Práctica Extra:

Emulación de videojuegos retro con la Raspberry Pi

Fundamentos de Sistemas Embebidos

Autor:
Rodrigo Iván Olvera Martínez

1. Objetivo

El alumno aprenderá a hacer las configuraciones necesarias para logras emular videojuegos retro en la Raspberry PI.

2. Introducción

La presente práctica resume los pasos a seguir para desplegar en pantalla videojuegos, utilizando el emulador Mednafen y haciendo algunas configuraciones podremos utilizar un control ya sea de Playstation o Xbox para lograr jugar sin la necesidad de un teclado.

2.1. Mednafen

Mednafen es un emulador de múltiples sistemas de videojuegos retro. Su nombre es una abreviatura de "My Emulator Doesn't Need A Frickin' Excellent Name" (Mi emulador no necesita un nombre excelente). Es un software de código abierto y gratuito que permite al usuario jugar y disfrutar de juegos de consolas retro.

Mednafen admite una amplia gama de sistemas, incluidos NES, SNES, Game Boy, Game Boy Advance, Sega Genesis, Sega Saturn, Atari Lynx y muchas otras plataformas populares. El emulador se enfoca en la precisión y la emulación de hardware, lo que significa que intenta replicar fielmente el funcionamiento de las consolas originales y sus juegos.

Al utilizar Mednafen, el usuario puede cargar imágenes de ROM o ISO de juegos en el emulador y jugarlos en su computadora. El software ofrece varias opciones de configuración, como gráficos, sonido y controles, que permiten a los jugadores personalizar su experiencia de juego según sus preferencias.

3. Material

Se asume que el alumno cuenta con un una Raspberry Pi con sistema operativo Raspbian e intérprete de Python instalado. Se aconseja encarecidamente el uso de *git* como programa de control de versiones.

- Control de PS4 o Xbox
- Cable usb v8 o el necesario si el control es inalámbrico.
- Cable HDMI a HDMI o HDMI a VGA si su monitor lo necesita
- Monitor y cable de alimentación
- Teclado y Mouse con entrada usb

4. Instrucciones

1. Instalar Mednafen.
2. Configuración.
3. Programa.

4.1. Paso 1: Instalar Mednafen.

Para este punto el alumno ya conectó debidamente a la red de internet la Raspberry, a continuación en la consolade Raspberry Pi OS insertar el comando “sudo apt update”

```
axel@raspberrypi:/home$ sudo apt update
Obj:1 http://security.debian.org/debian-security bullseye-security InRelease
Obj:2 http://deb.debian.org/debian bullseye InRelease
Obj:3 http://deb.debian.org/debian bullseye-updates InRelease
Obj:4 http://archive.raspberrypi.org/debian bullseye InRelease
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Todos los paquetes están actualizados.
axel@raspberrypi:/home$
```

Figura 1: comando update

Posteriormente insertar el comadno “ sudo apt install mednafen”

```
axel@raspberrypi:/home$ sudo apt install mednafen
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
mednafen ya está en su versión más reciente (1.26.1+dfsg-1).
El paquete indicado a continuación se instaló de forma automática y ya no es nec
esario.
  libfuse2
Utilice «sudo apt autoremove» para eliminarlo.
0 actualizados, 0 nuevos se instalarán, 0 para eliminar y 0 no actualizados.
axel@raspberrypi:/home$
```

Figura 2: comando para instalar mednafen

Dar si a la instalación y por último reiniciar la raspberry con “reboot”.

4.2. Paso 2: Configuración de archivos.

Vamos a crear un servicio para que la Raspberry inicie con una imagen o video personalizado. Creamos un archivo en el directorio raíz con el comando “sudo nano reproducción_arranque.sh”.
Y se escribe lo siguiente:

```
#!/bin/bash
fbi -T 1 -noverbose -a /home/Inicio/splash.png
```

```
  GNU nano 5.4                  reproduccion_arranque.sh
#!/bin/bash
fbi -T 1 -noverbose -a /home/Inicio/splash.png

[ El fichero «reproduccion_arranque.sh» no es de escritura ]
^G Ayuda      ^O Guardar    ^W Buscar     ^K Cortar     ^T Ejecutar   ^C Ubicación
^X Salir      ^R Leer fich. ^\ Reemplazar ^U Pegar      ^J Justificar ^_ Ir a línea
```

Figura 3: creación del archivo para la ejecución de imagen personalizada

Guardamos el archivo. Prestar atención en la dirección o ruta del archivo, para posteriormente guardarlo.

Después habilitar y activar el servicio de inicio

“sudo systemctl enable reproduccion_arranque”“sudo
systemctl start reproduccion_arranque”

Reiniciar la Raspberry, para que los cambios puedan verse, al reiniciar debería mostrarse en pantalla la imagen de arranque personalizada.

Figura 4: Imagen de presentación de nuestro sistema embebido

4.3. Paso 3: Copiar las ROMs a una ruta.

Buscar ROMs de la web y copiarlas a la ruta que sea de preferencia, en este caso "/home/rms/", puede ser la que sea de conveniencia.

```
axel@raspberrypi:/home/rms$ ls
 CAPAME.nes
'Chip '\''n Dale - Rescue Rangers 2 (USA).nes'
'Donkey Kong Classics (USA, Europe).nes'
'DuckTales (USA).nes'
'Flintstones 2 - The Surprise at Dinosaur Peak!, The (U).nes'
'Legend of Zelda, The (U) (PRG1) [!].nes'
'Metroid (U).nes'
'Ninja Gaiden (USA).nes'
 SF3.nes
'Shadow Warriors 2 (E) [!].nes'
'Simpsons, The - Bartman Meets Radioactive Man (USA).nes'
 SMB3.nes
 SMW.nes
 splash.png
'Super Mario Advance 2 - Super Mario World (USA, Australia).gba'
'Super Mario Kart (USA).sfc'
'Teenage Mutant Ninja Turtles (USA).nes'
'Terminator, The (USA, Europe).nes'
'Tiny Toon Adventures (USA).nes'
'Zelda 2 - The Adventure of Link (U).nes'
axel@raspberrypi:/home/rms$
```

Figura 5: ROMs copiadas a la ruta específica

4.4. Paso 4: Programa

El programa propuesto se guarda en la misma ruta o en la de conveniecia. Ejecutamos el comando "sudo nanorun.py"

```
import pygame
import sys
import os
import subprocess
import inputs
import keyboard


# Inicializar Pygame
pygame.init()

# Configuración de la pantalla
screen = pygame.display.set_mode((1800, 1000))
pygame.display.set_caption('Menú con mando de PlayStation')

#Cargar sonido
button_sound = pygame.mixer.Sound("/home/sonido2.wav")
button_sound2 = pygame.mixer.Sound('/home/sonido1.wav')


# Colores
BLACK = (0, 0, 0)
WHITE = (255, 255, 255)
RED = (255, 0, 0)
GREEN = (0, 255, 0)

# Fuente y texto
font = pygame.font.Font(None, 36)
options = ["1. Super Mario Bros 3", "2. Super Mario World", "3. Captan America", "4. Chip 'n Dale - Rescue Rangers 2",
           "5. Donkey Kong Classics", "6. DuckTales", "7. Flintstones 2", "8. Ninja Gaiden",
           "9. Shadow Warriors 2", "10. Simpsons, The - Bartman Meets Radioactive Man", "11. Teenage Mutant Ninja Turtles", "12. Terminator",
           "13. Tiny Toon Adventures", "14. Legend of Zelda", "15. Metroid", "APAGAR","REINICIAR"]
selected_option = 0
change_speed = 1
delay = 160
BUTTON_TO_EXIT = 6

def close_mednafen():
    subprocess.run(['pkill', 'mednafen'])

def play_rom(rom_path):
```

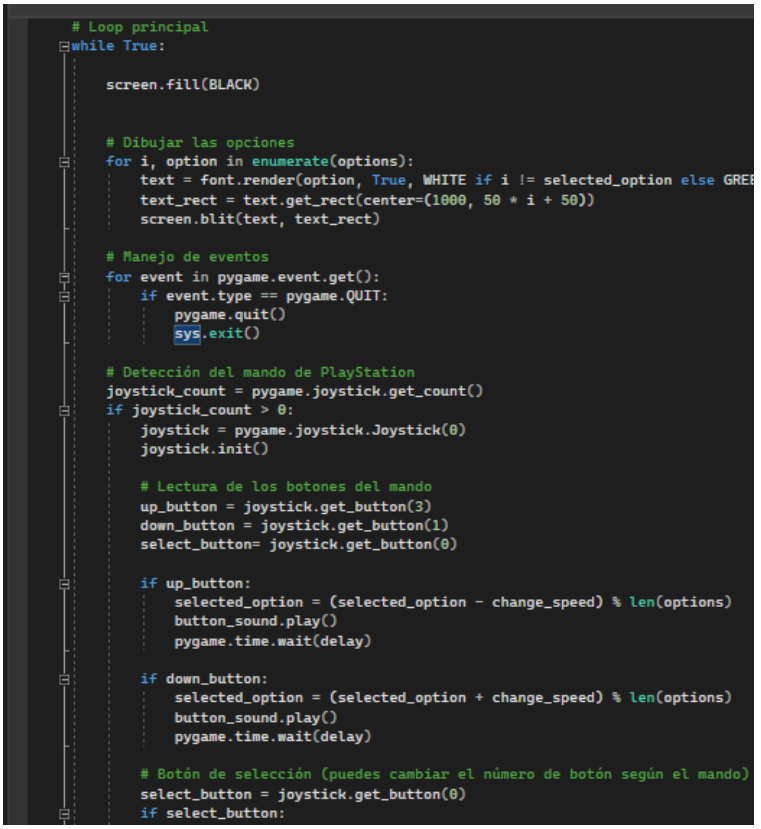
Figura 6.0: Menú

Inicialice las librerías que nos ayudarían a realizar nuestro menú, "pygame" se utilizo para poder controlar nuestras instrucciones desde el mando de consola X-box one e identifique el mando desde Python, "subprocess" principalmente fue útil para inicializar y poder abrir nuestros archivos .NES con ayuda del emulador

```
BUTTON_TO_EXIT = 6

def close_mednafen():
    subprocess.run(['pkill', 'mednafen'])

def play_rom(rom_path):
    print("Iniciando Mednafen...")
    subprocess.call(["mednafen", rom_path])  # Ejecutar Mednafen con la ruta de la ROM

# Código correspondiente al botón que deseas asignar como F12

BOTON_A_ASIGNAR = 'BTN_SELECT'

def verificar_botones():
    events = inputs.get_gamepad()
    for event in events:
        if event.ev_type == 'Key' and event.code == BOTON_A_ASIGNAR:
            if event.state == 1:  # Verifica si el botón está siendo presionado
                #keyboard.press_and_release('F12')
                close_mednafen()

# Loop principal
while True:

    screen.fill(BLACK)


    # Dibujar las opciones
    for i, option in enumerate(options):
        text = font.render(option, True, WHITE if i != selected_option else GREEN)
        text_rect = text.get_rect(center=(1000, 50 * i + 50))
        screen.blit(text, text_rect)

    # Manejo de eventos
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            pygame.quit()
            sys.exit()

    # Detección del mando de PlayStation
    joystick_count = pygame.joystick.get_count()
    if joystick_count > 0:
```

Figura 6.1: Menú

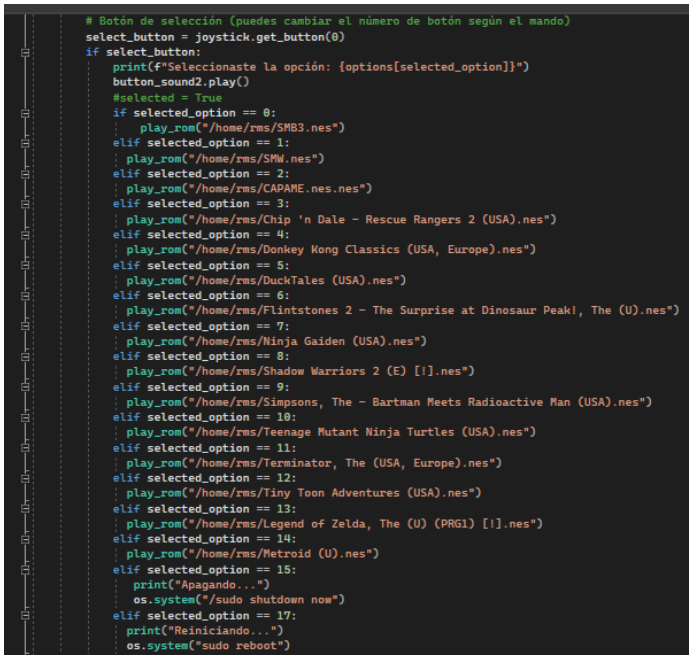
Se crearon las funciones close_mednafen y play_rom para abrir y cerrar el proceso del emulador “mednafen”.



```
# Loop principal
while True:

    screen.fill(BLACK)


    # Dibujar las opciones
    for i, option in enumerate(options):
        text = font.render(option, True, WHITE if i != selected_option else GREE
        text_rect = text.get_rect(center=(1000, 50 * i + 50))
        screen.blit(text, text_rect)

    # Manejo de eventos
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            pygame.quit()
            sys.exit()

    # Detección del mando de PlayStation
    joystick_count = pygame.joystick.get_count()
    if joystick_count > 0:
        joystick = pygame.joystick.Joystick(0)
        joystick.init()

        # Lectura de los botones del mando
        up_button = joystick.get_button(3)
        down_button = joystick.get_button(1)
        select_button= joystick.get_button(0)

        if up_button:
            selected_option = (selected_option - change_speed) % len(options)
            button_sound.play()
            pygame.time.wait(delay)

        if down_button:
            selected_option = (selected_option + change_speed) % len(options)
            button_sound.play()
            pygame.time.wait(delay)

        # Botón de selección (puedes cambiar el número de botón según el mando)
        select_button = joystick.get_button(0)
        if select_button:
```

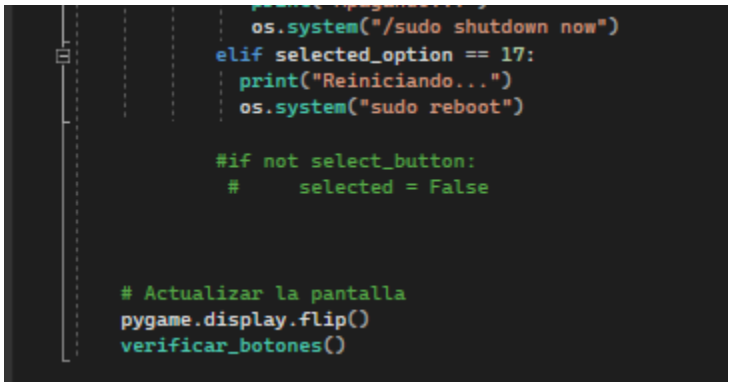
Figura 6.2: Menú

Se creo un loop principal donde se inicializaron los botones que seleccionaríamos para manejar el menú y el corrimiento del listado de los juegos, se tuvo que agregar un delay ya que se movía demasiado rápido



```
        # Botón de selección (puedes cambiar el número de botón según el mando)
        select_button = joystick.get_button(0)
        if select_button:
            print(f"Seleccionaste la opción: {options[selected_option]}")
            button_sound2.play()
            #selected = True
            if selected_option == 0:
                play_rom("/home/rms/SMB3.nes")
            elif selected_option == 1:
                play_rom("/home/rms/SMW.nes")
            elif selected_option == 2:
                play_rom("/home/rms/CAPAME.nes.nes")
            elif selected_option == 3:
                play_rom("/home/rms/Chip 'n Dale - Rescue Rangers 2 (USA).nes")
            elif selected_option == 4:
                play_rom("/home/rms/Donkey Kong Classics (USA, Europe).nes")
            elif selected_option == 5:
                play_rom("/home/rms/DuckTales (USA).nes")
            elif selected_option == 6:
                play_rom("/home/rms/Flintstones 2 - The Surprise at Dinosaur Peak!, The (U).nes")
            elif selected_option == 7:
                play_rom("/home/rms/Ninja Gaiden (USA).nes")
            elif selected_option == 8:
                play_rom("/home/rms/Shadow Warriors 2 (E) [!].nes")
            elif selected_option == 9:
                play_rom("/home/rms/Simpsons, The - Bartman Meets Radioactive Man (USA).nes")
            elif selected_option == 10:
                play_rom("/home/rms/Teenage Mutant Ninja Turtles (USA).nes")
            elif selected_option == 11:
                play_rom("/home/rms/Terminator, The (USA, Europe).nes")
            elif selected_option == 12:
                play_rom("/home/rms/Tiny Toon Adventures (USA).nes")
            elif selected_option == 13:
                play_rom("/home/rms/Legend of Zelda, The (U) (PRG1) [!].nes")
            elif selected_option == 14:
                play_rom("/home/rms/Metroid (U).nes")
            elif selected_option == 15:
                print("Apagando...")
                os.system("/sudo shutdown now")
            elif selected_option == 17:
                print("Reiniciando...")
                os.system("sudo reboot")
```

Figura 6.3: Menú



```
                os.system("/sudo shutdown now")
            elif selected_option == 17:
                print("Reiniciando...")
                os.system("sudo reboot")

            #if not select_button:
             #    selected = False



        # Actualizar la pantalla
        pygame.display.flip()
        verificar_botones()
```

Figura 6.4: Menú

En esta parte al seleccionar una opción mandamos a llamar a la función “play_rom” junto con la dirección de nuestros archivos .NES y por último la proyección de nuestra impresión grafica.

Este menú se encarga de cargar las roms y con llamadas al sistema arrancamos mednafen automáticamente,verificar muy bien si las rutas y nombres de las roms son correctas, de lo contrario, no se ejecutará el juego. Y todo esto controlado desde el mando de X-box

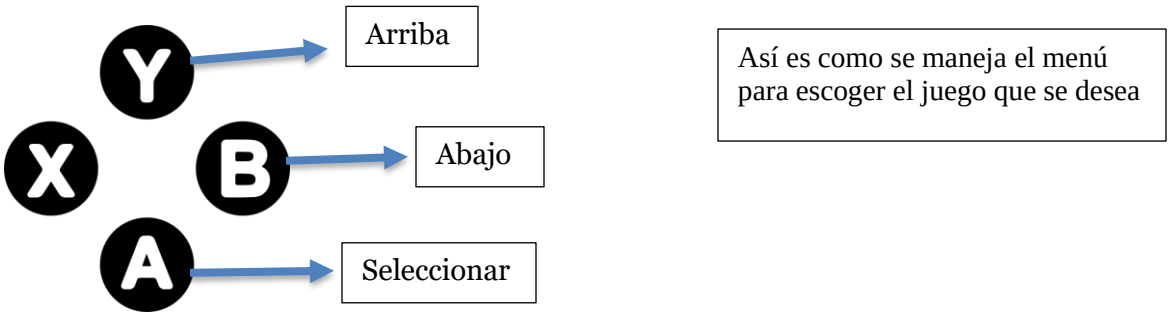


Figura 7: Diagrama del control del Menú

4.5. Paso 4: Configuración de archivo de arranque

Como podemos observar antes de que se ejecute el programa (run.py) el cual es nuestro menú, le damos un sleep de 30 segundos ya que anteriormente el programa no se podía ejecutar en ese momento ya que necesitaba que se cargaran recursos que posteriormente el programa iba a necesitar esto se creo en la ruta /etc/systemd/system/programa.service

```
[Unit]
 Description=programa

[Service]
 ExecStartPre=/bin/sleep 30
 ExecStart=/usr/bin/python3 /etc/init.d/run.py
 Restart=always
 User=root

[Install]
 WantedBy=multi-user.target
```

Figura 11 : Archivo de arranque.5

Después de esto recargamos systemd con sudo systemctl daemon-reload , habilitamos el servicio sudo systemctl enable mi_programa.service y por ultimo podemos iniciar el servicio manualmente para probarlo.

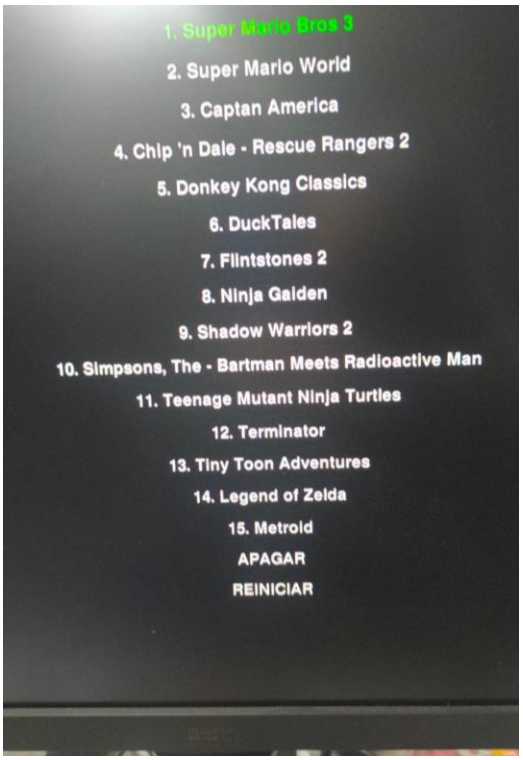


Figura 8: Resultado de la interfaz gráfica del menú.

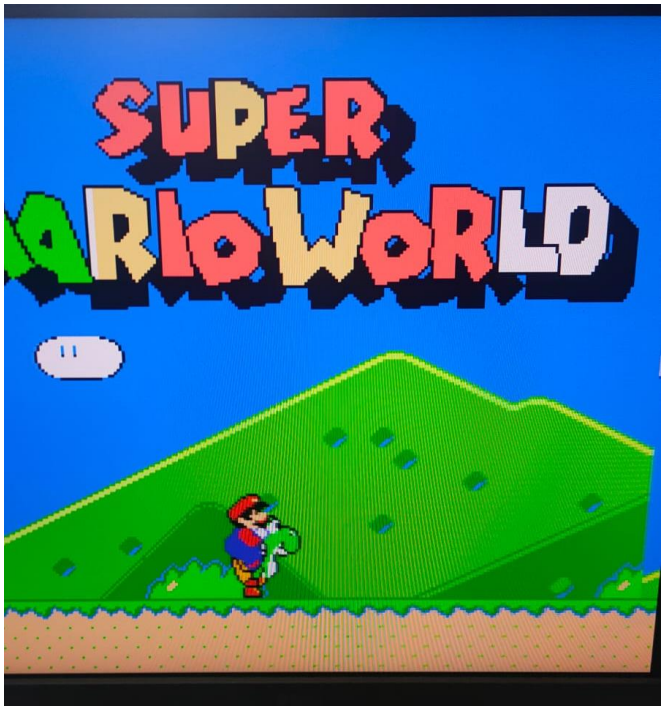


Figura 9: Visualización de los videojuegos.

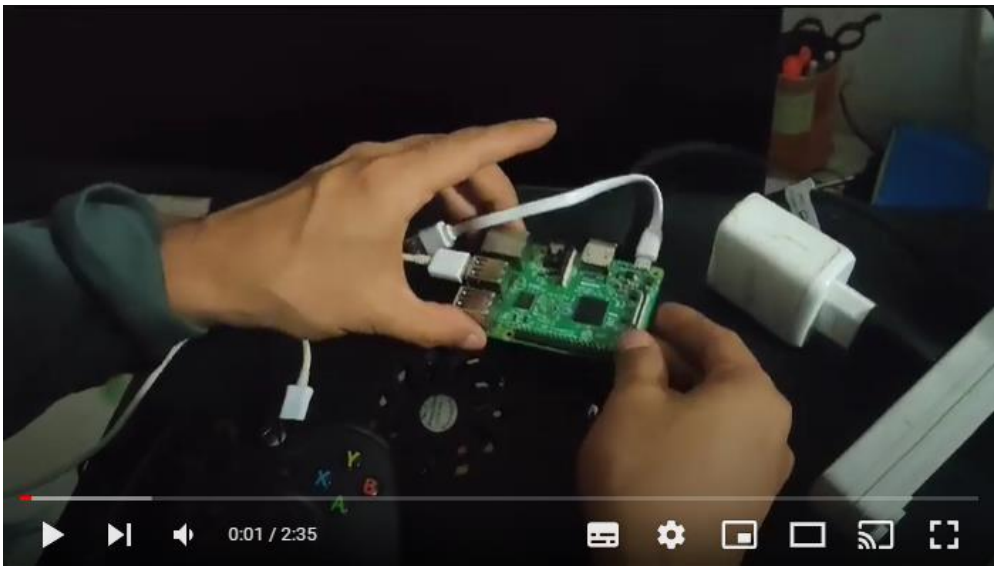


Figura 10: Enlace al video subido a YouTube

https://youtu.be/6l1N-wfPnZo

4.6. Paso 5: Configuración de mando o control.

Al iniciar cualquier juego presionamos Ctrl+Shift+1 y seguimos las instrucciones en pantalla, se deben de presionar los botones mostrados para configurarlos y se guarden. Este paso solo se hace la primera vez, posteriormente ya no será necesario.

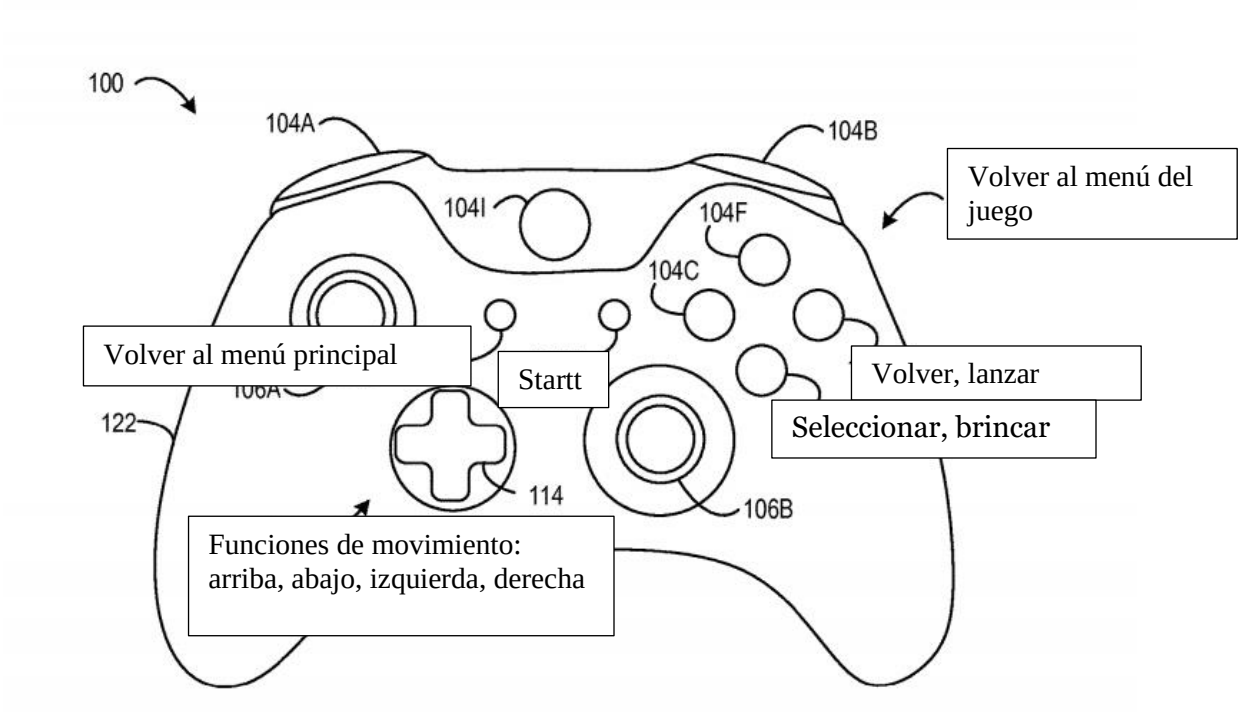


Figura 11: Configuración del mando dentro del emulador

1. Experimentos

[5pt] Ejecute un juego y configure los botones para el control o mando que se haya elegido

[2pt] Modifique el programa para que al cerrar un juego se presente nuevamente todos los juegos disponibles. [3pt]

Agregue mínimo 5 ROMs más, se muestren en la interfaz y ejecute los juegos.

[+3pt] Haga las configuraciones para que al encender la raspberry se ejecute el programa sin necesidad de arrancarlomanualmente.

2. Cuestionario

¿Qué otras herramientas de emulación existen, para el propósito de esta práctica?

¿Podemos cargar juegos de playstation o gamecube con mednafen?

¿Se considera ilegal el reproducir copias de juegos de Nintendo de la Manera en la que se hacen en esta práctica?

3. Referencias

[1] Foro insertmorecoins - Como se configura el mando en Mednafen? (s. f.). https://insertmorecoins.es/foro/consolas/como-se-configura-el-mando-en-mednafen/Ludmila Maceková. 1-wire-the technology for sensor networks. *Acta Electrotechnica et Informatica*, 12(4):52,2012. Last accessed 14/06/2023

[2] *Mednafen General Documentation*. (s. f.). https://mednafen.github.io/documentation/. Last Accessed 13/11/2023

[3] Emulators - Raspberry Pi Gaming - page 62. (s. f.). http://what-when-how.com/Tutorial/topic-213lnr/Raspberry-Pi-Gaming-90.html Last Accessed 11/06/2023

[4] Ellingwood, J. (2020). Cómo usar Systemctl para gestionar servicios y unidades de Systemd. DigitalOcean. https://www.digitalocean.com/community/tutorials/how-to-use-systemctl-to-manage-systemd-services- and-units-es. Last Accessed 14/11/2023

4.Enlaces.

https://youtu.be/6l1N-wfPnZo

https://github.com/Rodrigoivan09/FSEm-2024-1.git