple control interface for moving the arm, and some configurable settings to act as inputs for the kinematics equations. In addition to this we will develop a code library which will serve as an interface for the end user for use in their own code.

In order to verify that each of these systems is functioning as expected, we will also design a series of tests to perform on them. Tests will be performed and documented for each subsystem and the entire system as a whole.

4 Methodology

4.1 Kit Components

We decided to break a robot arm down into its component parts. We came up with the main parts of our kit: sticks, joints, a base, and end of arm tools. This breakdown was to try to maximize modularity while keeping the pieces relatively simple. By separating the joint from the stick, we can have multiple sticks, which are easy to manufacture, of different types and lengths to offset a few types of joints, which are difficult to manufacture. The base is necessary to send and receive computation control from a computer. Multiple end of arm tools are needed to provide functionality to the arm besides movement.

4.2 Connectors

The connectors are vitally important to the functionality of this project. A good connector will need to make solid mechanical and electrical connection between parts while also providing the ability to quickly connect and disconnect parts. In addition, the type of connection we choose will affect the modularity of the system as a whole. The important things to note while deciding on criteria for the connector are how/if they will be keyed, how they will pass electrical signals, where exactly the connector will be on the joints, and how the connector will be secured.

4.2.1 Connector Position on Joints

Connector position is the first major design choice we had to make. They can be positioned either on the axis of rotation of the joint or off the axis of rotation of the joint, and selecting one method versus the other vastly changes the way that the connector would work. Putting the connector ON the axis of rotation means that the connector would connect to the joint

axis directly, while putting it OFF the axis of rotation means the connector would connect to a piece that is connected to the joint axis.

The advantage that placing the connector off the axis of rotation has over placing the connector on the axis of rotation is that the joint will be a solid unit. Having the joint be a solid unit seems a better design choice than splitting it in half, so we decided to go with putting connectors off the axis of rotation.

4.2.2 Securing the Connection

One of the important aspects of a modular system is how easy it is to connect or disconnect parts to or from that system. The main options for quick connections are requiring no additional hardware, requiring a single screw, and requiring slots and pins.

The most obvious solution is to connect parts with no additional hardware required. This creates a very complicated design challenge since using no tools means the user would have to secure any connection with just their hands. This can weaken the joint mechanically. The advantage to this method has is that it is fairly quick.

A step down in the simplicity solution is to require a single screw to join two pieces together. This is still simple and pieces can be connected somewhat quickly, but does require a tool to connect pieces. The main advantage is that screws hold parts together very well and the mechanical integrity of the connection should be held.

Another option is to design the joints and sticks in such a way that they slot together and are held in place with pins. This requires additional hardware, but no tools. This should keep pieces together fairly well while still allowing connections to be made quickly and easily.

4.2.3 Keying the Connector

Keying the mechanical connection between joints, sticks, and the base changes how modular the system is overall as well as how many unique components will be needed in each kit. Not keying the connection is not an option since this would allow the user to connect the pieces together in any orientation and the orientation needs to be known in order to accurately control the arm. This leaves two main options for the connections: keying for 1 orientation and keying for 4 orientations.

Keying the connectors for 1 orientation means that three different kinds of revolute joint must be design to fully represent the ways a revolute joint can move in 3D space. Essentially, this would mean each different joint would rotate about a different axis relative to the connector axis. The modularity of the kit is impacted quite negatively by doing this, since each joint can only connect in one way and therefore cannot be used where another type of rotation is needed. This design is quite simple, however, since the connectors don't need to be

rotationally symmetric about any axis.

Keying the connectors for 4 orientations presents a slightly more challenging design problem, however. The connectors would need to be evenly rotationally symmetric 4 ways about the axis of connection in order for this design to work. 4-way keyed connectors will bring the number of unique joints down from 3 to 2. Doing this does help with the modularity of the design, though, since the rotational joint can be implemented to rotate in either axis perpendicular to the connector axis. A disadvantage of this configuration is an increase in complexity.

Since additional complexity when designing is less important that the overall modularity, we decided to go with a 4-position keyed connection. This allows a single rotational joint and only 1 right angle stick design so that the user can construct many different kinds of arms from these simple parts.

4.2.4 Passing Signals

Connectors also need to pass the power and signal buses through from joint to stick or joint. This can be done in one of a few ways, including: rigid mechanical connectors, loose wires running along the outside of parts, wires running inside of parts, and wires connecting internal bus bars.

Rigid mechanical connectors for passing signals would make connection when parts are connected together. These connections would have to be evenly rotationally symmetric 4 ways about the axis of rotation since the mechanical connectors are. A disadvantage of these connectors is that they rely on the integrity of the mechanical connection to pass electrical signals properly. If the connection flexes or bends too much then the electrical connection could break even though the mechanical connection is still mostly intact. Another disadvantage is that this is the most costly option for passing electrical signals, requiring 4 connectors per connection.

Loose wires along the outside of the parts have several advantages over rigid mechanical connectors. The first of which is that they only require one set of connections per connector. This reduces the cost of each connector significantly. The main disadvantage of this kind of electrical connection is that the wires could get snagged on something since the system is supposed to be active and moving. Another disadvantage is that the wires need to have enough slack to move with the arm without limiting the arm's movement.

Wires running through the parts that pop out at the connectors is another option or passing signals along the system. This option is practically the same cost as external wires, but doesn't have the problem of wires snagging on the environment. Unfortunately this doesn't solve the problem of wires needing lots of slack to allow for movement of the whole system.

Short wires that connect some internal bus bars provide a more expensive solution to this

problem. This would remove the problem of wires needing slack for the entire system. Instead, wires would only have enough slack for one joint. Doing this does bring some complexity issues, however, since the bars would have to be designed into the system and not added on at the end.

For our design we decided to use internal wires running the length of the system. The low cost and simplicity of this solution outweighs the negatives of having to add lots of extra wire to account for movement of the system.

4.3 Sticks

Sticks are the things that connect joints together and space joints out. They do not have any electronics on board; they simply pass power and communication wires along to the rest of the arm. They need to be strong, light weight, and cheap.

Sticks have an input side and an output side. Two kinds of sticks will need to be created: one will be straight, and one will have a right angle at the input side.

Stick Material	Cost/kg	Cost/ 20mm	Rigidity	Complexity	Weight
3D-printed PLA[1]	$20/kg	$3	Might break	Low: Very few constraints on possible designs	150g [1]
PVC[2]	$7	$0.70	Bends over time	Connect/ Disconnect easily	106g
Carbon Fiber[3]	$66	$6	Strong, but possibly too thin		58g
80/20[4]	$2.46	$2	Not going anywhere	Nice connecting options	154g

Table 4: Comparison of materials to construct sticks

We choose to use 3D-printed PLA. While it's not the absolute cheapest option, nor is it the lightest one, its high reconfigurability makes it the ideal material for our needs - especially given its availability for potential customers; anybody with a 3D-printer would be able to make one of our arms. Also, while aesthetic concerns should not be the only factor, we're allowed to consider the way the final product would look. An arm made from PVC would reflect poorly on all the involved parties.

4.4 Motor Selection

Motor with Encoder	Voltage(V)	Speed No Load (rpm)	Current No Load (A)	Stall Current (A)	Stall Torque (oz-in)	Gear Ratio	Weight (oz)	Price ($)
Actobotics Planetary Gear Motor	12	26	0.21	4.9	583	455:1	???	49.99
Lynxmotion Brushed DC Motor	12	10.5	0.22	2.5	500	264:1	???	55.76
HD Premium Planetary Gearmotor	12	313	0.52	20	416.6	27:1	11.64	59.99
HD Premium Planetary Gearmotor (2)	12	12	0.54	20	8110.2	721:1	13.4	59.99
HD Premium Planetary Gearmotor (3)	12	84	0.53	20	1347.1	100:1	???	59.99
Motor Without Encoder								
Precision Planetary Gearmotor	12	26	0.21	4.9	583	455:1	3.53	27.99
Premium Planetary Gearmotor	12	32	0.21	4.9	472.1	370:1	3.53	27.99

Table 5: Comparison of Possible Motors

To choose our motors, we looked for high-torque, low-cost DC motors. We chose to go with DC Brushed motors to control our arm because they are quiet, low-cost, vibration free and fairly efficient. We also considered using Dc Brushless motors as well as Stepper Motors, but each had their own pros and cons. Brushless motors cost much more than comparable brushed motors, and require complicated control logic to operate. Stepper motors were a good option due to their ability to be backdriven and their built in discrete steps for controlling. But, they do not operate well under conditions where the load changes significantly in a short period of time and also require external control to keep track of the position.

Once we decided to use brushed DC motors, our next step was to find suitable motors that fit our criteria of high torque and low cost. We found 2 categories of motors that seemed to fill these requirements, planetary gearbox motors and spur gearbox motors. Planetary gearboxes work by having multiple "planet" gears revolving around a central "sun" gear that rotates in place. They are named for their resemblance of the planets orbiting around the sun. All of the "planet" gears are held in place by an outer "ring" gear that acts to keep the "planet" gears in contact with both the "sun" and the "ring". By having these idler gears rotating around a central axis, you can have torque transferred linearly, without the

[1]This number assumes 100% infill. The actual number will almost certainly be lower.

[2]http://www.homedepot.com/p/Formufit-1-in-x-5-ft-Furniture-Grade-Sch-40-PVC-Pipe-in-White-P001FGP-WH 205171542?cm_mmc=Shopping%7cTHD%7cG%7c0%7cG-BASE-PLA-D26P-Plumbing%7c&gclid= CjOKCQjwx8fOBRD7ARIsAPVq-Nlw_xbuCOf-QHORvUW4gQ4Dx7SiZt_vqQ3OvxBdTW-eckQhdp5WWFYaAs9DEALw_ wcB&gclsrc=aw.ds&dclid=CIagtKLB09YCFUuraQodN_MAOQ

[3]https://www.rockwestcomposites.com/45552?gclid=CjOKCQjwx8fOBRD7ARIsAPVq-NmCNUg6ULxgd9udG-xSuPtJuHKgO zXPDgRr2CmKUOtSXX-waAgb9EALw_wcB

[4]https://8020.net/1010.html

need for offset shafts, greatly reducing the total size of the gearbox. With multiple gears transferring the torque load at one time, the individual load on each tooth is lowered making these perfect for high torque applications. Spur gearboxes on the other hand use linear offset shafts that transfer the entire torque from one gear to the next until the output shaft in a direct chain. This means that they wear out much faster since the torque load is much higher on individual gears and teeth. Therefore, since we need a reliable high torque motor, we decided to go with planetary gear motors.

Once we had made the decision to go with a planetary gearbox brushed DC motor, we made a chart as seen in Table 5 of possible motors that had high stall torques. One final decision that we had to make was whether or not to purchase a motor with a rotary shaft encoder. Rotary shaft encoders provide easy control over DC motors by relaying the position of the shaft before the gearbox on the motor. This allows for high resolution control in the case of high gear reductions but also costs a fair amount extra to purchase with the motor. Considering that the motor is just a part of the joint and we care more about the position of the overall joint rather than the motor itself, we decided to save the money and go with a cheaper non-encoder motor. By doing this we are moving the point at which we control the joint system from the motor to the joint if we use an absolute encoder on the joint shaft. This results in a closed loop control system which is optimal for our situation and cost-effective.

4.5 Joint Control Board Part selection

Selecting the types of sensors to use for the control board was a very important step of the control board design. There are many different types of sensors to accomplish each major goal that the control board must accomplish.

4.5.1 Joint Angle Sensor

Potentiometers seem like a good choice due to their simplicity and high accuracy capabilities. However, they do not lend themselves well to this application because of how quickly they wear out. Over time, as the joints move to different positions, the potentiometers will wear out quickly and cause inaccurate readings. Additionally, long lifespan and high resolution potentiometers can be very expensive. Furthermore, potentiometers are large and can be difficult to mount. Finally, the hard stop on the potentiometer means the joint angles will be limited to a certain range (typically about 270 °for single turn potentiometers).

The next obvious solution is to use optical encoders because they will not wear out and offer very high resolution capabilities. These sensors are not well suited for this application, however, since they are typically expensive, especially for high resolution encoders - and ones that are capable of reading absolute position. Additionally these sensors are somewhat bulky and would take up too much space in the closed environment of a joint.

This leaves us with hall effect sensors. These sensors are very small and moderately high resolution while also being a contact-free sensor, so wearing them out will not be a concern. A main concern with hall effect sensors is that they need to be mounted somewhat precisely and carefully. Traditional machining methods make this difficult to accomplish, but 3D printing allows us to easily overcome this challenge. Another concern is external electromagnetic interference, but with somewhat careful circuit board design, we should be able to minimize this issue.

4.5.2 Motor Current Sensor

A shunt resistor seems practical due to the simplicity of the design, but careful designing must be done in order to get the noise levels down to a reasonable amount. In addition to this, the power loss when using a shunt resistor could cause the arm to stall before anticipated. When the shunt resistor takes power from the motor, the whole motor curve slides inward, decreasing the maximum power output. Trace resistance would be a good alternative, but requires calibration after the circuit is constructed.

Instead of these, we decided to use a hall effect current sensor. Hall effect current sensors are ready-made sensors that give low noise, properly calibrated outputs, are not very expensive, and are easy to integrate into a circuit design. These sensors have extremely small power losses to the motor. The main drawback of these sensors is that they have a low bandwidth, but we are using DC motors so this should not be a problem. Some care will need to be taken when placing these on the circuit, however, since they are sensitive to external magnetic fields.

4.5.3 Off-board Communication

SPI and I^2C are mostly used for on-board, controller-to-peripheral communications and therefore are not a good choice for the base to control board communication. RS232 is not a good solution for this problem either because it is a single transmitter and single receiver per line. This leaves RS485 and CAN.

RS485 and CAN are similar in many ways, but with a few key differences that separate them. RS485 is very fast to transmit and simple to implement, but takes a lot of the controller's time to send packets. CAN has the advantage because the controller and transceiver control the transmission independent of the controller so the controller has more free time to process data. Another advantage CAN has over RS485 is the amount of error checking that goes on to ensure proper message transmission. For these reasons, we decided to use CAN to communicate between the base and control boards.

5 Joint Control Board

5.1 Motor Driver

The DC motor driver selected was the DRV8872. This driver takes in two inputs, in1 and in2, which affect the output much like inputs to an H-bridge.The exception is when both inputs are high. In this case, the inputs are pulled together as a motor break. A truth table can be seen in Table 6.

Table 6: DRV8872 truth table

IN1	IN2	OUT1	OUT2	Description
0	0	Z	Z	Coast
0	1	L	H	Reverse
1	0	H	L	Forward
1	1	L	L	Brake

To measure the speed of the motor, a servo horn with 6 spokes was attached and a beam break sensor was mounted on the motor with the spoke traveling through the beam. The signal line of the beam break sensor was connected to an Arduino that measured the frequency by incrementing a count in an interrupt triggered on a pin change. The ISR just incremented a count that was printed out and reset every 5 seconds. Some issues arose with this system, however, since there was a small amount of bouncing on the rising and falling edges, leading to multiple readings for each beam break. This was solved by placing a 10nF capacitor from the signal line to ground. Since this number was triggered 12 times per revolution and printed out every 5 seconds, the actual printed value happened to be in revolutions per minute, as shown in 1.

$$rpm = \frac{1rev}{12ticks} * \frac{1}{5} * \frac{60seconds}{1minute} \tag{1}$$

During initial testing, the motor driver was wired up with V_m of 8.4V, logic voltage of 5V, a 10kΩ pull up resistor on nFault, Isen grounded, and the motor outputs connected to a DC motor. During this test, one of the inputs was connected to an Arduino Uno, outputting a constant PWM wave using the `analogWrite()` function with a duty cycle of approximately 50%. The motor turned, but very slowly and with a high pitched whine. When a 100μF capacitor was placed from V_m to ground, the motor spun up to full speed and the whining sound went away.

Further testing revealed a strange behavior when increasing the frequency of the input PWM signal. The motor spun normally at low frequencies of around 500Hz, but at around 1kHz the motor started slowing down and making a whining sound. The problem worsened with increasing frequency. Eventually, this problem was fixed by using a power

supply that could output 3A and adding capacitors from in1 and in2 to ground. With these additions, the motor driver functioned as expected.

5.2 DC Motor

5.3 Demultiplexer

The original demultiplexer selected for the joint board (SN74LVC1G19) did not output the correct values to drive the motor driver (DRV8872). The demultiplexer output can be seen in Table and the motor driver inputs can be seen in Table . When the EN pin was pulled high, both outputs would also be driven high. This effect is undesirable since, when given a PWM signal, this would cause the motor driver to turn then brake then turn again as opposed to the desired turn then coast then turn. To solve this problem, a different chip (SN74LVC1G18) was selected. The truth table for this chip can be seen in Table 7.

Table 7: SN74LVC1G18 truth table

Inputs		Outputs	
S	A	Y0	Y1
0	0	L	Z
0	1	H	Z
1	0	Z	L
1	1	Z	H

5.4 INA332

The INA332 instrumentation amplifier is used to measure the force applied to a load cell. The amplification of this amplifier is given from the datasheet as Equation 2. Since the load cell needed an amplification of at least 100, the calculated resistances were $R_1 = 10\text{k}\Omega$ and $R_2 = 195\text{k}\Omega$. The actual values selected for R_1 and R_2 were 10kΩ and 200kΩ, respectively. This gives the amplifier an expected gain of 105 V/V.

$$G = 5 + 5(\frac{R_1}{R_2}) \tag{2}$$

The INA332 needs a voltage reference to use as the 0V differential output. Initially, two 1kΩ resistors were used to apply this voltage. This caused the output to change non-linearly with the input voltage. The resistors were replaced with a LM358 dual operational amplifier, configured as a voltage buffer with the input connected to a potentiometer. The INA332 inputs were connected to ground and the LM358 buffer potentiometer was adjusted

to set the 0V output to $\frac{1}{2}V_{cc}$. This voltage was 1.846V. Instead of a potentiometer, two resistors were used to create this 1.846V offset.

5.5 TM4C123GXL Launchpad

Several peripherals were needed to achieve the desired functionality from our microcontroller. A test board was set up in order to test and verify that each of these peripherals was setup properly and working as expected. The test board consisted of a potentiometer connected to an ADC pin, an SPI controlled ADC (MCP3202), the 1:2 demultiplexer (SN74LVC1G18), CAN transceiver (TC332), and some LEDs.

As a temporary stand in for the AS5055 absolute hall effect encoder to test the SSI peripheral, a MCP3202 12-bit, 2 channel ADC was used. Both devices use SPI to communicate their sensor data back to the MCU, and the packets are similar in structure. Some differences between the two that can be changed are a maximum sample rate for the AS5055 of 1ms as opposed to the few SCLK cycle delays for the MCP3202. The AS5055 has a maximum SCLK frequency of up to 10MHz at 3.3V while the MCP3202 has a limit of 900kHz at 3.3V.

The potentiometer was connected to PB? which was enabled at AIN3. The ADC was set to sample at 1kHz with hardware oversampling 16x enabled.

5.6 Hall Effect Encoder

5.7 CAN Bus

The preliminary CAN setup consisted of two TM4C Launchpads, one with transmit code and one with receive code.

5.8 Arm Structure

Arm structure is not something we wanted to define, since the end user is supposed to create their own arms, but there were some basic things we needed to define. The first of these is that every arm must begin with a base module and have some combination of up to four additional joints connected. This allows the end-user flexibility in how they want to construct the arm without allowing them to add too many joints.

5.9 Fourth Joint

In order to show that our control system can be integrated with other arms, we designed a fourth link to connect to an existing arm. By showing that we can add a newly designed link to our existing arm, we are creating a proof-of-concept that shows the versatility of our control system. The design of the fourth link was done in Solidworks, a software that allows users to design objects in a 3-D space. Once we had designed the fourth link, we moved forward to the rapid-prototyping stage where we took our parts designed in Solidworks and converted them to 3-D printable models. We then used Cura and the Lulzbot Taz 3-D printer to fabricate a first iteration of our fourth link. From here we continued to refine our parts and re-print pieces with tighter tolerances until they came together to make a fourth joint that interfaces with our control system.

5.9.1 Fourth Joint Design

Our first decision for creating the fourth joint was to create a joint that is actuated using a brushed DC motor to prove that we can control DC motors in addition to the servos that already exist on the arm. Next, we decided that since most of the joints currently on the arm provide rotation perpendicular to the z-axis of the actuator, we wanted a joint that provides parallel rotation. To accomplish this we removed the control logic from a servo that already was used on the arm, converting it into a brushed DC motor. Next, we designed a direct-drive mount for the motor we fabricated so that the output shaft would rotate along the motor's axis of rotation. We built a mounting for the motor and a structure to provide support for both radial and axial load on the shaft. This structure made use of multiple radial bearings, placed to keep the shaft in line and eliminate any friction between moving and static pieces of the joint. We also included a thrust bearing on the main shaft in order to stop thrust loads from being placed directly on the servo. Finally, we designed an idler-shaft that sits in two bearings so that it can free rotate with negligible friction. The purpose of this idler shaft is to rotate in a 1:1 gear ratio with the main shaft using a timing belt to connect the two shafts. At the bottom of the idler shaft there is a magnet whose changing magnetic field is read by a hall effect sensor mounted onto the bottom of the link so that it sits at a fixed distance from the magnet.

5.9.2 Fourth Joint Prototyping

The process of prototyping our fourth joint started with creating an interface so that we can connect it to the already existing arm. We decided for the sake of simplicity to attach our fourth joint where the current end-of-arm-tooling would normally connect. MOVE UP

To create a prototype of our fourth joint, we determined the mechanical requirements of our arm and worked backwards to create a rapid prototyping model (RPM). RPMs are usually CAD models which are able to be turned into a functional model using a 3-D

printer or other method. Once the functional model was 3-D printed, we would assemble the parts and test how they all fit together. With each new functional model we revised our RPM and printed a new functional model in order to meet the requirements for our fourth joint.

5.9.3 3-D Printing

taz6

5.10 Arm Base

5.11 End-of-Arm Tooling

5.12 Maven

Maven is a utility for Java-based projects that seeks to provide a uniform build system for the project. It accomplishes this by defining a project object model and a set of plugins that each Maven project shares. Therefore, Maven can provide a streamlined build environment for every instance of the project, allowing users to have the same build process across multiple different devices and environments. Maven could be compared to a flexible template for how a project should be arranged and what files should be included. We chose Maven for our Java project because it makes it much easier for our team to collaborate on the front-end side of the code. It also allows us to package into our program libraries that we used in our project so that there are fewer dependencies that the end-user must download in order to use our software.

6 Capstone Design Requirements

Our project will fulfill the Robotics capstone design requirement by touching on all of the major cornerstones of robotic technology. Mechanical Engineering will be fulfilled through designing a new link for the 3001 arm and developing the software required to control that link. We will need to use methods such as DH parameters, inverse and forward kinematics, and motor torque requirements to ensure the mechanical ability of this arm and control it. Dynamic control algorithms will be explored. We will also need to use computer programs such as ROS and Code Composer Studio in order to control the arm. Additionally we will design a software package and create an interactive GUI so that end-users are able to easily interact with the arm. We will need to write lower level embedded code in order to control the electronics on the master controller and each link's microcontroller. With all of

the electrical component design in addition to the communications protocols we cover most topics in microelectronic circuit design and implementations. Finally, Systems engineering will be used for the designing the electrical, mechanical and computer science portions of the project. We will need to use systems engineering skills that we have learned through our various robotics projects in order to complete the project.

6.1 Computer Science

In order to properly communicate with the arm, we will need to implement communication protocols. The communications will need to be efficient. The problem of sending information to each of the links of the Robotic Arm will rely heavily on Information Theory.

In addition to communications, we will be developing a set of programs to interface with the arm. The suite will include a library to offload the computationally heavy task of solving kinematic equations to direct the motion of the arm. It will also include an application so the user can interact with the functions inside the mathematics library. This application will have a GUI so the user can specify the arm's configuration. It will also provide the user with some basic control functions, such as jogging the arm's position (either joint-by-joint or linearly with respect to the base coordinate frame) and opening/closing the gripper. Another feature will be the ability to record poses for the arm to go to.

As the arm is designed with modularity in mind, the software will need to be dynamic enough to leave the user with plenty of room for configuration. One of the deliverables of this project will be a set of files that can be included in other people's work so that anybody has access to code that they can use to control the robotic arm.

Finally, we'll need to write firmware that will run on each of the arm's joints, and libraries that people will be able to use to interact with those joints.

6.2 Electrical and Computer Engineering

The design and implementation of each joint's control board involves core topics from Electrical Engineering. The controller board designed for each link must be carefully designed and tested rigorously. Electrical systems design skills will be used to select compatible parts to produce the desired functionality.

Once constructed, the controller board will need embedded code to control the motor. This code will need to be tested and optimized for better performance. Some communications protocols will be used to send data from chip to chip and board to board.

7 Constraints

Budget will likely be the biggest constraint with this project. The stipend given to students by WPI may not cover all of the costs incurred when constructing this robot, and the rest will be paid out of pocket by the student team. Another large constraint will be access to a 3D printer for prototyping. It will be crucial to start prototyping early to accomplish all of the design goals. Time will also be against a major concern, since there are many time consuming aspects to this project.

8 Acceptance Criteria

Acceptance criteria for this project will be broken into 5 major categories: Joints, End effector, Base, Software application, Code library

8.1 Joint Board

- Receive initialization information and joint angles from base
- Moves joint to angles
- Send position updates back to base
- Pass power and signal buses
- Capable of powering logic without powering motors
- Control board is the same for each joint

8.2 Modify RBE3001 Arm

- Remove control system and replace with our own
- Add a link to the existing arm
- Replace currently implemented servo motors with brushed DC motors

8.3 End Effector

- Receives power and signal buses
- Keyed connection

- One input connector
- Terminate CAN bus
- Uses a joint board

8.4 Base

- Sends and receives joint angles to/from Personal Computer (PC)
- Receives initialization information from PC, then sends it to all joints on signal bus
- Outputs power and signal buses
- Converts AC wall power to system power bus
- Power supply and arm on/off switch
- Capable of powering logic without powering motors
- Array of indicator LEDs

8.5 Software Application

- Sends configuration information to the Code Library
- Sends individual joint angles or pose commands to robot through the Code Library
- GUI to adjust current arm configuration parameters
- Record and play back sequence of poses
- Acts as a front-end for code library
- Stretch goal: 3D model of arm moving in real-time

8.6 Code Library

- Receive configuration information from user, selects control constants, sends to base
- Able to control the robot: Receive joint status, send joint angles
- Calculate joint angles using kinematics
- Stretch Goal: Written so that it can interface with multiple languages

9 Testing

9.1 Motor Driver

In order to determine whether the performance of the motor driver was dependent on input frequency or other factors, the input frequency was increase again from 200Hz to 10kHz. This time, the motor performed much better. The RPM actually increased with an increase in frequency, as can be seen in Table .

Table 8: PWM frequency input at 50% duty cycle vs motor speed

PWM Frequency (Hz)	Motor RPM (rpm)
200	80
400	79
1k	81
2k	82
5k	86
10k	90

9.2 DC Motor

9.3 Demultiplexer

9.4 INA332

In order to generate a small input voltage to the amplifier, a potentiometer was used with a larger resistor in series, creating a variable voltage divider. To create the small input voltage necessary, a 100kΩ resistor and a 1kΩ potentiometer were used. With the configuration given in 5.4,

9.5 TM4C123

Several peripherals were needed to achieve the desired functionality from our microcontroller. A test board was set up in order to test and verify that each of these peripherals was setup properly and working as expected. The test board consisted of a potentiometer connected to an ADC pin, an SPI controlled ADC (MCP3202), the 1:2 demultiplexer (SN74LVC1G18), CAN transceiver (TC332), and some LEDs.

As a temporary stand in for the AS5055 absolute hall effect encoder to test the SSI peripheral, a MCP3202 12-bit, 2 channel ADC was used. Both devices use SPI to communicate their sensor data back to the MCU, and the packets are similar in structure. Some differences between the two that can be changed are a maximum sample rate for the AS5055 of 1ms as opposed to the few SCLK cycle delays for the MCP3202. The AS5055 has a maximum SCLK frequency of up to 10MHz at 3.3V while the MCP3202 has a limit of 900kHz at 3.3V.

The potentiometer was connected to PB? which was enabled at AIN3. The ADC was set to sample at 1kHz with hardware oversampling 16x enabled.

9.6 Hall Effect Encoder

9.7 CAN Bus

Initial CAN testing consisted of

References

[1] . F. STU, "Robotick rameno abb irb 120."

[2] "Robotworx: Expert industrial robot integrator."

[3] "Reconfigurable modular manipulator (rmm)."

[4] D. Calzada-mariaca, M. Preston, and Y. Zhou, "Modular robotic arm," tech. rep., April 26, 2015.

[5] "robolink robot components— igus."

[6] P. Cain, "Pot vs. sensor," *Electronic Products*, pp. 44,46, 2010.

[7] "Choosing the right sensor technology."

[8] M. Howard, "Choosing the right position sensor," tech. rep., Zettlex.

[9] S. Ziegler, R. C. Woodward, H. H. C. Iu, and L. J. Borle, "Current sensing techniques: A review," *IEEE Sensors Journal*, vol. 9, no. 4, pp. 354–376, 2009.

[10] J. Patrick, "Serial protocols compared," *Embedded Systems Programming*, 2002.

[11] N. Murphy, "Can we talk?," *Embedded Systems Programming*, 2003.

[12] C. Watterson, "Controller area network (can) implementation guide," tech. rep., Analog Devices, February, 2012.

[13] S. Corrigan, "Controller area network physical layer requirements," tech. rep., Texas Instruments, January 2008.

Appendix A Kit Components

- 3x Twist Joints
- 3x Revolving Type A Joints
- 3x Revolving Type B Joints
- 3x 75mm Links
- 3x 150mm Links
- 3x 225mm Links
- 1x Claw Gripper End-of-Arm Tool
- 1x Hook End-of-Arm Tool
- 1x Capacitive Stylus/Pointer End-of-Arm Tool
- 1x Master Controller
- 1x Power Supply Unit
- Stretch Goals:
  - Prismatic Joint(s)
  - 1x Universal Gripper End-of-Arm Tool

Appendix B Gantt Chart