Experiment No._05

Title: Arduino and R-Pie interfacing.

Batch: B2 **Roll No.: 16010421059** **Experiment No.:05**

Aim: Arduino and R-Pie interfacing.

Resources needed: Internet, Arduino and Raspberry Pi module, and Jumper Wires.

Theory:

Pre Lab/ Prior Concepts:

The Raspberry Pi and Arduino are two popular and versatile platforms for hobbyists, makers, and engineers. The Raspberry Pi is a small, powerful single-board computer that runs a full Linux operating system, while the Arduino is a microcontroller board ideal for controlling sensors and actuators. By combining these two platforms, you can create projects that leverage the strengths of both.

Connection:

Serial Connection: Establish a serial connection between the Raspberry Pi and Arduino.

1. Connect the Arduino's TX (transmit) pin to the Raspberry Pi's RX (receive) pin.

2. Connect the Arduino's RX (receive) pin to the Raspberry Pi's TX (transmit) pin.

3. Connect a ground (GND) pin from the Arduino to a ground (GND) pin on the Raspberry Pi.

Software Setup

Raspberry Pi:

Install the Python serial library: sudo apt-get install python3-serial (If not already installed).

Arduino:

Open the Arduino IDE and create a new sketch.

Arduino Code:

```
void setup() {
  Serial.begin(9600); // Initialize serial communication at 9600 baud rate
}
void loop() {
  if (Serial.available()) {
    String command = Serial.readString();
    if (command == "print_university") {
```

```
    Serial.println("Somaiya Vidyavihar University");
  }
 }
}
```

Raspberry Pi Code:

```
import serial
import time
port = "/dev/ttyACM0" # Adjust the port name if necessary
arduino = serial.Serial(port, 9600, timeout=1)
time.sleep(2) # Allow time for connection to establish
arduino.write(b"print_university")  # Send the command to the Arduino
```

Running the Code:

1. Upload the Arduino code to your board.

2. Save the Python code on your Raspberry Pi.

3. Run the Python code: python your_script_name.py

You should see "Somaiya Vidyavihar University" printed on your Raspberry Pi's command prompt.

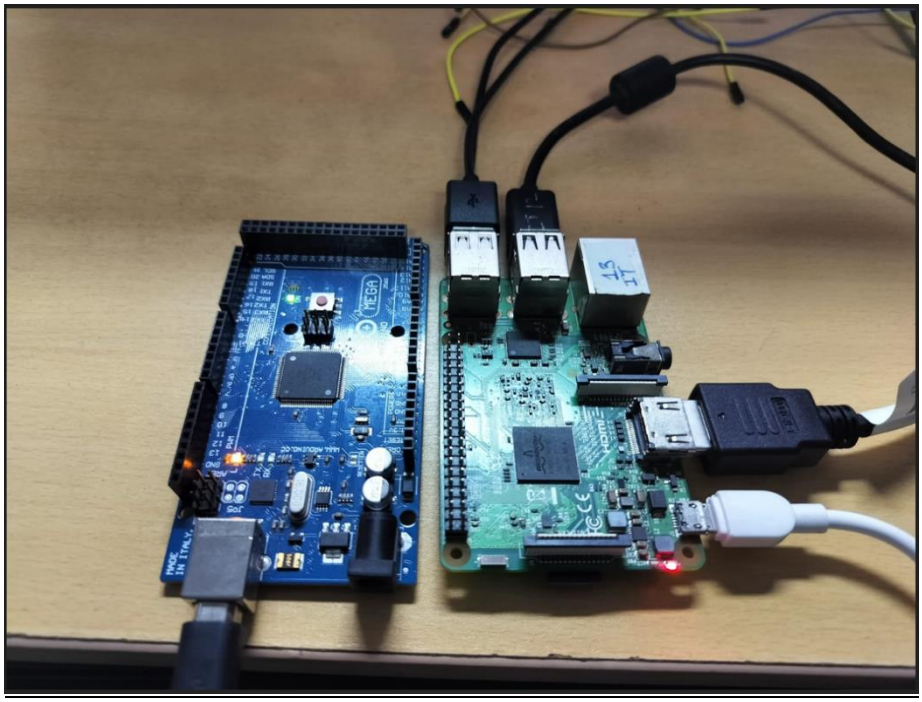
Activity:

1. Interfacing Arduino with R-Pi.
2. sending a command from the Raspberry Pi to the Arduino, which will trigger the Arduino to print "Somaiya Vidyavihar University" on the Raspberry Pi's command prompt.

Results: (Program printout with output / Document printout as per the format)

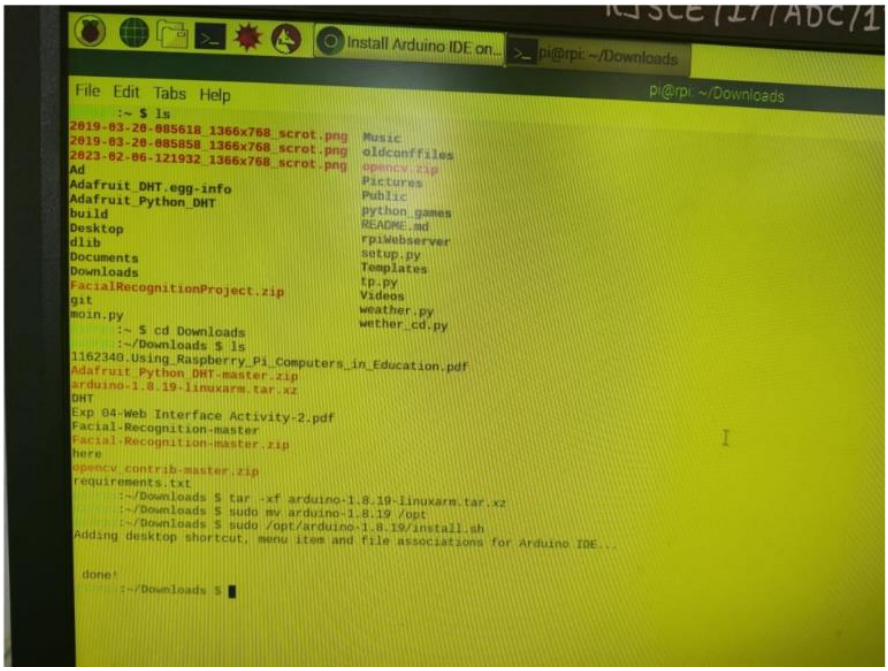
1. Printing "Hello, World!" using Arduino IDE:

Setup:



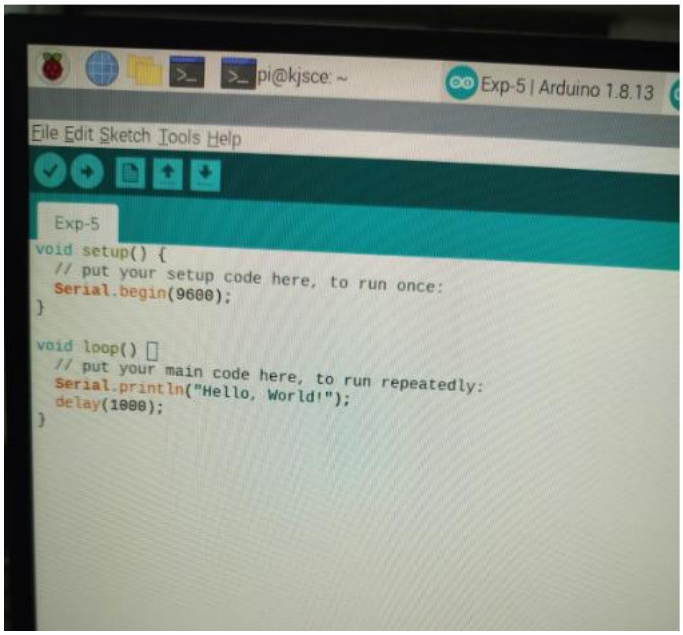
Terminal Commands:

$ sudo apt-get install Arduino



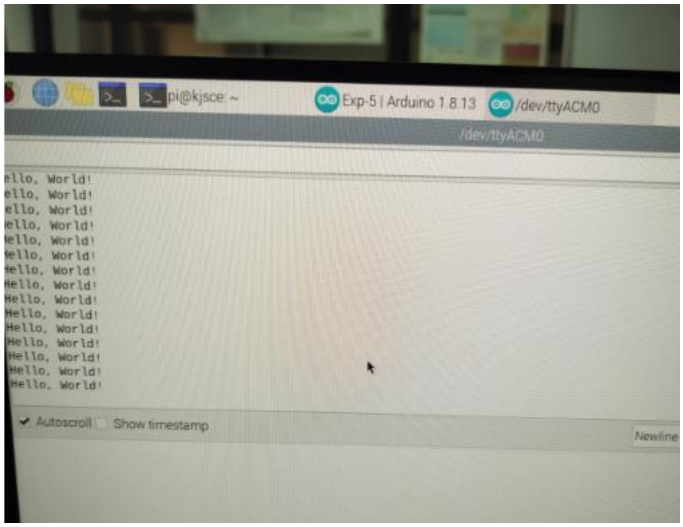
Upload Code in Arduino:'

```
void setup() {
   Serial.begin(9600);
}
void loop() {
    Serial.println("Hello");
    delay(1000);
}
```



Serial Monitor Output:

It prints "Hello, World!" after every 1 second.



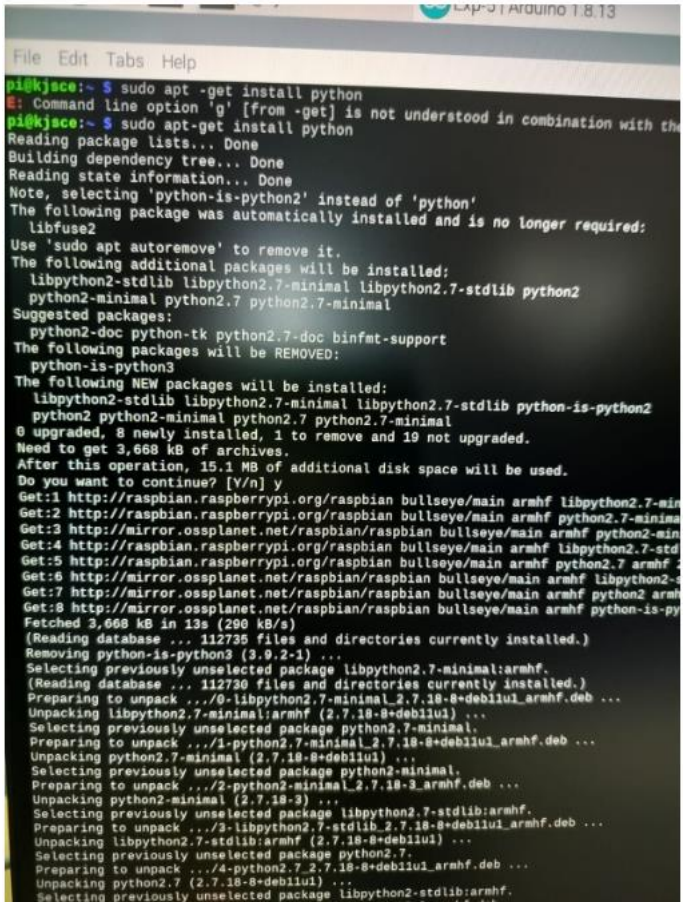
2. Printing "Somaiya Vidyavihar University" using python GPIO:

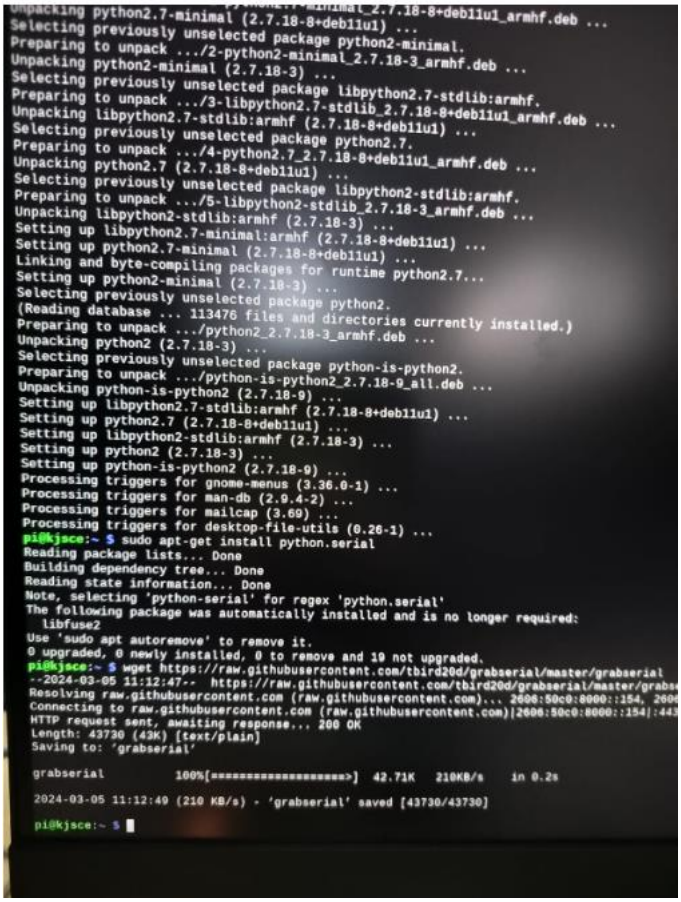
Terminal commands:

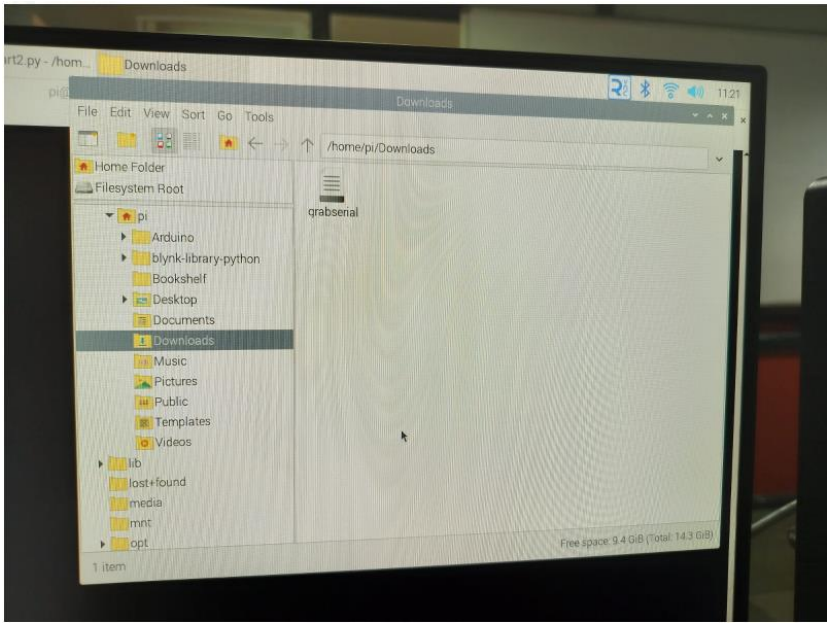
$ sudo apt -get install python

$ sudo apt -get install python.serial

$ wget https://raw.githubusercontent.com/tbird20d/grabserial/master/grabserial




Unpacking python2.7-minimal (2.7.18-8+deb11u1) ...
Selecting previously unselected package python2-minimal.
Preparing to unpack .../2-python2-minimal_2.7.18-3_armhf.deb ...
Unpacking python2-minimal (2.7.18-3) ...
Selecting previously unselected package libpython2.7-stdlib:armhf.
Preparing to unpack .../3-libpython2.7-stdlib_2.7.18-8+deb11u1_armhf.deb ...
Unpacking libpython2.7-stdlib:armhf (2.7.18-8+deb11u1) ...
Selecting previously unselected package python2.7.
Preparing to unpack .../4-python2.7_2.7.18-8+deb11u1_armhf.deb ...
Unpacking python2.7 (2.7.18-8+deb11u1) ...
Selecting previously unselected package libpython2-stdlib:armhf.
Preparing to unpack .../5-libpython2-stdlib_2.7.18-3_armhf.deb ...
Unpacking libpython2-stdlib:armhf (2.7.18-3) ...
Setting up libpython2.7-minimal:armhf (2.7.18-8+deb11u1) ...
Setting up python2.7-minimal (2.7.18-8+deb11u1) ...
Linking and byte-compiling packages for runtime python2.7...
Setting up python2-minimal (2.7.18-3) ...
Selecting previously unselected package python2.
(Reading database ... 113476 files and directories currently installed.)
Preparing to unpack .../python2_2.7.18-3_armhf.deb ...
Unpacking python2 (2.7.18-3) ...
Selecting previously unselected package python-is-python2.
Preparing to unpack .../python-is-python2_2.7.18-9_all.deb ...
Unpacking python-is-python2 (2.7.18-9) ...
Setting up libpython2.7-stdlib:armhf (2.7.18-8+deb11u1) ...
Setting up python2.7 (2.7.18-8+deb11u1) ...
Setting up libpython2-stdlib:armhf (2.7.18-3) ...
Setting up python2 (2.7.18-3) ...
Setting up python-is-python2 (2.7.18-9) ...
Processing triggers for gnome-menus (3.36.0-1) ...
Processing triggers for man-db (2.9.4-2) ...
Processing triggers for mailcap (3.69) ...
Processing triggers for desktop-file-utils (0.26-1) ...
pi@kjsce:~ $ sudo apt-get install python.serial
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Note, selecting 'python-serial' for regex 'python.serial'
The following package was automatically installed and is no longer required:
libfuse2
Use 'sudo apt autoremove' to remove it.
0 upgraded, 0 newly installed, 0 to remove and 10 not upgraded.
pi@kjsce:~ $ wget https://raw.githubusercontent.com/tbird20d/grabserial/master/grabserial
--2024-03-05 11:12:47-- https://raw.githubusercontent.com/tbird20d/grabserial/master/grabs
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 2606:50c0:8000::154, 260
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|2606:50c0:8000::154|:443
HTTP request sent, awaiting response... 200 OK
Length: 43730 (43K) [text/plain]
Saving to: 'grabserial'
grabserial 100%[==========>] 42.71K 210KB/s in 0.2s
2024-03-05 11:12:49 (210 KB/s) - 'grabserial' saved [43730/43730]
pi@kjsce:~ $

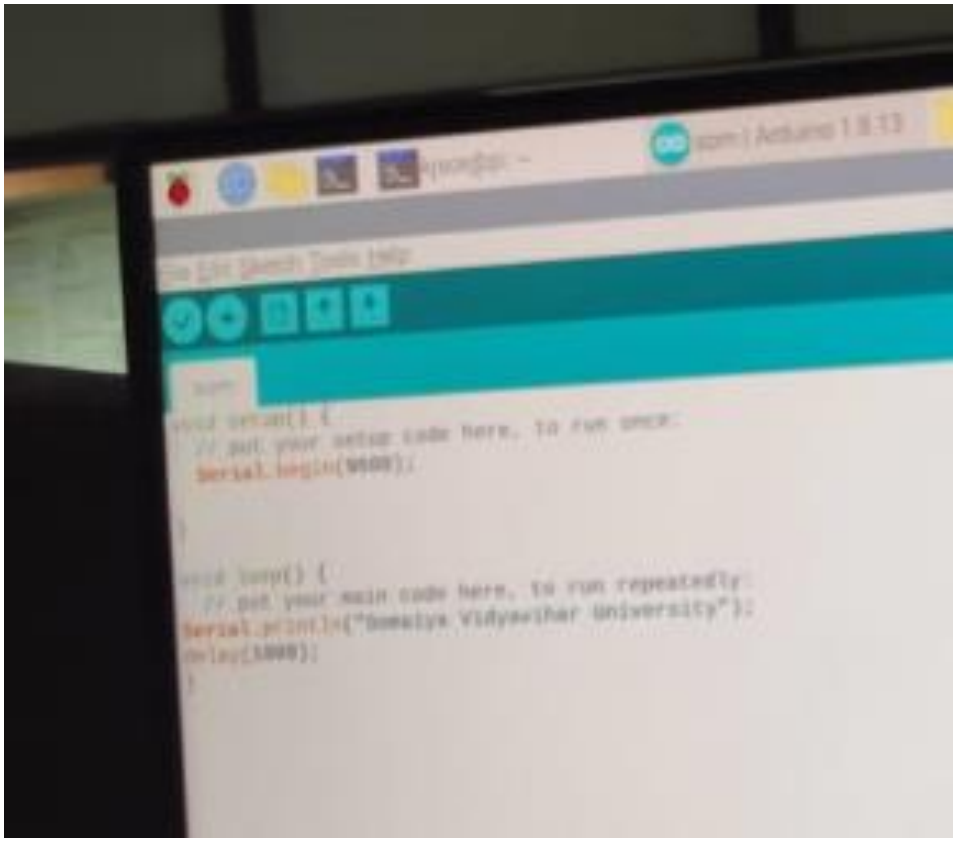

Downloads
File Edit View Sort Go Tools
/home/pi/Downloads
Home Folder
Filesystem Root
pi
Arduino
blynk-library-python
Bookshelf
Desktop
Documents
Downloads
Music
Pictures
Public
Templates
Videos
lib
lost+found
media
mnt
opt
grabserial
1 item
Free space: 9.4 GiB (Total: 14.3 GiB)

b. Code for Raspberry Pi:

```
import serial
import RPi.GPIO as GPIO
import time
ser=serial.Serial("/dev/ttyACM0",9600)
ser.baudrate=9600
def blink(pin):
GPIO.output(pin,GPIO.HIGH)
```

c. Upload code in Arduino:

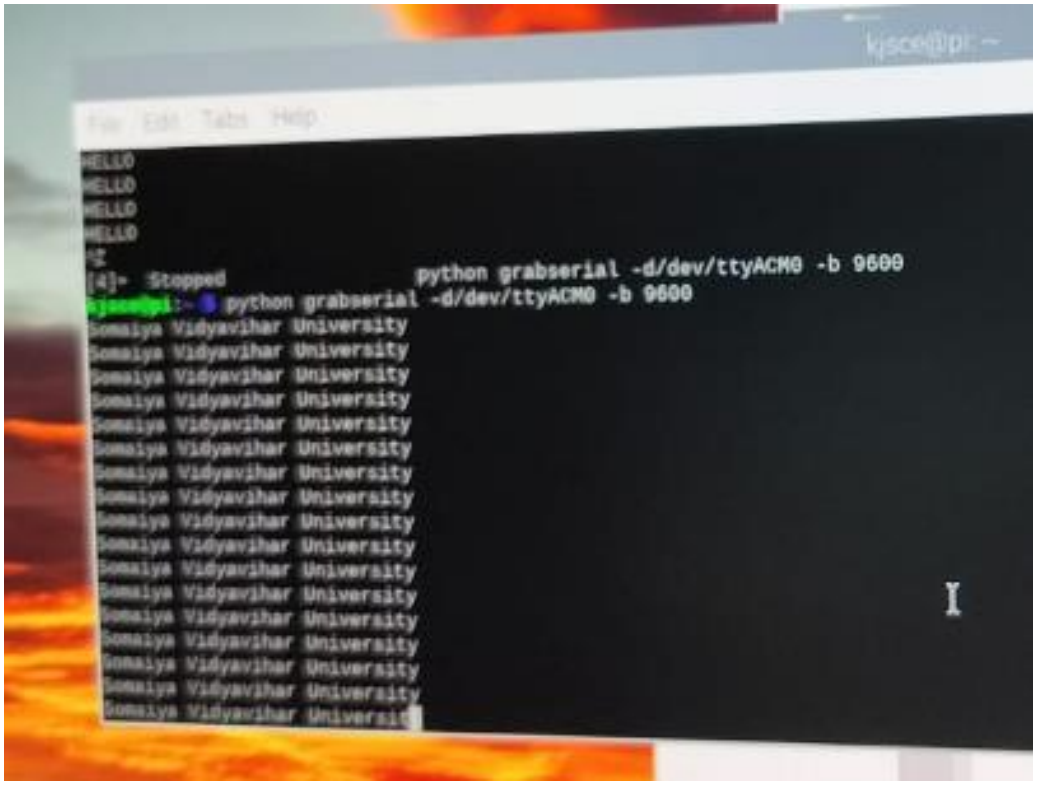
```
void setup() {
Serial.begin(9600);
}
void loop() {
Serial.println("Somaiya Vidyavihar University ");
delay(1000);
}
```



d. Display output in terminal:

It prints "***Somaiya Vidyavihar University***" after a delay of 1 second.

$ python grabserial –d /dev/ttyACM0 –b 9600



Questions:

1. **Case study: Application of Arduino and R-Pie synchronization.?**

Application of Arduino and Raspberry Pi Synchronization for Smart Home Automation

In recent years, there has been a surge in interest in smart home automation systems, where various devices are interconnected to enhance convenience, efficiency, and security. Arduino and Raspberry Pi are two popular platforms widely used in DIY projects and professional applications due to their versatility, affordability, and ease of use. In this case study, we explore the seamless integration of Arduino and Raspberry Pi to create a synchronized smart home automation system.

The primary goal of this project is to develop a smart home automation system that leverages the strengths of both Arduino and Raspberry Pi platforms for synchronized operation. The system should be capable of controlling various devices such as lights, temperature sensors, security cameras, and door locks.

Components Used:

1. Arduino Uno: Used for controlling low-level hardware peripherals and interfacing with sensors and actuators.
2. Raspberry Pi 4: Acts as the central processing unit and handles higher-level tasks such as data processing, networking, and user interface.
3. Sensors and Actuators: Including motion sensors, temperature sensors, LED lights, servo motors, and relays.
4. Power Supplies: To provide power to Arduino, Raspberry Pi, and connected peripherals.
5. Communication Interface: Such as USB, GPIO pins, or wireless modules for communication between Arduino and Raspberry Pi.

Implementation:

1. Sensor Data Acquisition: Arduino is responsible for interfacing with sensors and collecting data, such as motion detection from PIR sensors, temperature readings from temperature sensors, etc.
2. Actuator Control: Based on the sensor data received, Arduino controls actuators like LED lights, servo motors for opening/closing doors, or relays for switching appliances.
3. Data Processing and Decision Making: Raspberry Pi receives data from Arduino and processes it to make decisions. For example, it may analyze sensor data to determine if there's any intruder detected, and accordingly trigger actions like sending alerts or activating security cameras.
4. User Interface: Raspberry Pi hosts a user interface accessible via a web application or a dedicated application running on smartphones or tablets. This interface allows users to monitor sensor data, control devices remotely, and configure automation rules.
5. Synchronization and Communication: Arduino and Raspberry Pi communicate with each other using a suitable communication protocol such as serial communication (UART), I2C, SPI, or MQTT. This ensures synchronized operation and seamless integration between the two platforms.

Benefits:

1. Scalability: The modular design allows for easy expansion by adding more sensors, actuators, or integrating additional functionalities.
2. Customization: Users can customize automation rules and behavior according to their specific requirements.
3. Cost-Effectiveness: Utilizing Arduino and Raspberry Pi reduces the overall cost compared to proprietary smart home automation solutions.
4. Open-Source: Both Arduino and Raspberry Pi platforms are open-source, fostering community support, and enabling extensive customization and innovation.
5. Integration with Existing Systems: The system can be integrated with existing smart home ecosystems or third-party services through APIs or custom integrations.

The integration of Arduino and Raspberry Pi in smart home automation offers a powerful and flexible solution for creating customized, synchronized systems tailored to individual needs. By leveraging the strengths of both platforms, users can build sophisticated automation systems that enhance convenience, comfort, and security in their homes.

Outcomes:

CO2: Comprehend IoT architecture, enabling technologies and protocols

---

Conclusion:

Understood the concept and use cases of Arduino and R-Pie interfacing and then successfully implemented the same.

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of faculty in-charge with date

References:

1. https://en.wikipedia.org/wiki/Internet_of_things

2. Introduction - Blynk Documentation

Books:

1. Jan Holler, Vlasios Tsiatsis, Catherine Mulligan, Stefan Avesand, Stamatis Karnouskos, David Boyle, "From Machine-to-Machine to the Internet of Things: Introduction to a New Age of Intelligence", 1st Edition, Academic Press, 2014.

2. Vijay Madisetti and Arshdeep Bahga, "Internet of Things (A Hands-on-Approach)", 1stEdition, VPT, 2014.

3. Dr. Ovidiu Vermesan, Dr. Peter Friess, "Internet of Things - From Research and Innovation to Market Deployment", River Publisher, 2014