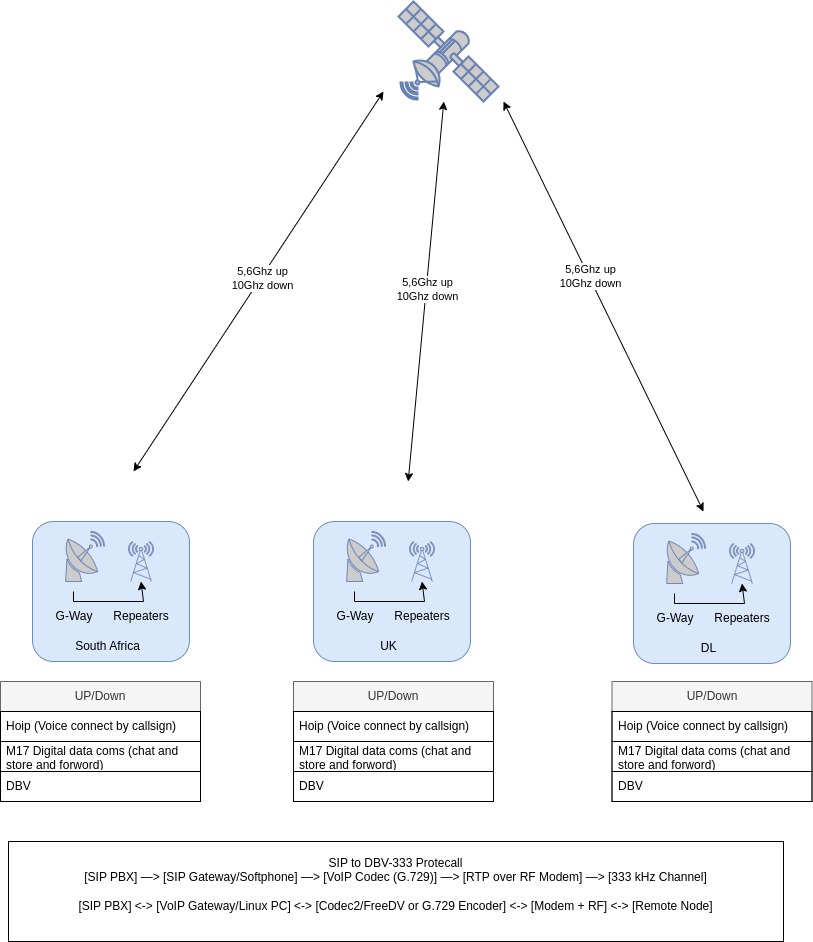
**New-Qo100-design criteria**

****

Proposed communication protocol stack

[SIP PBX] <-> [VoIP Gateway/Linux PC] <-> [Codec2/FreeDV or G.729 Encoder] <-> [Modem + RF] <-> [Remote Node]

# PROTOCOL STACK (Simplified)

| Layer | Function | Protocol/Tech |
| --- | --- | --- |
| Application | SIP call control + voice data | SIP, RTP, SIP proxy |
| Transport | Real-time transport of audio | UDP |
| Codec | Compresses audio to fit narrowband | **Codec2**, G.729, Opus |
| Framing | Frames data for modem | SLIP, KISS, custom framing |
| Modulation | Converts digital to analog RF | BPSK, QPSK, 16-QAM |
| RF Physical | Transmit over the 333 kHz channel | Custom/SDR/Analog RF |

## HARDWARE SETUP OPTIONS

### Option A: ****Using PC + SDR (Software-Defined Radio)****

#### 🖥️ At SIP PBX side:

* **Software**:
  + SIP server (Asterisk, FreeSWITCH)
  + Softphone (Linphone, Zoiper) or SIP client daemon
  + FreeDV or Codec2 encoder (freedv\_tx, c2enc)
* **RF Interface**:
  + SDR device (e.g., HackRF, LimeSDR, PlutoSDR)
  + GNU Radio or freedv modem integration

#### 📻 At RF side:

* Use **GNU Radio** flow graph to:
  + Modulate compressed audio (BPSK or QPSK)
  + Keep channel usage under 333 kHz
  + Handle framing, preamble, sync

#### Example SDR flow graph:

* Codec2 Input (1400 bps)
* BPSK Modulator
* Root Raised Cosine Filter
* Transmit via SDR (333 kHz bandwidth)

### 🛠️ Option B: ****Embedded Setup (Low-power field unit)****

* **Raspberry Pi or ESP32** for codec and packet logic
* **G.729 or Codec2 running via** linphonec **or** freedv
* **Modem**:
  + Custom FSK/PSK modem via Si4463 or CC1101 (with 300–500 kbps modes)
  + RF front-end tuned for your channel (e.g., 433 MHz/868 MHz/2.4 GHz)
* **Power**:
  + LiFePO4 battery + solar panel
* **Antenna**:
  + Helical or yagi for link reliability

## 📞 4. SOFTWARE COMPONENTS (SIP + VOIP + RF)

### 🖥️ On Linux PC (SIP Gateway):

# Install SIP client and audio tool

sudo apt install linphone-cli sox freedv gstreamer1.0-tools

# SIP call to destination (auto-answer mode)

linphonec -c /path/to/config -a -s sip:1002@pbx.server

# Pipe audio to Codec2 encoder

linphonec > pipe | sox -t raw -r 8k -e signed -b 16 -c 1 - -t raw -r 8k -e signed -b 16 -c 1 - | c2enc 1400 - - | modem\_tx

Replace modem\_tx with your SDR or RF module interface script.

### 📡 Modem (e.g., GNU Radio or Embedded)

* Sample baud rate: 20–60 kbps
* Use **differential BPSK or QPSK**
* Frame structure:
  + [Preamble][Sync Word][Header][Compressed Voice][CRC]

## 📶 5. EXAMPLE: Using ****Codec2 + FreeDV over RF****

You can use **FreeDV** mode 700D or 1600 (700–1600 bps), perfect for 333 kHz:

freedv\_tx 700D mic\_input.raw > /dev/ttyUSB0

Or with freedv-gui (audio + waterfall + modem all-in-one).

# 📜 6. Summary Table

| Component | Tech/Tool |
| --- | --- |
| SIP PBX | Asterisk or FreeSWITCH |
| VoIP Codec | Codec2 (700D, 1300, 1600) or G.729 |
| Audio Interface | ALSA, SoX, GStreamer |
| Modulation | BPSK/QPSK/16-QAM |
| Radio Interface | SDR (HackRF, LimeSDR) or CC1101 |
| Bandwidth Used | 2.4–32 kbps payload + framing, well within 333 kHz |
| Full Duplex Mode | TDD switching or dual RF links |

Shell script exsample

#!/bin/bash

# Description: End-to-end SIP VoIP over RF using Codec2 and FreeDV modem

# Requirements: linphonec, sox, freedv, asterisk (optional), rtl\_sdr/hackrf, gnuradio, codec2

# ---- CONFIG ----

SIP\_USER="1001"

SIP\_PASS="your\_password"

SIP\_DOMAIN="your.sipserver.local"

REMOTE\_USER="1002"

CODEC\_MODE="700D" # Options: 700D, 1600, 1300

# Audio paths

AUDIO\_INPUT="/tmp/mic\_in.raw"

CODEC\_OUT="/tmp/codec2\_out.bin"

RF\_STREAM="/tmp/rf\_stream.bin"

# ---- START SIP CALL ----

echo "Starting SIP call with $REMOTE\_USER@$SIP\_DOMAIN"

linphonecsh init

linphonecsh generic 'register --host $SIP\_DOMAIN --username $SIP\_USER --password $SIP\_PASS'

linphonecsh generic "call sip:$REMOTE\_USER@$SIP\_DOMAIN"

# ---- CAPTURE AUDIO FROM CALL ----

# Assumes audio is routed via PulseAudio or ALSA virtual device

arecord -f S16\_LE -r 8000 -c 1 -t raw > "$AUDIO\_INPUT" &

REC\_PID=$!

# ---- ENCODE AUDIO WITH CODEC2 ----

echo "Encoding with Codec2 - Mode $CODEC\_MODE"

c2enc "$CODEC\_MODE" "$AUDIO\_INPUT" "$CODEC\_OUT" &

ENC\_PID=$!

# ---- MODULATE AND TRANSMIT OVER RF ----

# Replace below with your SDR or GNU Radio transmitter command

echo "Transmitting via GNU Radio/SDR..."

cat "$CODEC\_OUT" | ./modem\_tx\_gnuradio.py &

TX\_PID=$!

# Wait until call ends or interrupted

trap "kill $REC\_PID $ENC\_PID $TX\_PID; linphonecsh generic 'terminate'; linphonecsh exit" INT

wait

he script and GNU Radio flowgraphs are now separated into clear code blocks:

1. 🖥️ **Shell script** for SIP call and Codec2 audio piping.
2. 📡 **GNU Radio TX flowgraph** for BPSK modulation.
3. 📻 **GNU Radio RX flowgraph** for BPSK demodulation.