📻 DAB+ Labs สำหรับ Raspberry Pi

คู่มือการเรียนรู้ Digital Audio Broadcasting Plus

พร้อม RTL-SDR และ PyQt5

เวอร์ชัน 1.1 | กันยายน 2025

🎯 วัตถุประสงค์โครงการ

📚 เรียนรู้เทคโนโลยี

- **DAB+** จากพื้นฐานจนถึงขั้นสูง
- **Python & PyQt5** GUI programming
- **Software Defined Radio** (SDR)
- **RF Signal Processing**

🛠️ สร้างแอปพลิเคชัน

- DAB+ Station Scanner
- Program Recorder
- Signal Analyzer
- Touch-Friendly GUI

🎯 **เป้าหมาย**: สร้างแอปที่ใช้งานได้จริงบน Raspberry Pi

ข้อกำหนดระบบ

Hardware

- **Raspberry Pi 4** (4GB+ RAM)
- **RTL-SDR V4** Dongle
- **หน้าจอสัมผัส 7"** (HDMI)
- **หูฟัง 3.5mm**
- **เสาอากาศ DAB+**

Software

- **Raspberry Pi OS Bookworm**
- **Python 3.11+**
- **PyQt5** GUI Framework
- **welle.io** DAB+ Decoder
- **RTL-SDR** Libraries

📚 ภาพรวมแล็บทั้งหมด

Lab	หัวข้อ	เวลา	ระดับ
0	Introduction to DAB+, Python และ PyQt5	75 นาที	⭐
1	การติดตั้งและทดสอบ RTL-SDR	90 นาที	⭐⭐
2	การใช้งาน welle.io ผ่าน Python	120 นาที	⭐⭐⭐
3	การควบคุม RTL-SDR โดยตรง	120 นาที	⭐⭐⭐
4	สร้าง DAB+ Station Scanner	150 นาที	⭐⭐⭐⭐
5	สร้าง DAB+ Program Recorder	150 นาที	⭐⭐⭐⭐
6	สร้าง DAB+ Signal Analyzer	180 นาที	⭐⭐⭐⭐⭐

รวมเวลา: ~12.25 ชั่วโมง

🎓 LAB 0: Introduction to DAB+, Python และ PyQt5

🕒 เวลารวม: 75 นาที (1 ชั่วโมง 15 นาที)

📋 ภาพรวมเนื้อหา

เป็นแล็บพื้นฐานสำหรับมือใหม่ ที่ยังไม่เคยใช้ Python หรือไม่รู้จัก DAB+

📡 ส่วนที่ 1: DAB+ Technology (15 นาที)

🆚 DAB+ vs FM Radio

- **เสียงดิจิทัล** ไม่มี static หรือสัญญาณรบกวน
- **คุณภาพคงที่** ไม่ขึ้นกับระยะทาง
- **Metadata** ชื่อเพลง, ศิลปิน แบบ real-time
- **MOT Slideshow** รูปภาพ album art
- **Multiplexing** หลายสถานีใช้ความถี่เดียว

🇹🇭 DAB+ ในประเทศไทย (2025)

การทดลองปัจจุบัน:

- **Block 9A** (202.928 MHz) - กรุงเทพฯ
  - สถานีธรรมะเพื่อประชาชน
  - สถานีวิทยุสันติ
- **Block 6C** (185.360 MHz) - ขอนแก่น
  - สถานีวิทยุขอนแก่นมหานคร

🛠️ เทคโนโลยีที่ใช้

🐍 ส่วนที่ 2: Python สำหรับมือใหม่ (30 นาที)

📝 Python Basics

```
# Variables และ Data Types
name = "สวัสดี"          # String
age = 25                 # Integer
height = 175.5           # Float
is_student = True        # Boolean

# Lists และการใช้งาน
fruits = ["แอปเปิ้ล", "กล้วย", "ส้ม"]
fruits.append("มะม่วง")
print(len(fruits))       # แสดง: 4
```

🔄 Control Flow

```
# Loops (การวนซ้ำ)
for fruit in fruits:
    print("ผลไม้:", fruit)

# Conditions (เงื่อนไข)
if age >= 18:
    print("เป็นผู้ใหญ่แล้ว")
else:
    print("เป็นเด็ก")

# Functions (ฟังก์ชัน)
def say_hello(name):
    return "สวัสดี " + name
```

🐍 Python: Classes และ Hardware Integration

🏗️ Object-Oriented Programming

```
class DABStation:
    def __init__(self, name, frequency):
        self.name = name
        self.frequency = frequency
        self.is_playing = False

    def start_playing(self):
        self.is_playing = True
        print(f"เริ่มเล่น {self.name}")

    def stop_playing(self):
        self.is_playing = False
```

🔧 Raspberry Pi GPIO

```
try:
    import RPi.GPIO as GPIO
    import time

    # ตั้งค่า GPIO pin 18
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(18, GPIO.OUT)

    # กะพริบ LED
    GPIO.output(18, GPIO.HIGH)
    time.sleep(1)
    GPIO.output(18, GPIO.LOW)

except ImportError:
    print("ทำงานบนคอมพิวเตอร์ทั่วไป")
```

ส่วนที่ 3: PyQt5 Hands-on (30 นาที)

PyQt5 Components

```python
from PyQt5.QtWidgets import *
import sys

class MainWindow(QMainWindow):
    def __init__(self):
        super().__init__()
        self.setup_ui()

    def setup_ui(self):
        # สร้าง central widget
        central_widget = QWidget()
        self.setCentralWidget(central_widget)

        # สร้าง layout
        layout = QVBoxLayout(central_widget)
```

Touch-Friendly Design

```python
# ปุ่มขนาดใหญ่สำหรับสัมผัส
button = QPushButton("กดที่นี่")
button.setMinimumSize(120, 60)

# Font ขนาดใหญ่
font = QFont()
font.setPointSize(14)
button.setFont(font)

# CSS Styling
button.setStyleSheet("""
    QPushButton {
        border-radius: 8px;
        background: #3498db;
        color: white;
    }
```

PyQt5: Signals & Slots

Event Handling

```
class DABPlayerWidget(QWidget):
    def __init__(self):
        super().__init__()
        self.setup_ui()
        self.setup_connections()

    def setup_connections(self):
        # เชื่อม signals กับ slots
        self.play_button.clicked.connect(self.on_play)
        self.volume_slider.valueChanged.connect(self.on_volume_change)

    def on_play(self):
        print("เริ่มเล่นเพลง!")
```

QTimer และ Updates

```
from PyQt5.QtCore import QTimer

class SignalMonitor(QWidget):
    def __init__(self):
        super().__init__()

        # Timer สำหรับ real-time update
        self.timer = QTimer()
        self.timer.timeout.connect(self.update_signal)
        self.timer.start(1000)  # อัพเดททุก 1 วินาที

    def update_signal(self):
        # อัพเดทค่าสัญญาณ
        signal_strength = self.get_signal_strength()
        self.signal_bar.setValue(signal_strength)
```

🎯 LAB 0: Demo Applications

📱 Demo 1: Basic Widgets

- **QLabel** แสดงข้อความและรูปภาพ
- **QPushButton** ปุ่มกดขนาดใหญ่
- **QLineEdit** ช่องใส่ข้อความ
- **QTextEdit** พื้นที่ข้อความหลายบรรทัด
- **QSlider** แถบเลื่อนค่า
- **QProgressBar** แสดงความคืบหน้า

📱 Demo 2: Touch Interface

- **ขนาดปุ่ม** อย่างน้อย 60x40 pixels
- **Font Size** 12-16pt สำหรับหน้าจอ 7"
- **Visual Feedback** เปลี่ยนสีเมื่อกด
- **Layout Management** responsive design
- **Error Handling** การจัดการข้อผิดพลาด

✅ สิ่งที่นักเรียนต้องเติม:

1. **อ่านชื่อ** จาก QLineEdit และแสดงใน QLabel

🏆 LAB 0: ผลลัพธ์ที่คาดหวัง

🎯 ความรู้ที่ได้รับ

DAB+ Technology:

- เข้าใจความแตกต่างจาก FM
- รู้จักเทคโนโลยี RTL-SDR
- เข้าใจ DAB+ ในประเทศไทย

Python Programming:

- Variables, functions, classes
- File handling และ modules
- GPIO programming พื้นฐาน

🛠️ Skills ที่พร้อมใช้

PyQt5 GUI Development:

- Widget การใช้งานพื้นฐาน
- Signals & Slots system
- Touch-friendly UI design
- Real-time updates ด้วย QTimer

เตรียมพร้อม สำหรับ Labs ขั้นสูง!

🔌 LAB 1: การติดตั้งและทดสอบ RTL-SDR

🛠️ การติดตั้ง

```
# ติดตั้ง RTL-SDR
sudo apt install rtl-sdr librtlsdr-dev

# Blacklist DVB drivers
sudo nano /etc/modprobe.d/blacklist-rtl.conf

# udev rules
sudo nano /etc/udev/rules.d/20-rtlsdr.rules
```

✅ การทดสอบ

```
# ตรวจสอบอุปกรณ์
lsusb | grep RTL

# ทดสอบการทำงาน
rtl_test -t

# ทดสอบการอ่านข้อมูล
rtl_test -s 2048000
```

🎯 **ผลลัพธ์: GUI App ทดสอบ RTL-SDR พร้อม Real-time Status**

📻 LAB 2: การใช้งาน welle.io ผ่าน Python

🔧 การติดตั้ง welle.io

```
# ติดตั้ง dependencies
sudo apt install qt5-qmake qtbase5-dev
sudo apt install libfaad-dev libmpg123-dev

# คอมไพล์จาก source
git clone https://github.com/AlbrechtL/welle.io.git
cd welle.io && mkdir build && cd build
cmake .. && make -j4 && sudo make install
```

📡 ความถี่ DAB+ ในไทย (2025)

การทดลองปัจจุบัน:

- **Block 9A**: 202.928 MHz (กทม.)
  - สถานีธรรมะเพื่อประชาชน
- **Block 6C**: 185.360 MHz (ขอนแก่น)
  - สถานีวิทยุขอนแก่นมหานคร

🎯 ผลลัพธ์: DAB+ Receiver App ที่ใช้งานได้จริง

🔬 LAB 3: การควบคุม RTL-SDR โดยตรง

📊 Signal Processing

```
from rtlsdr import RtlSdr
import numpy as np

sdr = RtlSdr()
sdr.sample_rate = 2.4e6
sdr.center_freq = 100e6

# อ่าน IQ samples
samples = sdr.read_samples(1024*1024)

# คำนวณ FFT
fft_data = np.fft.fft(samples)
power = 20 * np.log10(np.abs(fft_data))
```

📈 Spectrum Analysis

- **FFT** แปลง Time → Frequency Domain
- **Power Spectrum** วิเคราะห์ความถี่
- **Real-time Plotting** ด้วย matplotlib
- **PyQt5 Integration** GUI + กราฟ

🎯 **ผลลัพธ์: RF Spectrum Analyzer แบบ Real-time**

🔍 LAB 4: สร้าง DAB+ Station Scanner

🗄️ Database Management

```
import sqlite3

class DatabaseManager:
    def init_database(self):
        # สร้างตาราง ensembles
        cursor.execute("""
            CREATE TABLE ensembles (
                frequency REAL,
                ensemble_label TEXT,
                scan_time TIMESTAMP
            )
        """)
```

🔍 Automatic Scanning

- **ความถี่ DAB+ Band III** (174-240 MHz)
- **SQLite Database** เก็บข้อมูลสถานี
- **Real-time Quality** monitoring
- **Advanced PyQt5** TreeWidget, TableWidget

🎯 **ผลลัพธ์: DAB+ Station Scanner พร้อมฐานข้อมูล**

LAB 5: สร้าง DAB+ Program Recorder

Scheduling System

```python
import schedule
from datetime import datetime

class RecordingScheduler:
    def add_schedule(self, station, start_time,
                     duration, repeat='once'):
        # เพิ่มตารางการบันทึก
        schedule_item = {
            'station': station,
            'start_time': start_time,
            'duration': duration
        }
```

File Organization

```
DAB_Recordings/
├── 2024-12-08/
│   ├── Thai_PBS/
│   │   ├── audio.wav
│   │   ├── slideshow/
│   │   └── metadata.json
│   └── NBT/
└── logs/
```

ผลลัพธ์: DAB+ Program Recorder พร้อม Scheduler

📊 LAB 6: สร้าง DAB+ Signal Analyzer

🔬 Advanced Analysis

- **OFDM Structure** วิเคราะห์
- **SNR, MER, BER** คุณภาพสัญญาณ
- **Constellation Diagram** I/Q แสดงผล
- **Waterfall Plot** spectrum ตามเวลา

📈 Visualization & Reports

- **Real-time Metrics** LCD displays
- **Professional Reports** PDF generation
- **Data Export** CSV, JSON formats
- **Advanced Matplotlib** integration

🎯 ผลลัพธ์: Professional DAB+ Signal Analyzer

📡 เทคโนโลยี DAB+ เบื้องลึก

🏗️ DAB+ Signal Structure

```
DAB+ Frame (96ms)
├── Null Symbol (sync)
├── PRS (Phase Reference)
├── FIC (Fast Info Channel)
└── MSC (Main Service Channel)
    ├── Audio Services
    └── Data Services
```

🔄 OFDM Technology

- **2048 Carriers** ใช้พร้อมกัน
- **Guard Interval** ป้องกัน multipath
- **DQPSK Modulation** ทนต่อ noise
- **Error Correction** Reed-Solomon

🎯 **ความเข้าใจ:** จากพื้นฐานไปถึงระดับ Professional RF Engineer

🛠️ การพัฒนาด้วย PyQt5

🖥️ Touch-Friendly GUI

```
# ปุ่มขนาดใหญ่
button.setMinimumSize(120, 60)

# Font สำหรับหน้าจอ 7"
font = QFont()
font.setPointSize(14)

# CSS Styling
button.setStyleSheet("""
    QPushButton {
        border-radius: 8px;
        background: #3498db;
        color: white;
        font-weight: bold;
    }
""")
```

🔗 Signals & Slots

```
# Built-in signals
button.clicked.connect(self.on_click)
slider.valueChanged.connect(self.update_value)

# Custom signals
class MyWidget(QThread):
    data_ready = pyqtSignal(dict)

    def emit_data(self):
        self.data_ready.emit({'value': 42})
```

📊 การประมวลผลสัญญาณ (DSP)

🔢 NumPy & SciPy

```
import numpy as np
from scipy import signal

# FFT Analysis
fft_result = np.fft.fft(iq_samples)
frequencies = np.fft.fftfreq(len(samples), 1/sample_rate)

# Power Spectrum
power_db = 20 * np.log10(np.abs(fft_result))

# Signal Quality
snr = signal_power / noise_power
```

📈 Real-time Visualization

```
import matplotlib.pyplot as plt
from matplotlib.backends.backend_qt5agg import FigureCanvasQTAgg

class SpectrumPlot(FigureCanvasQTAgg):
    def update_spectrum(self, freq, power):
        self.axes.clear()
        self.axes.plot(freq/1e6, power)
        self.draw()
```

🔧 การแก้ไขปัญหาทั่วไป

🚫 RTL-SDR Issues

```
# ตรวจสอบการเชื่อมต่อ
lsusb | grep RTL

# แก้ driver conflicts
sudo modprobe -r dvb_usb_rtl28xxu
lsmod | grep dvb

# Permissions
sudo usermod -a -G plugdev $USER
```

🔇 Audio Issues

```
# ตั้งค่าเสียงออก 3.5mm
sudo raspi-config
# Advanced Options > Audio > Force 3.5mm

# ทดสอบเสียง
speaker-test -t wav -c 2

# PulseAudio restart
pulseaudio -k
```

💡 **เคล็ดลับ:** อ่านคู่มือแต่ละ LAB ก่อนเริ่มโค้ด

📈 เส้นทางการเรียนรู้

🎯 ระดับเริ่มต้น

1. **Lab 0**: PyQt5 พื้นฐาน
2. **Lab 1**: RTL-SDR ติดตั้ง
3. **Lab 2**: DAB+ รับฟัง

⏱️ **เวลา**: ~4-5 ชั่วโมง
🎯 **เป้าหมาย**: สร้าง DAB+ radio ใช้ได้

🚀 ระดับสูง

4. **Lab 3**: Signal Processing
5. **Lab 4**: Database & Scanning
6. **Lab 5**: Recording & Scheduling
7. **Lab 6**: Professional Analysis

⏱️ **เวลา**: ~8-10 ชั่วโมง
🎯 **เป้าหมาย**: Professional RF Tools

🎉 ผลลัพธ์ที่ได้รับ

🛠️ แอปพลิเคชันที่สร้างได้

- DAB+ Radio Receiver
- Station Scanner
- Program Recorder
- RF Spectrum Analyzer
- Signal Quality Monitor

✨ พัฒนาจาก ผู้เริ่มต้น → RF Engineer

📚 ความรู้ที่ได้รับ

- **DAB+ Technology** เชิงลึก
- **Python & PyQt5** ขั้นสูง
- **RF & DSP** ระดับมืออาชีพ
- **Raspberry Pi** Embedded Systems

🌐 แหล่งข้อมูลเพิ่มเติม

📖 Documentation

- welle.io GitHub
- RTL-SDR.com
- PyQt5 Docs
- DAB+ Standard (ETSI)

🎓 Learning Resources

- **GNU Radio** สำหรับ SDR ขั้นสูง
- **DSP Course** Signal Processing
- **RF Engineering** คลื่นวิทยุ
- **Embedded Linux** สำหรับ IoT

🌟 Next Steps - ขั้นต่อไป

🔧 พัฒนาเพิ่มเติม

- **Web Interface** ควบคุมผ่าน browser
- **Mobile App** Android/iOS remote
- **Cloud Integration** upload recordings
- **AI/ML** automatic classification

🎯 Career Paths

- **RF Engineer** วิศวกรคลื่นวิทยุ
- **SDR Developer** Software Defined Radio
- **IoT Developer** Internet of Things
- **Embedded Systems** ระบบฝังตัว

🚀 **จาก Hobby Project → Professional Career**

🙏 ขอบคุณและสนับสนุน

🎉 ขอบคุณที่เรียนรู้กับเรา!

DAB+ Labs เป็นโครงการ Open Source สำหรับการศึกษาและพัฒนาเทคโนโลยี

🌟 **ขอให้สนุกกับการเรียนรู้!**

📞 ติดต่อและสนับสนุน

- 💬 **Issues**: GitHub Issues
- 📧 **Email**: project contact
- 📱 **Community**: Forum discussion
- ⭐ **Star**: ถ้าชอบโครงการ

MIT License - ใช้ได้อย่างอิสระ

📝 สรุป: การเดินทาง DAB+ Learning

🎯 สิ่งที่เราได้เรียนรู้:

- **DAB+ Technology** จาก 0 ถึง Hero
- **Python & PyQt5** สำหรับ Professional GUI
- **RTL-SDR & RF Engineering** ด้วยมือ
- **Project Development** จาก Idea ถึง Working App

🏆 Achievement Unlocked:

📻 DAB+ Expert | 🐍 Python GUI Master | 📡 RF Engineer | 🔧 Maker

🎋 พร้อมสำหรับการผจญภัยใหม่แล้ว! 🎋