

Manual Técnico

RetroBox Pi

Hernández Zamora José Gael
Lara Acevedo Cristian Alexis
Sanchez Alvirde Andrés Iván

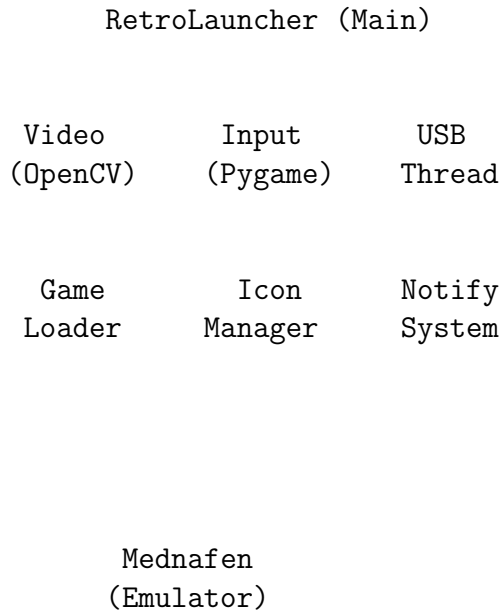
Noviembre 2025

Índice

1. Arquitectura del Sistema	2
1.1. Diagrama de Componentes	2
1.2. Stack Tecnológico	2
2. Estructura del Código	2
2.1. Clase Principal: RetroLauncher	2
2.2. Sistema de Video	3
2.3. Carga de Juegos	3
2.4. Detección USB	3
3. Configuración del Sistema	4
3.1. Variables de Configuración	4
3.2. Mapeo de Botones	4
4. Gestión de Procesos	4
4.1. Subprocess para Mednafen	4
4.2. Monitoreo y Terminación	5
5. Optimización	5
5.1. Control de FPS	5
5.2. Reducción de Carga	5
6. Debugging	5
6.1. Logs de Sistema	5
6.2. Test de Gamepad	5
7. Seguridad	6

1. Arquitectura del Sistema

1.1. Diagrama de Componentes



1.2. Stack Tecnológico

Componente	Tecnología	Versión	Función
Lenguaje	Python	3.7+	Core de aplicación
UI Framework	Pygame	2.0+	Renderizado gráfico
Video	OpenCV (cv2)	4.5+	Video de fondo
Emulador	Mednafen	1.26+	Ejecución de ROMs
Concurrencia	threading	stdlib	Detección USB

2. Estructura del Código

2.1. Clase Principal: RetroLauncher

```
1 class RetroLauncher:
2     def __init__(self):
3         # Inicializacion de Pygame y pantalla completa
4         # Carga de recursos (video, fuentes, iconos)
5         # Escaneo de juegos
6         # Inicio de hilo USB
```

Variables de instancia principales:

- `screen_width`, `screen_height`: Resolución detectada automáticamente
- `screen`: Superficie Pygame para renderizado
- `video_cap`: Capturador OpenCV para video de fondo
- `games`: Lista de juegos detectados
- `selected_index`: Índice del juego seleccionado
- `joystick`: Dispositivo de entrada detectado

2.2. Sistema de Video

Propósito: Renderizar video de fondo en loop infinito

```

1 def update_video_frame(self):
2     ret, frame = self.video_cap.read()
3     if not ret:
4         self.video_cap.set(cv2.CAP_PROP_POS_FRAMES, 0)
5     frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
6     # Convertir a superficie Pygame y escalar

```

2.3. Carga de Juegos

```

1 def load_games(self):
2     extensions = ['*.nes', '*.sfc', '*.smc', '*.gba', '*.gb', '*.gbc']
3     for ext in extensions:
4         for file_path in glob.glob(os.path.join(ROMS_PATH, ext)):
5             # Extraer metadata y agregar a lista

```

Estructura de datos:

```

1 {
2     'name': 'Nombre sin extension',
3     'path': '/ruta/completa/archivo.nes',
4     'ext': '.nes'
5 }

```

2.4. Detección USB

Algoritmo:

1. Verificar cada 5 segundos si existe `/dev/sda1`
2. Si detectado y no procesado:
 - Montar en `/mnt/usb`
 - Escanear archivos ROM
 - Copiar nuevos juegos a `/home/pi/roms/`

- Desmontar USB
- Recargar lista de juegos
- Mostrar notificación

3. Si USB removido, resetear flag

3. Configuración del Sistema

3.1. Variables de Configuración

```

1 # Rutas del sistema
2 ROMS_PATH = "/home/pi/roms"
3 ICONS_PATH = "/home/pi/icons"
4 LOGO_PATH = "/home/pi/LogoConsola.png"
5 VIDEO_PATH = "/home/pi/FondoConsola.mp4"
6 FONT_PATH = "/home/pi/retro.ttf"
7
8 # Configuración USB
9 USB_DEVICE = "/dev/sda1"
10 USB_MOUNT_POINT = "/mnt/usb"
11
12 # Paleta de colores (RGB)
13 COLOR_WHITE = (255, 255, 255)
14 COLOR_YELLOW = (255, 215, 0)
15 COLOR_RED = (255, 50, 50)
16 COLOR_GREEN = (50, 255, 50)

```

3.2. Mapeo de Botones

```

1 BTN_A = 0
2 BTN_B = 1
3 BTN_X = 2
4 BTN_Y = 3
5 BTN_LB = 4
6 BTN_RB = 5
7 BTN_SELECT = 6
8 BTN_START = 7
9 BTN_XBOX = 8

```

Para otros gamepads: Usar `jstest /dev/input/js0` y ajustar valores

4. Gestión de Procesos

4.1. Subprocess para Mednafen

```

1 launch_cmd = [
2     'mednafen',
3     '-video.fs', '1',
4     '-command.exit', '0',
5     game_path
6 ]
7 process = subprocess.Popen(launch_cmd)

```

4.2. Monitoreo y Terminación

```

1 while game_running:
2     if process.poll() is not None:
3         game_running = False
4
5     if SELECT and START pressed:
6         process.terminate()
7         try:
8             process.wait(timeout=2)
9         except subprocess.TimeoutExpired:
10            process.kill()

```

5. Optimización

5.1. Control de FPS

Raspberry Pi 3: 24 FPS
Raspberry Pi 4/5: 30 FPS

5.2. Reducción de Carga

Si hay lag, reducir resolución del video antes de procesarlo.

6. Debugging

6.1. Logs de Sistema

```
python3 /home/pi/retro_launcher.py 2>&1 | tee debug.log
```

6.2. Test de Gamepad

```
sudo apt install joystick
jstest /dev/input/js0
```

7. Seguridad

El archivo `/etc/sudoers` permite operaciones específicas:

```
pi ALL=(ALL) NOPASSWD: /bin/mount, /bin/umount, /sbin/poweroff
```

Solo permite estos comandos, no acceso root completo.