## **Audio Studio Link with SRT**

**Service Diagram**

รูปภาพประกอบด้วย ข้อความ, ภาพหน้าจอ, ตัวอักษร, เครื่องหมาย

เนื้อหาที่สร้างโดย AI อาจไม่ถูกต้อง

**Audio Encoder**

Hardware Requirements

1. Raspberry Pi

2. Audio Injector Card

Software Requirements

1. FFMPEG with Support srt , alsa

## ขั้นตอนการตั้งค่า Streaming เสียงผ่าน SRT

### 1. ตรวจสอบว่า ffmpeg รองรับ libsrt

ให้ใช้คำสั่งนี้เพื่อเช็ค:

bash

คัดลอกแก้ไข

ffmpeg -protocols | grep srt

ถ้าเห็นคำว่า srt แสดงว่ารองรับแล้ว

### 2. ค้นหาอุปกรณ์เสียงให้แน่ใจ

จากที่คุณมีอยู่:

bash

คัดลอกแก้ไข

card 1: Device [USB Advanced Audio Device], device 0: USB Audio [USB Audio]

อุปกรณ์นี้ใน ffmpeg จะเป็น:

makefile

คัดลอกแก้ไข

hw:1,0

### 3. เริ่มสตรีมเสียงผ่าน SRT

ตัวอย่างคำสั่ง ffmpeg:

bash

คัดลอกแก้ไข

ffmpeg -f alsa -i hw:1,0 -acodec libmp3lame -b:a 128k -f mpegts "srt://0.0.0.0:1234?mode=listener"

#### อธิบายคำสั่ง:

* -f alsa: ใช้ ALSA เป็นแหล่งเสียง (Linux sound API)
* -i hw:1,0: ระบุ Sound Card ที่จะใช้
* -acodec libmp3lame: เข้ารหัสเสียงเป็น MP3
* -b:a 128k: บิทเรตเสียง 128kbps
* -f mpegts: ใช้ container แบบ MPEG-TS
* srt://0.0.0.0:1234?mode=listener: ฟังที่พอร์ต 1234 ด้วย SRT

## 📲 ฝั่งผู้รับ: เชื่อมต่อเพื่อฟัง

### ถ้าใช้ ffplay (เช่นบน Windows หรือ Linux):

bash

คัดลอกแก้ไข

ffplay "srt://<ip-raspberry-pi>:1234"

### หรือถ้าจะเล่นบนมือถือ:

ใช้แอปเช่น **VLC Media Player** (รองรับ SRT ในบางเวอร์ชัน) แล้วเปิด stream:

cpp

คัดลอกแก้ไข

srt://<ip-raspberry-pi>:1234

## ✅ เคล็ดลับเพิ่มเติม

* ถ้าเสียงขาด ๆ หาย ๆ ลองเพิ่ม buffer:

bash

คัดลอกแก้ไข

srt://0.0.0.0:1234?mode=listener&latency=2000

* อย่าลืมเปิด port 1234 ใน firewall หรือ router ถ้ามีการเข้าถึงจาก network อื่น

หากคุณอยากใช้ WebRTC หรือ HTTP Live Streaming (HLS) แทนในภายหลังก็ทำได้เหมือนกันนะครับ แต่ SRT ถือว่า latency ต่ำและเสถียรมากสำหรับเสียงแบบ real-time

##### คุณพูดว่า: สำหรับฝั่งผู้รับ เชื่อมต่อให้กี่เครื่อง

###### ChatGPT พูดว่า:

สำหรับ **SRT ในโหมด listener (แบบ unicast)** โดยค่าเริ่มต้น **สามารถรับการเชื่อมต่อจากได้เพียงครั้งละ 1 เครื่องเท่านั้น** ต่อ 1 stream/pipeline ครับ เพราะ ffmpeg ที่เป็น listener จะรับเพียง client เดียวต่อหนึ่งกระบวนการเท่านั้น

## 🎯 วิธีแก้/ทางเลือกในการให้หลายเครื่องรับฟังได้พร้อมกัน

### ✅ ตัวเลือกที่ 1: เปิดหลายพอร์ต SRT (หลาย instance)

รัน ffmpeg หลายอัน โดยแต่ละอันเปิด srt listener คนละพอร์ต

bash

คัดลอกแก้ไข

ffmpeg -f alsa -i hw:1,0 -acodec libmp3lame -b:a 128k -f mpegts "srt://0.0.0.0:1234?mode=listener"

ffmpeg -f alsa -i hw:1,0 -acodec libmp3lame -b:a 128k -f mpegts "srt://0.0.0.0:1235?mode=listener"

...

ข้อดี: ง่าย ข้อเสีย: เปลือง resource และไม่ scalable ถ้าต้องการหลายเครื่อง

### ✅ ตัวเลือกที่ 2: ใช้ **SRT in caller mode + relay server** (เช่น OBS, ffmpeg, SRT-live-transmit, Haivision)

* ฝั่ง Raspberry Pi สตรีมไปยัง SRT server (แบบ caller)
* เครื่องอื่นๆ ค่อยมาเชื่อมต่อกับ server นั้น (listener หลายคน)

เช่น สตรีมไปยังเซิร์ฟเวอร์กลางที่รัน srt-live-transmit:

bash

คัดลอกแก้ไข

ffmpeg -f alsa -i hw:1,0 -acodec libmp3lame -b:a 128k -f mpegts "srt://<server-ip>:9999?mode=caller"

### ✅ ตัวเลือกที่ 3: ใช้ ffmpeg + nginx + HLS หรือ Icecast สำหรับ Audio Streaming

ถ้าต้องการให้หลายคนเปิดฟังเหมือนวิทยุออนไลน์

* Encode เสียงเป็น AAC/MP3
* ส่งไปยัง Icecast หรือ nginx-rtmp-module ที่รองรับ HLS/HTTP
* แล้วให้ผู้ใช้ฟังผ่านเบราว์เซอร์/มือถือได้ง่าย

### ✅ ตัวเลือกที่ 4: ใช้ SRT Multipoint relay (SRT Gateway)

มีโซลูชันเชิงพาณิชย์ เช่น:

* Haivision SRT Gateway
* GStreamer (รองรับ multicast ด้วย SRT)
* หรือใช้งานผ่าน SRT Hub/relay ของ Wowza หรือ OBS ด้วย script

## 🔚 สรุปง่ายๆ

| **วิธี** | **รองรับหลายเครื่อง** | **เหมาะกับ** |
| --- | --- | --- |
| เปิดหลาย port | ✔️ แต่ยุ่ง | ทดสอบเล็ก ๆ |
| SRT + Relay Server | ✔️ ✔️ ✔️ | ใช้งานจริงหลาย client |
| Icecast / HLS | ✔️✔️✔️ | ใช้กับ browser, มือถือ |
| Multicast (ขั้นสูง) | ✔️✔️✔️ | ระบบ broadcast ภายใน LAN |

ถ้าคุณบอกจำนวน client ที่ต้องรองรับ หรือว่าใช้งานใน LAN หรือผ่าน internet ได้ ผมช่วยออกแบบให้เหมาะได้เลยครับ 😎

4o

##### คุณพูดว่า: จะ run SRT-live-transmit บน Raspberry pi เพื่อรองรับผู้ฟังหลายคนได้หรือไม่

###### ChatGPT พูดว่า:

เยี่ยมเลยครับ! คำถามนี้ตรงจุดมาก 👍

## ✅ สรุปสั้น ๆ:

**ใช่ครับ! srt-live-transmit สามารถรันบน Raspberry Pi ได้** และสามารถทำตัวเป็น **SRT relay server** เพื่อรองรับผู้ฟังหลายคนได้ **พร้อมกัน** โดยให้ Raspberry Pi รับเสียง (stream ต้นทาง) และ forward ไปยังผู้รับหลายคน

## 🔁 ภาพรวมแนวคิด

* ffmpeg จะส่งเสียงจาก USB Sound Card ไปยัง srt-live-transmit ที่รันเป็น **listener**
* srt-live-transmit จะเปิดพอร์ตสำหรับ **ผู้ฟังหลายคน (listener)** ได้เชื่อมต่อมาพร้อมกัน

## 🔧 ตัวอย่างการตั้งค่า

### 1. ติดตั้ง srt-live-transmit บน Raspberry Pi

ถ้ายังไม่มี ให้ติดตั้งผ่าน SRT library:

bash

คัดลอกแก้ไข

sudo apt update

sudo apt install libsrt-tools

เช็คว่าใช้ได้:

bash

คัดลอกแก้ไข

srt-live-transmit --help

### 2. เริ่มต้น SRT Relay Server (บน Raspberry Pi)

ให้ srt-live-transmit รับสัญญาณที่พอร์ต 9000 แล้ว broadcast ไปยังพอร์ต 9001

bash

คัดลอกแก้ไข

srt-live-transmit "srt://:9000?mode=listener" "srt://:9001?mode=listener"

หมายความว่า:

* ffmpeg จะส่งไปที่ srt://localhost:9000
* ฝั่งผู้ฟังจะรับจาก srt://<ip-raspberry-pi>:9001
* หลาย client เชื่อมต่อ 9001 ได้พร้อมกัน!

### 3. สตรีมเสียงด้วย ffmpeg

bash

คัดลอกแก้ไข

ffmpeg -f alsa -i hw:1,0 -acodec libmp3lame -b:a 128k -f mpegts "srt://127.0.0.1:9000?mode=caller"

### 4. เชื่อมต่อฝั่งผู้ฟังหลายเครื่อง

บนคอมพิวเตอร์หรือมือถือ (ที่รองรับ SRT):

bash

คัดลอกแก้ไข

ffplay "srt://<ip-raspberry-pi>:9001"

หรือ VLC (ในบางเวอร์ชันที่รองรับ SRT):

bash

คัดลอกแก้ไข

srt://<ip-raspberry-pi>:9001

## 📈 หมายเหตุด้านประสิทธิภาพ

* Raspberry Pi 4 ขึ้นไปควรพอรองรับ client ได้ **หลายสิบคนใน LAN**
* ใช้ codec ที่เบา เช่น mp3 หรือ aac เพื่อไม่ให้ CPU ทำงานหนัก
* ถ้า client เยอะมาก (100+): แนะนำใช้ relay ต่ออีกชั้นบน server ภายนอก

## ✅ สรุปสุดท้าย:

| **ใช้ Raspberry Pi รัน SRT Relay (srt-live-transmit)** | **ได้ ✔️** |
| --- | --- |
| รองรับผู้ฟังหลายคนพร้อมกัน | ได้ ✔️ |
| ทำได้แบบ Low-latency (เสียงแบบเรียลไทม์) | ใช่ ✔️ |

SRT Relay Server

srt-live-transmit "srt://:9000?mode=listener" "srt://:9001?mode=listener"

=====================================

start\_stream() {

while true; do

echo "Starting $1..."

srt-live-transmit "$1" "$2"

echo "$1 stopped. Restarting in 5 seconds..."

sleep 5

done

}

start\_stream "srt://:9000?mode=listener&latency=300" "srt://:9001?mode=listener&latency=300" &

start\_stream "srt://:9002?mode=listener&latency=300" "srt://:9003?mode=listener&latency=300" &

start\_stream "srt://:9004?mode=listener&latency=300" "srt://:9005?mode=listener&latency=300" &

start\_stream "srt://:9006?mode=listener&latency=300" "srt://:9007?mode=listener&latency=300" &

start\_stream "srt://:9008?mode=listener&latency=300" "srt://:9009?mode=listener&latency=300" &

start\_stream "srt://:9010?mode=listener&latency=300" "srt://:9011?mode=listener&latency=300" &

wait

==============================

ffmpeg ecoder

1. for aac : ffmpeg -f alsa -i hw:1,0 -acodec aac -profile:a aac\_low -b:a 192k -ar 48000 -ac 2 -f mpegts "srt://159.138.230.150:9000?mode=caller"

2. for mp3 : ffmpeg -f alsa -i hw:1,0 -acodec libmp3lame -b:a 192k -f mpegts "srt://159.138.230.150:9000?mode=caller"

3. for mp3 and latency : ffmpeg -f alsa -i hw:1,0 -acodec libmp3lame -b:a 192k -f mpegts "srt://159.138.230.150:9000?mode=caller&latency=120"

4. for Opus : ffmpeg -f alsa -i hw:1,0 -acodec libopus -b:a 192k -f matroska "srt://159.138.230.150:9000?mode=caller"

5. scrip for loop encoding

while true; do

echo "Starting FFmpeg stream..."

ffmpeg -f alsa -i hw:1,0 -acodec aac -profile:a aac\_low -b:a 164k -ar 44000 -ac 2 -f mpegts "srt://159.138.230.150:9000?mode=caller"

echo "FFmpeg exited. Restarting in 5 seconds..."

slep 5

done

=================

\*\*\*\* List of CAPTURE Hardware Devices \*\*\*\*ffmpeg -f alsa -i hw:1,0 -acodec aac -profile:a aac\_low -b:a 192k -ar 48000 -ac 2 -f mpegts "srt://159.138.230.150:9000?mode=caller"

card 2: Device [USB Advanced Audio Device], device 0: USB Audio [USB Audio]

Subdevices: 1/1

Subdevice #0: subdevice #0

ffmpeg -f alsa -i hw:2,0 -acodec libmp3lame -ar 44100 -ac 2 -b:a 128k -f mpegts - | srt-live-transmit - "srt://159.138.230.150:9000?mode=caller&latency=120" &

[in#0 @ 0x55a95214d0] Unknown input format: 'alsa'

Error opening input file hw:2,0.

Error opening input files: Invalid argument

arecord -l

\*\*\*\* List of CAPTURE Hardware Devices \*\*\*\*

card 2: Device [USB Advanced Audio Device], device 0: USB Audio [USB Audio]

Subdevices: 1/1

Subdevice #0: subdevice #0

========================================

1.

sudo apt-get remove --purge ffmpeg

sudo apt-get autoremove

2.

sudo apt-get update

sudo apt-get install -y build-essential giffmpeg -f alsa -i hw:1,0 -acodec aac -profile:a aac\_low -b:a 192k -ar 48000 -ac 2 -f mpegts "srt://159.138.230.150:9000?mode=caller"

t autoconf automake libtool pkg-config \

libasound2-dev libpulse-dev libssl-dev cmake yasm nasm \

libx264-dev libx265-dev libvpx-dev libfdk-aac-dev libmp3lame-dev \

libopus-dev libass-dev libwebp-dev zlib1g-dev

3.

sudo apt-get install -y tclsh pkg-config cmake libssl-dev

git clone https://github.com/Haivision/srt.git

cd srt

./configure

make

sudo make install

sudo ldconfig

4.

git clone https://git.ffmpeg.org/ffmpeg.git ffmpeg

cd ffmpeg

5.

./configure \

--enable-gpl \

--enable-libmp3lame \

--enable-libopus \

--enable-libx264 \

--enable-libx265 \

--enable-libass \

--enable-libfdk-aac \

--enable-libpulse \

--enable-alsa \

--enable-libsrt \

--enable-nonfree \

--extra-ldflags="-L/usr/local/lib" \

--extra-cflags="-I/usr/local/include"

6.

make -j$(nproc)

sudo make install

sudo ldconfig

7.

ขั้นตอนที่ 7: ตรวจสอบการติดตั้ง

bash

Copy

ffmpeg -version

ตรวจสอบว่ามีการรองรับฟีเจอร์ที่ต้องการ:

bash

Copy

ffmpeg -formats | grep -E 'alsa|pulse|srt'

ffmpeg -codecs | grep -E 'mp3lame|alsa|pulse'

ขั้นตอนที่ 8: ทดสอบคำสั่ง

ลองใช้คำสั่งเดิมของคุณ:

bash

Copy

ffmpeg -f alsa -i hw:2,0 -acodec libmp3lame -ar 44100 -ac 2 -b:a 128k -f mpegts - | srt-live-transmit - "srt://159.138.230.150:9000?mode=caller&latency=120" &

หรือใช้ PulseAudio:

bash

Copy

ffmpeg -f pulse -i default -acodec libmp3lame -ar 44100 -ac 2 -b:a 128k -f mpegts - | srt-live-transmit - "srt://159.138.230.150:9000?mode=caller&latency=120" &