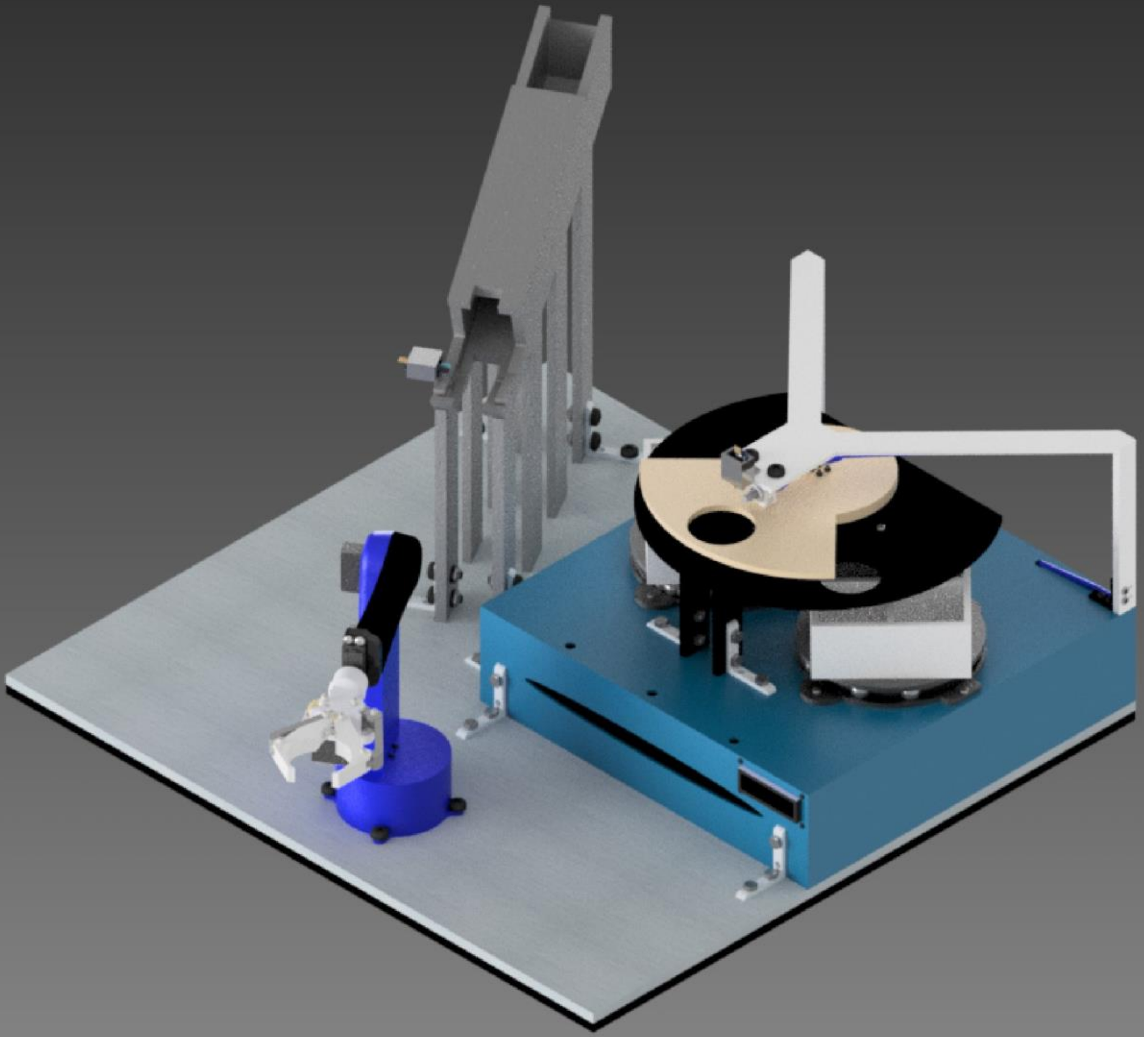
MCT331 Design of Mechatronics Systems I



Team 32

Amal Amr AbdelHamid Zewita	19P1504
Malak Tarek Mohamed Abdel-Wahed	19P1594
Marawan Abd El Rahman Ali Mohmoud Ashry	19P1447

Faculty of Engineering - Ain Shams University
جامعة عين شمس
1839
كلية الهندسة - جامعة عين شمس

Contents

Table of figures

ABSTRACT

A production line is a collection of coordinated procedures set up in a factory where parts are joined together to create finished goods or where raw materials are processed to create finished goods fit for consumption.

Usually, in order to make raw materials functional, such as metal ores, agricultural products like foodstuffs, or textile source plants like cotton and flax, a series of treatments is needed. The procedures for metal include crushing, smelting, and more refinement. For plants, the valuable material needs to be removed from the husks or other impurities before being processed for sale.

In our project, we will design a production line consists of three main stations: Feeding, Sorting and packaging. By the help of a 3-DOF robotic arm which contains motors and sensors, we will be able to do the feeding station so that the robotic arm can take our product from the magazine and put it in the sorting-by material station. By placing the Inductive Proximity Switch- metal sensor, we could successfully sort our products by material by detect if the product is metal or not. In addition, our rotary-designed sorting mechanism will be able to do the packaging process which is the final station in our production line.

A lot of motors, sensors, gears, bearings.. will be used in our unique production lin. So that we will briefly discuss each station by its sensors and electrical and mechanical components and how they will work efficiently. Most common electrical and mechanical components used are:

- Servo motors
- IR sensor
- Inductive proximity sensor
- Arduino UNO
- Thrust Bearings
- L supports

By the end of this project, a well-organized production line will successfully be able to do the main three stations that we have already mentioned above by the help of a robotic arm with three degrees of freedom. So, we will now discus each station and how it works with the aid of the a well-organized and accurate sketches and images of the mechanical design by also showing the electrical parts and the reason for each usage.

1.0 INTRODUCTION

We should comprehend the stages involved in creating a mechatronic system on both a micro and macro cycle level for this project, as well as how to incorporate product design criteria like utilizing materials, utilizing a human-machine interface, and using modular components The successful testing of the product in accordance with each station's specifications, as well as the stations' integration and safety features, will all be evaluated.

The Design of Mechatronics Systems standard VDI 2206 must be followed while creating a production line. The manufacturing line's primary function is to handle cylindrical or cuboid-shaped products and sort them according to their shape or substance before packing each sorted product category into its own packaging.

So, Our project is a production line that consists of THREE main stations: Feeding, Sorting and packaging. By the aid of motors and sensors, we will be able to do the feeding station so that the robotic arm can take our product from the magazine and put it in the rotary sorting-by material station.

Multiple programs or threads can run simultaneously on the majority of operating systems. This is referred to as multitasking. At any given time, each processor core can only have one program running on it. The scheduler, a component of the operating system, selects which program to run at what time and creates the appearance of simultaneous operation by swiftly switching between each program.

A Real Time Operating System (RTOSscheduler)'s is created to offer a predictable (often referred to as deterministic) pattern of execution. Given that embedded systems, like the Arduino devices, frequently have real-time requirements, this is especially intriguing for embedded systems.

Traditional real-time schedulers, like the one included in FreeRTOS, accomplish determinism by letting the user give each execution thread a priority. The priority is then used by the scheduler to decide which thread of execution should be executed next. So, our software will be done by RTOS.

Figure 1:Production line

2.0 PRODUCT JOURNEY

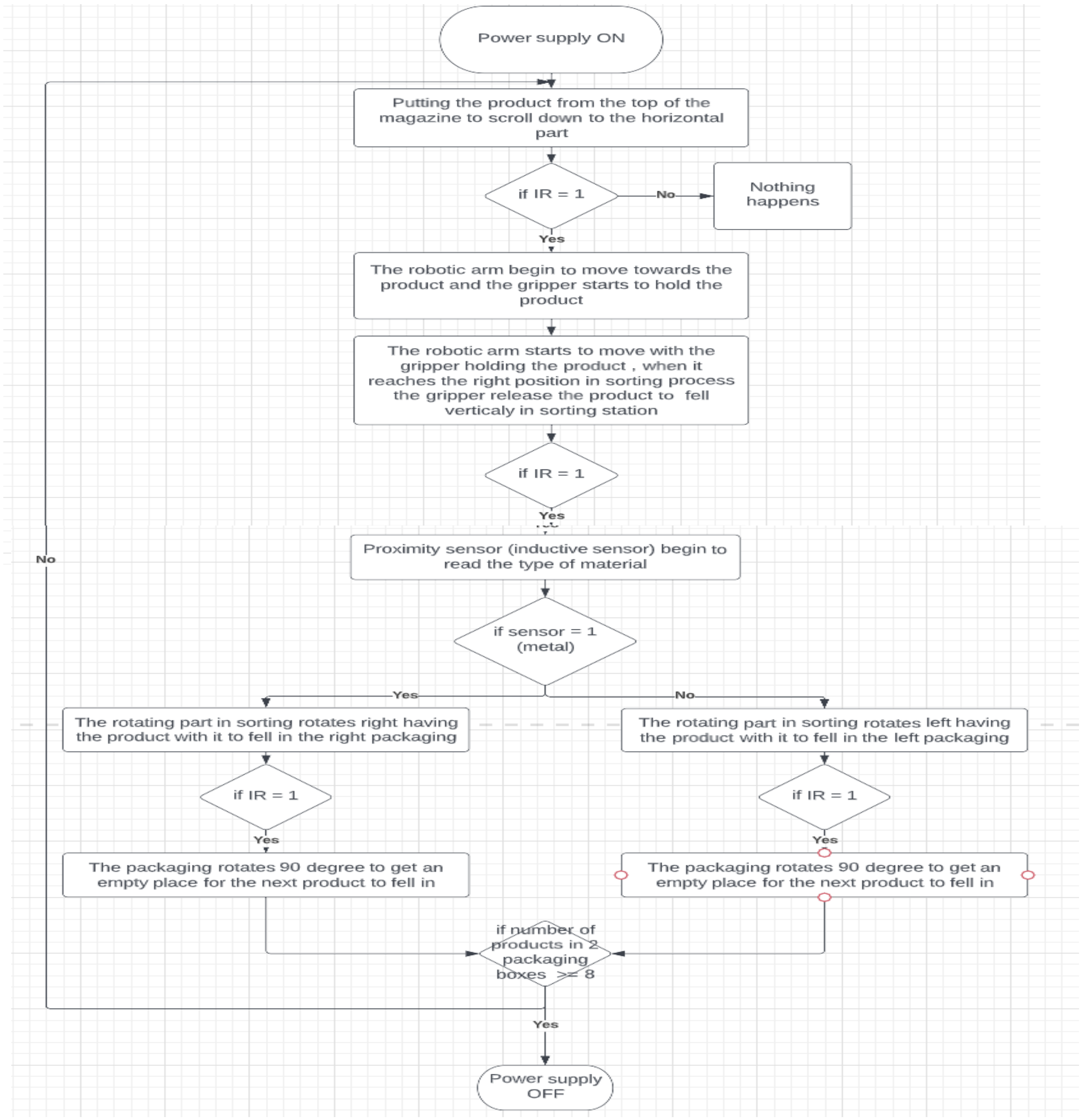

Power supply ON
Putting the product from the top of the magazine to scroll down to the horizontal part
if IR = 1
No
Nothing happens
Yes
The robotic arm begin to move towards the product and the gripper starts to hold the product
The robotic arm starts to move with the gripper holding the product , when it reaches the right position in sorting process the gripper release the product to fell verticaly in sorting station
if IR = 1
Yes
Proximity sensor (inductive sensor) begin to read the type of material
if sensor = 1 (metal)
Yes
No
The rotating part in sorting rotates right having the product with it to fell in the right packaging
The rotating part in sorting rotates left having the product with it to fell in the left packaging
if IR = 1
if IR = 1
Yes
Yes
The packaging rotates 90 degree to get an empty place for the next product to fell in
The packaging rotates 90 degree to get an empty place for the next product to fell in
if number of products in 2 packaging boxes >= 8
No
Yes
Power supply OFF

Figure 2:Product journey flow chart

3.0 FEEDING STATION

3.1 Mechanical System

The magazine is our pure mechanical feeding station, It is an inclined box so that our cylindrical product will be able to roll over until the product is stop by a vertical surface and it is also our feeding station. Having an Infra-Red sensor at the end, this allows the robotic arm to pick the product through the opened-front legs and move it the sorting station. The magazine has three legs on each side which are fixed by grooves in both sides with the magazine and also the base. For more fixation, we added 4 L-supports using M8 bolts and nuts , 2 in each side to surely fix the magazine in base. Material: Wooden. Our magazine will be manufactured by a laser cutting machine. But the final part that stops the products will be made by 3D printing.
The IR sensor is placed at the end of the magazine which the horizontal part that the product will stay at until the robotic arm comes and grips it. A wooden box is designed by a laser cutting machine to cover the IR sensor. A M3 bolt is used to fix the IR sensor in the box and the box itself to the magazine.

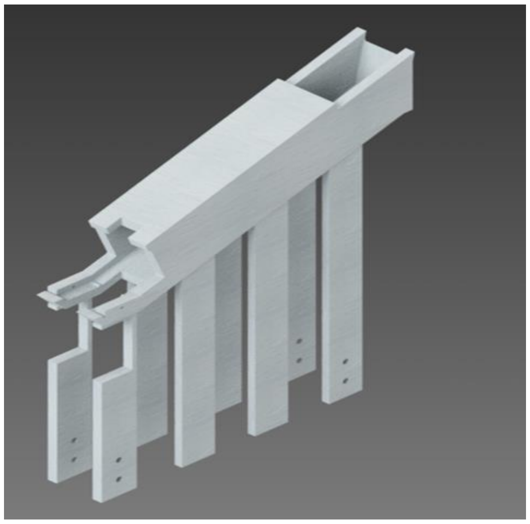


Figure 3:Feeding Station

4.0 SORTING STATION

4.1 Mechanical System

Our sorting-by material station is made of two plates and a thrust bearing. When the robotic arm takes the product from the magazine and then puts it into the sorting top plate hole, the proximity sensor reads the product's material. Then, the plate moves either right or left by the help of a servo motor to drop the product into the packaging boxes via a hole in bottom plate. Due to the high weight on the sorting shaft, we add a thrust bearing to carry the axial load. We designed our thrust bearing to have a large diameter to support the top plate. We fixed the bottom thrust bearing plate into the bottom sorting plate with 4 M4 bolts. The supports that hold the fixed sorting plate are fixed by grooves in electrical box and L supports. Also, fixed with grooves and L supports also from the upper side with the fixed part of sorting and bolts from top and bottom. Material: Wooden. Our two plates of sorting mechanism were manufactured by a laser cutting machine.

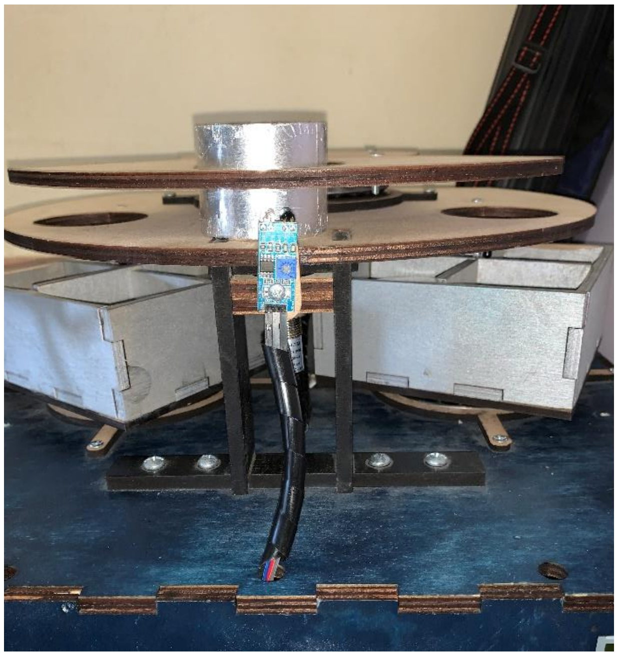


Figure 4: Sorting station

4.2 Actuator Sizing

Formula Sheet for motor

$$T_{motor} = T_R + J_{total}\ddot{\theta}$$
$$J_{total} = 2\,J_{L\,(effictive)} + J_{b\,(effictive)} + J_{p\,(effictive)}$$

Where;

J_{total} total inertia,

$J_{L\,(effictive)}$ inertia load,

$J_{b\,(effictive)}$ inertia belt,

$J_{p\,(effictive)}$ inertia pully.

$T_R = T_{Load} + T_{Friction} + T_{Damping}$

Where;

T_R resisting torque

T_{Load} load torque,

$T_{Friction}$ friction torque,

$T_{Damping}$ damping torque.

Assuming $T_{Friction} = 0$ / $T_{Damping} = 0$

$$J_{total} = 2\,J_{L\,(effictive)} + J_{b\,(effictive)} + J_{p\,(effictive)}$$

$m_L = 0.227\,kg$ / $r_p = 0.1775\,m$

$$J_{L\,(effictive)} = \frac{0.5m_L(r_p)^2}{\boldsymbol{\eta}} = \frac{0.5 \times 0.227 \times (0.1775)^2}{1} = 3.6 \times 10^{-3} kg.m^2$$
$$J_{total} = 2(J_{L\,(effictive)})$$
$$J_{total} = 7.15 \times 10^{-3} kg.m^2$$

$T_R = T_{Load}$

$m_L = 0.227\,kg$ / $r_p = 0.1775\,m$

$$T_{Load} = \frac{F_L\, r_p}{\eta} = \frac{m_L \times g \times r_p}{\eta} = \frac{0.227 \times 9.81 \times 0.1775}{1}$$

$$T_R = 0.395\ N.m$$

Assuming $t_a = 0.3\ sec$

$$\theta = \Pi/4\ rad$$

$$\ddot{\theta} = \frac{\partial\dot{\theta}}{\partial t} = \frac{\frac{5\Pi}{46} - 0}{0.3 - 0}$$

$$\ddot{\theta} = \pm 1.138\ rad/sec^{\,2}$$

$$T_{motor} = T_R + J_{total}\ddot{\theta}$$

Assuming $t_a = t_d = 0.3\ sec$ / $t_c = 2\ sec$

$T_R = 0.395\ N.m / J_{total} = 7.15 \times 10^{-3} kg.m^2 / \ddot{\theta} = \ \pm 1.138\ rad/sec^{\,2}$

$T_a = T_R + J_{total}\ddot{\theta} = 0.395 + (1.138 \times 7.15 \times 10^{-3}) = 0.403\ N.m$

$$T_c = T_R = 0.395\ N.m$$

$$T_d = T_R + J_{total}\ddot{\theta} = 0.395 + (-1.138 \times 7.15 \times 10^{-3}) = 0.386\ N.m$$

$$T_{rms} = \sqrt{\frac{((T_a)^2 \times t_a) + ((T_c)^2 \times t_c) + ((T_d)^2 \times t_d)}{t_a + t_c + t_d}}$$

$$T_{rms} = \sqrt{\frac{((0.403)^2 \times 0.3) + ((0.395)^2 \times 2) + ((0.386)^2 \times 0.3)}{2.6}}$$

$$T_{rms} = 0.3949\ N.m < T_{min} = 0.9\ N.m$$

$$(used\ motor\ rated\ torque\ after\ multiplying\ by\ safety\ factor\ )$$

5.0 PACKAGING STATION

5.1 Mechanical System

Our Packaging Mechanism is designed to be a box divided into 4 pieces that rotates 90° degrees with a servo motor whenever the IR sensor, planted into the electrical box beneath the packaging boxes, reads that there is a product that was dropped from the sorting station just to expose the empty space for the sorting to drop the new product. Each packaging box contains slots for the IR sensor. To fix the load exerted on the servo motor's shaft, we used a thrust bearing to reduce the axial load on servo motor's shaft. The servo motor's shaft is inserted into horn to fix the servo motor into the packaging. The packaging IR sensors are nested in the fixed sorting plate and will be bended to read the products. Material: Wooden. Our packaging boxes will be manufactured by a laser cutting machine. And as every face has fingers so every face will be manufactured and then meshed together to do the box.

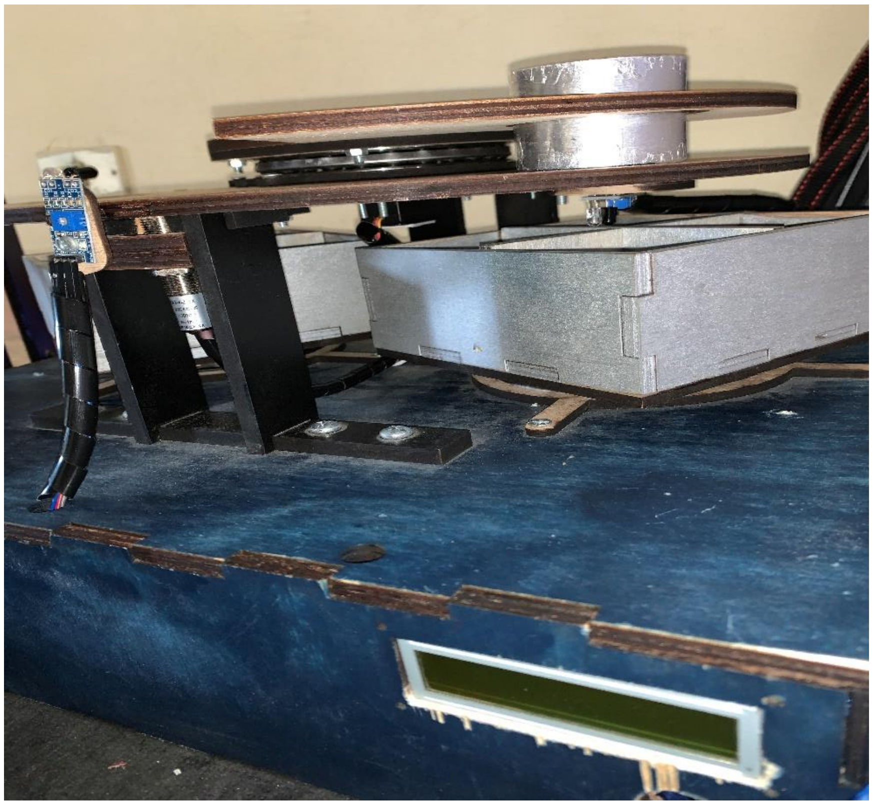


Figure 5:Packaging station

5.2 Actuator Sizing

- **Motor of the package of metal products**

$$J_{total} = 2\,J_{L\,(effictive)} + J_{b\,(effictive)} + J_{p\,(effictive)}$$

$m_L = 0.56\,kg$ / $r_p = 0.0975\,m$

$$J_{L\,(effictive)} = \frac{0.5 m_L (r_p)^2}{\eta} = \frac{0.5 \times 0.56 \times (0.0975)^2}{1} = 2.6 \times 10^{-3} kg.m^2$$

$$J_{total} = 2(J_{L\,(effictive)})$$

$$J_{total} = 5.6\ \times 10^{-3} kg.m^2$$

$T_R = T_{Load}$

$m_L = 0.56\,kg$ / $r_p = 0.0975\,m$

$$T_{Load} = \frac{F_L\,r_p}{\eta} = \frac{m_L\ \times g \times r_p}{\eta} = \frac{0.56 \times 9.81 \times 0.0975}{1}$$

$$T_R = 0.5356\,N.m$$

Assuming $t_a = 0.3\,sec$

$$\theta = \Pi/2\,rad$$

$$\ddot{\theta} = \frac{\partial\dot{\theta}}{\partial t} = \frac{\frac{5\Pi}{18} - 0}{0.3 - 0}$$

$$\ddot{\theta} = \pm 2.91\,rad/sec^{\,2}$$

$$T_{motor} = T_R + J_{total}\ddot{\theta}$$

Assuming $t_a = t_d = 0.3\,sec$ / $t_c = 1.5\,sec$

$T_R = 0.5356\,N.m / J_{total} = 5.2 \times 10^{-3} kg.m^2 / \ddot{\theta} = \ \pm 2.91\,rad/sec^{\,2}$

$T_a = T_R + J_{total}\ddot{\theta} = 0.5356 + (2.91 \times 5.2 \times 10^{-3}) = 0.55\,N.m$

$$T_c = T_R = 0.5356\,N.m$$

$$T_d = T_R + J_{total}\ddot{\theta} = 0.395 + (-1.138 \times 7.15 \times 10^{-3}) = 0.52\,N.m$$

$$T_{rms} = \sqrt{\frac{((T_a)^2 \times t_a) + ((T_c)^2 \times t_c) + ((T_d)^2 \times t_d)}{t_a + t_c + t_d}}$$

$$T_{rms} = \sqrt{\frac{((0.55)^2 \times 0.3) + ((0.5356)^2 \times 1.5) + ((0.52)^2 \times 0.3)}{2.1}}$$

$$T_{rms} = 0.5356\ N.m < T_{min} = 0.9\ N.m$$

$(used\ motor\ rated\ torque\ after\ multiplying\ by\ safety\ factor\ )$

- **Motor of the package of nonmetal products**

$$J_{total} = 2\, J_{L\,(effictive)} + J_{b\,(effictive)} + J_{p\,(effictive)}$$

$m_L = 0.375\ kg\ /\ r_p = 0.0975\ m$

$$J_{L\,(effictive)} = \frac{0.5 m_L (r_p)^2}{\eta} = \frac{0.5 \times 0.375 \times (0.0975)^2}{1} = 1.78 \times 10^{-3} kg.m^2$$

$$J_{total} = 2(J_{L\,(effictive)})$$

$$J_{total} = 3.565 \times 10^{-3} kg.m^2$$

$T_R = T_{Load}$

$m_L = 0.375\ kg\ /\ r_p = 0.0975\ m$

$$T_{Load} = \frac{F_L\, r_p}{\eta} = \frac{m_L \times g \times r_p}{\eta} = \frac{0.375 \times 9.81 \times 0.0975}{1}$$

$$T_R = 0.359\ N.m$$

Assuming $t_a = 0.3\ sec$

$$\theta = \Pi/2\ rad$$

$$\ddot{\theta} = \frac{\partial \dot{\theta}}{\partial t} = \frac{\frac{5\Pi}{18} - 0}{0.3 - 0}$$

$$\ddot{\theta} = \pm 2.91\ rad/sec^{\ 2}$$

$$T_{motor} = T_R + J_{total}\ddot{\theta}$$

Assuming $t_a = t_d = 0.3\ sec\ /\ t_c = 1.5\ sec$

$T_R = 0.359\ N.m/\ J_{total} = 3.565 \times 10^{-3} kg.m^2/\ \ddot{\theta} = \ \pm 2.91\ rad/sec^{\ 2}$

$T_a = T_R + \ J_{total}\ddot{\theta} = 0.359 + (2.91 \times 3.565 \times 10^{-3}) = 0.364\ N.m$

$$T_c = T_R = 0.359\ N.m$$

$$T_d = T_R + \ J_{total}\ddot{\theta} = 0.395 + (-1.138 \times 7.15 \times 10^{-3}) = 0.354\ N.m$$

$$T_{rms} = \sqrt{\frac{((T_a)^2 \times t_a) + ((T_c)^2 \times t_c) + ((T_d)^2 \times t_d)}{t_a + t_c + t_d}}$$

$$T_{rms} = \sqrt{\frac{((0.364)^2 \times 0.3) + ((0.359)^2 \times 1.5) + ((0.354)^2 \times 0.3)}{2.1}}$$

$$T_{rms} = 0.359\ N.m < T_{min} = 0.9\ N.m$$

$(used\ motor\ rated\ torque\ after\ multiplying\ by\ safety\ factor\ )$

6.0 ROBOTIC ARM

6.1 Mechanical System

Our robotic arm is 3-DOF structure. So, this robotic arm is able to grip the product from the magazine and put it which it will be able to pick the cylinders and move them to the sorting station. Each gear is fixed with a link and the link is fixed with a grip that's fixed also with the base of the gripper. The grip is designed to be half circled from inside to hold the cylindrical product and make enough contact with its surface.

ARM 1: It is a rotating arm connected to the servo motor in the base by a horn fixed with bolts. BASE: It is fixed to the base by 4 M3 bolts and where a servo motor is fixed inside to rotate Arm 1.

GRIPPER: A rotating gripper connected to Arm 2 by a servo motor with horn and bolts.

ARM 2: It is fixed to Arm 1 by a servo motor using a horn with bolts. It is the initial position and it can rotates 180 degrees.

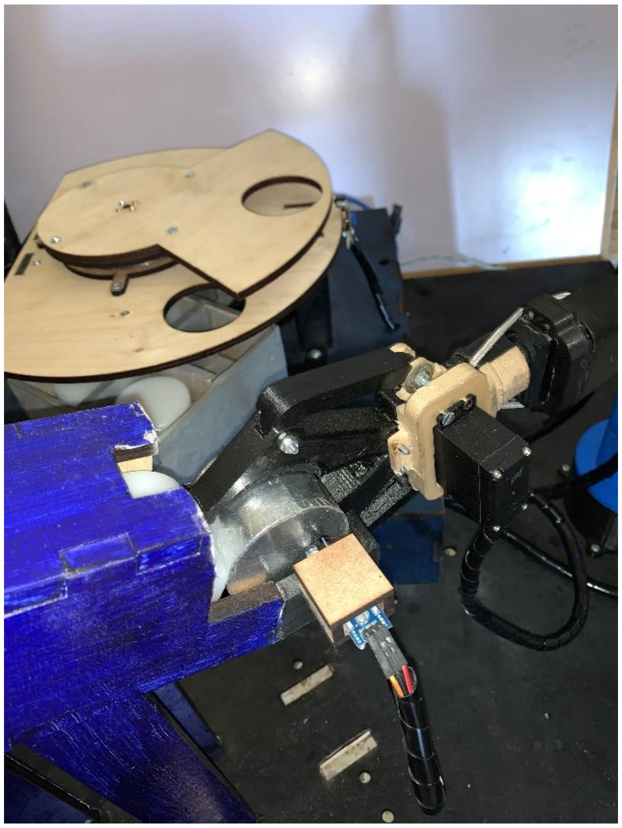


Figure 6:Robotic arm

Material: PLA Our robotic arm will be manufactured by a 3D Printing machine.

6.2 Actuator Sizing

- **Motor of the base of robotic arm**

$$J_{total} = 2\,J_{L\,(effictive)} + J_{b\,(effictive)} + J_{p\,(effictive)}$$

$m_L = 0.955\ kg$ / $r_p = 0.05\ m$

$$J_{L\,(effictive)} = \frac{0.5 m_L (r_p)^2}{\boldsymbol{\eta}} = \frac{0.5 \times 0.955 \times (0.05)^2}{1} = 1.19 \times 10^{-3}\ kg.m^2$$

$$J_{total} = 2(1.19 \times 10^{-3})$$

$$J_{total} = 2.38 \times 10^{-3} kg.m^2$$

$T_R = T_{Load}$

$m_L = 0.955\ kg$ / $r_p = 0.05\ m$

$$T_{Load} = \frac{F_L\, r_p}{\boldsymbol{\eta}} = \frac{m_L \times g \times r_p}{\boldsymbol{\eta}} = \frac{0.955 \times 9.81 \times 0.05}{1}$$

$$T_R = 0.468\ N.m$$

Assuming $t_a = 0.4\ sec$

$$\theta = 7\Pi/9\ rad$$

$$\ddot{\theta} = \frac{\boldsymbol{\partial}\dot{\theta}}{\boldsymbol{\partial}t} = \frac{\frac{35\Pi}{198} - 0}{0.4 - 0}$$

$$\ddot{\theta} = \pm 1.388\ rad/sec^{\,2}$$

$$T_{motor} = T_R + J_{total}\ddot{\theta}$$

Assuming $t_a = t_d = 0.4\ sec$ / $t_c = 4\ sec$

$T_R = 0.468\ N.m / J_{total} = 2.38 \times 10^{-3} kg.m^2 / \ddot{\theta} = \pm 1.388\ rad/sec^{\,2}$

$T_a = T_R + J_{total}\ddot{\theta} = 0.468 + (1.388 \times 2.38 \times 10^{-3}) = 0.471\ N.m$

$$T_c = T_R = 0.468\ N.m$$

$$T_d = T_R + J_{total}\ddot{\theta} = 0.468 + (-1.388 \times 2.38 \times 10^{-3}) = 0.464\ N.m$$

$$T_{rms} = \sqrt{\frac{((T_a)^2 \times t_a) + ((T_c)^2 \times t_c) + ((T_d)^2 \times t_d)}{t_a + t_c + t_d}}$$

$$T_{rms} = \sqrt{\frac{((0.471)^2 \times 0.4) + ((0.468)^2 \times 4) + ((0.464)^2 \times 0.4)}{4.8}}$$

$$T_{rms} = 0.4679\ N.m < T_{min} = 0.9\ N.m$$

$(used\ motor\ rated\ torque\ after\ multiplying\ by\ safety\ factor\ )$

- **Motor of the arm of robotic arm**

$$J_{total} = 2\,J_{L\,(effictive)} + J_{b\,(effictive)} + J_{p\,(effictive)}$$

$m_L = 0.493\ kg\ /\ r_p = 0.31\ m$

$$J_{L\,(effictive)} = \frac{0.5 m_L (r_p)^2}{\boldsymbol{\eta}} = \frac{0.5 \times 0.493 \times (0.31)^2}{1} = 0.024\ kg.m^2$$

$$J_{total} = 2(0.024)$$

$$J_{total} = 0.047 kg.m^2$$

$T_R = T_{Load}$

$m_L = 0.493\ kg\ /\ r_p = 0.31\ m$

$$T_{Load} = \frac{F_L\, r_p}{\eta} = \frac{m_L \times g \times r_p}{\eta} = \frac{0.493 \times 9.81 \times 0.31}{1}$$

$$T_R = 1.5\ N.m$$

Assuming $t_a = 0.3\ sec$

$$\theta = 2\Pi/9\ rad$$

$$\ddot{\theta} = \frac{\partial\dot{\theta}}{\partial t} = \frac{\frac{10\Pi}{81} - 0}{0.3 - 0}$$

$$\ddot{\theta} = \pm 1.29\, rad/sec^{\,2}$$

$$T_{motor} = T_R + J_{total}\ddot{\theta}$$

Assuming $t_a = t_d = 0.3\, sec$ / $t_c = 1.5\, sec$

$T_R = 1.5\, N.m / J_{total} = 0.047 kg.m^2 / \ddot{\theta} = \pm 1.29\, rad/sec^{\,2}$

$T_a = T_R + J_{total}\ddot{\theta} = 1.5 + (1.29 \times 0.047) = 1.56\, N.m$

$$T_c = T_R = 1.5\, N.m$$

$$T_d = T_R + J_{total}\ddot{\theta} = 1.5 + (-1.29 \times 0.047) = 1.439\, N.m$$

$$T_{rms} = \sqrt{\frac{((T_a)^2 \times t_a) + ((T_c)^2 \times t_c) + ((T_d)^2 \times t_d)}{t_a + t_c + t_d}}$$

$$T_{rms} = \sqrt{\frac{((1.56)^2 \times 0.3) + ((1.5)^2 \times 1.5) + ((1.439)^2 \times 0.3)}{2.1}}$$

$$T_{rms} = 1.51\, N.m < T_{min} = 2\, N.m$$

$(used\ motor\ rated\ torque\ after\ multiplying\ by\ safety\ factor\ )$

- **Motor of the wrist of robotic arm**

$$J_{total} = 2\, J_{L\,(effictive)} + J_{b\,(effictive)} + J_{p\,(effictive)}$$

$m_L = 0.288\, kg$ / $r_p = 0.035\, m$

$$J_{L\,(effictive)} = \frac{0.5 m_L (r_p)^2}{\boldsymbol{\eta}} = \frac{0.5 \times 0.288 \times (0.035)^2}{1} = 1.6 \times 10^{-5} kg.m^2$$

$$J_{total} = 2(1.6 \times 10^{-5})$$

$$J_{total} = 3.175\ \times 10^{-5} kg.m^2$$

3. Motor of the wrist of robotic arm

$T_R = T_{Load}$

$m_L = 0.288\, kg$ / $r_p = 0.035\, m$

$$T_{Load} = \frac{F_L\, r_p}{\eta} = \frac{m_L \times g \times r_p}{\eta} = \frac{0.288 \times 9.81 \times 0.035}{1}$$

$$T_R = 0.0988\ N.m$$

Assuming $t_a = 0.3\ sec$

$$\theta = \Pi/2\ rad$$

$$\ddot{\theta} = \frac{\partial \dot{\theta}}{\partial t} = \frac{\frac{5\Pi}{23} - 0}{0.3 - 0}$$

$$\ddot{\theta} = \pm 2.277\ rad/sec^{\,2}$$

$$T_{motor} = T_R + J_{total}\ddot{\theta}$$

Assuming $t_a = t_d = 0.3\ sec$ / $t_c = 2\ sec$

$T_R = 0.0988\ N.m / J_{total} = 3.175 \times 10^{-5} kg.m^2 / \ddot{\theta} = \pm 2.277\ rad/sec^{\,2}$

$T_a = T_R + J_{total}\ddot{\theta} = 0.0988 + (2.277 \times 3.175 \times 10^{-5}) = 0.0989\ N.m$

$$T_c = T_R = 0.0988\ N.m$$

$$T_d = T_R + J_{total}\ddot{\theta} = 0.0988 + (-2.277 \times 3.175 \times 10^{-5}) = 0.0987\ N.m$$

$$T_{rms} = \sqrt{\frac{((T_a)^2 \times t_a) + ((T_c)^2 \times t_c) + ((T_d)^2 \times t_d)}{t_a + t_c + t_d}}$$

$$T_{rms} = \sqrt{\frac{((0.0989)^2 \times 0.3) + ((0.0988)^2 \times 2) + ((0.0987)^2 \times 0.3)}{2.6}}$$

$$T_{rms} = 0.098\ N.m < T_{min} = 0.3\ N.m$$

$$(used\ motor\ rated\ torque\ after\ multiplying\ by\ safety\ factor\ )$$

- **Motor of the gripper of robotic arm**

$$J_{total} = 2\,J_{L\,(effictive)} + J_{b\,(effictive)} + J_{p\,(effictive)}$$

$m_L = 0.174\,kg$ / $r_p = 0.0155\,m$

$$J_{L\,(effictive)} = \frac{0.5m_L(r_p)^2}{\boldsymbol{\eta}} + \frac{0.5m_g(r_g)^2}{\boldsymbol{\eta}}$$

$$= \frac{0.5 \times 0.174 \times (0.0155)^2}{1} + \frac{0.5 \times 0.0.0068 \times (0.0155)^2}{1} = 2.49 \times 10^{-8} kg.m^2$$

$$J_{total} = 2(J_{L\,(effictive)})$$

$$J_{total} = 4.984\ \times 10^{-8} kg.m^2$$

4. Motor of the gripper of robotic arm

$T_R = T_{Load}$

$m_L = 0.174\,kg$ / $r_p = 0.0155\,m$

$$T_{Load} = \frac{F_L\,r_p}{\eta} = \frac{m_L \times g \times r_p}{\eta} = \frac{0.174 \times 9.81 \times 0.0155}{1}$$

$$T_R = 0.026\,N.m$$

Assuming $t_a = 0.3\,sec$

$$\theta = 7\Pi/18\,rad$$

$$\ddot{\theta} = \frac{\partial\dot{\theta}}{\partial t} = \frac{\frac{35\Pi}{162} - 0}{0.3 - 0}$$

$$\ddot{\theta} = \pm 2.26\,rad/sec^{\,2}$$

$$T_{motor} = T_R + J_{total}\ddot{\theta}$$

Assuming $t_a = t_d = 0.3\,sec$ / $t_c = 1.5\,sec$

$T_R = 0.026\,N.m/ J_{total} = 4.984 \times 10^{-8} kg.m^2/\ \ddot{\theta} = \pm 2.26\,rad/sec^{\,2}$

$$T_a = T_R + J_{total}\ddot{\theta} = 0.026 + (2.26 \times 4.984 \times 10^{-8}) = 0.0261\ N.m$$

$$T_c = T_R = 0.026\ N.m$$

$$T_d = T_R + J_{total}\ddot{\theta} = 0.026 + (-2.26 \times 4.984 \times 10^{-8}) = 0.0259\ N.m$$

$$T_{rms} = \sqrt{\frac{((T_a)^2 \times t_a) + ((T_c)^2 \times t_c) + ((T_d)^2 \times t_d)}{t_a + t_c + t_d}}$$

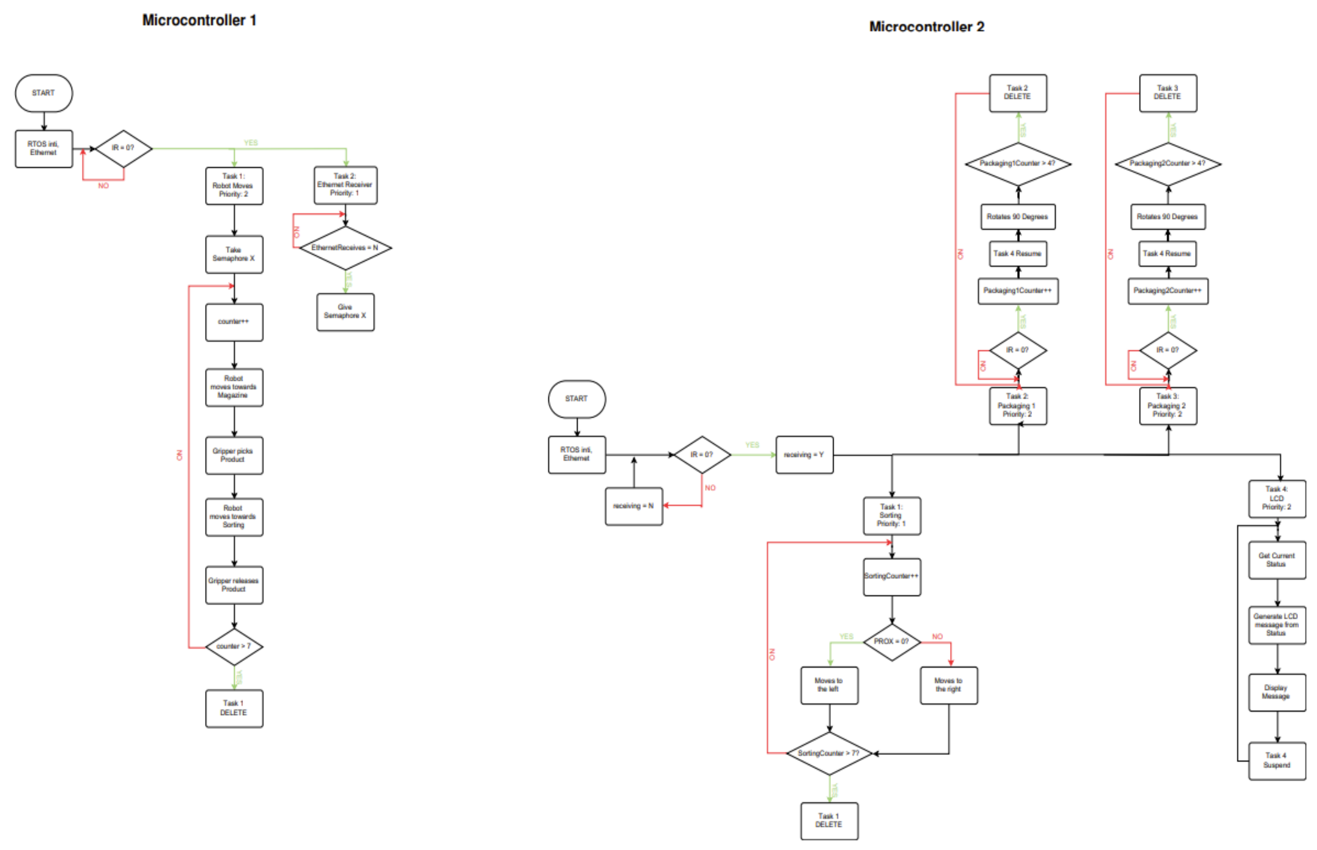
$$T_{rms} = \sqrt{\frac{((0.0261)^2 \times 0.3) + ((0.026)^2 \times 1.5) + ((0.0259)^2 \times 0.3)}{2.1}}$$

$$T_{rms} = 0.026\ N.m < T_{min} = 0.3\ N.m$$

$(used\ motor\ rated\ torque\ after\ multiplying\ by\ safety\ factor\ )$

7.0 SOFTWARE FLOW CHART

A well organized and accurate flowchart of the whole project software using RTOS, showing our two microcontrollers and how we connect them by the ethernet commun



8.0 CIRCUITS AND WIRING

8.1 Control Box

Our wooden control box with 400x100x400 mm that contains all the electronics. All the wires coming out from each part is going through that control box holes and slot. In addition, for more professional and cleaner look we made holes in the control box bottom to pass the wires beneath the base and on witch is fixed our sorting and packaging stations.

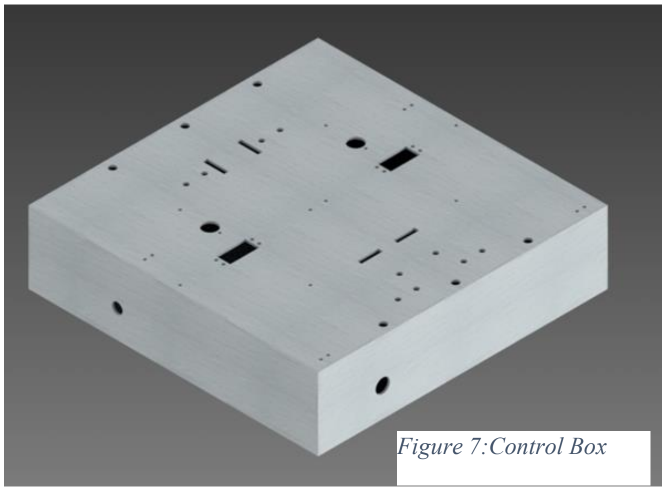


Figure 7:Control Box

8.2 Wiring Tracks

All jumpers are well wired so that they will be assembled in the control box.

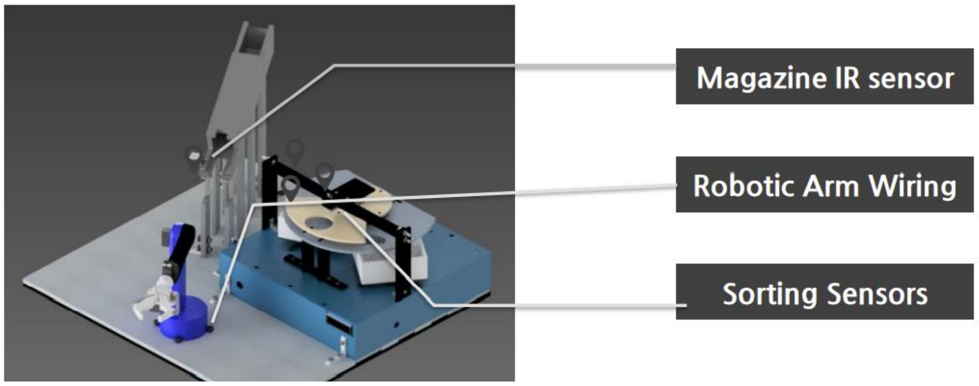


Figure 8:Wiring Tracks

8.3 Electronic Components

- 12 volts-15A Power Supply.
- Two Arduino UNO
- Two Bucks Step down module to get a 6 Volts output to power all servo motors.
- Seven Servo motors (4: Robotic arm, 2 for each packaging box, 1 for sorting).
- Four Infra-Red sensors (1: agazine, 1 for sorting, 2 for each packaging box).
- One Proximity inductive sensor to detect metal products.
- LCD
- I2C Module to reduce the lacd pins.
- Router for the ethernet communication.
- PCBs

Figure 9:Power Supply

Figure 10:Buck module

Figure 12:Proximity sensor

Figure 11:I2C module

8.4 PCB Layout for Microcontroller 1

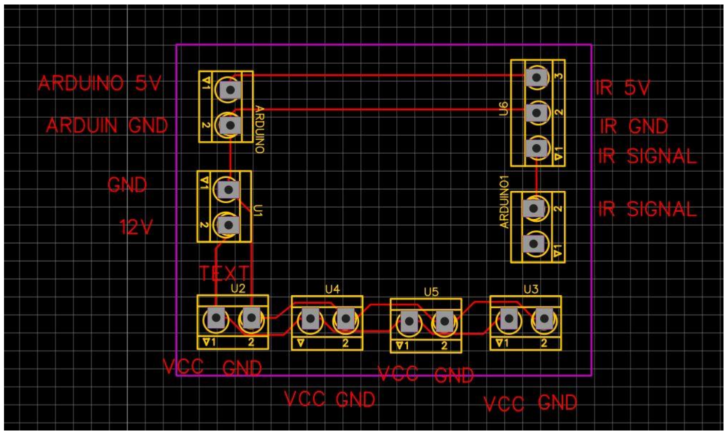


Figure 14:Detailed robotic arm PCB layout

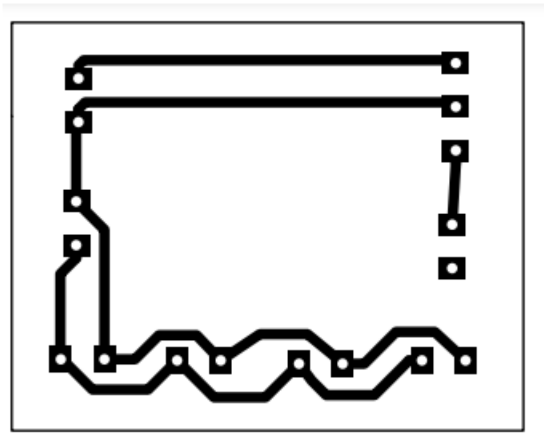


Figure 13:Microcontroller 1 PCB layout

8.5 PCB Layout for Microcontroller 2

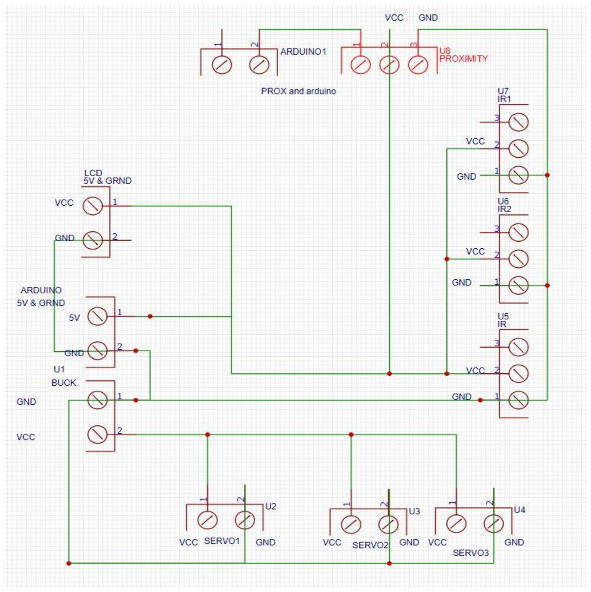


Figure 15: Microcontroller 2 PCB Schematic

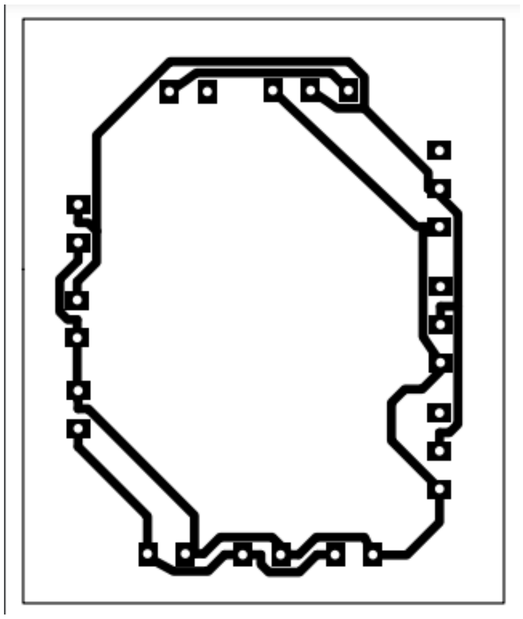


Figure 16:Microcontroller 2 PCB layout

9.0 ETHERNET COMMUNICATION

Using common Ethernet protocols in an industrial setting is known as industrial Ethernet (IE). Since IE systems make use of the physical layer that is already present in the majority of contexts and is easily expandable, there is less of a need to create additional networks. A new level of freedom in application configuration and system design is made possible by industrial Ethernet communication. We can build networks that are more effective and versatile thanks to Ethernet connectivity. There are Ethernet networks everywhere.

9.1 Sender

```
#include <Arduino_FreeRTOS.h>
#include "LiquidCrystal_I2C.h"
#include <EtherCard.h>
#include "semphr.h"
#include <Servo.h>


  const int n = 1; //if n = 0 textToSend is send (Metal) if n = 1 textToSend2 is send (Non Metal)
  /* Addresses */
  static byte mymac[] = { 0x1A,0x2B,0x3C,0x4D,0x5E,0x6F };  // mac address
  static byte myip[] = { 192, 168, 1, 2 }; // ethernet interface ip address
  static byte gwip[] = { 192, 168, 1, 1 }; // gateway ip address Must be same in both Sender and
reciver
  static byte mask[] = { 255, 255, 255, 0 }; // subnet mask
  static byte dstIp[] = { 192, 168, 1, 3 }; // destination ip-address (reciver ip address)
  // ports
```

```
const int dstPort PROGMEM = 1234;
const int srcPort PROGMEM = 4321;

/* Data to send via Ethernet */
char noProduct[] = "NO"; // NO text that will be send
char Product[] = "YES";  // YES text that will be send

/* Buffer and timer */
byte Ethernet::buffer[700];
static uint32_t timer;  // dummy incrementor that'll later be used for delays

// Calling LCD PINS
LiquidCrystal_I2C lcd(0x3F,16,2);

// for if condition in LCD
boolean state = true;

// calling task functions
static void SortingTask(void* parameters);
static void PackagingTask(void* parameters);
static void LCDTask(void* parameters);

// Global Variables
// Packaging 1
int i = 0;
// Packaging 2
int x = 0;
```

```
// Servo calling
Servo SortingServo;
Servo Packaging1Servo;
Servo Packaging2Servo;

// defining servos here cause they're classes
int pinServo_Sorting = 3;
int pinServo_Packaging1 = 5;
int pinServo_Packaging2 = 6;

// defining the sorting proximity sensor
int pinPROX = A0;

// defining IR sensors
int pinIR_Sorting = 2; // Sorting IR
int pinIR_Packaging1 = 4; // Packaging 1 IR
int pinIR_Packaging2 = 7; // Packaging 2 IR

// initialization values of each sensor
int valIR_Packaging1 = 0;
int valIR_Packaging2 = 0;
int valIR_Sorting = 0;
int valPROX = 0;

// counters
int counterSorting = 0;
int counterPackaging = 0;
int counterPackaging1_LCD = 0;
int counterPackaging2_LCD = 0;

//create a binary semaphore
SemaphoreHandle_t xBinarySemaphore;
SemaphoreHandle_t xBinary1Semaphore;

// Task handeling
TaskHandle_t SortingTask_handle;
TaskHandle_t PackagingTask_handle;
TaskHandle_t LCDTask_handle;

/* Ethernet peripheral initialization function */
void initializeEthernet()
{
  // Begin Ethernet communication
  if (ether.begin(sizeof Ethernet::buffer, mymac, 10) == 0)  // if using Mega use value 53 if using
uno use value 10
  { Serial.println("Failed to access Ethernet controller");
    return;}

  // Setup static IP address
  ether.staticSetup(myip, gwip, 0, mask);

  // Log configuration
  Serial.println(F("\n[Sender]"));
  ether.printIp("IP:  ", ether.myip);
  ether.printIp("GW:  ", ether.gwip);
}
```

```
void setup() {
  // Initializing Serial Monitor
  Serial.begin(57600);
  // Initializing LCD
  lcd.init();
  lcd.backlight();
  // Initialize Ethernet
  initializeEthernet();

  // For packaging 1
  lcd.setCursor(0,0);
  lcd.print("Metal Count: ");
  // For packaging 2
  lcd.setCursor(0,1);
  lcd.print("Artelon Count: ");

  // Creating task semaphore
  xBinarySemaphore = xSemaphoreCreateBinary(); // sorting to packaging
  xBinary1Semaphore = xSemaphoreCreateBinary(); // packaging to LCD

  // Set the classes
  SortingServo.attach(pinServo_Sorting);
  Packaging1Servo.attach(pinServo_Packaging1);
  Packaging2Servo.attach(pinServo_Packaging2);

  // Set the digital pins
  pinMode(pinPROX,          INPUT);
  pinMode(pinIR_Sorting,      INPUT);
  pinMode(pinIR_Packaging1,   INPUT);
  pinMode(pinIR_Packaging2,   INPUT);

  // Create tasks and assign their priorities
  xTaskCreate(SortingTask,   "Sorting_Task",     200, NULL, 1, &SortingTask_handle);
  xTaskCreate(PackagingTask, "Packaging_Task",   200, NULL, 2, &PackagingTask_handle);
  xTaskCreate(LCDTask,       "LCD_Task",         100, NULL, 3, &LCDTask_handle);

  // starts tasks with its priorities
  vTaskStartScheduler();

} // END of setup function

void loop() { } // END of loop function

// function that's responsible for product sorting
static void SortingTask(void* parameters) {
  while(1) {
    valIR_Sorting = digitalRead(pinIR_Sorting);
    valPROX = digitalRead(pinPROX);

    // ethernet sending
    for(int i = 0; i <= 3; i++){
      if (millis() > timer) {
        timer = millis() + 100;
        // testing funtion if changed the global variable from the top from zero or one as to identify
metal or non metal this should be your function
```

```
    if(valIR_Sorting == 1)
    {ether.sendUdp(noProduct,sizeof(noProduct),srcPort,dstIp,dstPort );}
    else if(valIR_Sorting == 0)
    {ether.sendUdp(Product,sizeof(Product),srcPort,dstIp,dstPort );}
  }
}

// sorting process
if(valIR_Sorting == 0 && valPROX == 1) {
  counterSorting++;
  for (int i = 94; i >= 45; i--) {
    SortingServo.write(i);
    vTaskDelay(10/portTICK_PERIOD_MS);
  }
  vTaskDelay(1000/portTICK_PERIOD_MS);
} // Metal Sorting
else if(valIR_Sorting == 0 && valPROX == 0) {
  counterSorting++;
  for (int i = 94; i <= 150; i++) {
    SortingServo.write(i);
    vTaskDelay(10/portTICK_PERIOD_MS);
  }
  vTaskDelay(1000/portTICK_PERIOD_MS);
} // Artelon Sorting
else {
  SortingServo.write(94);
  vTaskDelay(1000/portTICK_PERIOD_MS);
```

```
    } // no product to be found

    SortingServo.write(94);
    vTaskDelay(1000/portTICK_PERIOD_MS);

    // gives semaphore to the packaging
    xSemaphoreGive(xBinarySemaphore);

    // deleting task after 8 products
    if(counterSorting > 7) {
      vTaskDelete(SortingTask_handle);
    }
  }
} // END of SortingTask function

// function that's responsible for Packaging 1
static void PackagingTask(void* parameters) {
  while(1) {
    // takes semaphore from the sorting
    xSemaphoreTake(xBinarySemaphore, portMAX_DELAY);
    valIR_Packaging1 = digitalRead(pinIR_Packaging1);
    valIR_Packaging2 = digitalRead(pinIR_Packaging2);

    if(valIR_Packaging1 == 0) {
      counterPackaging++;
      Packaging2Servo.write(0);
      vTaskDelay(273/portTICK_PERIOD_MS);
      Packaging2Servo.write(90);
      vTaskDelay(273/portTICK_PERIOD_MS);
    }
    if(valIR_Packaging2 == 0) {
      counterPackaging++;
      Packaging1Servo.write(0);
      vTaskDelay(273/portTICK_PERIOD_MS);
      Packaging1Servo.write(90);
      vTaskDelay(273/portTICK_PERIOD_MS);
    }

    // gives semaphore to the LCD
    //xSemaphoreGive(xBinary1Semaphore);

    if(counterPackaging > 7) {
      vTaskDelete(PackagingTask_handle);
    }
  }
} // END of SortingTask function

// function that's responsible for LCD
static void LCDTask (void* parameters) {
  while(1) {
    // takes semaphore from the packaging
    //xSemaphoreTake(xBinary1Semaphore, portMAX_DELAY);
    valIR_Packaging1 = digitalRead(pinIR_Packaging1);
    valIR_Packaging2 = digitalRead(pinIR_Packaging2);

    if (valIR_Packaging1 == 0 && state == true){
      counterPackaging1_LCD++;
      state = false;
      lcd.setCursor(15,0);
      lcd.print(counterPackaging1_LCD);
      vTaskDelay(1000/portTICK_PERIOD_MS);
    }
    else if (valIR_Packaging2 == 0 && state == true){
      counterPackaging2_LCD++;
      state = false;
      lcd.setCursor(15,1);
      lcd.print(counterPackaging2_LCD);
      vTaskDelay(1000/portTICK_PERIOD_MS);
    }

    if (valIR_Packaging1 == 1 || valIR_Packaging2 == 1 ) {
      state = true;
      vTaskDelay(1000/portTICK_PERIOD_MS);
    }

    if(counterPackaging1_LCD > 3 && counterPackaging2_LCD > 3 ) {
      vTaskDelete(LCDTask_handle);
    }
  }
} // END of LCDTask function
```

9.2 Receiver

```
#include <Arduino_FreeRTOS.h>
#include <IPAddress.h>
#include <EtherCard.h>
#include <Servo.h>

  // Global variables
  // base
  int i = 0;
  int j = 0;
  int k = 0;
  // arm
  int x = 0;
  int y = 0;
  int z = 0;
  // wrist
  int w = 0;
  int r = 0;
  // gripper
  int g = 0;
```

```
// Servo calling
Servo BaseServo;
Servo ArmServo;
Servo WristServo;
Servo GripperServo;

// defining servos here cause they're classes
int pinServo_Base = 3;
int pinServo_Arm = 5;
int pinServo_Wrist = 6;
int pinServo_Gripper = 9;

// defining IR sensors
int pinIR_Magazine = 8; // Sorting IR

// initialization values of each sensor
int valIR_Magazine = 0;
//int valIR_Sorting = 0;

// counters
int Counter = 0;

// Task handeling
TaskHandle_t RoboticArmTask_handle = NULL;

const int n;
/* Addresses */
```

```
static byte mymac[] = { 0x70,0x69,0x69,0x2D,0x30,0x31 }; // mac address
static byte myip[] = { 192, 168, 1, 3 }; // ethernet interface ip address
static byte gwip[] = { 192, 168, 1, 1 }; // gateway ip address
static byte mask[] = { 255, 255, 255, 0 };// subnet mask
// ports
const int dstPort PROGMEM = 1234;

/* Buffer */
byte Ethernet::buffer[700];

void initializeEthernet()
{
  // Begin Ethernet communication
  if (ether.begin(sizeof Ethernet::buffer, mymac, 10) == 0)  // if using Mega use value 53 if using
uno use value 10
  { Serial.println("Failed to access Ethernet controller");
    return;}

  // Setup static IP address
  ether.staticSetup(myip, gwip, 0, mask);

  // Log configuration
  Serial.println(F("\n[Receiver]"));
  ether.printIp("IP:  ", ether.myip);
  ether.printIp("GW:  ", ether.gwip);
}

/* callback that prints received packets to the serial port */
```

```
 void udpSerialPrint(uint16_t dest_port, uint8_t src_ip[IP_LEN], uint16_t src_port, const char
*data, uint16_t len){
   IPAddress src(src_ip[0],src_ip[1],src_ip[2],src_ip[3]);

   Serial.println(data);   // print the data send );
   if(data=='NO')       // dummy condition just for testing
    {int n =0;
   Serial.println(n);
 }}

 void setup() {
   // Initializing Serial Monitor
   Serial.begin(9600);

   // Initialize Ethernet
   initializeEthernet();

   //register udpSerialPrint() to destination port (callback function)
   ether.udpServerListenOnPort(&udpSerialPrint, dstPort);

   // Set the classes
   BaseServo.attach(pinServo_Base);
   ArmServo.attach(pinServo_Arm);
   WristServo.attach(pinServo_Wrist);
   GripperServo.attach(pinServo_Gripper);

   // Set the digital pins
   pinMode(pinIR_Magazine,      INPUT);
   // Create tasks and assign theor priorities
   // Create the Idle task for when there is no task in operation
   // Idle task has the highest priority
   xTaskCreate(RoboticArmTask,          "Robotic   Arm   Task",              1000,   NULL,   1,
&RoboticArmTask_handle);

  } // END of setup function



  void loop() { } // END of loop function



  // function that's responsible for taking  products to sorting
  static void RoboticArmTask(void* parameters) {
   while(1) {
     int p = 0;
     // receive packets, get ready for callbacks
     ether.packetLoop(ether.packetReceive());
     valIR_Magazine = digitalRead(pinIR_Magazine);
     if(valIR_Magazine == 1) {
       Counter++;
       // between magazine and sorting
       for (i = j; i <= 120; i++) {
         BaseServo.write(i);
         vTaskDelay(60/portTICK_PERIOD_MS);
       }
       vTaskDelay(1000/portTICK_PERIOD_MS);
```

```
// arm gonna move down for the magazine
for (x = y; x <= 20; x++) {
  ArmServo.write(x);
  vTaskDelay(20/portTICK_PERIOD_MS);
}
vTaskDelay(1000/portTICK_PERIOD_MS);
// base centering into the magazine
for (j = i; j <= 133; j++) {
  BaseServo.write(j);
  vTaskDelay(60/portTICK_PERIOD_MS);
}
vTaskDelay(1000/portTICK_PERIOD_MS);
i = 0;
// gripper grips
for (g = p; g <= 48.5; g++) {
  GripperServo.write(g);
  vTaskDelay(20/portTICK_PERIOD_MS);
}
vTaskDelay(1000/portTICK_PERIOD_MS);
// arm gonna move up to pick the product
for (y = x; y >= 0; y--) {
  ArmServo.write(y);
  vTaskDelay(20/portTICK_PERIOD_MS);
}
vTaskDelay(1000/portTICK_PERIOD_MS);
// wrist rotating 90
for (w = r; w <= 90; w++) {
  WristServo.write(w);
```

```
    vTaskDelay(20/portTICK_PERIOD_MS);
  }
  vTaskDelay(1000/portTICK_PERIOD_MS);
  // base going to sorting
  for (k = j; k >= 60; k--) {
    BaseServo.write(k);
    vTaskDelay(60/portTICK_PERIOD_MS);
  }
  vTaskDelay(1000/portTICK_PERIOD_MS);
  j = 0;
  // arm moving down again for sorting
  for (x = 0; x <= 15; x++) {
    ArmServo.write(x);
    vTaskDelay(20/portTICK_PERIOD_MS);
  }
  vTaskDelay(3000/portTICK_PERIOD_MS);
  // gripper releases
  for ( p = g; p >= 0; p--) {
    GripperServo.write(p);
    vTaskDelay(20/portTICK_PERIOD_MS);
  }
  vTaskDelay(3000/portTICK_PERIOD_MS);
  WristServo.write(0);
} // Taking product to sorting
else { // everything sets to zero
  for (int a = k; a >= 0; a--) {
    k--;
    BaseServo.write(a);
```

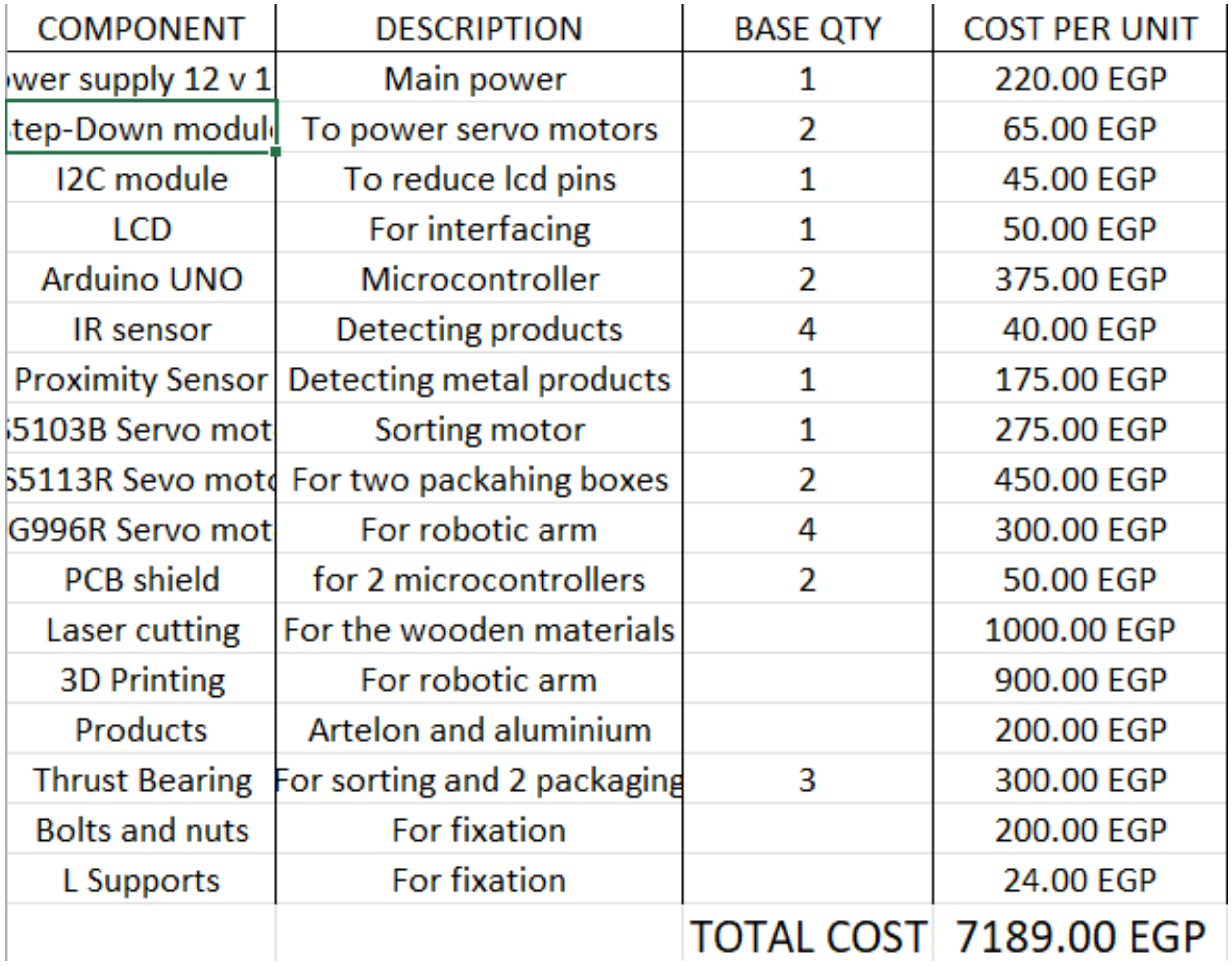
```
      vTaskDelay(60/portTICK_PERIOD_MS);
     }
     for (int a = y; a >= 0; a--) {
      y--;
      ArmServo.write(a);
      vTaskDelay(20/portTICK_PERIOD_MS);
     }
     for (int a = w; a >= 0; a--) {
      w--;
      WristServo.write(a);

      vTaskDelay(20/portTICK_PERIOD_MS);
     }
    } // no product to be found

    BaseServo.write(0);
    ArmServo.write(0);
    WristServo.write(0);
    GripperServo.write(0);

    if(Counter > 7) {
     vTaskDelete(NULL);
    }
   }
  } // END of Base Task function
```

10.0 BILL OF MATERIALS



COMPONENT	DESCRIPTION	BASE QTY	COST PER UNIT
wer supply 12 v 1	Main power	1	220.00 EGP
tep-Down modul	To power servo motors	2	65.00 EGP
I2C module	To reduce lcd pins	1	45.00 EGP
LCD	For interfacing	1	50.00 EGP
Arduino UNO	Microcontroller	2	375.00 EGP
IR sensor	Detecting products	4	40.00 EGP
Proximity Sensor	Detecting metal products	1	175.00 EGP
5103B Servo mot	Sorting motor	1	275.00 EGP
5113R Sevo mot	For two packahing boxes	2	450.00 EGP
G996R Servo mot	For robotic arm	4	300.00 EGP
PCB shield	for 2 microcontrollers	2	50.00 EGP
Laser cutting	For the wooden materials		1000.00 EGP
3D Printing	For robotic arm		900.00 EGP
Products	Artelon and aluminium		200.00 EGP
Thrust Bearing	For sorting and 2 packaging	3	300.00 EGP
Bolts and nuts	For fixation		200.00 EGP
L Supports	For fixation		24.00 EGP
		TOTAL COST	7189.00 EGP

Figure 17:Bill of materials