PICO SNAKE A FUN GAME

A PROJECT REPORT

Submitted

In the partial fulfillment of the requirements for the Mid-Term Project Assignment Evaluation of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

by

BALI BABU CHAUHAN [19410121182]

RAJSHRI GUPTA [1941012812]

UNDER THE GUIDANCE OF

Dr. Hara Prasad Tripathy

Assistant Professor

Dr. Siddharth Sahany

Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INSTITUTE OF TECHNICAL EDUCATION AND RESEARCH

SIKSHA 'O' ANUSANDHAN (DEEMED TO BE UNIVERSITY)

BHUBANESWAR-751003, ODISHA, INDIA.

DECEMBER-2022

DECLARATION

I hereby declare that the work described in this thesis "**PICO SNAKE A FUN GAME**" which is being submitted in partial fulfillment for the award of **Bachelor of Technology** in the Department of **Computer Science and Engineering** affiliated to Siksha 'O' Anusandhan (Deemed to be University), Bhubaneswar (Odisha) is the result of investigations carried out by me under the Guidance of **Dr. Siddharth Sahany Assistant Professor Centre for Internet of Things and** of **Dr. Hara Prasad Tripathy Assistant Professor Centre for Internet of Things**, **Institute of Technical Education and Research (ITER), Bhubaneswar.**

.

The work is original and has not been submitted for any Degree/Diploma of this or any other university.

Place: Bhubaneswar

Date: 17th December, 2022

BALI BABU CHAUHAN [19410121182]

RAJSHRI GUPTA [1941012812]

CERTIFICATE

This is to certify that the project report entitled "***PICO SNAKE A FUN GAME***" being submitted by **BALI BABU CHAUHAN [19410121182]** and **RAJSHRI GUPTA [1941012812]**

in partial fulfillment for Mid-Term Project Assignment Evaluation of **Bachelor of Technology in** the Department of **Computer Science and Engineering** affiliated to Siksha 'O' Anusandhan (Deemed to be University), Bhubaneswar (Odisha)**,** is a record of bonafide work carried out by them during the academic year 2022-2023 under our guidance and supervision.

The results embodied in the report have not been submitted to any other University or Institution for the award of any degree or diploma.

(Dr. Hara Prasad Tripathy)
Project Guide

(Dr. Siddharth Sahany)
Project Guide

ACKNOWLEDGEMENTS

I thank Dr. Hara Prasad Tripathy, Associate Professor of Center for Internet of Things (CIoT), Institute of Technical Education and Research (ITER), Siksha 'O' Anusandhan (Deemed to be University), Bhubaneswar for his valuable advice, support and help during my project work.

I thank Dr. Siddharth Sahany, Assistance Professor of Center for Internet of Things (CIoT), Institute of Technical Education and Research (ITER), Siksha 'O' Anusandhan (Deemed to be University), Bhubaneswar for her valuable comments and suggestions that greatly helped in improving quality of report.

I thank to all my teachers and professors for their valuable comments after reviewing my project reports.

I wish to extend my special thanks to all my colleagues and friends who helped directly or indirectly to complete my project assignment work.

I extend my thanks to my parents and all my family members for their unceasing encouragement and support who wished me a lot to complete this work.

Signature of Students

ABSTRACT

The main purpose of the project is make a full fledged computer mobile game in raspberry pi pico. The basic premise of the game remains the same – you die if you hit the walls or run into yourself. And the snake speeds up as it grows longer. What makes this implementation of the Snake game unique is that the snake slithers as it moves during gameplay. This game shows that what big things can be achieved with the given limited resources. Also it involves the gui and interfaces designed for the interactions.

INDEX

TOPICS

- **Declaration** i
- **Certificates** ii
- **Acknowledgement** iii
- **Abstract** iv

CHAPTER 1: INTRODUCTION

1.1 Introduction of the project …… 1

1.2 Project overview…… 2

CHAPTER 2: INTERNET OF THINGS

2.1 Introduction to IOT…… 4

2.2 Need of Internet of Things…… 7

2.3 Applications of Internet of Things…… 10

CHAPTER 3: HARDWARE DESCRIPTION

3.1 Introduction with block diagram…… 13

3.2 Raspberry Pi Pico …… 16

3.3 OLED Display…….. 25

3.4 Push Button…….

3.5 Buzzer…….

CHAPTER 4: SOFTWARE DESCRIPTION

4.1 Thonny IDE……28

4.3 Wokwi online simulator working and operations ……

CHAPTER 5: Project Description……

CHAPTER 6: Advantages, Disadvantages and Applications……

CHAPTER 7: Results, Conclusion, Future Prospects……

CHAPTER 8: Team Work……

CHAPTER 1: INTRODUCTION

1.1 **Introduction:**

The worst thing about modern smartphones and handheld videogame consoles is the lack of Snake. If you can't play a quick game of Snake during meetings, then what's the point?.

I remember my first dumb phone, which was called a "phone" back then. It could play simple games like Snake. That was bleeding edge. Prior to that, you'd have to carry a separate handheld that played Tetris or a more dynamic gaming system like a Gameboy. The Raspberry Pi Pico is a pretty powerful microcontroller compared to what handhelds ran on in the 90s. Naturally, it offers us the ability to play games from that era.

This is a fun snake game where the snake moves around and eats the food. The size of snake and position of food changes as the game proceeds further. In the pico snake game snake moves around the oled screen and the movements are made so that the snake can eat the food. As the snake eats the food, the size of the snake increases and the position of the food is changed.

This goes on repeatedly until the snake collides with any end if the screen or the snake eats it's own tail and when that happens "Game Over" is displayed on the screen and a sound comes from the buzzer

The main objectives of the project are:

1. Movement of snake are controlled by the four push buttons arranged in hand compatible manner.
2. The visual affects are maintained with the codes like snake eating foods and random food generation.
3. Displaying the status if you strike the wall or when the snake tries to eat itself.
4. Controlling the snake with the buttons with haptic feedback.
5. Show the scores when the game finishes to know the status.

1.2 Project Overview:

The project "**Snake Game "** using raspberry pi pico and micropython.

The project explains the implementation of **"Snake Game"** using micropython . The organization of the thesis is explained here with:

Chapter 1 Presents introduction to the overall thesis and the overview of the project. In the project overview a brief introduction of Mcropython, OLED ssd1306, push buttons, and its applications are discussed.

Chapter 2 Presents the topic Internet Of Things. It explains about what is Internet Of Things, Need for Internet Of Things and its applications.

Chapter 3 Presents the hardware description. It deals with the block diagram of the project and explains the purpose of each block. In the same chapter the explanation of

Chapter 4 Presents the software description. It explains the implementation of the project using Thonny IDE for micropython.

Chapter 5 Presents the project description along with visualizations.

Chapter 6 Presents the advantages, disadvantages and applications of the project.

Chapter 7 Presents the results, conclusion and future scope of the project.

CHAPTER 2: INTERNET OF THINGS (IOT)

Introduction:

The Internet of Things (IoT) is an important topic in technology industry, policy, and engineering circles and has become headline news in both the specialty press and the popular media. This technology is embodied in a wide spectrum of networked products, systems, and sensors, which take advantage of advancements in computing power, electronics miniaturization, and network interconnections to offer new capabilities not previously possible. An abundance of conferences, reports, and news articles discuss and debate the prospective impact of the “IoT revolution”—from new market opportunities and business models to concerns about security, privacy, and technical interoperability. It is basically connecting the electronic items to the cloud and in other words internet in order to make them more useful and worthful.

The large-scale implementation of IoT devices promises to transform many aspects of the way we live. For consumers, new IoT products like Internet-enabled appliances, home automation components,there are many uses to household people and energy management devices are moving us toward a vision of the “smart home’’, offering more security and energy efficiency. Other personal IoT devices like wearable fitness and health monitoring devices and network enabled medical devices are transforming the way healthcare services are delivered. This technology promises to be beneficial for people with disabilities and the elderly, enabling improved levels of independence and quality of life at a reasonable cost. IoT systems like networked vehicles, intelligent traffic systems, and sensors embedded in roads and bridges move us closer to the idea of “smart cities’’, which help minimize congestion and energy consumption. IoT technology offers the possibility to transform agriculture, industry, and energy production and distribution by increasing the availability of information along the value chain of production using networked sensors. There are lots of sensors in this electronic world using iot we can make them more worthful than ever.However, IoT raises many issues and challenges that need to be considered and addressed in order for potential benefits to be realized.

A number of companies and research organizations have offered a wide range of projections about the potential impact of IoT on the Internet and the economy during the next five to ten years. Cisco, for example, projects more than 24 billion Internet–connected objects by 2019; Morgan Stanley, however, projects 75 billion networked devices by 2020. Looking out further and raising the stakes higher, Huawei forecasts 100 billion IoT connections by 2025. McKinsey Global Institute suggests that the financial impact of IoT on the global economy may be as much as $3.9 to $11.1 trillion by 2025. While the variability in predictions makes any specific number questionable, collectively they paint a picture of significant growth and influence.There are enormous number of things we could see in future.

History:

The term “Internet of Things” (IoT) was first used in 1999 by British technology pioneer Kevin Ashton to describe a system in which objects in the physical world could be connected to the Internet by sensors. Ashton coined the term to illustrate the power of connecting Radio-Frequency Identification (RFID) tags used in corporate supply chains to the Internet in order to count and track goods without the need for human intervention. Today, the Internet of Things has become a popular term for describing scenarios in which Internet connectivity and computing capability extend to a variety of objects, devices, sensors, and everyday items.

While the term “Internet of Things” is relatively new, the concept of combining computers and networks to monitor and control devices has been around for decades. By the late 1970s, for example, systems for remotely monitoring meters on the electrical grid via telephone lines were already in commercial use. In the 1990s, advances in wireless technology allowed “machine–to–machine” (M2M) enterprise and industrial solutions for equipment monitoring and operation to become widespread. Many of these early M2M solutions, however, were based on closed purpose–built networks and proprietary or industry–specific standards, rather than on Internet Protocol (IP)–based networks and Internet standards.

Using IP to connect devices other than computers to the Internet is not a new idea. The first Internet “device”—an IP–enabled toaster that could be turned on and off over the Internet—was featured at an Internet conference in 1990. Over the next several years, other “things” were IP–enabled, including a soda machine at Carnegie Mellon University in the US and a coffee pot in the Trojan Room at the University of Cambridge in the UK (which remained Internet–connected until 2001). From these whimsical beginnings, a robust field of research and development into “smart object networking” helped create the foundation for today’s Internet of Things.

From a broad perspective, the confluence of several technology and market trends is making it possible to interconnect more and smaller devices cheaply and easily:

• **Ubiquitous Connectivity**—Low–cost, high–speed, pervasive network connectivity, especially through licensed and unlicensed wireless services and technology, makes almost everything “connectable’’.

• **Widespread adoption of IP–based networking**— IP has become the dominant global standard for networking, providing a well–defined and widely implemented platform of software and tools that can be incorporated into a broad range of devices easily and inexpensively.

• **Computing Economics**— Driven by industry investment in research, development, and manufacturing, Moore’s law continues to deliver greater computing power at lower price points and lower power consumption.

• **Miniaturization**— Manufacturing advances allow cutting-edge computing and communications technology to be incorporated into very small objects. Coupled with greater computing economics, this has fueled the advancement of small and inexpensive sensor devices, which drive many IoT applications.

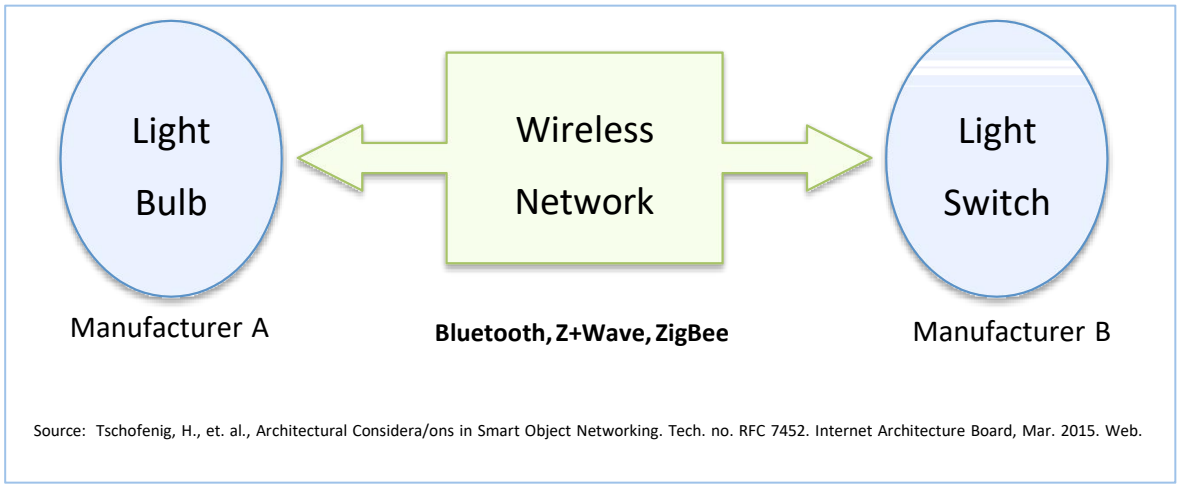
• **Advances in Data Analytics**— New algorithms and rapid increases in computing power, data storage, and cloud services enable the aggregation, correlation, and analysis of vast quantities of data; these large and dynamic datasets provide new opportunities for extracting information and knowledge.

• **Rise of Cloud Computing**– Cloud computing, which leverages remote, networked computing resources to process, manage, and store data, allows small and distributed devices to interact with powerful back-end analytic and control capabilities.

From this perspective, the IoT represents the convergence of a variety of computing and connectivity trends that have been evolving for many decades. At present, a wide range of industry sectors – including automotive, healthcare, manufacturing, home and consumer electronics, and well beyond are considering the potential for incorporating IoT technology into their products, services, and operations.

Internet of Things Communications Models:

The device-to-device communication model represents two or more devices that directly connect and communicate between one another, rather than through an intermediary application server. These devices communicate over many types of networks, including IP networks or the Internet. Often, however these devices use protocols like Bluetooth, Z-Wave, or ZigBee to establish direct device-to-device communications, as shown in Figure 1.



Source: Tschofenig, H., et. al., Architectural Considera/ons in Smart Object Networking. Tech. no. RFC 7452. Internet Architecture Board, Mar. 2015. Web.

Figure 1. Example of device-to-device communication model.

Device-to-Cloud Communications:

In a device-to-cloud communication model, the IoT device connects directly to an Internet cloud service like an application service provider to exchange data and control message traffic. This approach frequently takes advantage of existing communications mechanisms like traditional wired Ethernet or Wi-Fi connections to establish a connection between the device and the IP network, which ultimately connects to the cloud service. This is shown in Figure 2.

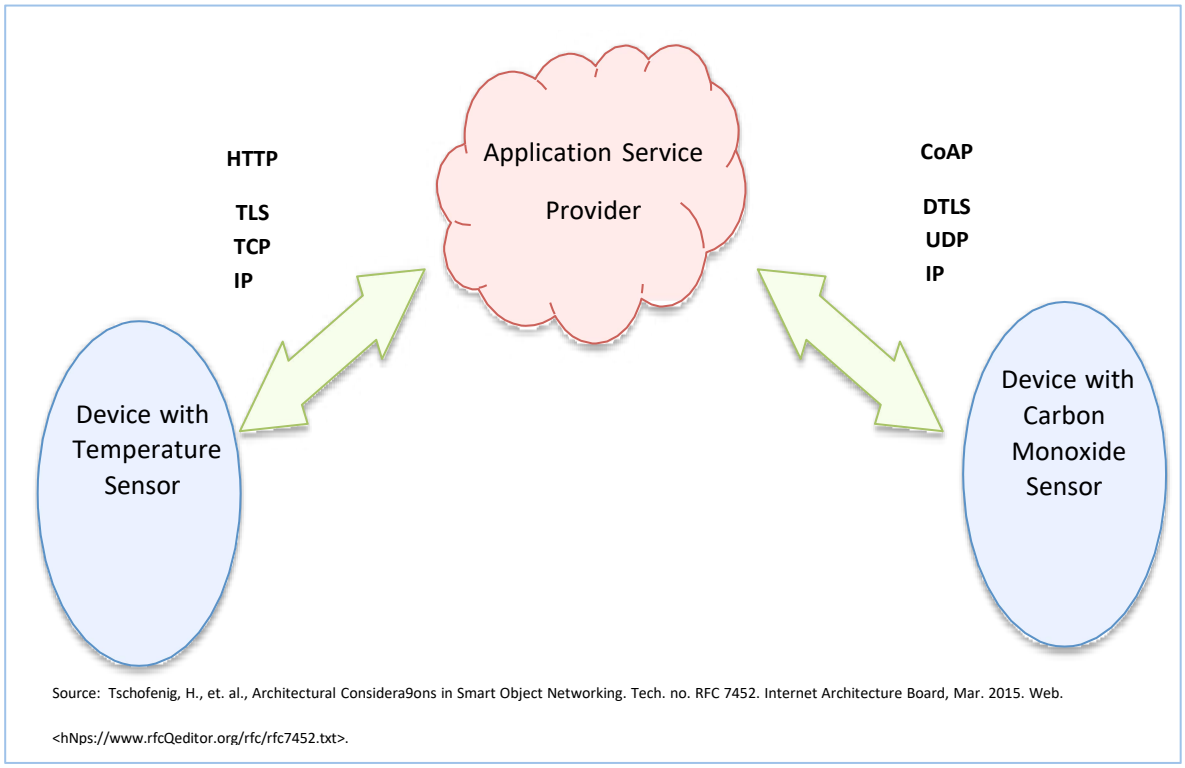


Figure 2. Device-to-cloud communication model diagram.

Device-to-Gateway Model:

In the device-to-gateway model, or more typically, the device-to-application-layer gateway (ALG) model, the IoT device connects through an ALG service as a conduit to reach a cloud service. In simpler terms, this means that there is application software operating on a local gateway device, which acts as an intermediary between the device and the cloud service and provides security and other functionality such as data or protocol translation. The model is shown in Figure 3

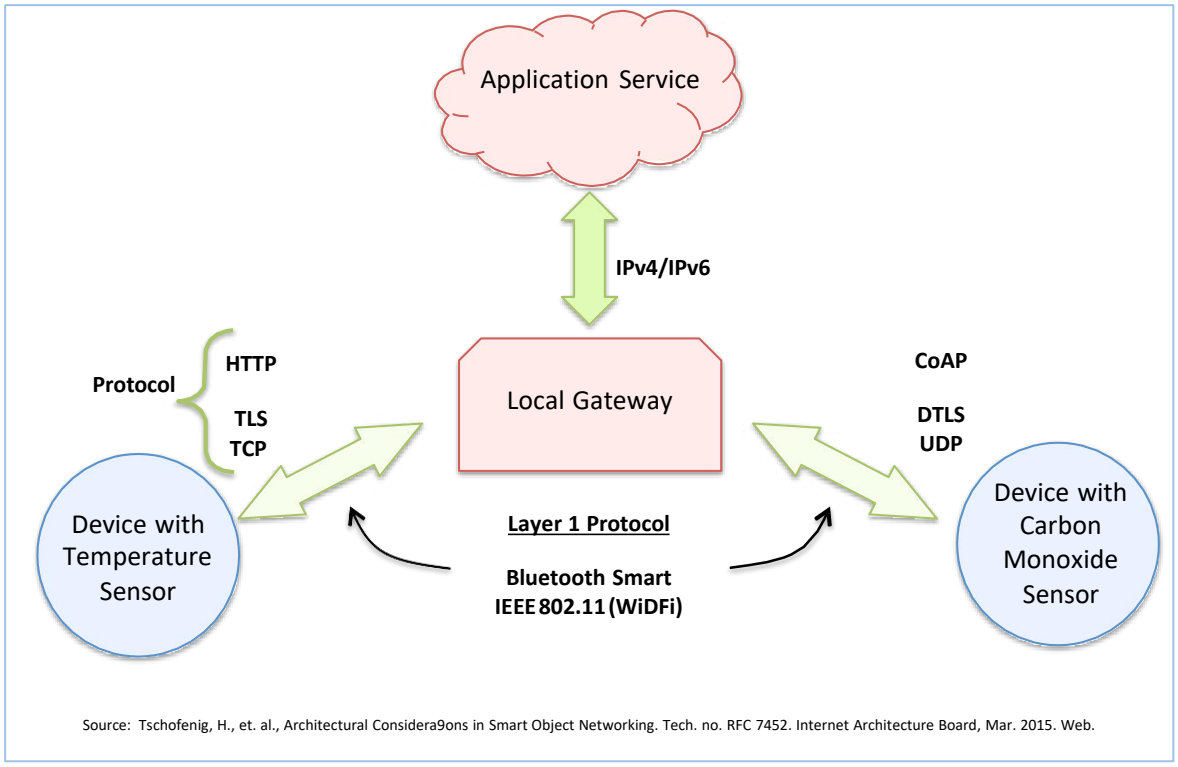


Figure 3. Device-to-gateway communication model diagram.

Back-End Data-Sharing Model:

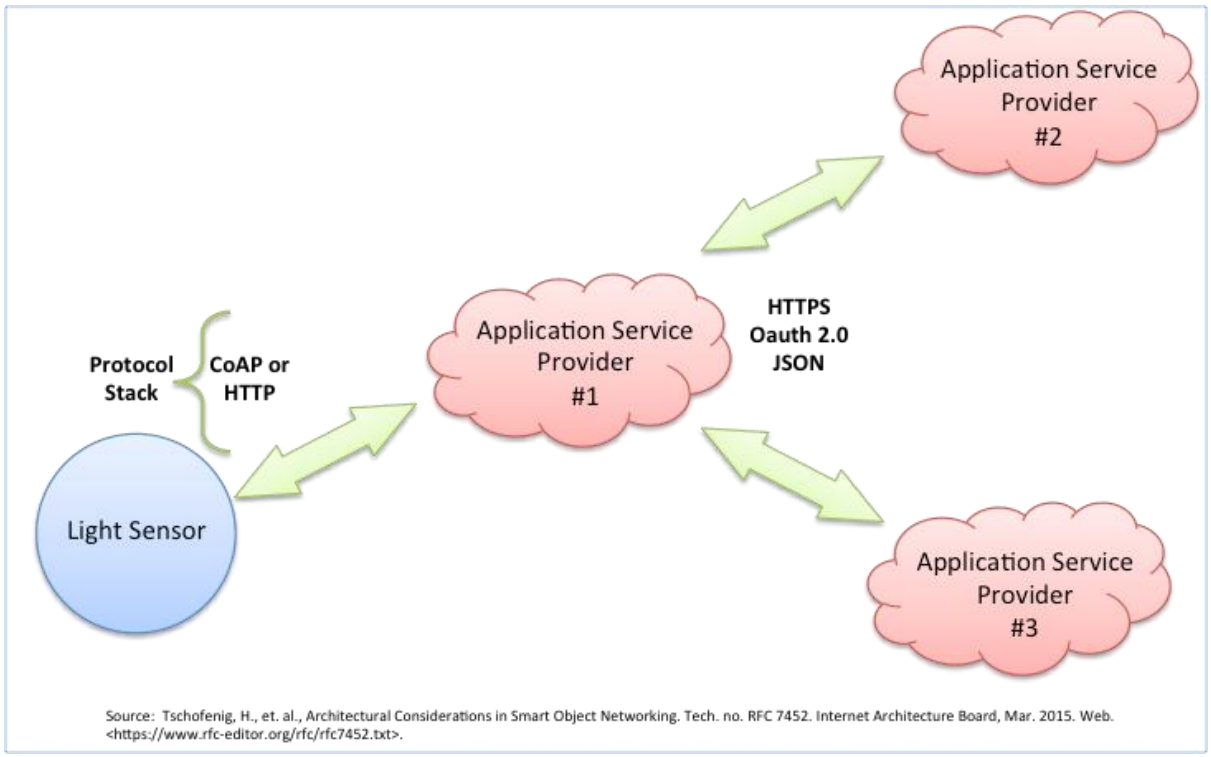


Figure 4. Back-end data sharing model diagram.

The back-end data-sharing model refers to a communication architecture that enables users to export and analyze smart object data from a cloud service in combination with data from other sources. This architecture supports “the [user’s] desire for granting access to the uploaded sensor data to third parties”. This approach is an extension of the single device-to-cloud communication model, which can lead to data silos where “IoT devices upload data only to a single application service provider’’. A back-end sharing architecture allows the data collected from single IoT device data streams to be aggregated and analyzed.

For example, a corporate user in charge of an office complex would be interested in consolidating and analyzing the energy consumption and utilities data produced by all the IoT sensors and Internet-enabled utility systems on the premises. Often in the single device-to-cloud model, the data each IoT sensor or system produces sits in a stand-alone data silo. An effective back-end data sharing architecture would allow the company to easily access and analyze the data in the cloud produced by the whole spectrum of devices in the building. Also, this kind of architecture facilitates data portability needs. Effective back-end data- sharing architectures allow users to move their data when they switch between IoT services, breaking down traditional data silo barriers.

The back-end data-sharing model suggests a federated cloud services approach or cloud applications programmer interfaces (APIs) are needed to achieve interoperability of smart device data hosted in the cloud. A graphical representation of this design is shown in Figure 4.

Different Definitions, Similar Concepts:

Despite the global buzz around the Internet of Things, there is no single, universally accepted definition for the term. Different definitions are used by various groups to describe or promote a particular view of what IoT means and its most important attributes. Some definitions specify the concept of the Internet or the Internet Protocol (IP), while others, perhaps surprisingly, do not. For example, consider the following definitions.

The Internet Architecture Board (IAB) begins RFC 7452, “Architectural Considerations in Smart Object Networking’’, with this description:

The term "Internet of Things" (IoT) denotes a trend where a large number of embedded devices employ communication services offered by the Internet protocols. Many of these devices,

often called "smart objects,'' are not directly operated by humans, but exist as components in buildings or vehicles, or are spread out in the environment.

Within the Internet Engineering Task Force (IETF), the term *"smart object networking"* is commonly used in reference to the Internet of Things. In this context, "smart objects" are devices that typically have significant constraints, such as limited power, memory, and processing resources, or bandwidth. Work in the IETF is organized around specific requirements to achieve network interoperability between several types of smart objects.

Published in 2012, the International Telecommunication Union (ITU) ITU–T Recommendation Y.2060, Overview of the Internet of things, discusses the concept of interconnectivity, but does not specifically tie the IoT to the Internet:

3.2.2 Internet of things (IoT): A global infrastructure for the information society, enabling advanced services by interconnecting (physical and virtual) things based on existing and evolving interoperable information and communication technologies.

Note 1—Through the exploitation of identification, data capture, processing and communication capabilities, the IoT makes full use of things to offer services to all kinds of applications, whilst ensuring that security and privacy requirements are fulfilled.

Note 2—From a broader perspective, the IoT can be perceived as a vision with technological and societal implications.

This definition in a call for papers for a feature topic issue of IEEE Communications Magazine links the IoT back to cloud services:

The Internet of Things (IoT) is a framework in which all things have a representation and a presence in the Internet. More specifically, the Internet of Things aims at offering new applications and services bridging the physical and virtual worlds, in which Machine-to-Machine (M2M) communications represents the baseline communication that enables the interactions between Things and applications in the cloud.

The Oxford Dictionaries offers a concise definition that invokes the Internet as an element of the IoT:

Internet of things (noun): The interconnection via the Internet of computing devices embedded in everyday objects, enabling them to send and receive data.

IoT Applications and Scenarios of Relevance:

"The major objectives for IoT are the creation of smart environments/spaces and self-aware things (for example: smart trans- port, products, cities, buildings, rural areas, energy, health, living, etc.) for climate, food, energy, mobility, digital society and health applications".

The outlook for the future is the emerging of a network of interconnected uniquely identifiable objects and their virtual representations in an Internet alike structure that is positioned over a network of interconnected computers allowing for the creation of a new platform for economic growth.

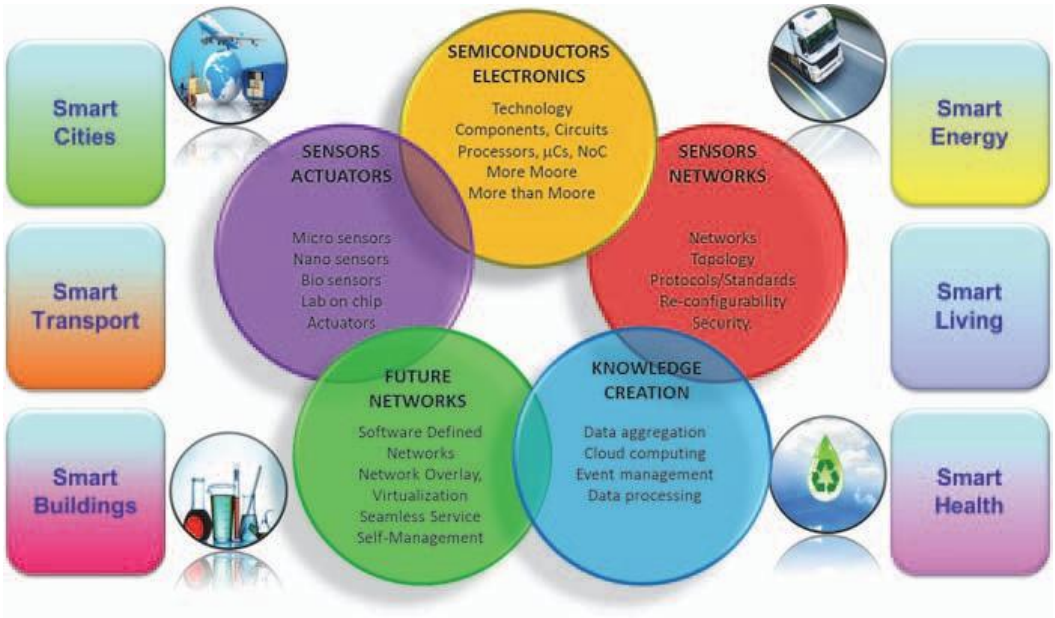


Fig. 5 Internet of Things — smart environments and smart spaces creation.

At the city level, the integration of technology and quicker data analysis will lead to a more coordinated and effective civil response to security and safety (law enforcement and blue light services); higher demand for outsourcing security capabilities.

At the building level, security technology will be integrated into systems and deliver a return on investment to the end-user through leveraging the technology in multiple applications (HR and time and attendance, customer behaviour in retail applications etc.).

There will be an increase in the development of “Smart” vehicles which have low (and possibly zero) emissions. They will also be connected to infrastructure. Additionally, auto manufacturers will adopt more use of “Smart” materials.

IoT Functional View

The Internet of Things concept refers to uniquely identifiable things with their virtual representations in an Internet-like structure and IoT solutions comprising a number of components such as:

- Module for interaction with local IoT devices (for example embedded in a mobile phone or located in the immediate vicinity of the user and thus contactable via a short range wireless inter- face). This module is responsible for acquisition of observations and their forwarding to remote servers for analysis and permanent storage.
- Module for local analysis and processing of observations acquired by IoT devices.
- Module for interaction with remote IoT devices, directly over the Internet or more likely via a proxy. This module is responsible for acquisition of observations and their forwarding to remote servers for analysis and permanent storage.
- Module for application specific data analysis and processing. This module is running on an application server serving all clients. It is taking requests from mobile and web clients and relevant IoT observations as input, executes appropriate data processing algorithms and generates output in terms of knowledge that is later presented to users.
- Module for integration of IoT-generated information into the business processes of an enterprise. This module will be gaining importance with the increased use of IoT data by enterprises as one of the important factors in day-to-day business or business strategy definition.
- User interface (web or mobile): visual representation of measurements in a given context (for example on a map) and interaction with the user, i.e. definition of user queries.
- It is important to highlight that one of the crucial factors for the success of IoT is stepping away from vertically-oriented, closed systems towards open systems, based on open APIs and standardized protocols at various system levels.

Data Management:

Data management is a crucial aspect in the Internet of Things. When considering a world of objects interconnected and constantly exchanging all types of information, the volume of the generated data and the processes involved in the handling of those data become critical.

A long-term opportunity for wireless communications chip makers is the rise of Machine-to-Machine (M2M) computing, which one of the enabling technologies for Internet of Things. This technology spans abroad range of applications. While there is consensus that M2M is a promising pocket of growth, analyst estimates on the size of the opportunity diverge by a factor of four . Conservative estimates assume roughly 80 million to 90 million M2M units will be sold in 2014, whereas more optimistic projections forecast sales of 300 million units. Based on historical analyses of adoption curves for similar disruptive technologies, such as portable MP3 players and antilock braking systems for cars, it is believed that unit sales in M2M could rise by as much as a factor of ten over the next five years.

There are many technologies and factors involved in the "data management" within the IoT context. Some of the most relevant concepts which enable us to understand the challenges and opportunities of data management are:

- Data Collection and Analysis
- Big Data
- Semantic Sensor Networking
- Virtual Sensors
- Complex Event Processing.

Data Collection and Analysis (DCA):

Data Collection and Analysis modules or capabilities are the essential components of any IoT platform or system, and they are constantly evolving in order to support more features and provide more capacity to external components (either higher layer applications leveraging on the data stored by the DCA module or other external systems exchanging information for analysis or processing).

The DCA module is part of the core layer of any IoT platform. Some of the main functions of a DCA module are:

User/customer data storing: Provides storage of the customer's information collected by sensors

User data & operation modelling: Allows the customer to create new sensor data models to accommodate collected information and the modelling of the supported operations

On demand data access: Provides APIs to access the collected data

Device event publish/subscribe/forwarding/notification: Provides APIs to access the collected data in real time conditions

Customer rules/filtering: Allows the customer to establish its own filters and rules to correlate events

Customer task automation: Provides the customer with the ability to manage his automatic processes.

Example: scheduled platform originated data collection,…

Customer workflows: Allows the customer to create his own workflow to process the incoming events from a device

Multitenant structure: Provides the structure to support multiple organizations and reseller schemes.

In the coming years, the main research efforts should be targeted to some features that should be included in any Data Collection and Analysis platform:

• **Multi-protocol.** DCA platforms should be capable of handling or understanding different input (and output) protocols and formats. Different standards and wrappings for the submission of observations should be supported.

• **De-centralization.** Sensors and measurements/observations captured by them should be stored in systems that can be de-centralized from a single platform. It is essential that different components, geographically distributed in different locations may cooperate and exchange data. Related with this concept, federation among different systems will make possible the global integration of IoT architectures.

• **Security.** DCA platforms should increase the level of data protection and security, from the transmission of messages from devices (sensors, actuators, etc.) to the data stored in the platform.

• **Data mining features.** Ideally, DCA systems should also integrate capacities for the processing of the stored info, making it easier to extract useful data from the huge amount of contents that may be recorded.

CHAPTER 3: HARDWARE DESCRIPTION

3.1 Introduction:

In this chapter the block diagram of the project and design aspect of independent modules are considered. Block diagram is shown in fig: 3.1

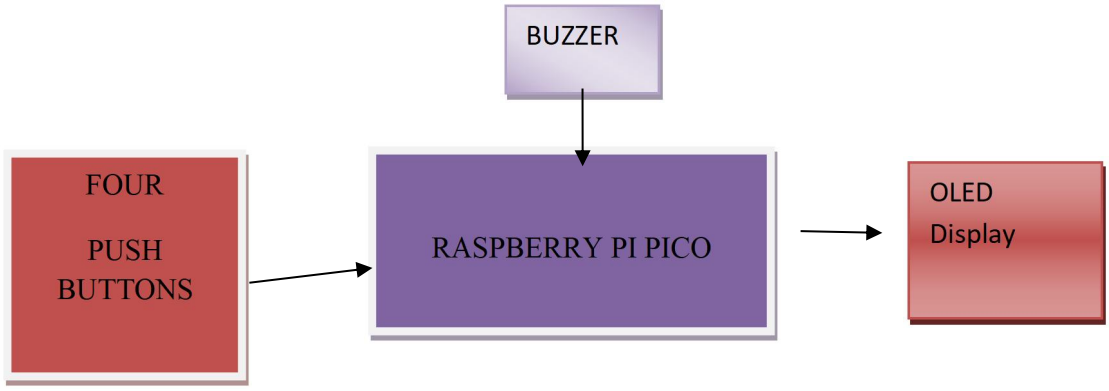


FIG 3.1: Block diagram of PICO SNAKE GAME

The main blocks of this project are:

- Raspberry Pi Pico
- OLED display
- Push Buttons
- Buzzer

Raspberry Pi Pico

Raspberry Pi Pico, an RP2040 microcontroller board with dual-core ARM Cortex-M0+ processor, 264k of internal RAM, and flexible Programmable I/O (PIO) feature. Pins GP0 to GP22 are digital GPIO pins. Pins GP26, GP27, and GP28 are digital GPIO pins with analog From light displays and IoT devices to signage and manufacturing processes, the Raspberry Pi Pico series gives you the power to control countless home, hobby, and industrial operations.

Programmable in C and MicroPython, Pico is adaptable to a vast range of applications and skill levels, and getting started is as easy as dragging and dropping a file.

More experienced users can take advantage of Raspberry Pi Pico's rich peripheral set, including SPI, I2C, and eight Programmable I/O (PIO) state machines for custom peripheral support.

Now available with wireless connectivity or pre-soldered headers, for even more flexibility in your projects. Designed by Raspberry Pi, RP2040 features a dual-core Arm Cortex-M0+ processor with 264kB internal RAM and support for up to 16MB of off-chip flash. A wide range of flexible I/O options includes I2C, SPI, and - uniquely - Programmable I/O (PIO). These support endless possible applications for this small and affordable package

Specification

- 21 mm × 51 mm form factor
- RP2040 microcontroller chip designed by Raspberry Pi in the UK
- Dual-core Arm Cortex-M0+ processor, flexible clock running up to 133 MHz
- 264kB on-chip SRAM
- 2MB on-board QSPI flash
- 2.4GHz 802.11n wireless LAN (Raspberry Pi Pico W and WH only)
- 26 multifunction GPIO pins, including 3 analogue inputs
- 2 × UART, 2 × SPI controllers, 2 × I2C controllers, 16 × PWM channels
- 1 × USB 1.1 controller and PHY, with host and device support
- 8 × Programmable I/O (PIO) state machines for custom peripheral support
- Supported input power 1.8–5.5V DC
- Operating temperature -20°C to +85°C (Raspberry Pi Pico and Pico H); -20°C to +70°C (Raspberry Pi Pico W and Pico WH)
- Castellated module allows soldering direct to carrier boards (Raspberry Pi Pico and Pico W only)
- Drag-and-drop programming using mass storage over USB
- Low-power sleep and dormant modes
- Accurate on-chip clock

Board:

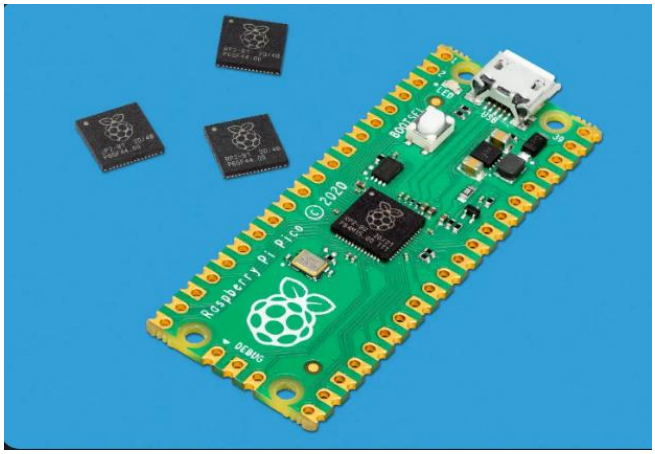
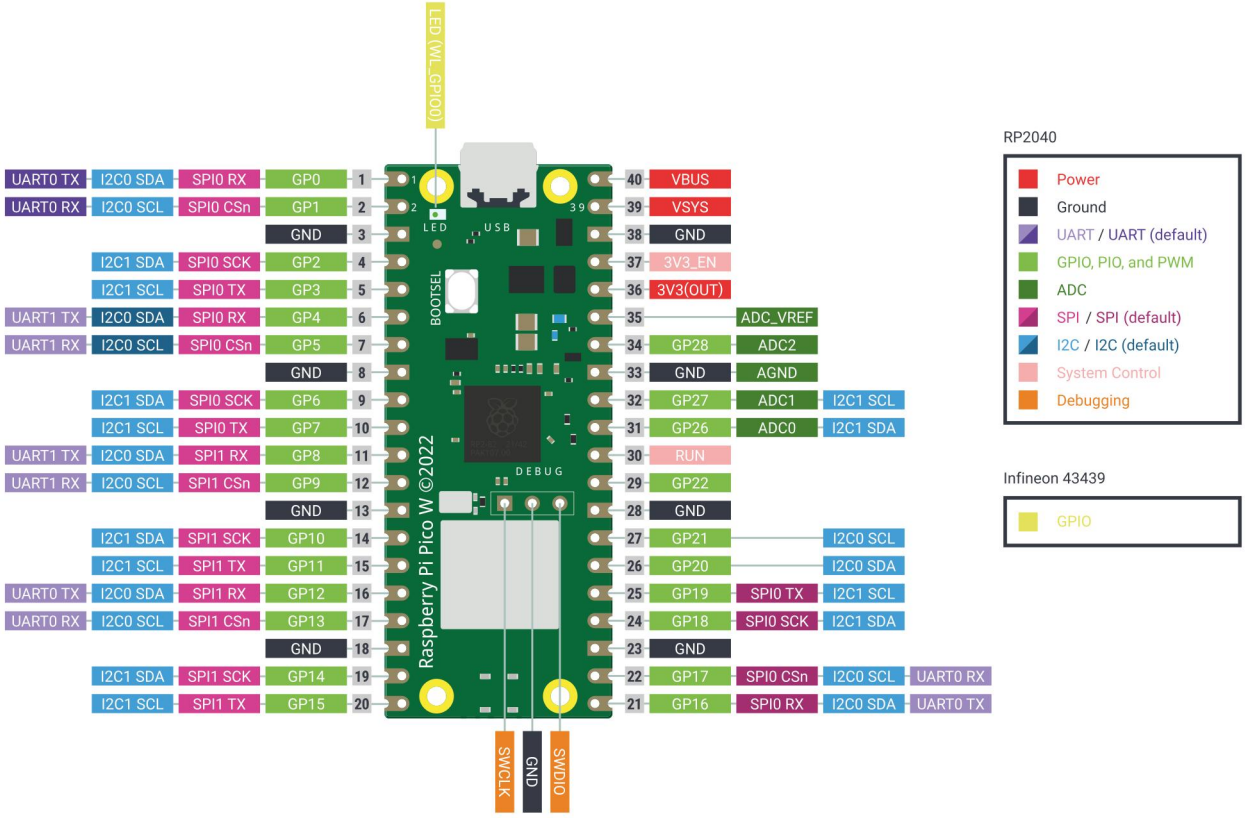


Fig.8 RP2040 board Description

Pin Definition:



OLED display:

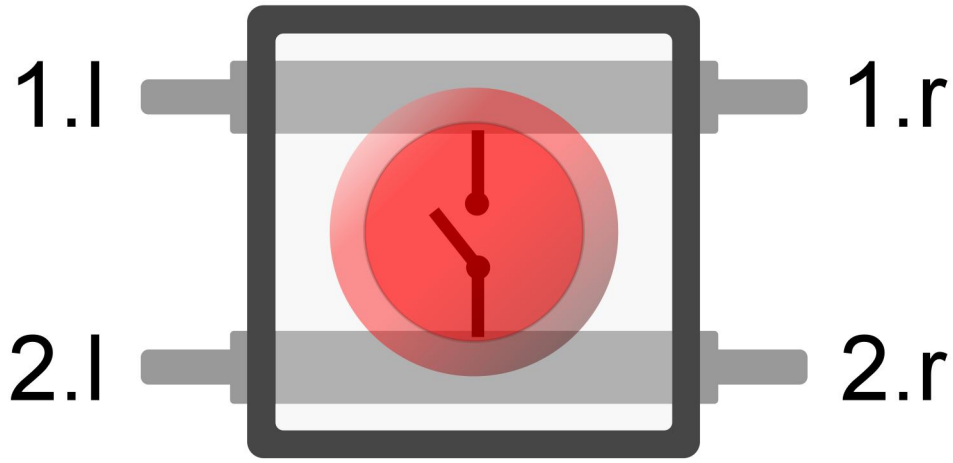
Transparent OLED is a breakthrough transparent display technology that displays dynamic or interactive information on a transparent surface glass. This revolutionary display allows users to view what is shown on a glass video screen while still being able to see through it. Designers can overlay text, digital images, and video content onto physical objects or scenes that sit behind the glass.

Transparent OLED displays are self-emitting and utilize cutting-edge Organic Light Emitting Diode (OLED) technology to eliminate the need for a backlight or enclosure, making it possible to create truly see-through installations in a virtually frameless glass design.

Working principle of OLED:

OLEDs work in a similar way to conventional diodes and LEDs, but instead of using layers of n-type and p-type semiconductors, they use organic molecules to produce their electrons and holes. A simple OLED is made up of six different layers. On the top and bottom there are layers of protective glass or plastic. The top layer is called the seal and the bottom layer the substrate. In between those layers, there's a negative terminal (sometimes called the cathode) and a positive terminal (called the anode). Finally, in between the anode and cathode are two layers made from organic molecules called the emissive layer (where the light is produced, which is next to the cathode) and the conductive layer (next to the anode)

Push Buttons



The push button has two set of pins (contacts), 1 and 2. When the push button is pressed, it connects these two contacts, thus closing an electrical circuit.

Each contact has a pin of the left side of the push button, and another pin on the right side of the push button. So pin 1.l is the left pin for first contact, and 1.r is the right pin for the first contact. Since both belong to the same contact, they are always connected, even when the button is not pressed.

Buzzer

The buzzer can operate in two modes: "smooth" (the default) and "accurate".

"smooth" sounds better and is suitable for simple, single-frequency tones. Use it when playing a melody or playing tones with Arduino's tone() function. Complex and polyphonic sounds may not play correctly (or not play at all) in "smooth mode"

Use the "accurate" mode when you need to play complex sounds. It will accurately play the sound you feed in. However, it'll add audible click noises to your sound. These noises are due to fluctuations in the simulation speed - it's not always able to provide the complete sound buffer in real time.

CHAPTER 4: SOFTWARE DESCRIPTION

Thonny IDE:

Thonny (/ˈθɒni/ THON-ee) is an integrated development environment for Python that is designed for beginners. It was created by Aivar Annamaa, an Estonian programmer. It supports different ways of stepping through the code, step-by-step expression evaluation, detailed visualization of the call stack and a mode for explaining the concepts of references and heap.

The program works on Windows, macOS and Linux. It is available as binary bundle including the recent Python interpreter[3] or pip-installable package.[6] It can be installed via the operating-system package manager on Debian, Raspberry Pi, Ubuntu and Fedora.

File:

- **New**
Creates a new instance of the editor, with the bare minimum structure of a sketch already in place.
- **Open**
Allows to load a sketch file browsing through the computer drives and folders.
- **Run**
Run the code either in raspberry pi or local system.
- **Close**
Closes the instance of the Arduino Software from which it is clicked.
- **Save**
Saves the sketch with the current name. If the file hasn't been named before, a name will be provided in a "Save as.." window.
- **Saveas...**
Allows to save the current sketch with a different name.
- **Preferences**
Opens the Preferences window where some settings of the IDE may be customized, as the language of the IDE interface.
- **Quit**
Closes all IDE windows. The same sketches open when Quit was chosen will be automatically reopened the next time you start the IDE.

Edit:

- **Undo/Redo**
Goes back of one or more steps you did while editing; when you go back, you may go forward with Redo.
- **Cut**
Removes the selected text from the editor and places it into the clipboard.

- **Copy**
  Duplicates the selected text in the editor and places it into the clipboard.
- **Paste**
  Puts the contents of the clipboard at the cursor position, in the editor.
- **SelectAll**
  Selects and highlights the whole content of the editor.
- **Comment/Uncomment**
  Puts or removes the // comment marker at the beginning of each selected line.
- **Increase/DecreaseIndent**
  Adds or subtracts a space at the beginning of each selected line, moving the text one space on the right or eliminating a space at the beginning.
- **Find**
  Opens the Find and Replace window where you can specify text to search inside the current sketch according to several options.
- **Find Next**
  Highlights the next occurrence - if any - of the string specified as the search item in the Find window, relative to the cursor position.
- **Find Previous**
  Highlights the previous occurrence - if any - of the string specified as the search item in the Find window relative to the cursor position.

- **AddFile...**
  Adds a source file to the sketch (it will be copied from its current location). The new file appears in a new tab in the sketch window. Files can be removed from the sketch using the tab menu accessible clicking on the small triangle icon below the serial monitor one on the right side o the toolbar.

- **Install Micropython or Circuit python**
  This ide provides to install desired type of language in our raspberry pi. After connecting raspberry pi pico with computer and in the presence of internet it show a bunch of languages that can be installed.

Wokwi Online Simulator

Wokwi is an online Electronics simulator. You can use it to simulate Arduino, ESP32, and many other popular boards, parts and sensors.

Unique Features WiFi simulation - Connect your simulated project to the internet. You can use MQTT, HTTP, NTP, and many other network protocols.Virtual Logic Analyzer - Capture digital signals in your simulation (e.g. UART, I2C, SPI) and analyze them on your computer. Advanced debugging with GDB - Powerful Arduino and Raspberry Pi Pico debugger for advanced users. SD card simulation - Store and retrieve files and directories from your code. Club members can also upload binary files (such as images). Wokwi compiles your code into a binary firmware, and then executes the binary firmware one instruction at a time, as a real microcontroller would. If you want to learn about the internals, check out the following resources: The diagram editor provides an interactive way to edit your diagram: add components to the simulation and

define the connections between them. It's a convenient alternative for editing the diagram.json file directly.

Editing parts

Adding a part

To add a new part, click on the purple "+" button at the top of the diagram editor.

You'll see a menu with a list of parts you can add. Choose a part to add it. The part will be added at position (0, 0), and then you can drag it to the desired position.

Not all parts are currently available through the menu. For example, MCU boards and micro-controllers such as the Arduino Nano or the ATtiny85 are missing. You can still add these parts by editing diagram.json directly.

Moving a part

Move a part by clicking on it and then dragging it with your mouse.

Rotating a part

Rotate a part by clicking on it (to select it) and then pressing "R". The part will rotate 90 degrees clockwise. If you need to rotate a part by a different amount (e.g. 45 degrees), you can achieve that by editing diagram.json.

Duplicating a part

Create a new copy of a part by clicking on it (to select it) and the pressing "D". You can press "D" several times to create multiple copies of the part.

Deleting a part

Delete a part by clicking on it (to select it) and then pressing the Delete button.

Selecting multiple parts

Select multiple parts by clicking on the parts with the Shift key pressed. You can then move all the parts together, duplicate them (using the "D" key), or delete them using the Delete key.

Copying and pasting parts

You can copy the selects part(s) by using the standard Copy keyboard shortcut (Ctrl+C or ⌘+C). If you selected multiple parts, all the wires that connect the selected parts are also copied. The parts you copied are stored in your system clipboard in a JSON format, similar to the diagram.json format.

To paste the parts you copied, click on the diagram and press the standard Paste keyboard (Ctrl+V or ⌘+V). In some cases, the parts will be pasted outside of the currently visible diagram area, so you may have to zoom out in order to find them. This will be fixed in the future.

You can use the copy-paste feature between different project, and quickly copy several parts (including all the internal connections) at once.

Editing wires

Creating a wire between two parts

To create a new wire between two parts, click on one of the pins that you'd like to connect. Then click on the second (target) pin. This will create the wire.

If you want the wire to go in a specific way, you can guide it by clicking where you want it to go after selecting the first pin.

To cancel a new wire (delete it without selecting a target pin) click the right mouse button or press Escape.

Changing the color of a wire

The color of new wires is automatically determined by the function of the pin: wires starting from ground pins are black, 5 V pins are red, and other wires are green.

CHAPTER 6: ADVANTAGES AND DISADVANTAGES

The advantages is that you can enjoy playing your childhood game which is very pleasant. Also the with this project you will be able to know how the things work like raspberry pi , oled screen or push buttons and buzzer. How they together make sense. The main advantage is this saves your time. When you play game in your phone you get distracted by other things and spend lots and lots of time scrolling your phone. But this device has only one feature and one purpose. Ie snake game nothing else.

DISADVANTAGES:

The main disadvantage is that the display is very small and sometimes it becomes difficult to see the food. Also the there is only one game which can be little bit boring but there is room for adding more games as the codes have followed the design pattern which helps in code reusablity and modification very easy, its like adding controller through usb.

CHAPTER 7: RESULTS

7.1 Results:

The project "**PICO SNAKE A FUN GAME**" was designed a system which gives entertainments and fun to play. We can play the games with the help using push buttons and

when snake eats the food a sweet beep sound is produce and also harsh sound when you hit the wall. Snake can grow as long as it eats the food.

7.2 Conclusion:

This project taught us how can we show visual effects oled screen and interact with the push buttons. With the help of raspberry pi we can build many different unusual things that can surprise the whole world.

7.3 Future Scope:

The codes for snake are so optimized that we can add multiple games without touching or modifying the original codes. So in near future if we wanna add Dino game or tetris game or bounce ball we can add. In fact there are are some games in beta phase like chrome dino game.

REFERENCES

The sites which were used while doing this project:

1. https://docs.wokwi.com/parts/wokwi-pi-pico

2.https://www.raspberrypi.com/documentation/microcontrollers/raspberry-pi-pico.html

3. https://components101.com/displays/oled-display-ssd1306

4. Programming with MicroPython [Book] - O'Reilly

CHAPTER 8: TEAMWORK

8.1. SUMMARY OF TEAM WORK

8.1.1 ATTRIBUTES

1	Attends group meetings regularly and arrives on time.
2	Contributes meaningfully to group discussions.
3	Completes group assignments on time.
4	Prepares work in a quality manner.
5	Demonstrates a cooperative and supportive attitude.
6	Contributes significantly to the success of the project.

8.1.2 SCORE

1=strongly disagree; **2**=disagree; **3**=agree; **4**=strongly agree

Student 1: _____

Student 2: _____

Student 1	Evaluated by	
	Attributes	Student 2
	1	
	2	
	3	
	4	
	5	
	6	
	Grand Total	

Student 2	Evaluated by	
	Attributes	Student 1
	1	
	2	
	3	
	4	
	5	
	6	
	Grand Total	

_____ _____

Signature of Student 1 **Signature of Student 2**

Appendix

CODES

```
from draw import Draw

import random, utime, _thread

from machine import Pin,PWM

class Snake:

    def __init__(self):

        self.buzzer=Pin(20,Pin.OUT)

        self.obj=Draw()

        self.head=(25,31)

        self.body=[(i,31) for i in range(self.head[0]-20,self.head[0]+1)]
self.border=[(i,0) for i in range(128)]+[(i,63) for i in range(128)]+[(0,i) for i in
range(64)]+[(127,i) for i in range(64)]

    def intro(self):

        self.obj.clear()

        self.obj.oled.text("Game Starts",20,22)

        for i in range(5,0,-1):

            self.obj.oled.text(f"in {i} seconds",15,38)

            self.obj.show()

            self.obj.oled.text(f"in {i} seconds",15,38,0)

            utime.sleep(1)

        self.obj.clear()

        for point in self.border:

            self.obj.point(*point)

    def render(self,food,visible=1):

        self.obj.glow=visible

        for point in self.body:

            self.obj.point(*point)

        self.obj.point(*self.head)

        self.obj.point(*food)

    def condtion(self):

        if self.head in self.border or self.head in self.body:
```

```
            return False
        return True
    def randomFood(self):
        while True:
            food=random.randrange(1,126),random.randrange(1,62)
            if not food in self.body:
                return food
    def move(self,direction,up,down,left,right):
        head=self.head
        if direction==up:
            return head[0],head[1]-1
        elif direction==down:
            return head[0],head[1]+1
        elif direction==left:
            return head[0]-1,head[1]
        elif direction==right:
            return head[0]+1,head[1]
    def play(self,up,down,left,right):
        def buzzerListner(pin,duration=0.3):
              Pin.value(1)
              Utime.sleep(duration)
              Pin.value(0)



        upBtn=Pin(up,mode=Pin.IN, pull=Pin.PULL_DOWN)
        downBtn=Pin(down,mode=Pin.IN,pull=Pin.PULL_DOWN)
        leftBtn=Pin(left,mode=Pin.IN,pull=Pin.PULL_DOWN)
        rightBtn=Pin(right,mode=Pin.IN,pull=Pin.PULL_DOWN)
        self.intro()
        direction=right
        food=(80,31)
        self.render(food)
        self.obj.show()
        while True:
```

```
    if upBtn.value() and direction!=down:
        direction=up
    elif downBtn.value() and direction!=up:
        direction=down
    elif leftBtn.value() and direction!=right:
        direction=left
    elif rightBtn.value() and direction!=left:
        direction=right
    self.render(food,0)
    self.head=self.move(direction,up,down,left,right)
    if not self.condtion():
        _thread.start_new_thread(buzzerListner,(self.buzzer,3))
        break
    self.body.append(self.head)
    utime.sleep(.01)
    if self.head==food:
        _thread.start_new_thread(buzzerListner,(self.buzzer,0.5))
        food=self.randomFood()
    else:
        self.body.pop(0)
    self.render(food)
    self.obj.show()
self.obj.clear()
self.obj.oled.text("Game Over",20,22,0)
self.obj.oled.text(f"Your Score:{len(self.body)-21}",10,38,0)
self.obj.show()
utime.sleep(5)
```