Proyecto Final:

Consola Arcade

Objetivo

El alumno realizará un sistema embebido basado en una consola de videojuegos retro.

Introducción

MAME

MAME es un framework de emulación multipropósito.

El propósito de MAME es preservar décadas de historia del software. A medida que la tecnología electrónica sigue avanzando, MAME evita que este importante software "antiguo" se pierda y se olvide. Esto se logra documentando el hardware y cómo funciona. Con el tiempo, MAME (originalmente significaba Multiple Arcade Machine Emulator) absorbió el proyecto hermano MESS (Multi Emulator Super System), por lo que MAME ahora documenta una amplia variedad de computadoras (en su mayoría antiguas), consolas de videojuegos y calculadoras, además de los videojuegos arcade que fueron su enfoque inicial.

Hypseus Singe

Es un programa para jugar juegos de arcade laserdisc en PC, Mac o Raspberry Pi

Lista de materiales

- Raspberry Pi 3 o 4
- Memoria microSD
- Fuente de alimentación regulada a 5V y al menos 2 amperios de Salida.

Desarrollo

Instalación del SO

Para comenzar con la implementación de este proyecto, se tiene que instalar el sistema operativo con el cual va a trabajar la Raspberry Pi y el sobre el cual se va a montar el emulador. Hay diferentes formas de hacerlo, pero por practicidad se opta por usar un programa en Windows para instalar el sistema operativo (Raspberry Pi OS) directo en la memoria microSD llamado *Raspberry pi os imager*.

La versión del sistema operativo que se va a ocupar y que es si o si necesario que sea específicamente la versión *Lite*, porque la parte gráfica para el emulador se instalará de manera independiente, y así se evitará que haya procesos que no se requieren para el sistema que se va a desarrollar. Descargar en la página oficial y vienen más a detalle como instalarlo de esta forma.

https://www.raspberrypi.com/software/

Configuración para trabajar de remotamente con SSH

Una vez hecho esto, se conecta la memoria microSD con el sistema operativo a la Raspberry y se procede a encender. Por ser la primera vez que se ocupa el sistema operativo desde que se

instaló, se debe hacer un login por medio de la consola con el siguiente *usuario* y *contraseña* que viene por defecto (se puede cambiar, pero no es necesario) de la siguiente manera:

```
login: pi
password: raspberry
```

Posterior a esto se habilita y se inicia el servicio de *SSH* para poder trabajar desde otro equipo por comodidad, pero si prefieres esto puede omitirse.

```
sudo systemctl enable ssh
sudo systemctl start ssh
```

En caso de que se opte por trabajar mediante SSH, se debe conocer la dirección IP para poder conectarse de manera exitosa.

```
hostname -I
```

Desde la computadora donde se desea trabajar vía remota, se ingresa el siguiente comando (recuerda sustituir lo que está con los pico-paréntesis) y previamente se tuvo que haber instalado un cliente SSH para poder trabajar de esta forma, uno de ellos puede ser PuTTY.

```
ssh pi@<RaspberryPi_IP>
```

Se actualizan los paquetes del sistema

```
sudo apt-get update && sudo apt-get upgrade -y
```

Creación de modos de trabajo

Para proceder a la instalación del emulador, primero hay que crear 2 modos de ejecución; un modo *Arcade* que es en el cual se va a ejecutar automáticamente el emulador y un modo *Servicio* para poder dar mantenimiento, configuraciones y otras posibles cosas por medio de consola.

Esto se logra editando el archivo *bash_aliases* y agregando unas líneas de la siguiente forma:

```
nano ~/.bash_aliases
```

```
alias update='sudo apt-get update && sudo apt-get upgrade -y'
alias cputemp='/usr/bin/vcgencmd measure_temp'
alias cpufreq="echo Clock Speed=$(($(/usr/bin/vcgencmd measure_clock arm | awk -F '=' '{print
$2}')/1000000)) MHz"
alias frontend='_frontend(){ if [[ "${1,,}" =~ ^(mame|attract|advance)$ ]]; then [ ! -f
/home/pi/settings ] && echo FRONTEND=mame>/home/pi/settings; sed -i
"s/^FRONTEND=.*$/FRONTEND="${1,,}"/g" $(readlink /home/pi/settings) && echo "Frontend set to:
"${1,,} " (reboot to apply)."; case "${1,,}" in mame) [ -z "$2" ] && (grep -q AUTOROM=
/home/pi/settings && sed -i "s/^AUTOROM=.*$/AUTOROM=/g " $(readlink /home/pi/settings)) || (grep
-q AUTOROM= /home/pi/settings && sed -i "s/^AUTOROM=.*$/AUTOROM="${2,,}"/g" $(readlink
/home/pi/settings) | | (echo AUTOROM="${2,,}" | tee -a /home/pi/settings; echo | tee -a
/home/pi/settings); echo "Automatic ROM Launch is set to: "${2 ,,}".") ;; attract) CFG=$(find
/home/pi/.attract/emulators -type f -iname "$2".cfg | head -n 1); ([ -z "$2" ] || [ ! -f $CFG ])
&& (grep -q AUTOROM= /home/pi/settings && sed -i "s/^AUTOROM=.*$/AUTOROM=/g" $(readlink
/home/pi/settings)); [! -z "$2" ] && [ ! -z "$3" ] && [ -f $CFG ] && (grep -q AUTOROM=
/home/pi/settings && sed -i "s/^AUTOROM=.*$/AUTOROM=\""${2^} " "${3,,}"\"/g" $(readlink
/home/pi/settings) || (echo AUTOROM=\""${2^}" "${3,,}"\" | tee -a /home/pi/settings; echo | tee -
a /home/pi/settings); echo "Automatic ROM Launch is set to: "${3,,}" (emulator "${2^}") .") ;;
bag; else echo "Invalid or missing argument. Try: mame [rom],

# Aliases to switch between Arcade Mode and Service Mode
alias arcademode='sudo systemctl enable mame-autostart.service'
alias servicemode='sudo systemctl disable mame-autostart.service'
```

```
alias mode='echo -n "The system is currently in " ; systemctl -q is-active mame-autostart.service
&& echo -n ARCADE || echo -n SERVICE; echo "mode."'
```

Para efectuar cambios hay que hacer logout y login de nuevo.

A continuación, se crea una carpeta en donde se encontrarán los scripts para instalar el emulador y el programa que permite jugar los juegos de arcade en la Raspberry, es decir, MAME (Multiple Arcade Machine Emulator) y Hypseus. Luego se creará un script para verificar las últimas versiones de estos programas.

```
mkdir ~/scripts
nano ~/scripts/versions-check.sh
```

El script se anexará, pero parte del contenido es el siguiente, en el cual solo verifica cual es la última versión disponible en el repositorio oficial de MAME,

```
function  mame-check {
  CHECKURL=https://github.com/mamedev/mame/releases/latest
  HTMLTAG= ' <title>Release MAME '

  [ -x /home/pi/mame/mame ] &&  export MAMEVER= $( /home/pi/mame/mame -version | cut -d '  ' -f1
)
   if [ !  -z  $MAMEVER ] ;  then
    LATESTMAMEVER= $( wget -q -O - $CHECKURL  | grep " $HTMLTAG "  | awk ' {print $3} ' )

    if [ -z  $LATESTMAMEVER ] ;  then  echo  $MAMEVER ERROR ;  exit ;  fi   # We make sure wget
was successful

    if [ $( version $LATESTMAMEVER )  -gt  $( version $MAMEVER ) ] ;  then
      echo  $MAMEVER  $LATESTMAMEVER
    else
      echo  $MAMEVER Latest
     fi
  fi
  }
```

Se le otorgan los permisos de ejecución a todos los usuarios del script anterior

```
chmod +x ~/scripts/versions-check.sh
```

Se hace configuración de la zona horaria, lenguaje de teclado y el servicio de NTP

```
sudo dpkg-reconfigure tzdata
sudo dpkg-reconfigure keyboard-configuration
sudo dpkg-reconfigure locales

sudo nano /etc/systemd/timesyncd.conf
NTP=ca.pool.ntp.org
```

Haciendo un cambio en el hostname, en los siguientes archivos se busca la línea que comience con *127.0.0.1* y se reemplaza *raspberrypi* con algún nombre que queramos

```
sudo nano /etc/hostname
sudo nano /etc/hosts
```

se hace un reinicio para aplicar cambios

```
sudo reboot
```

Gráficos con SDL2

Después de reiniciar, se crea un grupo llamado *render* para la parte de los gráficos que se ocuparan de SDL2.

```
sudo usermod -a -G render pi
```

Algo opcional también es deshabilitar algunos servicios para tener un mejor rendimiento al momento de prender la raspberry como wifi, dns, bluetooth (si se ocupará un mando de este tipo no lo deshabiliten), ipv6, entre otros.

Regresando a SDL2, se debe habilitar o crear una forma de que el emulador no se ejecute en modo ventana, sino en tamaño pantalla completa y para esto hay que quitar la parte del soporte de X11 con ayuda de un script llamado *sdl2-latest.sh* (se anexa también el script), se le da permisos de ejecución a todos los usuarios y se procede a compilar.

```
nano ~/scripts/sdl2-latest.sh
chmod +x ~/scripts/sdl2-latest.sh
~/scripts/sdl2-latest.sh
```

El tipo de fuente en los textos que ocupará SDL2 se descargan e instalan de manera independiente de la siguiente forma:

```
cd~
wget https://libsdl.org/projects/SDL_ttf/release/SDL2_ttf-2.0.15.tar.gz

tar zxvf SDL2_ttf-2.0.15.tar.gz
cd SDL2_ttf-2.0.15
./configure
make -j $(nproc)
sudo make install
sudo ldconfig -v
```

De igual manera que con SDL2, se debe hacer un ajuste al emulador para que no se ejecute con X11 mediante la creación de un parche llamado *drawbgfx.cpp.*

```
nano ~/scripts/drawbgfx.cpp.patch
```

Se crea un script para que el emulador se compile de manera automática (se anexa el script), se asignan permisos de ejecución.

```
nano ~/scripts/mame-updater.sh
chmod +x ~/scripts/mame-updater.sh
```

Posteriormente se instalan las dependencias para el emulador

```
sudo apt-get install fontconfig libfontconfig-dev libx11-dev libpulse-dev -y
```

Ahora pasando con Hypseus-Singe, se crea un script para compilar (se anexa) y se asignan permisos de ejecución y se compila.

```
nano ~/scripts/hypseus-build.sh
chmod +x ~/scripts/hypseus-build.sh
~/scripts/hypseus-build.sh
```

Autostart

Para diferenciar la consola arcade de un emulador en algún dispositivo hay que hacer que se inicie de manera automática el emulador, es decir, un autostart.

```
nano ~/scripts/autostart.sh
chmod +x ~/scripts/autostart.sh
```

Además de este script, se debe crear el servicio que se ejecute en segundo plano siempre con ayuda de systemd y que además es el encargado de realizar esta función con diferentes programas. Si el servicio se encuentra activo esto hace que se encuentre en modo Arcade la raspberry, del o contrario se encontrará en modo Servicio.

```
sudo systemctl edit mame-autostart.service --force --full
```

El contenido debe ser el siguiente:

```
[Unit]
Description=MAME Appliance Autostart service
Conflicts=getty@tty1.service smbd.service nmbd.service rng-tools.service cron.service mame-
artwork-mgmt.service
Requires=local-fs.target
After=local-fs.target
ConditionPathExists=/home/pi/settings

[Service]
User=pi
Group=pi
PAMName=login
Type=simple
EnvironmentFile=/home/pi/settings
ExecStart=/home/pi/scripts/autostart.sh
Restart=on-abort
RestartSec=5
TTYPath=/dev/tty1
StandardInput=tty

[Install]
WantedBy=multi-user.target
Also=shutdown.service
```

Donde el apartado *description* nos dará información sobre el servicio y es lo que se muestra en la consola. El apartado *service* es para indicar las rutas, a que usuarios pertenecen, entre otras. Y la sección *install* es como systemd instala el servicio.

También se debe crear un servicio para que cuando se cierre el emulador, se apagué la raspberry.

```
sudo systemctl edit shutdown.service --force --full
```

Cuyo contenido debe ser el siguiente:

```
[Unit]
Description=Shutdown and poweroff service
After=mame-autostart.service
```

```
[Service]
TTYPath=/dev/tty1
ExecStart=/sbin/poweroff
StandardInput=tty

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

Logo al iniciar y apagar Raspberry

Hay que recordar que para que tenga una apariencia más cercana a los arcades, al encender se debe ver un logo/imagen o un pequeño video alusivo a lo que representa el dispositivo, para hacer esto se debe hacer lo siguiente:

Entrar a *config.txt*

```
sudo nano /boot/config.txt
```

Agregar la siguiente línea al final para deshabilitar el logo que viene por defecto y colocar otro

```
disable_splash=1
```

En el archivo *cmdline.txt* se deshabilitará el logo que aparece de la Raspberry pi

```
sudo nano /boot/cmdline.txt
```

Se agrega para quitar el logo de Raspberry pi

```
logo.nologo
```

Crear una carpeta donde se encontrará el nuevo logo, puede ser una imagen plana o video, se le colocará un nombre distintivo como mame.jpg

Se lo mismo para la imagen cuando se apaga la Raspberry

```
mkdir ~/splash
sudo mv ~/mame.jpg ~/splash
ln -s ~/splash/mame.jpg ~/splash/mame-boot.jpg
```

Se crea el servicio

```
sudo systemctl edit mame-bootsplash.service --force --full
```

Contiene lo siguiente y posteriormente se activa el servicio

```
[Unit]
Description=MAME Boot Splash Screen service
DefaultDependencies=no
After=local-fs.target

[Service]
Type=simple
StandardInput=tty
StandardOutput=tty
ExecStart=/usr/bin/fim -q --no-history /home/pi/splash/mame-boot.jpg
SuccessExitStatus=42
```

```
[Install]
WantedBy=sysinit.target


sudo systemctl enable mame-bootsplash.service
```

Para al caso de cuando se apaga se crea el siguiente servicio y su contenido correspondiente

```
sudo systemctl edit mame-shutdownsplash.service --force --full


[Unit]
Description=MAME Shutdown/Poweroff/Reboot Splash Screen service
DefaultDependencies=no
Before=halt.target
Conflicts=mame-bootsplash.service

[Service]
Type=simple
ExecStart=/usr/bin/fim -q --no-history /home/pi/splash/mame-shutdown.jpg
SuccessExitStatus=42

[Install]
WantedBy=reboot.target halt.target poweroff.target


sudo systemctl enable mame-shutdownsplash.service
sudo systemctl start mame-shutdownsplash.service
```

Se reinicia

```
sudo reboot
```

Se pasa a modo arcade y nuevamente reiniciar para comenzar a utilizar

```
arcademode
sudo reboot
```

Conclusión

La implementación del arcade fue complicada en un inicio, porque nos atoramos en la parte de la ejecución automática del emulador al encender la Raspberry por unos días, hasta que pudimos encontrar una solución y notamos que estábamos implementando mal el autor un. Por otro lado, se pudo verificar que si es importante las versiones de las dependencias o programas que se requieren para el desarrollo de proyectos, porque pueden ocasionar errores al momento de trabajar con otros programas o incluso puede que esas versiones no se puedan descargar correctamente, ya que las quitan de los repositorios.

Bibliografía.

DirtBagXon. "GitHub - DirtBagXon/hypseus-singe: Hypseus is a SDL2 version of Daphne and Singe. Laserdisc game emulation." GitHub. https://github.com/DirtBagXon/hypseus-singe (accedido el 11 de enero de 2023).

"MAMEdev.org | Home of The MAME Project". MAMEdev.org | Home of The MAME Project. https://www.mamedev.org (accedido el 10 de enero de 2023).

Raspberry Pi Foundation. "Raspberry Pi OS". Raspberry Pi. https://www.raspberrypi.com/software/ (accedido el 11 de enero de 2023).

sonicprod. "How to make a dedicated MAME Appliance on a Raspberry Pi 4B | Comment réaliser un système MAME dédié sur un Raspberry Pi 4/Pi 400". Github Gist. https://gist-github-com.translate.goog/sonicprod/f5a7bb10fb9ed1cc5124766831e120c4?_x_tr_sl=fr&_x_tr_tl=en&_x_tr_hl=fr (accedido el 12 de enero de 2023).