

**KULLIYYAHOFENGINEERING**

**END-OF-SEMESTER EXAMINATION**

**SEMESTER 1, 2020/2021 SESSION**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Program | : | **Mechatronics Engineering** | Level of Study | : | **UG 4** |
| Time | : | **9.00 am – 9.00 am (16 Jan 2021)** | Date | : | **15 January 2021** |
| Duration | : | **24 hours** | Section(s) | : | **1** |
| Course Code | : | **MCT 4334 / MCTE 4342** | Name | : | MouazAlsamman |
| Course Title | : | **Embedded System Design**  **GitHub Link:**  [moath974/EmbeddedSystem (github.com)](https://github.com/moath974/EmbeddedSystem) | Matric | : | 1524241 |

This Question Paper Consists of **Three (3)** Printed Pages (Including Cover Page) with **Two (2)** Questions.

**INSTRUCTION(S) TO CANDIDATES**

* Total mark of this examination is **40 marks.**
* This examination is worth **40 %** of the total course assessment. ● This is an open book, open notes examination. Answer **ALL QUESTIONS** ● Marks assigned to each question are listed in the margin.
* **Note that one of the conditions to pass the course is to obtain at least 50 % of this examination.**

|  |
| --- |
| **DECLARATION**  By answering this final examination, I hereby declare that:   * The whole answer of this final examination is my own work. * I do not receive any help from any other parties in answering on any part of this final examination. * I do not give any clue, hint or work to other students in answering on any part of this final examination. * I understand that any form of cheating or attempt to cheat is a serious offence which may lead to dismissal. |

**QUESTION 1 (10 marks)**

Smart electronic products are becoming fundamental to the way people live, and embedded systems now permeate everyday life. Microprocessors and microcontrollers can be found in practically every electrically powered product that we use. This ubiquity is helping to shape some of the key trends in embedded systems development, driving both functionality and ease of use.

Here are the top 5 emerging technologies in an embedded system application:

1. Smart Farming
2. Intelligent Transport Systems
3. Edge AI
4. Wearable Devices
5. Building Automation

Choose any 2 emerging technologies and briefly discuss the pushing factors of the trends and the issues behind it with examples in order to illustrate your answer.

**(5)** a) **Smart farming:**

The concept of smart farming is advancing with a good pace in agricultural business. A networked farm can offer several advantages such as useful data collection, high-precision crop control, and automated farming techniques. Out of many advantages offered by Internet of Things (IoT), its ability of innovating the landscape of present farming methods is ground-breaking. IoT sensors provide information on crop yields, pest infestation, soil nutrition, and rainfall. IoT offers precise data which can be utilized to better improve farming techniques.

When the global population surpasses 10 billion, catering to food demands is expected to be the most crucial challenge to the world, according to a recent report on [smart agriculture solutions](http://www.futuremarketinsights.com/reports/smart-agriculture-solution-market). To prevent global starvation, governments across the world have started compelling ago-industrialists and farmers to boost their farm yield. Robust adoption of various smart agriculture solutions is helping them reach that goal. The global smart agriculture solutions market is also gaining prevalence across farming sectors riddled by challenges of water and soil degradation. Smart agriculture solutions are being adopted worldwide, ensuring stability in crop productivity around the world, witnessing a rise in prevalence of sustainable practices for cropland management. Moreover, implementing Smart Farming can help to reduce the expenses such as reduce the number of workers in farm, which is nowadays in some countries takes a big portion of the capital.

Here is a good example for utilizing Embedded system in Smart Farming. The Embedded Decision Support System for Smart Farming, this technical assistance system will help farmers to improve the crop yield by providing information with regard to soil testing based on sensor network in farming such as monitoring ecological conditions like soil moisture, soil temperature, soil fertility, also providing information about weather predictions. This technical system maintains database of farmer, their field with present, previous, and standard crop details. Using “Embedded Decision Support System for Smart Farming” farmers are guided to improve their agricultural production by predicting current crop and also, we can analyze agricultural parameters on Thing Speak through internet. With the proposed work crop health and yield shall be improved and farmers are updated regularly with cultivation information. By using this system farmers can grow the suitable crop and can get desirable yield & profit.

The Embedded Decision Support System for Smart Farming uses simple components like sensors, display units, ZigBee Tx & Rx, ARM7 processors, Voice Bank, and speaker etc., which are easily available in market and smaller in size which makes it portable. The sensor section used in this system gives accurate values of all the parameters necessary to test the soil fertility. When the user presses the button on the sensing section, the respective responses are obtained quickly and display the result on GLCD. The necessary assistance information stored in voice bank is triggered by RF unit as a voice output. Then the necessary parameter values are obtained at Technical Base Station via ZigBee Modems and those values are display on monitor for database purpose. Also, the data is send to IoT site Thing Speak through GSM modem.

**(5)** b)

**What is Build Automation?**

[Build automation](https://flexagon.com/flexdeploy/build-automation/) is the process of creating and building software without direct human intervention. With build automation, tasks that were traditionally the responsibility of a developer are standardized, to become scripted, repeatable, automated steps to moving a new software forward to its final form.

Without automation, software development follows a single-track, step-by-step progression from architecture to deployment. However, this results in a cumbersome, slow, manual process that slows the lifecycle and leads to errors.

With automation, however, an organization can become flexible, agile, and responsive to the changing demands of the business landscape. Because many software build processes are repetitive, they are excellent candidates for automation. And build automation sets the stage for a number of benefits, while providing the foundation for more sophisticated DevOps processes.

Often, build automation involves creating repeatable, standardized processes for compiling source code into binary code, packaging or compressing binary code, creating installers, and running automated tests, and updating to a centralized repository.

**What are the benefits of build automation?**

Build automation is critical to DevOps, as it must be established before an organization can implement other DevOps processes, including continuous integration, continuous delivery, and continuous deployment. In addition to advancing DevOps processes toward maturity and accelerating its benefits, build automation on its own provides a number of benefits, including:

1. Fewer Errors

Manual processes have more variables, and therefore, a higher number of errors than automated, standardized processes.

1. Faster Cycle

When steps are automated to follow one after the other, it streamlines the software delivery process and eliminates time wasted waiting on a developer to move a project forward to the next stage. Additionally, with fewer errors, less time is required to investigate and address problems in the software build.

1. Efficiency

A precursor to automation is standardizing and processes, eliminating unnecessary steps and duplication of effort. This makes the build process more efficient, and resources can be channeled to higher-impact tasks.

1. Transparency

An automated build provides visibility into the process, making it easier to analyze, and identify potential roadblocks or opportunities for improvement. It also expands process visibility to other stakeholders, such as project managers, who can access granular information about different process stages without waiting for employee updates.

1. Scalability

Automated processes are infinitely more scalable than manual processes. As an application grows in size or complexity, build automation can support and enable that growth without requiring enormous investments in additional resources.

**Challenges for Build Automation:**

1. Builds Longer:

It will take more time to run longer builds, increase the wait time of the developer and thus decrease productivity.

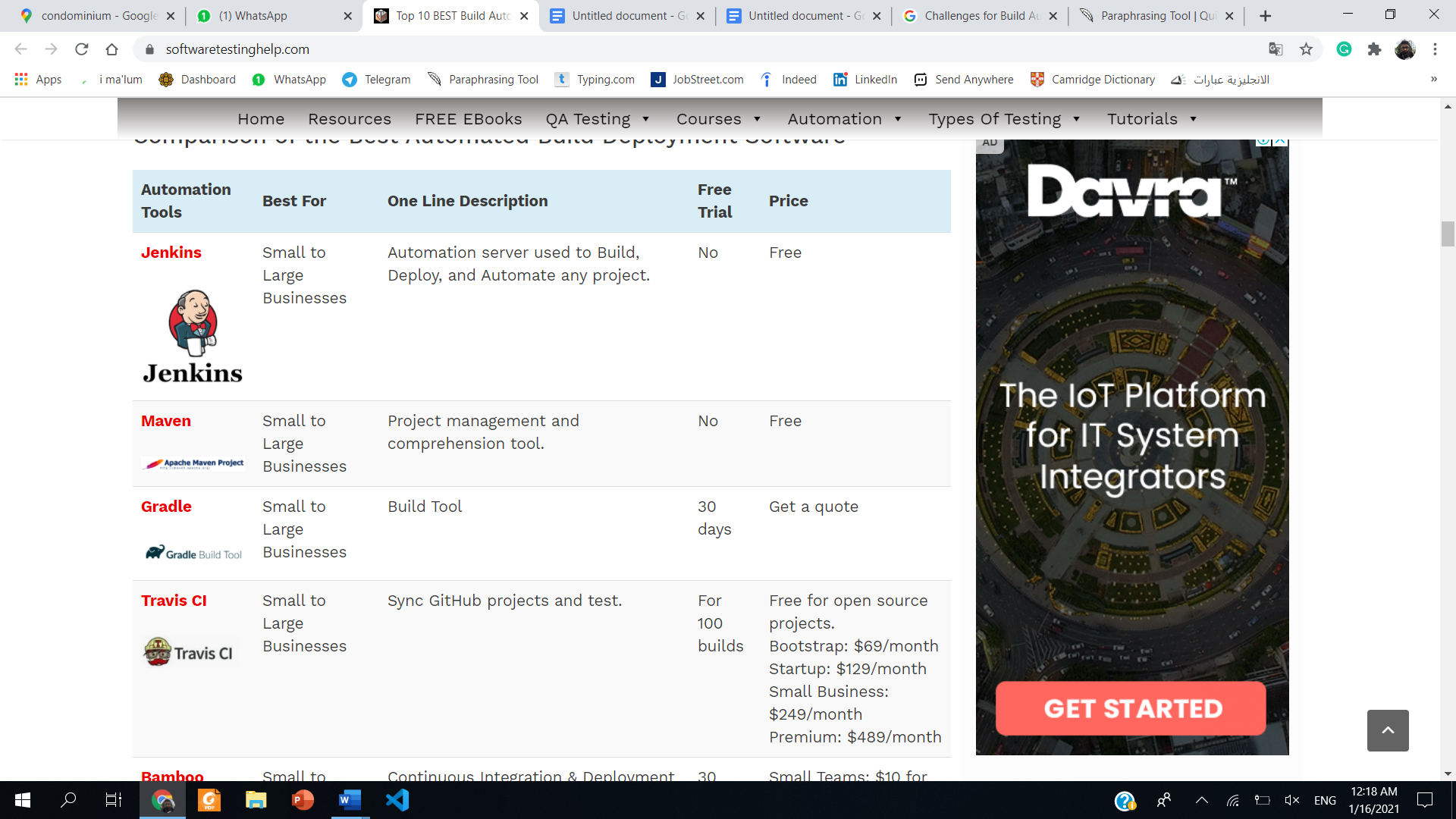
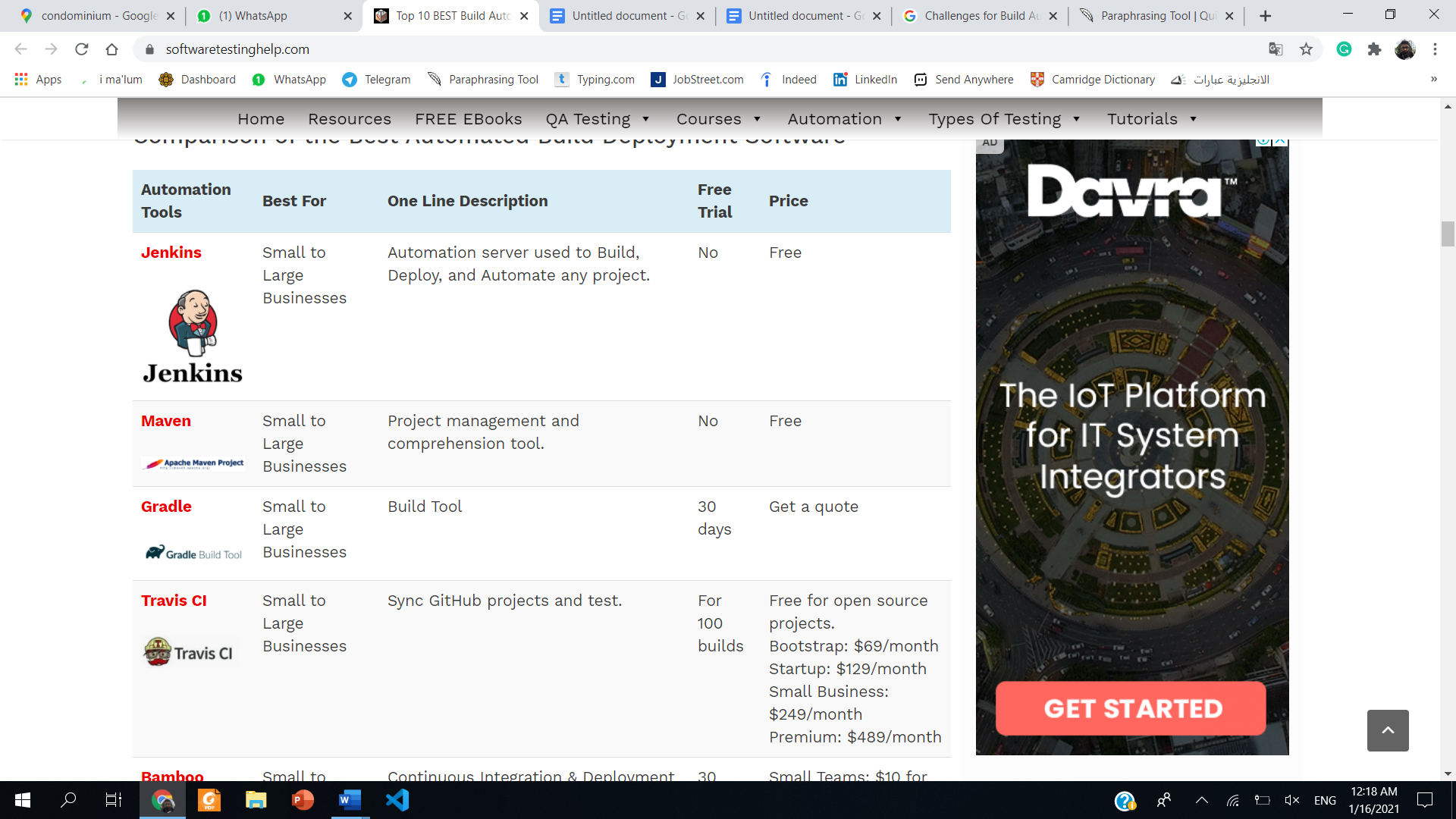
1. Builds Large Volumes:

If a large number of builds are running, then for that specific period, you will have limited access to the build servers.

1. Complex builds:

Complex construction may require extensive manual work and may decrease flexibility.

**Example of the top build automation tools:**



**QUESTION 2 (30 marks)**

The video games are a very powerful force for getting the youths interested in technology. It has way bigger knock-on effects than people may realize.

At age 12, having mastered BASIC, Musk sold the code for his PC game [Blastar](https://www.theverge.com/2015/6/9/8752333/elon-musk-blastar-pc-game) to a PC magazine for approximately $500 (Figure 1). Eleven years later, he and his brother founded Zip2, a company that provided city guides, maps, and yellow pages for the newspaper industry, and which they eventually sold to Compaq for $307 million. Musk says he did most of the coding for Zip2, mostly at night when the software wasn't in use.



**Figure 1: Elon Musk and his Blaster Game.**

“Part of the reason, maybe the reason, I probably wouldn’t have started programming if it weren’t for video games or wouldn’t have been as interested in computers and tech if it weren’t for video games. “Musk said.

Meanwhile, Breakout Game was one of the most popular games in the long history of the Atari 2600. Head Engineer Al Alcorn assigned Atari employee Steve Jobs, a lowlevel technician working a nightshift to design the single-player game, which was similar to Pong (Figure 2). At a time when microchips were extremely expensive, Atari.

**2**

were anxious to reduce the cost of each Breakout coin-op and offered Jobs a $100 bonus for every chip he managed to eliminate from the initial concept. Jobs in turn asked his friend Steve Wozniak to do it together.



**Figure 2: Steve Jobs and Wozniak working for Atari Breakout Game.**

Video game and arcade consoles are embedded systems, comprising of many components all serving a specific function, allowing the system to take input from the player and relay the outputs on a screen display. Present-day video game console systems generally consist of several embedded components.

Nowadays, there are many integrated embedded systems development tools (software/hardware) are available in the market, but your choice is dependent on the type of microcontroller and processor you are using for your development.

1. Discuss the tools that are available based on the above stories. **(2)**

Atari Video Computer System (VCS) changed video game history with its incorporation of microprocessors within its infrastructure. Before this, video games relied on a core board with transistors and diodes.

the Atari 2600 was accredited with popularizing the incorporation of microprocessors within consoles. For this console, the MOS 6502 microprocessor was used. The Atari 2600 also incorporated 128 bytes of RAM and 4-kilobyte ROM (read-only memory) chips loaded with software, and these ROM chips were stored in removable cartridges, allowing users to easily swap various games using the same hardware. The Atari VCS also comprised of a custom graphics chip called Stella, which allowed the system to sync with the television generating screen display and sound effects.

6502 assembly language Batari Basic. As the 2600 uses the 6507, a variant of the MOS Technology 6502 chip, as its CPU, most homebrews released are written in 6502 assembly language.

Hardware:



1. Console:

The Atari 2600's CPU is the MOS Technology 6507, running at 1.19 MHz.

The console has 128 bytes of RAM for scratch space.

1. Graphics: the video device provides two 8-pixel bitmapped sprites, two 1-pixel "missile" sprites, a 1-pixel "ball", and a 40-pixel "playfield" that is drawn by writing a bit pattern for each line into a register just before the television scans that line. The Atari 2600 was designed to be compatible with the cathode-ray tube television sets produced in the late 1970s and early 1980s and uses different color palettes depending on the television signal format.
2. Controllers: The VCS originally shipped with two types of controllers: a joystick (part number CX10) and pair of rotary paddle controllers (CX30). Driving controllers, which are similar to paddle controllers but can be continuously rotated, shipped with the Indy 500 launch game. After less than a year, the CX10 joystick was replaced with the CX40 model.

Some internals components:

* The motherboard of the 2600 is dominated by an ominous metal box, likely the EMI shield covering the ICs. 1.19MHz 8-bit processor
* 128 bytes RAM
* 192 x 160-pixel resolution
* 128 colors, with max 4 colors per line
* 2 channel mono sound
* Atari's custom chip, the Television Interface Adapter (TIA) is the moneymaker of the 2600, as it allowed for multiple colors, increased graphic capabilities, and sound.
* There are six components of video that the TIA can create: The playing field, Two sprites (8-pixel lines), a "ball" (single pixel), and two "missiles" (two-pixel lines). Combinations of these elements allowed for the complex video games witnessed in the 2600.

1. Design and develop an Arduino-based games project with the following steps:
   1. Describe the game and the uniqueness of the proposed system. **(3)**

I am proposing a game that is going to be a fun game using very basic components I chose this project since it can be easy and fun since this year we stayed at home a lot it should be easy for a mechatronic student to apply some of the knowledge by understanding a simple code and apply it in a real life even though this code is about a game but since we can make this can and understand it very well it will be easy for us to do more complicated projects:

**Features:**

* Splash screen that waits until you are ready.
* When you are holding the SELECT button to jump, *score* is stopped from incrementing so that you can't just hold down the button the entire time.
* Custom characters for the dinosaur, cacti, birds, and blocks.
* Death screen.
* High score saved in internal EEPROM.

**How to Play**

* Get closer to the box so the box will open, and the Arduino will be initialized.
* Press SELECT to start.
* Pres SELECT to jump.
* When you die, reset using the SELECT button.
  1. Design the system architecture or block diagram of the operation. **(3)**

LCD and Pushbutton

Servo motor

software

Ultra-sonic sensor

Arduino uno

CONTROL UNIT

O/I

O/I

O/I

I/O

LCD

I/O

Buzzer

Servo

Push Button

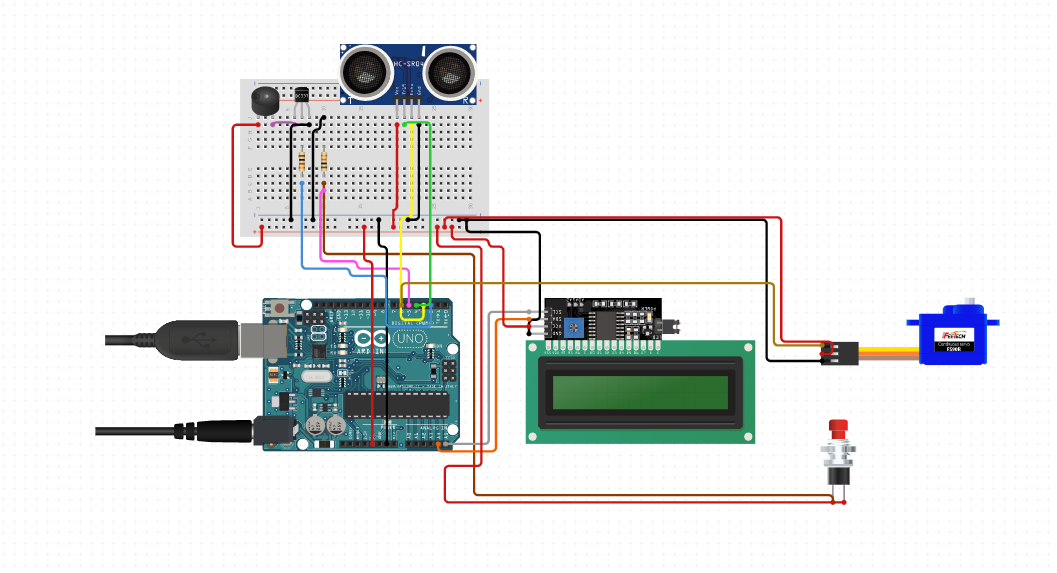
Ultra-Sonic sensor

The program starts as following:

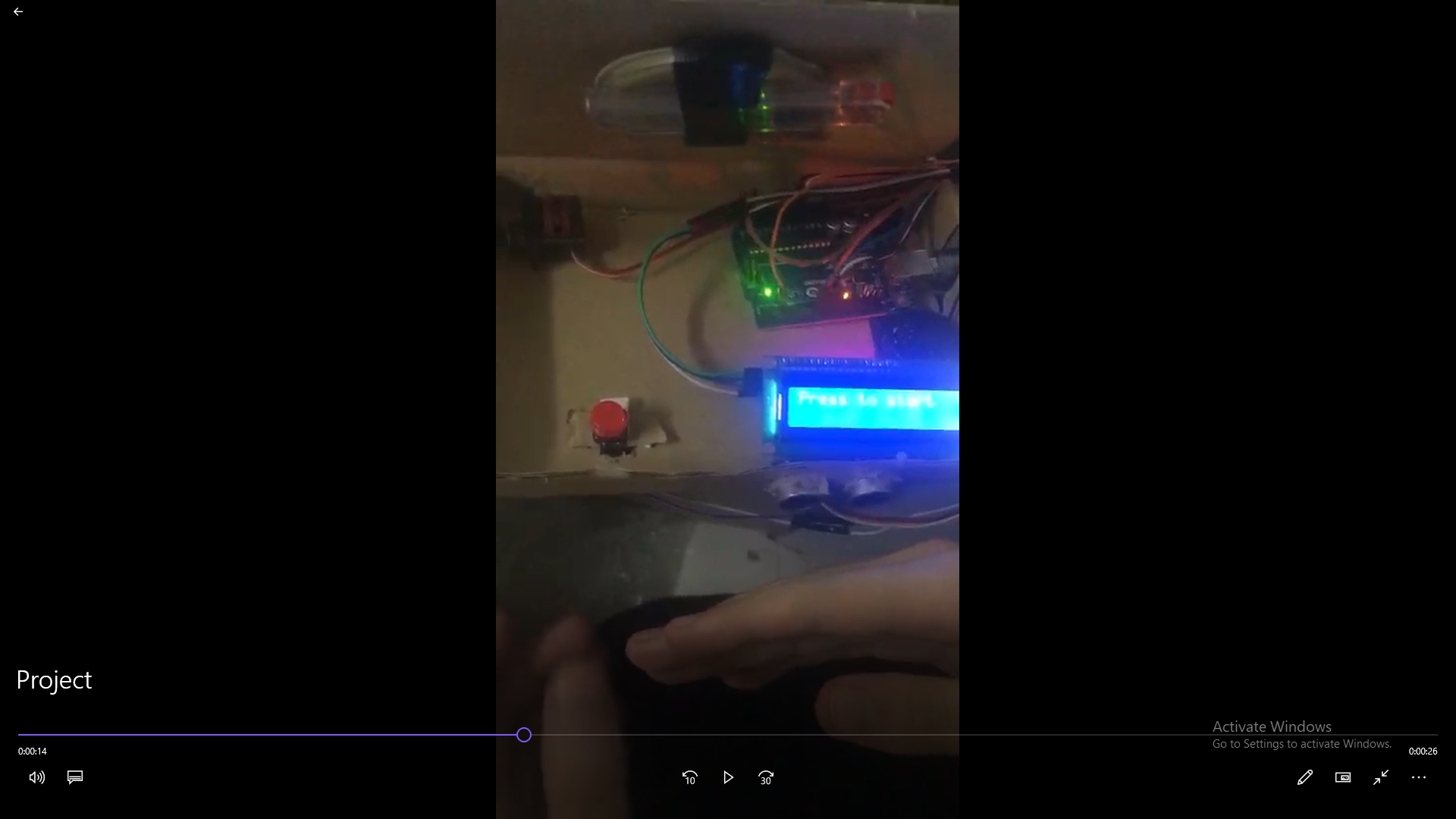
1. When we want to open the box, we go near at and wave to the ultra-sonic sensor so the box will open, and we can use the LCD and pushbutton.
2. After opening the box LCD will be on and we can see the display sentence “Press Select to start”
3. To start the game, we press on the PUSHBUTTON and start playing the game.
4. The game will start by sending a few trees as an obstacle and the number will increase as long as we are winning.
5. We have to press the PUSHBUTTON whenever we see a tree (Press it until u are above the tree)
6. If we hit any tree the game will finish and “Game Over” will display.
7. If we want to play again, we can press the PUSHBUTTON again.
8. If we are bored and we want to stop just leave the game and the box will automatically closed since the ultra-sonic cannot recognize any objective near it.
9. Thank you it was a great time creating a game for you.
   1. Built a programming flowchart. **(3)**



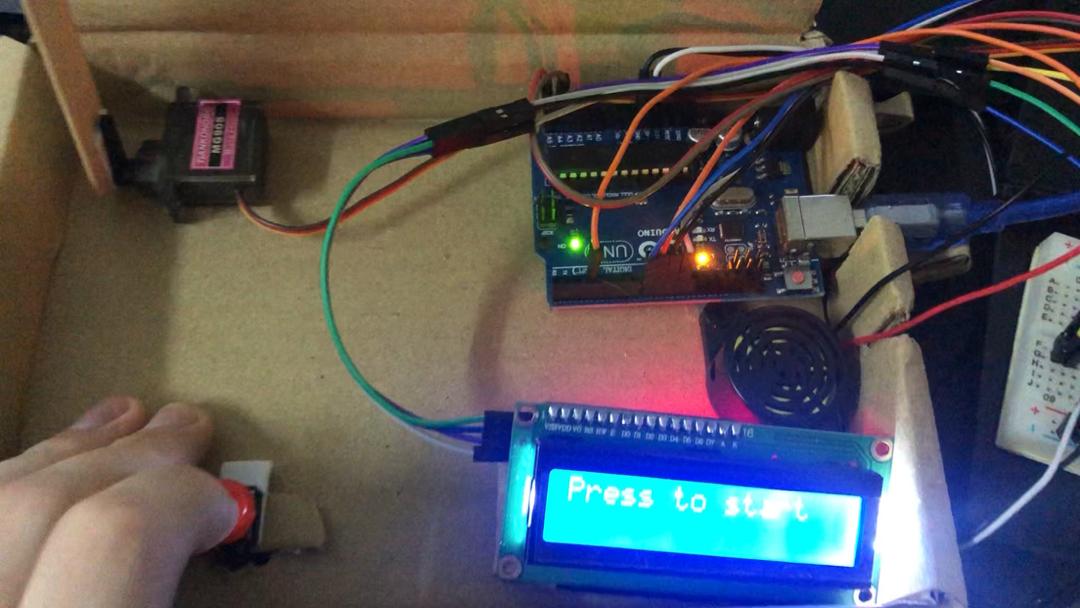
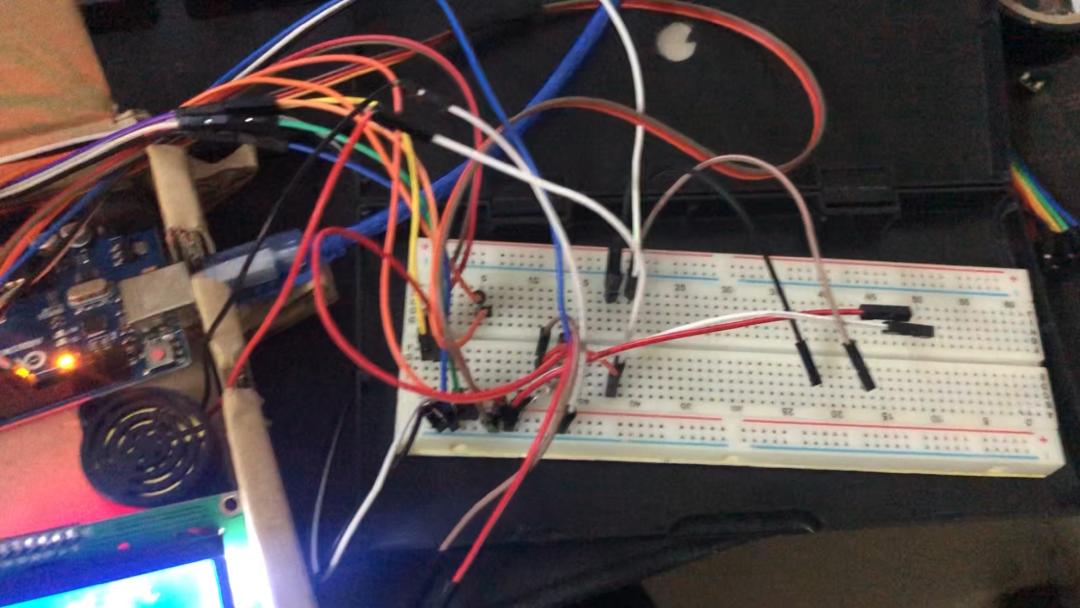
* 1. Construct the circuit design which includes interfaces with the sensors, actuators, and LCD. **(3)**



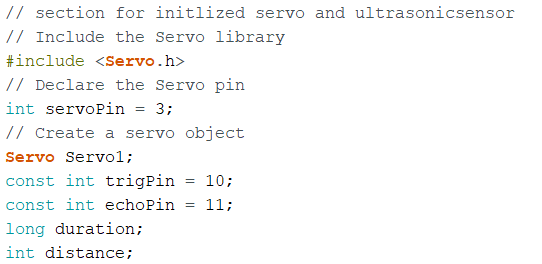
|  |  |
| --- | --- |
| Equipment | price |
| Arduino uno | 25 |
| LCD 16x2 12c | 13 |
| Servo motor | 10 |
| Buzz | 2.5 |
| Bush button | 0.9 |
| Ultra-sonic sensor | 3.5 |
| Resistance | 1 |
| total | 55.9 |



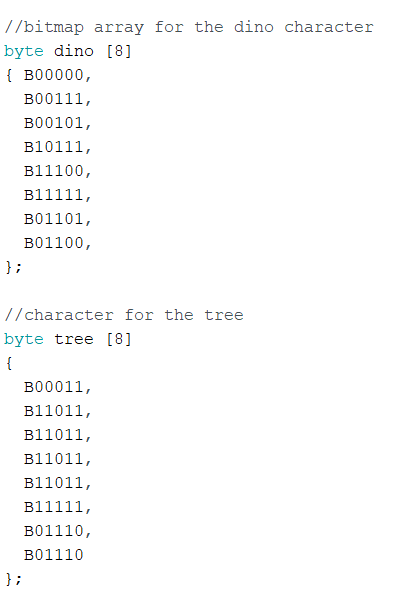




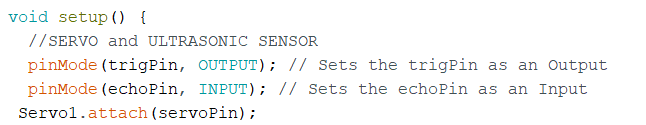
* 1. Write the programming codes with details comments. **(8)**

****

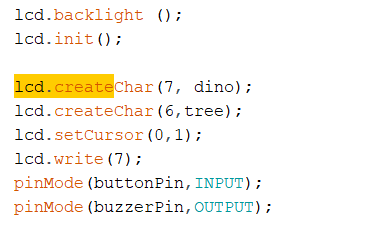
Those commands for the servo and the ultra-sonic sensor



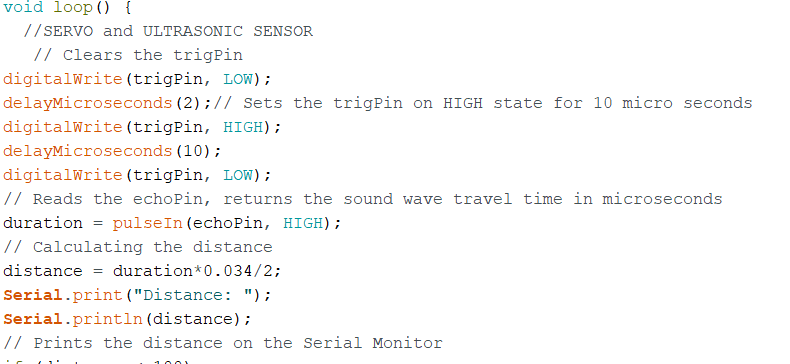
This is used to define the shape of the character for the tree (obstacle) or the player.



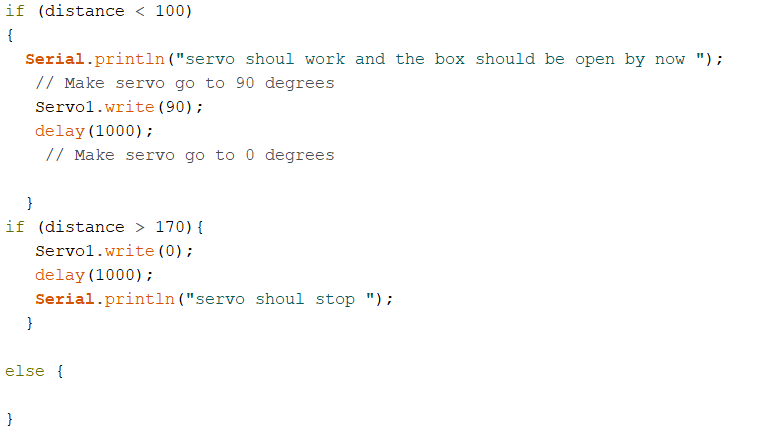
Evaluate which pin is output and which pin is input for servo and ultrasonic sensor

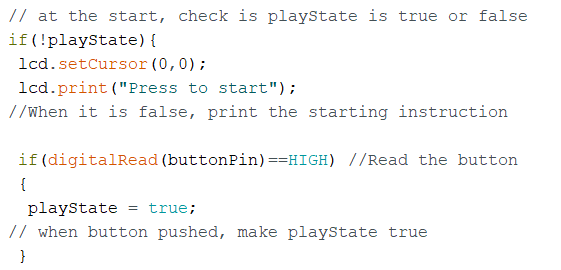


These commands are for creating the array which you wrote as a character and for further displaying it in the lcd screen. And for buzz pin and button pin.



Those command to get the value from ultrasonic sensor and calculate into distance.

those conditions to open the box so when I reach the box the box should be opened and when I leave it, it should closed.

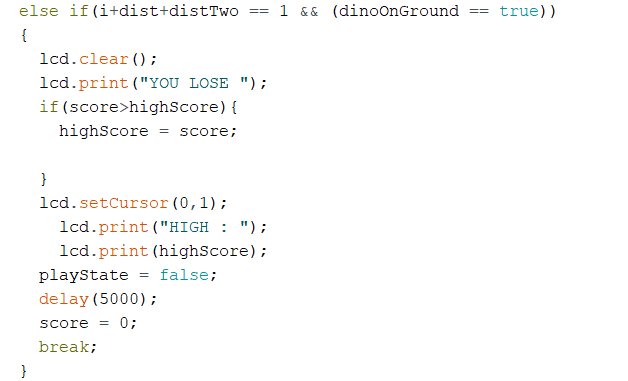


Print on the 0,0 value from screen which is the top right corner Press To Start

And read the value from the pushbutton.



Generate random value for dist and distTwo between 4 and 9 so this distance considers the gap between the trees. Then reed the pushbutton if pressed and jump and generate sound.

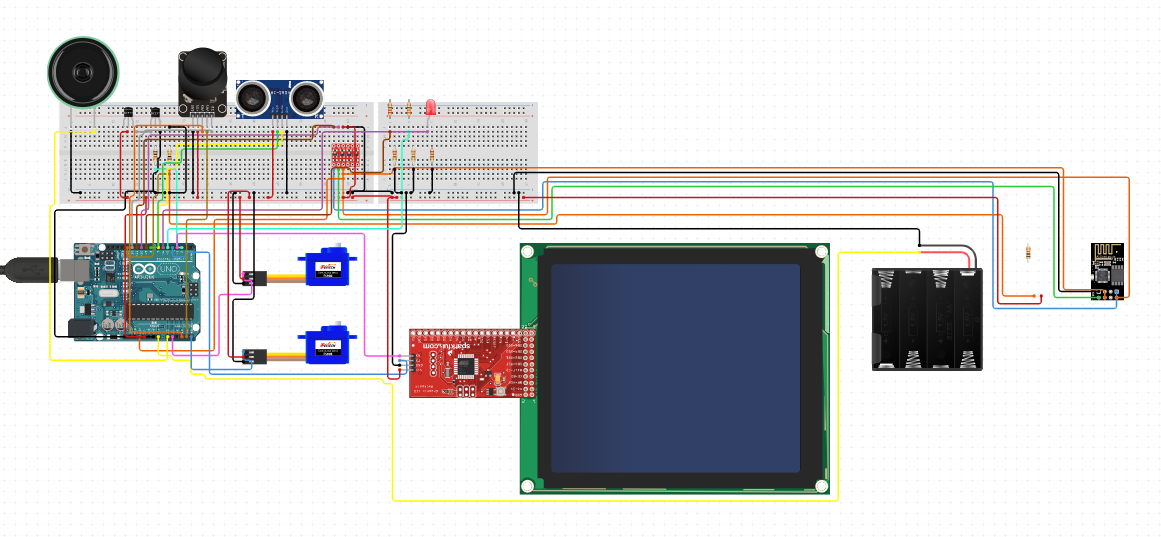
those condition for losing and if I got high score save it in the EEPROM memory.

The Code File will be provided in the GitHub Link.

* 1. Record the demo video and upload to GitHub (Folder name “ESD/Final Exam”). **(4)**
  2. Analyze the system limitations and recommendations for future development. **(4)**

It can be seen from the videos that the project has so many limitations and can be improve a lot if we have more equipment and more time. Some of the obvious limitation is by having only one servo and using a broken carton as a frame for the project I was planning to connect this project to IOT using a ESP32 NODEMCO, but I could not do it due to missing equipment I also could not found a Wi-Fi module for Arduino uno. The ultra-sonic sensor used is not in a perfect condition it was very tiring dealing with it since I use it a lot before, so I prefer using more accurate sensor. We can improve this project more by adding a prize for the one who get a new high score. This can be done by adding a motor and a simple mechanism to through the prize to the new winner. It also can be improved if we make a suitable frame and good wiring. This can help us to teach other kids specially since we are doing games and those kids will love to know how to create games using simple components. It also so easy for me now to upgrade on the game and build more stages since the code have already been done and need some modification.

**Future work**



I added a lot of component that’s will make the game more realistic and it will make it more portable as we can see we can take it anywhere since we use batteries now and for the movement we can see by changing from the push button to the controller its more obvious that it will be easier and we used a bigger screen so we can make the game with higher lever I also connected a Wi-Fi module in case I wanted to connected online and also add speaker so we can play the game using voice recognition and for servo motor we will try to create the mechanism to show the price to the new winner.

**END OF PAPER**