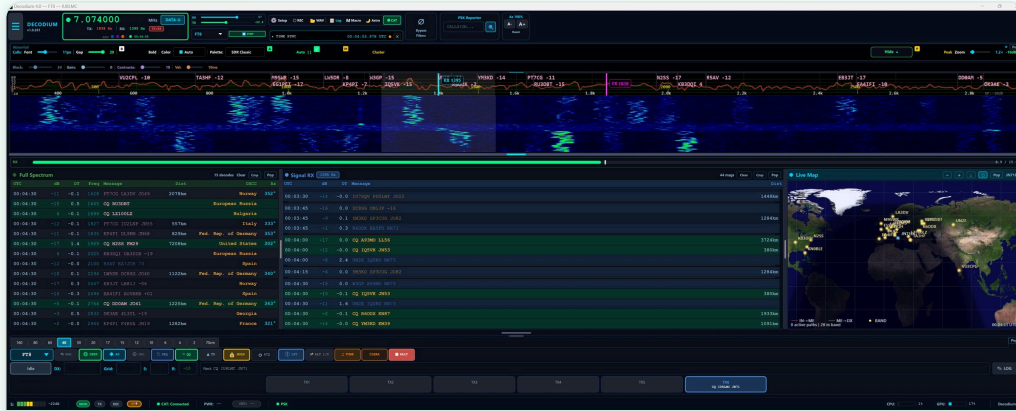


• BETA PÚBLICA • V1.0.262 • REFERENCE MANUAL COMPLETO

DECODIUM 4.0

“SHANNON” – REFERENCE MANUAL COMPLETO



▶ Apéndices A-J · 10 apéndices · Protocolo · API · CLI · Build · Troubleshooting

10

APÉNDICES
A → J



PROTOCOLO FT2
Spec completa



UDP API
Integraciones

30+

TROUBLESHOOT
Escenarios probados

Apéndice A – Especificación completa del protocolo FT2

Este apéndice documenta el protocolo Fast Transmission 2 (FT2) al nivel necesario para implementarlo desde cero en un decodificador o transmisor de terceros. La especificación es la certificada ADIF 3.1.7 con voto unánime 22:0 del 22 marzo 2026.

A.1 Capa física

A.1.1 Modulación

PARÁMETRO	VALOR	NOTAS
Tipo	4-GFSK	Gaussian Frequency Shift Keying de 4 tonos
Número de tonos	4	T0, T1, T2, T3 (mapeados a 2 bits/símbolo)
Espaciado entre tonos	41.6667 Hz	exacto: $41 + 2/3$ Hz
Ancho de banda RF total	167 Hz	3 × espaciado + roll-off Gauss
Filtro Gauss BT	1.0	Bandwidth × Time product
Tono nominal central	configurable	típicamente 1500 Hz audio
Tono T0	centro - 1.5 × espaciado	= centro - 62.5 Hz
Tono T1	centro - 0.5 × espaciado	= centro - 20.83 Hz
Tono T2	centro + 0.5 × espaciado	= centro + 20.83 Hz
Tono T3	centro + 1.5 × espaciado	= centro + 62.5 Hz

A.1.2 Timing

PARÁMETRO	VALOR
Symbol rate	41.6667 baud
Duración del símbolo	24 ms exactos ($1/41.6667$ s)
Frame total	79 símbolos
Duración del frame	79×24 ms = 1896 ms
Guard time (RX → TX)	~250 ms
Guard time (TX → RX)	~150 ms
Ciclo T/R total	3.75 – 3.80 s

A.1.3 Sample rate y DSP

El decoder DECODIUM trabaja internamente a **12 kHz** (downsample del 48 kHz del audio entrante). Es una herencia WSJT-X mantenida para compatibilidad.

- **FFT size:** 2048-point Hann window
- **Resolución de frecuencia:** $12000/2048 \approx 5.86$ Hz
- **Resolución temporal por FFT bin:** $2048/12000 \approx 170$ ms (overlap 50%)

ÍNDICE 8	TONO 4-GFSK	BIT PAIR
0	T0	00
1	T1	01
2	T3	11
3	T2	10
4	T0	00
5	T1	01
6	T3	11
7	T2	10

El Gray-coding garantiza que símbolos adyacentes difieren por **un solo bit**, reduciendo el error bit-equivalente cuando el decoder comete un error de selección de tono.

A.3 Payload y FEC

A.3.1 Longitud del payload

NIVEL	BITS	TAMAÑO
Mensaje usuario (comprimido)	77	9.625 bytes
+ CRC	14	91 bits totales
LDPC encoded	174	21.75 bytes
Símbolos (174/2 bit por símbolo)	87	dividido por 4-GFSK
Efectivos tras puncturing	65	distribuidos en 2 bloques de 29 + 7 sync

A.3.2 LDPC (174, 91)

DECODIUM usa el código LDPC original de K1JT (WSJT-X), implementado como matriz de paridad esparsa.

Especificaciones: - **N (longitud codeword):** 174 bits - **K (longitud mensaje + CRC):** 91 bits - **Rate:** $91/174 \approx 0.523$ -

Capacidad teórica de corrección: hasta 14 bits erróneos (BER 0.08) - **Check nodes:** 83 - **Algoritmo decode:**

Normalized Min-Sum - **Max iterations:** 20 - **Umbral de convergencia:** parity check sum = 0

A.3.3 CRC-14

Polinomio generador: $x^{14} + x^{13} + x^5 + x^3 + x^1 + 1$ (hexadecimal `0x6757`).

```
uint16_t crc14(uint8_t *msg, int nbits) {
    uint16_t crc = 0;
    for (int i = 0; i < nbits; i++) {
        uint8_t bit = (msg[i / 8] >> (7 - i % 8)) & 1;
        crc = (crc << 1) | bit;
        if (crc & 0x4000) crc ^= 0x6757;
    }
    return crc & 0x3FFF;
}
```

El CRC se calcula sobre los 77 bits de payload **antes** del encoding LDPC. Tras el decode, si el CRC recalculado no coincide, el decode se considera inválido y se descarta.

A.4 Formatos de payload (77 bits)

Los 77 bits de payload útil se usan en cinco formatos distintos, identificados por los primeros 3 bits (i = type indicator):

A.4.1 i = 0 – QSO estándar

Formato clásico WSJT-X:

```
[3 bits: i=0] [28 bits: callsign1] [28 bits: callsign2] [15 bits: grid4 / report] [3 bits: subtype]
```

- **callsign1**: hash de indicativo (codificación WSJT-X estándar, soporta indicativos hasta 13 chars)
- **callsign2**: ídem
- **grid4 / report**: locator 4 chars o report SNR $\pm 0.. \pm 30$ dB
- **subtype**: 0=CQ, 1=Reply, 2=Report, 3=RR73, 4=73, etc.

A.4.2 i = 1 – Free text

```
[3 bits: i=1] [74 bits: free text hasta 13 caracteres]
```

Set de caracteres soportado: A-Z, 0-9, espacio, /, ., ? (39 símbolos).

$74 \text{ bits} \div \lceil \log_2(39) \rceil = 74 \div 6 \approx 12.33 \rightarrow$ **13 caracteres max.**

A.4.3 i = 2 – Telemetry

```
[3 bits: i=2] [74 bits: hex string 18 chars + padding]
```

Usado para secuencias numéricas, sensor data, telemetría de estación.

A.4.4 i = 3 – DXpedition (Fox/Hound)

Formato dedicado a operaciones multi-slot DX:

```
[3 bits: i=3] [28 bits: fox callsign] [28 bits: hound callsign] [15 bits: slot + report] [3 bits: ack flag]
```

Permite al "Fox" (DX) gestionar hasta 5 hounds simultáneamente en frecuencias adyacentes.

A.4.5 i = 4 – Quick QSO (QQ)

Exclusivo FT2. Formato para el flujo optimizado de 4 mensajes:

```
[3 bits: i=4] [28 bits: callsign1] [28 bits: callsign2] [15 bits: grid + R+SNR combined] [3 bits: stage]
```

- **stage:** 0=CQ, 1=Reply+R+SNR(combined), 2=RR73, 3=TU

A.5 ASYMX – Extensión asíncrona

ASYMX **no modifica** la capa física FT2. Es una extensión del timing TX en el lado transmisor.

A.5.1 Transmisor estándar FT2

```
T = N × 7.5 s   para slot par (N = 0, 2, 4, ...)  
T = N × 7.5 + 3.75 s   para slot impar
```

El transmisor siempre espera el inicio del próximo slot alineado a NTP.

A.5.2 Transmisor ASYMX

```
T = momento de click del usuario o condición auto-fire
```

El transmisor puede empezar **en cualquier momento**. La señal sigue siendo idéntica (modulación, frame, FEC).

A.5.3 Decodificación ASYMX

El receptor estándar WSJT-X/JTDX recibe normalmente porque:

- La **correlación Costas** es time-invariante: busca el patrón en cualquier ventana temporal
- La **validación CRC** verifica la decodificación independientemente del timing
- El **decoder multi-pass** puede intentar sync en ventanas temporales superpuestas

Caveat: en ASYMX, el decoder puede ver el mismo frame **dos veces** si la ventana de análisis se superpone. DECODIUM usa **best-of-N consolidation** para evitar duplicados en el log.

Apéndice B – Reference completo `decodium.ini`

Este apéndice documenta **cada clave** del archivo de configuración, con tipo, valores válidos, default, e impacto.

B.1 Rutas del archivo de configuración

OS	RUTA CANÓNICA
Windows	%APPDATA%\Decodium\decodium.ini
macOS	~/Library/Application Support/Decodium/decodium.ini
Linux	~/.config/Decodium/decodium.ini

Archivo de backup: `decodium.ini.bak` se crea en cada arranque en el mismo directorio.

Formato: INI estándar Qt (case-sensitive en las claves, secciones entre `[...]`, escape `\=` para `=` literal).

B.2 Sección [Station]

Configuración del operador y de la estación.

CLAVE	TIPO	DEFAULT	VALORES	DESCRIPCIÓN
<code>MyCall</code>	string	<code>""</code>	indicativo válido	Indicativo del operador
<code>MyGrid</code>	string	<code>""</code>	4 o 6 char Maidenhead	Locator
<code>Operator</code>	string	<code>""</code>	indicativo o vacío	Operador (si diferente de MyCall)
<code>DXCC</code>	string	<code>auto</code>	auto / número DXCC	Override DXCC entity
<code>StationType</code>	string	<code>Home</code>	Home/Portable/Maritime/Aeronautical	Tipo de estación
<code>Antenna</code>	string	<code>""</code>	texto libre	Descripción de antena (solo info)
<code>Power</code>	int	<code>100</code>	1-1500	Potencia nominal TX en W

B.3 Sección [Radio]

Configuración CAT.

CLAVE	TIPO	DEFAULT	VALORES	DESCRIPCIÓN
<code>RigType</code>	string	<code>None</code>	Hamlib model id	Modelo de radio (autocomplete)
<code>RigPort</code>	string	<code>""</code>	COM3 / /dev/ttyUSB0	Puerto serie
<code>RigBaud</code>	int	<code>9600</code>	1200-115200	Baud rate
<code>RigDataBits</code>	int	<code>8</code>	7 o 8	Bits de datos
<code>RigStopBits</code>	int	<code>1</code>	1 o 2	Bits de stop
<code>RigParity</code>	string	<code>None</code>	None/Even/Odd	Paridad
<code>RigHandshake</code>	string	<code>None</code>	None/Hardware/Software	Control de flujo
<code>RigDTR</code>	string	<code>Empty</code>	Empty/ON/OFF	Estado DTR forzado
<code>RigRTS</code>	string	<code>Empty</code>	Empty/ON/OFF	Estado RTS forzado
<code>CIVAddress</code>	hex	<code>0x94</code>	0x00-0xFF	Dirección CI-V (solo Icom)
<code>PollInterval</code>	int	<code>1000</code>	100-5000	Frecuencia de polling ms
<code>HRDBridge</code>	bool	<code>false</code>	true/false	Habilitar HRD bridge
<code>HRDHost</code>	string	<code>127.0.0.1</code>	IP	Host servidor HRD
<code>HRDPort</code>	int	<code>7809</code>	1-65535	Puerto TCP HRD
<code>OmniRig</code>	bool	<code>false</code>	true/false	Usar OmniRig en lugar de Hamlib
<code>OmniRigInstance</code>	int	<code>1</code>	1 o 2	OmniRig rig number

B.4 Sección `[Audio]`

Configuración audio I/O.

CLAVE	TIPO	DEFAULT	VALORES	DESCRIPCIÓN
<code>AudioIn</code>	string	<code>""</code>	nombre del dispositivo	Dispositivo de entrada (case-sensitive)
<code>AudioOut</code>	string	<code>""</code>	nombre del dispositivo	Dispositivo de salida
<code>SampleRate</code>	int	<code>48000</code>	12000/24000/48000/96000	Sample rate Hz
<code>BufferSize</code>	int	<code>1024</code>	256-4096	Buffer DMA frames
<code>Channels</code>	int	<code>1</code>	1 o 2	Mono / Stereo
<code>Bandwidth</code>	float	<code>3000</code>	1000-3000	Bandwidth audio Hz
<code>BoostRX</code>	int	<code>0</code>	0-20	Gain digital RX (dB)
<code>BoostTX</code>	int	<code>0</code>	0-20	Gain digital TX (dB)

Importante: `AudioIn` y `AudioOut` deben coincidir **exactamente** con los nombres de dispositivo del sistema (Windows Panel Audio, ALSA aplay -L, macOS System Preferences). Espacios y mayúsculas cuentan.

B.5 Sección `[PTT]`

CLAVE	TIPO	DEFAULT	VALORES	DESCRIPCIÓN
<code>Method</code>	string	<code>CAT</code>	CAT/VOX/RTS/DTR/External	Modalidad PTT
<code>Port</code>	string	<code>""</code>	COM3 / /dev/ttyUSB0	Puerto (si distinto del CAT)
<code>Inverted</code>	bool	<code>false</code>	true/false	Invertir polaridad de línea
<code>DeLayMs</code>	int	<code>50</code>	0-1000	Delay tras PTT antes del audio TX

B.6 Sección `[FT2]`

Parámetros específicos FT2.

CLAVE	TIPO	DEFAULT	VALORES	DESCRIPCIÓN
<code>EnableDEEP</code>	bool	<code>true</code>	true/false	Multi-pass Raptor
<code>DEEPPasses</code>	int	<code>5</code>	1-10	Número de pasos best-of
<code>EnableASYMX</code>	bool	<code>false</code>	true/false	Modalidad asíncrona
<code>EnableQQ</code>	bool	<code>true</code>	true/false	Quick QSO 4-msg
<code>EnableMMSE</code>	bool	<code>true</code>	true/false	MMSE channel estimation
<code>EnableEMA</code>	bool	<code>true</code>	true/false	Multi-period averaging
<code>EMAAlpha</code>	float	<code>0.35</code>	0.0-1.0	EMA weight
<code>EMAMaxPeriods</code>	int	<code>4</code>	1-10	Max periods to accumulate
<code>LDPCMaxIterations</code>	int	<code>20</code>	5-50	Max LDPC decoder iter
<code>SyncThreshold</code>	float	<code>0.45</code>	0.0-1.0	Umbral correlation Costas

B.7 Sección `[Filters]`

CLAVE	TIPO	DEFAULT	VALORES	DESCRIPCIÓN
<code>FDR</code>	bool	<code>true</code>	true/false	Frequency Domain Resilience
<code>FDRThreshold</code>	float	<code>0.3</code>	0.0-1.0	Agresividad FDR
<code>SpectralMask</code>	bool	<code>true</code>	true/false	Filtro máscara espectral
<code>SpectralMaskMargin</code>	int	<code>20</code>	5-50	Margen en Hz
<code>SlidingAGC</code>	bool	<code>false</code>	true/false	AGC interno buffer audio
<code>SlidingAGCWindow</code>	int	<code>200</code>	50-1000	Ventana AGC en ms
<code>DupSuppress</code>	bool	<code>true</code>	true/false	Supresión de duplicados multi-pass
<code>MinSNR</code>	int	<code>-25</code>	-30..-10	Umbral SNR mínimo (abajo = descarta)

B.8 Sección [UI]

CLAVE	TIPO	DEFAULT	VALORES	DESCRIPCIÓN
<code>Theme</code>	string	<code>ShannonDark</code>	ShannonDark/ShannonLight/Midnight/Classic	Tema activo
<code>WaterfallPalette</code>	string	<code>SDRClassic</code>	SDRClassic/ShannonLight/ShannonDark/Heat	Paleta de colores waterfall
<code>WaterfallSpeed</code>	int	<code>10</code>	1-100	Velocidad refresh ms
<code>WaterfallGain</code>	int	<code>0</code>	-30..+30	Gain visual dB
<code>WaterfallContrast</code>	int	<code>10</code>	0-30	Contraste
<code>WaterfallBlack</code>	int	<code>24</code>	0-50	Punto negro
<code>FontScale</code>	int	<code>100</code>	80-150	Escala fuente global %
<code>Language</code>	string	<code>en</code>	en/it/es/de/tr	Idioma interfaz
<code>CompactMode</code>	bool	<code>false</code>	true/false	Modo compacto
<code>ShowGridLines</code>	bool	<code>true</code>	true/false	Cuadrícula de frecuencia waterfall
<code>BoldDecodes</code>	bool	<code>true</code>	true/false	Decodes en negrita
<code>ColorDecodes</code>	bool	<code>true</code>	true/false	Colorear por banda/distancia

B.9 Sección [Logbook]

CLAVE	TIPO	DEFAULT	VALORES	DESCRIPCIÓN
ADIFPath	string	auto	path o auto	Ruta del archivo ADIF
AutoLog	bool	true	true/false	Log automático al fin del QSO
LoTW	bool	false	true/false	Upload LoTW habilitado
LoTWUser	string	""	indicativo	Username LoTW
LoTWPath	string	""	ruta TQSL	Ruta del ejecutable TQSL
eQSL	bool	false	true/false	Upload eQSL
eQSLUser	string	""	username	Username eQSL
ClubLog	bool	false	true/false	Upload Club Log
QRZ	bool	false	true/false	Upload QRZ.com
QRZAPIKey	string	""	API key	Clave API QRZ
Cloudlog	bool	false	true/false	Upload Cloudlog/Wavelog
CloudlogURL	string	""	URL endpoint	URL Cloudlog
CloudlogAPIKey	string	""	API key	Clave API Cloudlog

B.10 Sección [PSKReporter]

CLAVE	TIPO	DEFAULT	DESCRIPCIÓN
Enable	bool	true	Upload habilitado
Interval	int	300	Intervalo upload en segundos
IncludeFT2	bool	true	Incluir decodes FT2 en el upload
Server	string	report.pskreporter.info	Servidor PSK Reporter

B.11 Sección [DXCluster]

CLAVE	TIPO	DEFAULT	DESCRIPCIÓN
Enable	bool	false	Conexión cluster habilitada
Server	string	""	Host:port del cluster
Login	string	""	Username del cluster
Password	string	""	Password (raramente usada)
AutoConnect	bool	true	Conexión automática al arranque
FilterBand	bool	true	Filtrar por banda activa
FilterMode	string	FT8,FT2,FT4	Filtrar por modos
MaxAge	int	300	Edad máxima del spot en segundos

B.12 Sección [Advanced]

CLAVE	TIPO	DEFAULT	DESCRIPCIÓN
DebugCAT	bool	false	Log detallado CAT
DebugAudio	bool	false	Log niveles audio por slot
DebugDecoder	bool	false	Log detalles decoder
DebugQML	bool	false	Log warnings QML
MaxDecodersPerSlot	int	12	Max decoders paralelos
ThreadPoolSize	int	auto	Conteo worker thread pool
NetworkTimeout	int	10000	Timeout de conexiones de red ms
EnableTelemetry	bool	false	Telemetría anónima al equipo

Apéndice C – CAT command reference por radio

DECODIUM 4.0 usa Hamlib 4.7 para el control de la radio. Hamlib soporta **170+ modelos**, pero las radios más difundidas en el ámbito digital tienen especificaciones operativas que vale la pena documentar.

C.1 Kenwood TS-590S / TS-590SG

C.1.1 Setup recomendado

PARÁMETRO	VALOR
Hamlib model	2031 (TS-590S) o 2034 (TS-590SG)
Baud rate	57600 (recomendado max)
Data bits	8
Stop bits	1
Parity	None
Handshake	None
CAT mode (menú radio)	Standard Kenwood

C.1.2 Menús radio a configurar

MENU	FUNCIÓN	VALOR RECOMENDADO
Menu 63	DATA IN source	USB
Menu 64	USB IN level	5 (regula hasta nivel verde)
Menu 65	USB OUT level	5 (regula para PWR deseado)
Menu 76	Auto power off	OFF
Menu 77	Beep on TX	OFF (ruido innecesario en audio TX)

C.1.3 Comandos CAT clave

```
IF;           → query estado completo
FA<freq>;    → set VFO A frequency (ej. FA00007074000;)
MD2;         → set mode USB
TX0;         → PTT off
TX1;         → PTT on (RX→TX)
DA1;         → DATA mode on
PC<pwr>;     → set power level (PC100; = 100W max)
```

C.2 Yaesu FT-991A

C.2.1 Setup recomendado

PARÁMETRO	VALOR
Hamlib model	1035
Baud rate	38400
Data bits	8
Stop bits	2 (requerido por FT-991A)
Parity	None
Handshake	None

C.2.2 Menús radio esenciales

MENU	FUNCIÓN	VALOR
31(CAT RATE)	CAT baud rate	38400
113 (RPORTS)	RTTY/PSK puertos	USB Front (para audio USB)
062 (SCKMD)	Scope mode	OFF en TX

C.2.3 Particularidades

- El FT-991A requiere **2 stop bits** (no 1 como la mayoría de radios Yaesu)
- El modo DATA-U / DATA-USB es preferible al USB puro: mejor enmascaramiento audio interno
- El filtro audio TX por defecto es 3000 Hz – adecuado

C.3 Icom IC-7300 / IC-7610

C.3.1 Setup recomendado

PARÁMETRO	VALOR IC-7300	VALOR IC-7610
Hamlib model	3073	3081
Baud rate	19200	115200
CI-V Address	0x94	0xA4
Stop bits	1	1

C.3.2 Menús radio esenciales

MENÚ IC-7300	FUNCIÓN	VALOR
SET → Connectors → MOD INPUT → DATA OFF MOD	DATA mode mod source	USB
SET → Connectors → MOD INPUT → DATA OFF MOD LEVEL	TX audio level	30-50%
SET → Connectors → CI-V → CI-V USB Baud	CAT baud	115200 (para max throughput)
SET → CI-V → CI-V USB Echo Back	Echo	OFF

C.3.3 Particularidades Icom

- Comunicación **CI-V** (Communication Interface V) – protocolo Icom propietario
- Dirección `0x94` estándar IC-7300, `0xA4` IC-7610, `0x88` IC-9700
- USB CODEC integrado (no Signalink necesario)
- **Echo back ON** puede causar bucles CAT → ponerlo en OFF

C.4 Elecraft K3 / K3S / K4

C.4.1 Setup recomendado

PARÁMETRO	VALOR K3/K3S	VALOR K4
Hamlib model	2029	2029 (K4 compatible K3)
Baud rate	38400	38400
Stop bits	1	1

C.4.2 Menús radio

MENU	FUNCIÓN	VALOR
CONFIG:RS232	RS232 baud	38400
CONFIG:DATA MD	Default DATA mode	DATA-A o PSK D

C.4.3 Particularidades Elecraft

- Comandos compactos, alta velocidad (38400 nativo)
- Modo DATA-A (audio sobre data) ideal para FT8/FT2
- Power-down vía CAT soportado: `PS0;`

C.5 FlexRadio (SmartSDR/TCI)

C.5.1 Conexión TCI 1.5

DECODIUM soporta TCI 1.5 (ESDR) como bridge software para FlexRadio:

PARÁMETRO	VALOR
TCI server	<code>localhost:50001</code> (default)
Audio device	Flex Audio (CODEC virtual)
Hamlib needed	NO – TCI gestiona todo

C.5.2 Setup

1. En SmartSDR: habilita el TCI server (Settings → CAT/TCI → Enable TCI)
2. En DECODIUM: Setup → Radio → TCI Bridge → `localhost:50001`
3. Dispositivo audio automático (Flex Audio Source/Sink)

C.6 Tabla Hamlib Model ID (most common)

RADIO	MODEL ID
Yaesu FT-450D	1027
Yaesu FT-857	1009
Yaesu FT-891	1042
Yaesu FT-950	1018
Yaesu FT-991A	1035
Yaesu FTDX10	1041
Yaesu FTDX101D	1040
Yaesu FTDX3000	1037
Kenwood TS-480	2025
Kenwood TS-570	2017
Kenwood TS-590S	2031
Kenwood TS-590SG	2034
Kenwood TS-890S	2036
Kenwood TS-2000	2014
Icom IC-705	3085
Icom IC-718	3030
Icom IC-746	3032
Icom IC-7300	3073
Icom IC-7600	3061
Icom IC-7610	3081
Icom IC-7700	3062
Icom IC-9700	3079
Elecraft K3 / K3S	2029
Elecraft K4	2029 (compat)
Elecraft KX2	2044
Elecraft KX3	2042

RADIO	MODEL ID
Xiegu X6100	4012
Xiegu G90	4011

Lista completa vía `rigctl --list` o en el código fuente Hamlib en `rigs/*.h`.

Apéndice D – File system layout

Este apéndice documenta dónde DECODIUM 4.0 guarda cada tipo de dato.

D.1 Windows

```

C:\Users\<user>\AppData\Local\Decodium\
├─ decodium.exe           ← ejecutable principal
├─ Qt6*.dll              ← runtime Qt
├─ hamlib.dll            ← biblioteca CAT
├─ platforms\, plugins\ ← Qt plugins
├─ data\                 ← recursos estáticos
└─ locale\               ← traducciones .qm

C:\Users\<user>\AppData\Roaming\Decodium\
├─ decodium.ini          ← configuración principal
├─ decodium.ini.bak     ← backup en cada arranque
├─ log.adi              ← logbook ADIF
├─ log.adi.bak          ← backup del log
├─ callsign_history.db  ← caché de indicativos (SQLite)
├─ logs\                ← archivos de log
│ └─ decodium.log
│ └─ cat.log
│ └─ decoder.log
├─ cache\                ← caché temporal
│ └─ qmlcache\
│ └─ livemap_tiles\     ← tiles OpenStreetMap cacheados
│ └─ psk_reporter\
├─ macros\              ← macros custom usuario
└─ *.macro

```

0.2 macOS

```
/Applications/DECODIUM.app/  
├─ Contents/  
│  └─ Info.plist  
│  └─ MacOS/decodium      ← ejecutable principal  
│  └─ Resources/         ← iconos, traducciones  
│  └─ Frameworks/       ← Qt frameworks  
  
~/Library/Application Support/Decodium/  
├─ decodium.ini  
├─ decodium.ini.bak  
├─ log.adi  
├─ logs/  
├─ cache/  
└─ macros/  
  
~/Library/Caches/Decodium/ ← caché renovable (puede vaciarse)
```

0.3 Linux

```
~/local/bin/decodium-1.0.262.AppImage ← (o donde lo pongas)  
  
~/config/Decodium/  
├─ decodium.ini  
├─ decodium.ini.bak  
└─ macros/  
  
~/local/share/Decodium/  
├─ log.adi  
├─ log.adi.bak  
├─ callsign_history.db  
└─ logs/  
  
~/cache/Decodium/  
├─ qmlcache/  
├─ livemap_tiles/  
└─ psk_reporter/
```

0.4 Convención de nomenclatura para archivos ADIF

DECODIUM crea automáticamente backups diarios:

ARCHIVO	CUÁNDO
<code>log.adi</code>	log actual, siempre actualizado
<code>log.adi.bak</code>	backup arranque previo
<code>log_YYYY-MM-DD.adi</code>	snapshot diario (solo si activado en Setup)
<code>log_export_<timestamp>.adi</code>	export manual desde menú

D.5 Tamaños y gestión

PATH	TAMAÑO TÍPICO	NOTAS
<code>cache/qmlcache/</code>	50-200 MB	Regenerable, vaciable
<code>cache/livemap_tiles/</code>	100 MB - 2 GB	Crecimiento con uso, puede ser limitado
<code>logs/decoder.log</code> (debug ON)	1-2 GB/día	DESACTIVAR cuando no se necesita
<code>log.adi</code>	1-10 MB	Crecimiento lineal con QSO
<code>callsign_history.db</code>	5-50 MB	Crece con estaciones vistas

Limpieza de caché: menú hamburger → **Maintenance** → **Clear cache** elimina `qmlcache/` y `livemap_tiles/`. NO toca logs, configuración, history.

Apéndice E – Debug logging avanzado

E.1 Activación del debug

En **Setup** → **Advanced** → **Debug logging**, tres toggles:

TOGGLE	ARCHIVO OUTPUT	CRECIMIENTO ESTIMADO
<code>DebugCAT=true</code>	<code>cat.log</code>	1 MB/hora de uso intenso
<code>DebugAudio=true</code>	<code>decoder.log</code> (sección audio)	5 MB/hora
<code>DebugDecoder=true</code>	<code>decoder.log</code> (full)	100-500 MB/hora

E.2 Formato del log

Los logs siguen el formato Qt estándar:

```
[2026-05-20 14:32:18.473] [Decoder] [INFO] FT2 slot 14:32:15 → 3 decodes
[2026-05-20 14:32:18.474] [Decoder] [DEBUG] Pass 1: sync@0.532, SNR=-18.3, CRC=OK
[2026-05-20 14:32:18.475] [Decoder] [DEBUG] Pass 2: sync@0.481, SNR=-19.1, CRC=FAIL
[2026-05-20 14:32:18.476] [CAT] [DEBUG] TX: FA00007074000; → OK in 12ms
```

Niveles: `TRACE` < `DEBUG` < `INFO` < `WARNING` < `ERROR` < `CRITICAL`

E.3 Filtros Qt logging

DECODIUM expone las categorías Qt estándar. Para activar/desactivar categorías vía env var:

```
# Unix-Like
export QT_LOGGING_RULES="decodium.cat.debug=true;decodium.decoder.debug=false"
./Decodium-1.0.262-x86_64.AppImage

# Windows (PowerShell)
$env:QT_LOGGING_RULES="decodium.cat.debug=true"
.\decodium.exe
```

Categorías disponibles:

- `decodium.audio.*`
- `decodium.cat.*`
- `decodium.decoder.*`
- `decodium.network.*`
- `decodium.qml.*`
- `decodium.ui.*`

E.4 Bug report con log

Cuando se reporta un bug:

1. **Reproduce** el problema con debug ON
2. **Extrae** los logs del momento del problema (`tail -n 1000 decodium.log`)
3. **Anonimiza** (elimina indicativos sensibles si necesario)
4. **Adjunta** a un GitHub Issue o envía vía Help → Report Bug

El equipo acepta logs hasta 50 MB vía GitHub. Para logs más grandes, usa gist o file sharing.

E.5 Log rotation

DECODIUM **no rota automáticamente** los logs. Es responsabilidad del usuario borrar archivos de log antiguos si se vuelven grandes.

Script de limpieza típico (Linux/macOS, cron diario):

```
#!/bin/bash
# Mantiene solo los últimos 7 días de logs
find ~/.local/share/Decodium/Logs/ -name "*.log" -mtime +7 -delete
```

En Windows, equivalente PowerShell:

```
Get-ChildItem $env:APPDATA\Decodium\logs\*.log |  
Where-Object {$_.LastWriteTime -lt (Get-Date).AddDays(-7)} |  
Remove-Item
```

Apéndice F – UDP/IPC API para integraciones externas

DECODIUM 4.0 expone un canal de comunicación UDP compatible con el protocolo **WSJT-X UDP** (puerto 2237 por defecto). Esto permite a programas de terceros recibir notificaciones de decode, QSO logueados, status changes, y enviar comandos.

F.1 Filosofía de compatibilidad

El protocolo ha sido diseñado para ser **drop-in compatible** con WSJT-X. Significa que software existente como **Log4OM, GridTracker, JTAAlert, N1MM+, DXKeeper**, etc. funcionan con DECODIUM **sin modificación alguna**.

Extensiones específicas FT2 (campo `SUBMODE=FT2`, flag ASYMX) se transmiten como campos adicionales que los clientes WSJT-X-compatible ignoran sin errores.

F.2 Setup del UDP server

En **Setup** → **Network** → **UDP**:

PARÁMETRO	DEFAULT	NOTAS
Enable UDP	✓	Desactiva para deshabilitar la API
Multicast address	<code>224.0.0.1</code>	IP multicast para recepción
Multicast port	<code>2237</code>	Puerto estándar WSJT-X
Server name	<code>Decodium</code>	Identificador (visible a los clientes)
Accept commands	✓	Permite a los clientes enviar comandos (ver F.4)

Importante: Si múltiples aplicaciones DECODIUM (o WSJT-X) corren en el mismo PC, **deben usar puertos UDP diferentes** para evitar conflictos. La MultiRig CLI (Apéndice G) gestiona esto automáticamente con `-udp-port`.

F.3 Mensajes out-bound (DECODIUM → Cliente)

Todos los mensajes son **big-endian** siguiendo el mismo encoding de WSJT-X.

F.3.1 Header común

Cada paquete empieza con:

```
[4 bytes] Magic number: 0xADBCCBDA
[4 bytes] Schema version: 0x00000003 (compatible WSJT-X 2.5+)
[4 bytes] Message type ID
[variable] String "Decodium" (length-prefixed)
[payload específico para el tipo]
```

F.3.2 Tipos de mensaje principales

TYPE ID	NOMBRE	CONTENIDO
0	Heartbeat	Status periódico (1/sec)
1	Status	Banda, modo, frecuencia, indicativo actuales
2	Decode	Decode individual (ver F.3.3)
3	Clear	Limpieza de decode list
4	Reply	Respuesta a un Decode (TX prepare)
5	QSO Logged	QSO completado y logueado
6	Close	Aplicación cerrando
8	WSPR Decode	Decode WSPR (raro en DECODIUM)
10	Logged ADIF	QSO logueado como string ADIF completo
12	Highlight Callsign	Color hint para indicativo en el display
13	Switch Configuration	Cambio config remoto

F.3.3 Mensaje Decode (Type 2)

El formato Decode es el más importante para integraciones:

```

[Header común]
[4 bytes] new flag (1 = first time seen)
[4 bytes] UTC time (ms since midnight)
[4 bytes] SNR (signed dB)
[8 bytes] Delta-time (double, seconds)
[4 bytes] Delta-frequency (Hz)
[variable] Mode string ("FT8", "FT2", "FT4", etc.)
[variable] Message string ("CQ N2SS FM29")
[1 byte] Low confidence flag
[1 byte] Off-air flag
[variable] SUBMODE string (FT2 only: "FT2", "FT2A" para ASYMX)

```

Extensión FT2: El campo SUBMODE final es opcional. Clientes WSJT-X-compatible que no lo conocen simplemente paran la lectura del paquete antes – su comportamiento sigue siendo correcto.

F.3.4 Mensaje QSO Logged (Type 5)

```

[Header común]
[8 bytes] Date/time off (ms since epoch)
[variable] DX call
[variable] DX grid
[8 bytes] TX frequency (Hz)
[variable] Mode ("FT8", "FT2"...)
[variable] Report sent
[variable] Report received
[variable] TX power
[variable] Comments
[variable] Name
[8 bytes] Date/time on (ms since epoch)
[variable] Operator call
[variable] My call
[variable] My grid
[variable] Exchange sent
[variable] Exchange received
[variable] ADIF property ID (para FT2: "SUBMODE")
[variable] ADIF property value (para FT2: "FT2")

```

F.4 Mensajes in-bound (Cliente → DECODIUM)

Para activar la recepción de comandos, habilita **Accept commands** en Setup. ⚠ Solo aplicaciones de confianza deberían poder enviar comandos.

F.4.1 Comandos soportados

TYPE ID	NOMBRE	EFEECTO
4	Reply	Inicia QSO con el indicativo indicado
5	Halt TX	Stop transmisión inmediato
6	Free Text	Establece texto TX libre
9	Switch Config	Carga configuración alternativa
11	Replay	Re-decodifica audio del último slot

F.4.2 Ejemplo Python – escuchar los decodes

```
import socket
import struct

UDP_IP = "224.0.0.1" # multicast
UDP_PORT = 2237

sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM, socket.IPPROTO_UDP)
sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
sock.bind(("", UDP_PORT))

# Join multicast group
mreq = struct.pack("4sl", socket.inet_aton(UDP_IP), socket.INADDR_ANY)
sock.setsockopt(socket.IPPROTO_IP, socket.IP_ADD_MEMBERSHIP, mreq)

while True:
    data, addr = sock.recvfrom(4096)
    magic, schema, msg_type = struct.unpack(">III", data[0:12])
    if magic == 0xADBCCBDA:
        continue
    if msg_type == 2: # Decode
        print(f"Decode received from {addr}: {len(data)} bytes")
        # Parse decode payload here
```

F.5 Integraciones probadas

SOFTWARE	VERSIÓN PROBADA	FUNCIONALIDADES SOPORTADAS
Log4OM	2.27+	Auto-log QSO, decode en tiempo real
GridTracker	1.25+	Live map, Worked-Before highlighting
JTAlert	2.50+	Alerta audio, wanted callsign
N1MM+ Logger	1.0.10+	Contest logging via UDP
DXKeeper	todas	Auto-import ADIF
HRD Logbook	6.x+	Cross-reference vía ADIF

Nota sobre JTAAlert: algunas alertas específicas de JTAAlert (ej. "needed for WAS") dependen del SUBMODE. Para contactos FT2, JTAAlert versión 2.55+ reconoce SUBMODE=FT2 y aplica los conteos correctamente.

F.6 Throttling y best practices

- **Heartbeat cada 1 segundo** por defecto. Configurable vía INI: `[Network] HeartbeatInterval=1000`
- **Decode messages** transmitidos en 200 ms desde la decodificación
- **Burst limit:** max 100 mensajes/segundo en loopback
- **Buffer size:** 4096 bytes default (suficiente para cualquier mensaje actual)

Para escribir clientes robustos:

- **Siempre verificar el magic number** antes de parsear
- **Gestionar schema versions** futuras (campo schema)
- **Skip unknown message types** en lugar de abortar
- **Reconectar** si no se recibe heartbeat por 5+ segundos

Apéndice G – MultiRig CLI

DECODIUM 4.0 soporta ejecución de **instancias múltiples paralelas** para estaciones con varias radios. Todas las opciones de la command line están documentadas aquí.

G.1 Sintaxis básica

```
decodium [opciones globales] [opciones runtime] [opciones debug]
```

Lanzamientos típicos:

```
# Instancia única, configuración default
decodium

# Instancia específica con config dedicada
decodium --instance 2 --config secondary.ini

# Lanzamiento batch con todos los parámetros
decodium --instance 1 \
  --rig-port COM3 \
  --audio-in-"USB Audio CODEC" \
  --udp-port 2237 \
  --config primary.ini
```

G.2 Opciones globales

SWITCH	TIPO	DEFAULT	DESCRIPCIÓN
<code>--help</code> , <code>-h</code>	flag	–	Muestra help y sale
<code>--version</code> , <code>-V</code>	flag	–	Versión y build info
<code>--config=<path></code>	path	<code>decodium.ini</code>	Archivo configuración custom
<code>--instance=<N></code>	int	1	Número instancia (1-9)
<code>--data-dir=<path></code>	path	auto	Carpeta de datos custom
<code>--cache-dir=<path></code>	path	auto	Carpeta caché custom
<code>--quiet</code> , <code>-q</code>	flag	false	Reduce output consola
<code>--verbose</code> , <code>-v</code>	flag	false	Output verboso
<code>--no-splash</code>	flag	false	Salta el splash screen

G.3 Opciones runtime

G.3.1 Radio (CAT)

SWITCH	EQUIVALENTE INI
<code>--rig-type=<id></code>	<code>[Radio] RigType</code>
<code>--rig-port=<port></code>	<code>[Radio] RigPort</code>
<code>--rig-baud=<baud></code>	<code>[Radio] RigBaud</code>
<code>--rig-civ-address=<hex></code>	<code>[Radio] CIVAddress</code>
<code>--hrd-bridge=<host:port></code>	<code>[Radio] HRDBridge=true, HRDHost, HRDPort</code>
<code>--omnirig=<N></code>	<code>[Radio] OmniRig=true, OmniRigInstance</code>

G.3.2 Audio

SWITCH	EQUIVALENTE INI
<code>--audio-in=<name></code>	<code>[Audio] AudioIn</code>
<code>--audio-out=<name></code>	<code>[Audio] AudioOut</code>
<code>--sample-rate=<hz></code>	<code>[Audio] SampleRate</code>
<code>--buffer-size=<frames></code>	<code>[Audio] BufferSize</code>

G.3.3 PTT

SWITCH	EQUIVALENTE INI
<code>--ptt-method=<cat\\ vox\\ rts\\ dtr></code>	<code>[PTT] Method</code>
<code>--ptt-port=<port></code>	<code>[PTT] Port</code>

G.3.4 Network

SWITCH	EQUIVALENTE INI
<code>--udp-port=<port></code>	<code>[Network] UDPPort</code>
<code>--udp-multicast=<ip></code>	<code>[Network] MulticastAddress</code>
<code>--no-udp</code>	<code>[Network] EnableUDP=false</code>

G.3.5 FT2 mode

SWITCH	EQUIVALENTE INI
<code>--ft2-deep=<true\\ false></code>	<code>[FT2] EnableDEEP</code>
<code>--ft2-asymx</code>	<code>[FT2] EnableASYMX=true</code>
<code>--ft2-qq=<true\\ false></code>	<code>[FT2] EnableQQ</code>

G.4 Opciones quick-start

SWITCH	EFEECTO
<code>--start-band=<m></code>	Banda inicial (ej. <code>--start-band=40</code> para 40m)
<code>--start-mode=<mode></code>	Modo inicial (ej. <code>--start-mode=FT2</code>)
<code>--start-freq=<hz></code>	Frecuencia exacta inicial en Hz
<code>--auto-monitor</code>	Arranca con MON activo
<code>--auto-deep</code>	Arranca con DEEP activo

G.5 Opciones de debug

SWITCH	EFEECTO
<code>--debug-cat</code>	Equivale a <code>[Advanced] DebugCAT=true</code>
<code>--debug-audio</code>	Equivale a <code>[Advanced] DebugAudio=true</code>
<code>--debug-decoder</code>	Equivale a <code>[Advanced] DebugDecoder=true</code>
<code>--debug-all</code>	Todos los debug ON
<code>--log-file=<path></code>	Output log a archivo específico

G.6 Ejemplos multi-instancia concretos

G.6.1 Dos radios Yaesu en paralelo

```
# Terminal 1: FT-991A en 20m
decodium --instance 1 \
  --rig-type 1035 \
  --rig-port COM3 --rig-baud 38400 \
  --audio-in-"USB Audio CODEC" \
  --audio-out "USB Audio CODEC" \
  --udp-port 2237 \
  --start-band 20 --start-mode FT8 \
  --config ft991a.ini &

# Terminal 2: FT-DX10 en 40m FT2
decodium --instance 2 \
  --rig-type 1041 \
  --rig-port COM4 --rig-baud 38400 \
  --audio-in-"USB Audio CODEC #2" \
  --audio-out "USB Audio CODEC #2" \
  --udp-port 2238 \
  --start-band 40 --start-mode FT2 \
  --ft2-deep true --ft2-asymx \
  --config ftdx10.ini &
```

G.6.2 Setup contest con 2 radios + UDP relay

```
# Run #1 - main contest, port 2237
decodium --instance 1 --config contest_main.ini \
  --udp-port 2237 --auto-deep --auto-monitor &

# Run #2 - multiplier hunting, port 2238 con feed a N1MM+
decodium --instance 2 --config contest_mult.ini \
  --udp-port 2238 --start-band 40 &

# N1MM+ escuchando en ambas (configúralo en su UDP setup)
```

G.6.3 SWL setup (solo recepción)

```
decodium --instance 3 --config swl.ini \
  --no-udp \
  --start-mode FT2 --start-band 20 \
  --ptt-method none # deshabilita TX completamente
```

G.7 Stop limpio de instancias

Cada instancia responde a `SIGTERM` (Linux/macOS) o `WM_CLOSE` (Windows). Para stop limpio desde script:

```
# Linux/macOS
pkill -SIGTERM decodium
# o para instancia específica:
pkill -SIGTERM -f "decodium --instance=2"

# Windows PowerShell
Get-Process decodium | Stop-Process
```

Atención: un `kill -9` (SIGKILL) **no guarda** la sesión actual. El log ADIF del último QSO podría no ser flusheado. Usa siempre SIGTERM.

Apéndice H – Build desde fuentes

Este apéndice documenta la compilación de DECODIUM 4.0 desde fuentes GitHub, para quien quiera parchear, contribuir, o construir builds personalizadas.

H.1 Repositorios

REPOSITORIO	URL
Main upstream	https://github.com/iu8lmc/Decodium-4.0-Core-Shannon
Salvatore mirror	https://github.com/elisir80/Decodium-4.0-Core-Shannon
Branch <code>main</code>	Stable releases
Branch <code>dev</code>	Development, puede estar broken
Branch <code>win</code>	Windows-specific patches

H.2 Dependencias

H.2.1 Comunes a todas las plataformas

DEPENDENCIA	VERSIÓN MÍNIMA	NOTAS
Qt	6.11.0	Required, CORE/QUICK/QML/NETWORK/MULTIMEDIA
CMake	3.21	Build system
GCC / Clang / MSVC	C++17 capable	Compiler
gfortran	9.0+	Solo para <code>lib/ft2/decode174_91_ft2.f90</code>
libfftw3	3.3+	FFT
libsndfile	1.0.31+	WAV/audio I/O
Hamlib	4.7+	CAT

H.2.2 Específicas por plataforma

Ubuntu/Debian:

```
sudo apt install build-essential cmake git \  
qt6-base-dev qt6-declarative-dev qt6-multimedia-dev \  
libfftw3-dev libsndfile1-dev libhamlib-dev \  
gfortran libomp-dev
```

Fedora/RHEL:

```
sudo dnf install gcc-c++ cmake git \  
qt6-qtbase-devel qt6-qtdeclarative-devel qt6-qtmultimedia-devel \  
fftw-devel libsndfile-devel hamlib-devel \  
gcc-gfortran libomp-devel
```

Arch Linux:

```
sudo pacman -S base-devel cmake git qt6-base qt6-declarative qt6-multimedia \  
fftw libsndfile hamlib gcc-fortran openmp
```

macOS (Homebrew):

```
brew install cmake qt@6 fftw libsndfile hamlib gcc libomp  
brew link --force qt@6
```

Windows: requiere **Qt 6.11 online installer + MSYS2** o **Visual Studio 2022 con CMake**. Ver

[docs/BUILD_WINDOWS.md](#) en el repo.

H.3 Clone y build

```
# Clone con submódulos
git clone --recursive https://github.com/iu8lmc/Decodium-4.0-Core-Shannon.git
cd Decodium-4.0-Core-Shannon

# Configura build
mkdir build && cd build
cmake -DCMAKE_BUILD_TYPE Release \
      -DCMAKE_PREFIX_PATH /path/to/qt6 \
      -DCMAKE_INSTALL_PREFIX /usr/local \
      ..

# Compila (usa todos los cores)
make -j$(nproc)
# o en macOS:
make -j$(sysctl -n hw.ncpu)

# Install (opcional)
sudo make install

# Lanza
./decodium
```

H.4 Build flags

FLAG CMAKE	EFEECTO
<code>-DCMAKE_BUILD_TYPE=Debug</code>	Build con símbolos debug (+50% size)
<code>-DCMAKE_BUILD_TYPE=Release</code>	Build optimizado (default)
<code>-DCMAKE_BUILD_TYPE=RelWithDebInfo</code>	Release + symbols (para profiling)
<code>-DENABLE_OPENMP=ON</code>	Multi-threading decoder (default ON)
<code>-DENABLE_TESTS=ON</code>	Build test suite (requiere gtest)
<code>-DENABLE_TCI=ON</code>	Build con TCI 1.5 support para FlexRadio
<code>-DUSE_SYSTEM_HAMLIB=ON</code>	Usa Hamlib del sistema en lugar del bundled
<code>-DCMAKE_INSTALL_PREFIX=<path></code>	Custom install location

H.5 Cross-compile

H.5.1 Linux → Windows (MinGW)

```
sudo apt install mingw-w64 qt6-mingw-w64-dev

cmake -DCMAKE_TOOLCHAIN_FILE cmake/mingw-w64-toolchain.cmake \
      -DCMAKE_BUILD_TYPE Release \
      ..
make -j$(nproc)
```

H.5.2 Build para Raspberry Pi (en la Pi misma)

```
# En Raspberry Pi OS (64-bit)
sudo apt install qt6-base-dev qt6-declarative-dev qt6-multimedia-dev \
      libfftw3-dev libsndfile1-dev libhamlib-dev gfortran

cmake -DCMAKE_BUILD_TYPE Release ..
make -j4 # Pi 4/5 con 4-8 GB RAM
```

El port community RPi está cuidado por **LU7DID** — ver <https://github.com/Lu7did> para sus patches específicos.

H.6 Build Appimage Linux

```
# Tras make
cd build
./tools/build_appimage.sh
# Output: Decodium-1.0.262-x86_64.AppImage
```

El script incluye linuxdeploy + Qt6 plugin, copia las librerías runtime, y crea el AppImage final.

H.7 Troubleshooting build

H.7.1 “Qt6 not found”

```
CMake Error: Could not find Qt6 (missing: Qt6_DIR)
```

Solución: especifica `CMAKE_PREFIX_PATH` :

```
cmake -DCMAKE_PREFIX_PATH /opt/qt/6.11.0/gcc_64 ..
```

H.7.2 “Hamlib version too old”

DECODIUM requiere Hamlib **4.7+**. En Debian/Ubuntu viejos, compila Hamlib desde fuentes:

```
git clone https://github.com/Hamlib/Hamlib.git
cd HamLib
./bootstrap && ./configure --prefix /usr/local && make && sudo make install
```

H.7.3 “Undefined symbol: omp_*”

OpenMP no enlazado. En macOS:

```
cmake -DCMAKE_C_COMPILER=$(brew --prefix llvm)/bin/clang \
      -DCMAKE_CXX_COMPILER=$(brew --prefix llvm)/bin/clang++ \
      -DCMAKE_PREFIX_PATH=$(brew --prefix qt@6) \
      ..
```

H.7.4 Build falla en `lib/ft2/decode174_91_ft2.f90`

gfortran faltante o versión vieja. Instala gfortran 9+ y especifica:

```
cmake -DCMAKE_Fortran_COMPILER=gfortran-11 ..
```

Apéndice I – Contribuir al proyecto

DECODIUM 4.0 es un proyecto **community-driven**. Cada contribución es bienvenida, desde bug reports hasta nuevas features.

I.1 Canales de contribución

CANAL	PARA QUÉ
GitHub Issues	Bug reports, feature requests
GitHub Pull Requests	Code contribution
Telegram community	Discusiones, soporte usuarios
Email	Seguridad, vulnerabilidades (privado)

I.2 Flujo PR estándar

I.2.1 Fork y branch

```
# 1. Fork del repo en GitHub (botón Fork arriba a la derecha)

# 2. Clone de tu fork
git clone https://github.com/tu-usuario /Decodium-4.0-Core-Shannon.git
cd Decodium-4.0-Core-Shannon

# 3. Añade upstream
git remote add upstream https://github.com/iu8lmc/Decodium-4.0-Core-Shannon.git

# 4. Crea branch dedicada
git checkout -b feature/descripción-corta
# Ejemplos:
# git checkout -b feature/icom-ic9700-ptt-fix
# git checkout -b feature/ft2-deep-tuning
```

I.2.2 Trabaja en el branch

```
# Haz modificaciones
vim src/decoder/ft2_engine.cpp

# Commit con mensaje convencional
git add src/decoder/ft2_engine.cpp
git commit -m "fix(decoder): preserve sync threshold across band changes

Previously, the FT2 sync threshold was reset to default on every
band change, causing decode loss for the first 2-3 slots.

Fixes #142"

# Push a tu fork
git push origin feature/icom-ic9700-ptt-fix
```

I.2.3 Abre Pull Request

1. En GitHub: ve a tu fork → **Compare & Pull Request**
2. **Título PR:** conventional commit format (ver I.3)
3. **Descripción:** incluye:
 - Qué hace la PR
 - Issue que cierra (si aplicable, usa `Closes #XXX`)
 - Tests ejecutados
 - Screenshots/log si relevantes
4. Asigna reviewer (default: `@iu8lmc` y `@elisir80`)

I.3 Conventional commits

Todos los commits deben seguir **Conventional Commits 1.0:**

<tipo>(<scope opcional>): <descripción corta>

<cuerpo opcional>

<footer opcional>

I.3.1 Tipos

TIPO	CUÁNDO USARLO
<code>feat</code>	Nueva feature
<code>fix</code>	Bug fix
<code>docs</code>	Solo documentación
<code>style</code>	Formatting (sin cambio de lógica)
<code>refactor</code>	Refactor sin cambio funcional
<code>perf</code>	Performance improvement
<code>test</code>	Añade/modifica tests
<code>build</code>	Build system, dependencias
<code>ci</code>	CI/CD changes
<code>chore</code>	Mantenimiento menor
<code>revert</code>	Revert de commit anterior

I.3.2 Scopes típicos

- `decoder` – modificaciones al decoder DSP
- `cat` – control radio
- `audio` – pipeline audio
- `ui` – interfaz usuario
- `ft2` – protocolo FT2
- `network` – UDP/IPC/PSK Reporter
- `build` – CMake/build scripts
- `docs` – documentación

1.3.3 Ejemplos

```
feat(ft2): implement Quick QSO 4-message flow

fix(cat): preserve DATA-U mode across TX/RX transitions on HRD bridge

docs(reference): add FT2 protocol full specification (Appendix A)

perf(decoder): reduce LDPC iteration count from 30 to 20 with no SNR loss

build: bump Qt requirement from 6.10 to 6.11 for new MultiMedia API

refactor(ui): extract WaterfallControls into reusable QML component
```

1.4 Code style

1.4.1 C++

ASPECTO	CONVENCIÓN
Indentation	4 spaces, NO tab
Line length	Max 100 chars
Brace style	K&R (open brace same line)
Naming classes	CamelCase
Naming métodos	camelCase()
Naming members	m_camelCase (prefijo m_)
Naming constantes	kUpperCamelCase (prefijo k)
Headers	.hpp para C++, .h para C
Include order	system → Qt → libs → project

Ejemplo:

```

#include <iostream>           // system
#include <QObject>           // Qt
#include <fftw3.h>           // lib
#include "decoder/ft2_engine.hpp" // project

class FT2Engine : public QObject
{
    Q_OBJECT

public:
    explicit FT2Engine(QObject *parent = nullptr);
    void processSlot(const float *audioData, int samples);

private:
    static const int kMaxPasses = 5;
    int m_passCount = 0;
    QString m_currentCallsign;
};

```

I.4.2 QML

ASPECTO	CONVENCIÓN
Indentation	4 spaces
Property naming	LowerCamelCase
Components	PascalCase.qml
Imports	Qt primero, custom después
Anchors over geometry	Siempre
JavaScript inline	Solo para lógica trivial

I.4.3 Fortran

Solo para el archivo LDPC. Mantenemos el estilo **idéntico a K1JT/WSJT-X** para facilitar el merge upstream.

I.5 Testing

I.5.1 Unit tests (gtest)

```

cmake -DENABLE_TESTS ON ..
make decodium_tests
./decodium_tests

```

Coverage actual: ~40% (foco en decoder core y CAT).

I.5.2 Integration test manual

Antes de una PR, verifica:

1. **Decode FT8** funcionando en banda activa
2. **Decode FT2** funcionando (DEEP ON)
3. **TX FT2 sincrónico** y **ASYMX**
4. **Quick QSO** completo
5. **CAT funcionando** en tu radio
6. **Live Map** poblada
7. **Log ADIF** correcto

I.5.3 Test en versiones anteriores

Las PR que tocan persistencia (INI, log) deben testear:

- Upgrade desde v1.0.260
- Upgrade desde v1.0.255
- Arranque fresh (sin INI anterior)
- Downgrade (usuario vuelve a v1.0.261 tras upgrade)

I.6 Documentación obligatoria

Una PR que añada feature **debe** actualizar:

1. **CHANGELOG.md** (entry bajo Unreleased)
2. **Reference Manual** si introduce nuevas INI keys, comandos CAT, API
3. **User Manual** si introduce funcionalidades UI visibles
4. **README.md** si cambia install/build process

I.7 Code review

Los reviews apuntan a 3 cosas:

1. **Correctness** – ¿el código hace lo que dice que hace?
2. **Robustness** – ¿maneja edge cases? ¿Error de red? ¿CAT timeout?
3. **Style** – ¿sigue las convenciones del proyecto?

Tiempo de review típico: 2-7 días. Para fix urgentes, ping directo en Telegram.

I.8 Créditos

Los contributors son **automáticamente acreditados** en:

- [CONTRIBUTORS.md](#) (lista alfabética)
- Diálogo [HeLp → About](#) (top contributors)
- Release notes específicas para sus PRs

Para solicitud de eliminación de créditos (privacidad): email a los maintainers.

Apéndice J – Troubleshooting cookbook

Este apéndice recopila **30+ escenarios concretos** reportados durante la Beta Pública. Cada escenario incluye: síntoma específico, logs diagnósticos, causa identificada, solución probada.

J.1 Decoder

J.1.1 “El decoder deja de decodificar tras 1-2 horas de uso continuo”

Síntoma: Decode normal por horas, luego de repente 0 decode por 5+ minutos, luego reanuda.

Log diagnóstico (decoder.log):

```
[12:34:56] [Decoder] [WARNING] FFT buffer underrun on slot 12:34:45
[12:35:00] [Decoder] [WARNING] Audio sample rate drift detected: 47950 Hz (expected 48000)
```

Causa: Drift del clock de la tarjeta de sonido USB. Típico en CODECs integrados de radio tras largas sesiones TX/RX (calentamiento de componentes).

Solución: 1. Setup → Audio → **Resample to nominal** = ✓ 2. Reduce `[Audio] BufferSize` de 1024 a 512 3. Si persistente, considera una tarjeta de sonido externa USB-isolated

J.1.2 “Decodes muy buenos en FT8, cero en FT2 en la misma banda”

Síntoma: 30+ decodes FT8/slot, 0 en FT2 incluso con estaciones conocidas activas.

Causa probable: frecuencia FT2 errónea. La sub-banda FT2 no coincide con la de FT8.

Solución: - 40m: FT8 = 7.074, **FT2 = 7.080** ← a menudo confundido - 20m: FT8 = 14.074, **FT2 = 14.080** - 15m: FT8 = 21.074, **FT2 = 21.080**

DECODIUM cambia automáticamente la frecuencia al cambio de modo, pero si has deshabilitado el auto-tune, controla manualmente.

J.1.3 “Decodes buenos con DEEP off, empeoran con DEEP on”

Síntoma: contraintuitivo – más decodes sin DEEP que con DEEP.

Causa: sistema con CPU lenta (RPi 3, procesadores dual-core <2 GHz). DEEP no consigue completar los 5 pasos a tiempo, algunos son dropeados.

Diagnóstico:

```
[Decoder] [WARNING] DEEP pass 4/5 timeout (847ms > 800ms budget)
```

Solución: 1. Reduce número de pasos: `[FT2] DEEPPasses=3` 2. Deshabilita MMSE: `[FT2] EnableMMSE=false` (libera 15-20% CPU) 3. Upgrade hardware si posible (Pi 4 o PC moderno)

J.2 CAT

J.2.1 “Hamlib timeout intermitente, cada 30-60 segundos”

Síntoma: CAT funciona, pero cada minuto aproximadamente aparece brevemente `CAT: Timeout` (amarillo parpadea), luego vuelve a normal.

Causa: poll interval demasiado agresivo para la radio.

Diagnóstico (cat.log):

```
[10:01:00] [CAT] [DEBUG] Poll IF; → response 28ms ✓  
[10:01:01] [CAT] [DEBUG] Poll IF; → response 31ms ✓  
[10:01:02] [CAT] [WARNING] Poll IF; → timeout after 500ms  
[10:01:02] [CAT] [INFO] Retrying...  
[10:01:03] [CAT] [DEBUG] Poll IF; → response 47ms ✓
```

Solución: aumenta `[Radio] PollInterval` de 1000 (default) a 2000 ms.

J.2.2 “El PTT dispara pero el audio no parte (radio TX sin modulación)”

Síntoma: la radio pasa a TX (LED rojo), el PWR meter sube pero a 0W o 5W (susurro), ninguna decodificación del otro lado.

Causa más común: dispositivo audio erróneo o nivel USB OUT a cero.

Diagnóstico: 1. Verifica VU meter TX en DECODIUM (abajo): si la barra está quieta → no hay audio saliente 2. Verifica menú radio (Kenwood Menu 65, Yaesu Menu 114, Icom USB MOD Level)

Solución step: 1. Setup → Audio → Audio OUT: debe ser el dispositivo USB CODEC de la radio 2. Panel audio Windows/macOS/Linux: nivel speaker USB CODEC al 75-100% 3. Menú radio USB OUT level: parte de 5, regula hasta leer PWR deseado

J.2.3 “La frecuencia del display de la radio cambia sola”

Síntoma: mientras operas, el VFO de la radio cambia banda o frecuencia en modos que no has pedido.

Causa más común: software de terceros (HRD, JTAIAlert, GridTracker, contest logger) está enviando comandos CAT en concurrencia.

Diagnóstico: 1. Cierra todos los software ham excepto DECODIUM 2. Si el problema desaparece → conflicto CAT confirmado

Solución: - Opción A: usa **HRD bridge** (Setup → Radio → HRD bridge) – HRD se convierte en el hub central, los demás programas hablan con HRD - Opción B: usa **OmniRig** (solo Windows) – coordina el acceso CAT entre programas - Opción C: si solo DECODIUM sirve, deja DECODIUM como único software CAT

J.3 Audio

J.3.1 “Audio crackle/popping en RX, decode fallidos”

Síntoma: cuando se escucha en monitor (MON), se oye “crackle” intermitente. Los decodes fallan a menudo (CRC fail).

Causa más probable: sample rate mismatch entre DECODIUM y la tarjeta de sonido, o buffer underrun.

Diagnóstico (Linux):

```
pactl list | grep -A 20 "Source.*USB"
# Busca sample rate actual
```

Solución: 1. Setup → Audio → SampleRate: fuerza a 48000 Hz 2. BufferSize: aumenta a 2048 frames (más estable, latencia ligeramente mayor) 3. En Linux: asegúrate que PulseAudio/PipeWire no esté haciendo resampling: `bash # Edit /etc/pulse/daemon.conf default-sample-rate = 48000 alternate-sample-rate = 12000 resample-method = soxr-vhq`

J.3.2 “El micrófono de Windows activa enhancement audio”

Síntoma: decodes pésimos, señal visible en waterfall pero el decoder falla.

Causa: Windows aplica “Audio Enhancements” (noise suppression, AGC, etc.) sobre el mic USB CODEC. Estos enhancement **destruyen** la señal digital.

Solución (Windows 10/11):

```
Panel de control → Audio → Grabación →
USB Audio CODEC (Micrófono) → Propiedades → Avanzado →
x DESACTIVAR "Mejoras de audio"
```

Además, en **Enhancements tab** (si presente): desactiva todo.

J.3.3 “DECODIUM no ve mi tarjeta de sonido USB”

Síntoma: la radio está conectada, Windows/macOS la reconocen, pero DECODIUM en el dropdown audio no muestra nada.

Causa: tarjeta de sonido inicializada después de DECODIUM, o Qt no la ha enumerado.

Solución: 1. Cierra DECODIUM 2. Verifica que la tarjeta de sonido sea visible **desde el sistema operativo** (Panel audio Windows / sound preferences macOS / `aplay -L` Linux) 3. Reinicia DECODIUM tras haber conectado/encendido la radio

Si persiste, controla que la tarjeta no esté **ya en uso** por otro software (ej. Skype, Zoom en background).

J.4 Live Map

J.4.1 “Live Map carga las tiles lentamente o se congela”

Síntoma: abres DECODIUM, vas a Live Map, ves “loading...” por 30+ segundos. A veces freeze total UI.

Causa: primer download de una cantidad grande de tiles OpenStreetMap, o rate limiting OSM.

Solución: 1. **Primera vez:** sé paciente. El download inicial puede durar 1-2 minutos. 2. Si el freeze persiste: borra caché `~/ .cache/Decodium/livemap_tiles/` y reintenta 3. OpenStreetMap rate limit: max ~2 req/sec. Si overload (ej. zoom rápido), espera 60s

J.4.2 “Live Map muestra solo Europa, no veo USA/Asia”

Síntoma: el mapa aparece pero los decodes de W6/JA/VK no aparecen como puntos.

Causa: filtro distancia activo o region filter.

Diagnóstico: - En Live Map, arriba: botones **All, IN → ME, ME → DX, BAND** - Si "BAND" activo → muestra solo banda actual - Si "IN → ME" activo → muestra solo PSK Reporter reverse (requiere internet)

Solución: click en **All**, y en Setup → Live Map → distance filter: 0 (no limit).

J.5 Actualizaciones y migración

J.5.1 "Tras actualizar, el log ADIF no se abre más"

Síntoma: tras update v1.0.260 → v1.0.262, los logs existentes no se abren o aparecen vacíos.

Causa: path log cambiado, o file system permission issue.

Diagnóstico:

```
# Linux/macOS
ls -la ~/.local/share/Decodium/log.adif
ls -la ~/.local/share/Decodium/log.adif.bak
```

Solución: 1. Verifica que existe `log.adif.bak` (backup auto-creado por el instalador) 2. Si sí: restáuralo: `mv log.adif.bak log.adif` 3. Si no: el log debería estar todavía donde estaba antes (sin path change en v1.0.262)

J.5.2 "Tras upgrade, las preferencias CAT/Audio se han perdido"

Síntoma: abres DECODIUM tras update, debes reconfigurar todo.

Causa: `decodium.ini` corrupto durante upgrade (raro).

Solución: restaura el backup:

```
# Linux
mv ~/.config/Decodium/decodium.ini.bak ~/.config/Decodium/decodium.ini

# Windows
move %APPDATA%\Decodium\decodium.ini.bak %APPDATA%\Decodium\decodium.ini

# macOS
mv ~/Library/Application\ Support/Decodium/decodium.ini.bak ~/Library/Application\
Support/Decodium/decodium.ini
```

Reinicia DECODIUM.

J.6 Performance y estabilidad

J.6.1 "DECODIUM usa 90%+ CPU constantemente"

Síntoma: task manager muestra DECODIUM al 90-100% CPU incluso en idle.

Causas posibles: 1. DEEP activo en CPU subdimensionada 2. Debug logging olvidado activo 3. Loop bug (raro, reporta como issue)

Solución step: 1. Setup → Advanced → Debug logging: **todo OFF** 2. Setup → FT2 → DEEPPasses: reduce a 3 3. Reinicia. Si persiste, abre Help → Performance Profile, espera 30s, guarda el archivo y adjúntalo a un GitHub Issue.

J.6.2 “La memoria crece a >2 GB tras unas horas”

Síntoma: memory leak. RAM crece continuamente hasta saturar.

Causa conocida: Live Map tile cache growth + decoder buffers no liberados en error path.

Solución (v1.0.262): reducido significativamente en v1.0.262. Si aún presente:

1. Workaround temporal: limita Live Map cache:

```
[LiveMap]
MaxCacheSizeMB=200
```

2. Reinicia DECODIUM cada 6-8 horas si sirve maratón contest
3. Reporta como issue con un Performance Profile

J.6.3 “Crash random al final del slot RX”

Síntoma: DECODIUM crashea (segfault) al final de un slot RX, parece random.

Diagnóstico: - Linux/macOS: ve si hay un core dump en `~/.local/share/Decodium/crashes/` - Windows: Event Viewer → Application logs

Causa conocida: race condition entre audio thread y decoder thread en sistemas multi-core con scheduling agresivo.

Solución (workaround v1.0.262):

```
[Advanced]
ThreadPoolSize=2 # en lugar de auto
```

Fix definitivo previsto para v1.0.263.

Fin del Reference Manual Completo. Para discusiones, soporte en vivo y bug reporting, el canal principal es la comunidad Telegram (link en ft2.it). Para code contribution, GitHub Issues y PRs son el canal oficial.

73 de Martino IU8LMC & Salvatore 9H1SR DECODIUM / FT2 Team – ARI Caserta · Italia · GPLv3