RT-5D DMR Transceiver

Programming Protocol Reverse-Engineering Report

Derived from CPS Source Code Analysis (JJCC-888DMR_CPS)
February 2026

1. Overview

This document presents a detailed analysis of the programming protocol used by the RT-5D DMR transceiver (also identified internally as the JJCC-888DMR). The analysis is derived entirely from the Customer Programming Software (CPS) source code supplied with the radio, specifically the C# .NET project "JJCC-888DMR_CPS".

The radio uses a custom, binary, framed serial protocol over a USB-serial adapter to exchange configuration data with a host PC. The protocol is symmetric in structure: both the PC and radio use the same packet framing. All configuration is transferred in a fixed sequence of typed data blocks, each covering a specific memory region or feature set within the radio.

2. Physical & Serial Layer

The CPS establishes the serial connection using the following fixed parameters, as seen in MainForm.cs:

Parameter	Value
`Baud Rate`	`115200`
`Data Bits`	`8`
`Parity`	`None (0)`
`Stop Bits`	`1`
`Flow Control`	`None (default)`
`Interface`	`USB-Serial (COM port, user-selectable)`

Note: *The baud rate of 115200 is hardcoded in all three communication call sites within MainForm.cs. There is no user-configurable baud rate option in the CPS.*

3. Packet Framing

Every message exchanged between the CPS and radio is encapsulated in a fixed-format binary frame. The structure is defined by the DataHandleHelper class. The frame is always transmitted and received in this exact layout:

Offset (bytes)	Field	Size (bytes)	Value / Notes
0	Frame Header (SOF)	1	0xA5 (165 decimal) - always this value
1	Command Byte	1	Identifies the operation (see Command Table)
2-3	Sequence / Page ID	2	Big-endian 16-bit. Used as packet/page counter
4-5	Payload Length	2	Big-endian 16-bit. Number of payload bytes (N)
6 ... 6+N-1	Payload Data	N	The data for this packet
6+N	CRC High Byte	1	CRC-16/CCITT, upper byte
6+N+1	CRC Low Byte	1	CRC-16/CCITT, lower byte

Total frame size = 6 (header) + N (payload) + 2 (CRC) = N + 8 bytes.

3.1 Start-of-Frame Detection (Receive Side)

The CPS receive state machine is three-stage:

- Stage 1 (ReceiveDataOne): Read bytes one at a time until 0xA5 is found. This byte marks the start of a frame.
- Stage 2 (ReceiveDataTwo): Read the next 5 bytes (bytes 1-5: command, sequence high, sequence low, length high, length low). Extract the payload length N from bytes 4-5.
- Stage 3 (HandleData): Wait until N + 2 bytes are available (payload + CRC), then read and validate.

3.2 CRC Algorithm

The CRC is CRC-16/CCITT (polynomial 0x1021, no initial value preset to 0xFFFF - the code XORs starting from 0). The CRC is computed over bytes 1 through 5+N (i.e., all frame fields except the SOF byte and the CRC itself). The algorithm from DataHandleHelper.CrcValidation():

```
int crc = 0;
for each byte in dat[offset .. offset+count-1]:
    crc ^= (byte << 8)
    for 8 iterations:
        if (crc & 0x8000) != 0: crc = (crc << 1) ^ 0x1021
        else:                   crc = (crc << 1)
crc = crc & 0xFFFF
```

CRC covers: cmd (byte 1) + seq_hi + seq_lo + len_hi + len_lo + payload (bytes 1 through 5+N).

> Note: *A reply with command byte 0xEE (238 decimal) is treated as a NAK/error response and is silently ignored by the receive handler.*

4. Command Set

The following command codes are defined in DataComCommandType.cs. The SOF byte (0xA5) is also defined there as 'FrameHeader'.

Command Name	Hex Value	Dec	Direction	Description
`CmdHandshake`	`0x02`	`2`	`Host -> Radio`	`Initial handshake, sends 15-byte ASCII string`
`CmdCheckPwd`	`0x05`	`5`	`Host -> Radio`	`Send 6-byte terminal password for auth`
`CmdGetVersion`	`0x46`	`70`	`Host -> Radio`	`Request 128-byte version block from radio`
`CmdOver`	`0x01`	`1`	`Host -> Radio`	`Session end / programming complete`
`CmdReadChMode`	`0x10`	`16`	`Host -> Radio`	`Read channel mode data block`
`CmdReadVfoMode`	`0x11`	`17`	`Host -> Radio`	`Read VFO mode data block`
`CmdReadOptionalFun`	`0x12`	`18`	`Host -> Radio`	`Read optional functions block`
`CmdReadAddrBook`	`0x13`	`19`	`Host -> Radio`	`Read address book (contacts) block`
`CmdReadRxGroup`	`0x14`	`20`	`Host -> Radio`	`Read receive group list block`
`CmdReadEncKey`	`0x15`	`21`	`Host -> Radio`	`Read encryption keys block`
`CmdReadDtmf`	`0x16`	`22`	`Host -> Radio`	`Read DTMF settings block`
`CmdReadBasicInfo`	`0x19`	`25`	`Host -> Radio`	`Read basic info (model name/ID) block`
`CmdWriteChMode`	`0x30`	`48`	`Host -> Radio`	`Write channel mode data block`
`CmdWriteVfoMode`	`0x31`	`49`	`Host -> Radio`	`Write VFO mode data block`
`CmdWriteOptionalFun`	`0x32`	`50`	`Host -> Radio`	`Write optional functions block`
`CmdWriteAddrBook`	`0x33`	`51`	`Host -> Radio`	`Write address book block`
`CmdWriteRxGroup`	`0x34`	`52`	`Host -> Radio`	`Write receive group list block`
`CmdWriteEncKey`	`0x35`	`53`	`Host -> Radio`	`Write encryption keys block`
`CmdWriteDtmf`	`0x36`	`54`	`Host -> Radio`	`Write DTMF settings block`
`CmdWriteBasicInfo`	`0x39`	`57`	`Host -> Radio`	`Write basic info (model/ID) - optional`
`(NAK/Error)`	`0xEE`	`238`	`Radio -> Host`	`Radio error response; CPS ignores and retries`

Note: *Additional image-related commands are defined (CmdImageHandshake=0x02, CmdIntoImageFlashEditMode=0x08, CmdSetImageAddress=0x03, CmdEraseImageOldData=0x04, CmdWriteImageNewData=0x57, CmdImageOver=0x06) for the boot screen image programming feature, which follows a separate flow not detailed in this document.*

5. Full Protocol Sequence

The communication always proceeds in a fixed, strict sequence regardless of direction (read or write). Each step is a request-response pair. Below is the complete ordered sequence:

Step #	Phase Name	Command	Sequence/Pages	Payload Size (bytes)	Notes
1	Handshake	0x02	0	15	ASCII: "PROGRAMJC8810DU"
2	Check Password	0x05	0	6	6 bytes: FF FF FF FF FF FF (default/blank)
3	Get Version	0x46	0	128	128-byte placeholder; radio returns version info
4	DTMF	0x16 / 0x36	0	272	1 packet only
5	Encryption Keys	0x15 / 0x35	0	264	1 packet only
6	Address Book	0x13 / 0x33	0-79	800 ea.	80 packets x 800 bytes = 64,000 bytes total
7	Rx Group List	0x14 / 0x34	0-3	1024 ea.	4 packets x 1024 bytes = 4,096 bytes total
8	Channel Mode	0x10 / 0x30	0-63	1024 ea.	64 packets x 1024 bytes = 65,536 bytes total
9	VFO Mode	0x11 / 0x31	0	128	1 packet (2 VFOs x 64 bytes each)
10	Optional Functions	0x12 / 0x32	0	64	1 packet
11	Basic Info	0x19 / 0x39	0	64	1 packet (write is conditional - see notes)
12	End Session	0x01	0	2	2 zero bytes; radio exits programming mode

Note: *For a READ session, the CPS sends Read commands (0x10-0x19) and receives the payload in the radio's response. For a WRITE session, the CPS sends Write commands (0x30-0x39) carrying the payload. In both cases the same packet structure applies.*

Note: *The BasicInfo write step (Step 11) is conditional: it only executes if the flag BasicInfoData.WriteModelNameAndId is true. This flag appears to be set only when writing model identity to the radio is explicitly desired.*

5.1 Timeout and Retry Mechanism

After transmitting a packet, the CPS starts a software timer with the following parameters:

- Timer interval: 200 ms per tick
- Timeout counter: 5 ticks (i.e., ~1 second timeout before flagging a retry)
- Maximum retry attempts: 3

On timeout, the CPS flushes any pending bytes from the receive buffer, retransmits the last packet exactly as sent, and decrements the retry counter. After 3 failed retries the session is aborted and CommFail is returned.

6. Data Block Formats

What follows is a byte-level description of each data block payload. All multi-byte integers are little-endian unless otherwise noted. Unused/padding bytes are filled with 0xFF. A leading 0xFF in a field generally indicates "empty" or "not programmed".

6.1 Handshake Block (15 bytes)

The handshake payload is the ASCII string PROGRAMJC8810DU, which encodes to exactly 15 bytes:

```
    50 52 4F 47 52 41 4D 4A 43 38 38 31 30 44 55
```

The radio is expected to reply with the same command byte (0x02) and a valid response payload. The CPS does not validate the content of the handshake reply, only that a valid framed response is received.

6.2 Password Block (6 bytes)

The password block sends 6 bytes representing the terminal password. The default/blank password is all 0xFF:

```
    FF FF FF FF FF FF
```

The radio validates this against its stored password. A NAK (0xEE) response would indicate a wrong password, though the CPS does not implement a specific password-error handler beyond the general NAK ignore.

6.3 Version Block (128 bytes)

The CPS sends 128 zero bytes as a placeholder. The radio replies with 128 bytes of version/identity information. The CPS does not parse this block beyond acknowledging receipt - it is informational only and used to confirm the radio type before proceeding.

6.4 DTMF Block (272 bytes, 1 packet)

The DTMF block is split into two sub-regions:

Offset	Size	Field	Encoding
0-4	5	Current ID	Each nibble = DTMF digit index into "0123456789ABCD*#"; 0xFF = empty
5	1	(reserved/pad)	0xFF
6	1	PTT ID Mode	Low nibble: 0=Off, 1=BOT, 2=EOT, 3=Both
7	1	DTMF Duration	Low nibble: 0-4 (index into duration table)
8	1	DTMF Interval	Low nibble: 0-4 (index into interval table)
9-31	23	(padding)	0xFF
32-271	240	Code Group List	15 groups x 16 bytes each

Each Code Group entry (16 bytes) layout:

Offset within entry	Size	Field	Encoding
0-5	6	DTMF Code	Each byte = digit index into "0123456789ABCD*#"; 0xFF = end/empty
6-15	10	(padding)	0xFF

6.5 Encryption Key Block (264 bytes, 1 packet)

Supports up to 8 encryption key entries, each 33 bytes:

Offset within entry	Size	Field	Encoding
0	1	Algorithm	Low nibble: 0=None, 1=Basic, 2=Enhanced
1-32	32	Key Data	Each byte = hex digit index into "0123456789ABCDEF"; 0xFF = empty

Total: 8 x 33 = 264 bytes. Entries marked 0xFF in byte 0 and byte 1 are treated as empty.

6.6 Address Book (800 bytes/packet, 80 packets = 4,000 contacts max)

The address book is the largest data region, transmitted in 80 consecutive packets. Each packet carries data for 50 contacts. Contact record layout (16 bytes each):

Offset	Size	Field	Encoding
0	1	Call Type	Low nibble: 0=Group, 1=Private, 2=All Call
1	1	(zero byte)	Always 0x00 for valid entries
2	1	Call ID byte 2 (MSB)	Big-endian 24-bit DMR ID (bytes 2-4)
3	1	Call ID byte 1	
4	1	Call ID byte 0 (LSB)	
5-14	10	Contact Name	GB2312 encoded string, null/0xFF terminated
15	1	(padding)	0xFF

An entry with bytes 0, 1, or 5 == 0xFF is considered unoccupied. The Call ID is validated to be within the DMR address range 1-16776415 (0x000001 to 0xFFFFFF).

6.7 Receive Group List (1024 bytes/packet, 4 packets = 32 groups max)

Each packet carries data for 8 receive groups. Each group occupies 128 bytes:

Offset within group entry	Size	Field	Encoding
0-95	96	Member ID List	Up to 32 members x 3 bytes each (24-bit DMR ID, big-endian)
96-107	12	Group Name	GB2312 encoded, null/0xFF terminated
108-127	20	(padding)	0xFF

Member IDs of 0x000000 indicate no further members. Each member is cross-referenced to the address book to resolve the contact name. An entry is skipped if byte 96 == 0xFF (name field empty).

6.8 Channel Mode (1024 bytes/packet, 64 packets = 1024 channels across up to 10 areas)

Channel data is the most complex block. Each packet carries 16 channels. Each channel occupies 64 bytes. The channel numbering is linear: packet N carries channels N*16 through N*16+15. Channels are mapped to areas using the formula: area = channel_index / channels_per_area.

Channel record layout (64 bytes):

Offset	Size	Field	Encoding / Notes
0-3	4	Rx Frequency	Little-endian 32-bit integer x 10 Hz (e.g., 0x002DC6C0 = 146.520 MHz). Stored as raw Hz/10 integer.
4-7	4	Tx Frequency	Same format as Rx Frequency
8-9	2	Rx Sub-Audio	See Sub-Audio encoding below
10-11	2	Tx Sub-Audio	See Sub-Audio encoding below
12	1	Signaling Code	Low nibble: index into DTMF code group list (0=none)
13	1	PTT ID	Low nibble: 0=Off, 1=BOT, 2=EOT, 3=Both
14	1	Channel Type flag 1	Low nibble: 0=Analog, 1=Digital
15	1	Channel Type flag 2	Low nibble: 0=DMR Tier I, 1=DMR Tier II (only if byte 14=1)
16	1	Tx Power	Low nibble: 0=Low, 1=Medium, 2=High
17	1	Scramble	Low nibble: 0-8 (scrambler type/code)
18	1	Encryption	Low nibble: 0=None, 1=Basic, 2=Enhanced, 3=AES
19	1	Busy Lockout	Low nibble: 0=Off, 1=On
20	1	Scan Add	Low nibble: 0=No, 1=Yes
21	1	Time Slot	Low nibble: 0=TS1, 1=TS2
22	1	Color Code	Low nibble: 0-15
23	1	Rx Group	Byte value: 0=None, 1-32=Rx group index
24	1	(reserved)	0xFF
25	1	Enc Key Index	Low nibble: index into encryption key list
26	1	DMR Mode	Low nibble: 0=Simplex, 1=Repeater
27	1	Learn FHSS	Low nibble: 0=Off, 1=On (Frequency Hopping Spread Spectrum)
28-31	4	FHSS Code	6 hex digits packed BCD, byte 31 = 0x00 if valid, 0xFF if unused
32-43	12	Channel Name	GB2312 encoded, null/0xFF terminated
44-45	2	Contact Index	Little-endian 16-bit: index into address book (0=none)
46-63	18	(padding)	0xFF

6.8.1 Frequency Encoding

Frequencies are stored as 32-bit little-endian integers representing the frequency in units of 10 Hz. For example, 146.520 MHz is stored as 14652000 (decimal) = 0x00DFC1A0, stored little-endian as A0 C1 DF 00.

The CPS validates frequencies against: VHF 18-300 MHz (exclusive of 300), UHF 300-1000 MHz. The step granularity is 10 Hz.

6.8.2 Sub-Audio (CTCSS/DCS) Encoding

The 2-byte sub-audio field encodes both CTCSS tones and DCS codes:

- OFF: Both bytes 0x00

- DCS code: Byte 0 = (DCS index + 1), Byte 1 = 0x00. The DCS table contains standard codes in order (D023N, D025N, etc.)
- CTCSS tone: Little-endian 16-bit integer = tone frequency x 10 (e.g., 88.5 Hz = 885 = 0x0375, stored as 75 03). Reconstructed by inserting decimal point before last digit.

6.8.3 FHSS Code Encoding

FHSS (Frequency Hopping Spread Spectrum) codes are 6 hex-digit values packed into 3 bytes (bytes 28-30), with byte 31 = 0x00 to signal a valid entry. The digits are stored in reverse order: byte 30 holds digits 1-2 (most significant), byte 29 holds digits 3-4, byte 28 holds digits 5-6 (least significant). Byte 28 ORed with the packed value.

6.9 VFO Mode Block (128 bytes, 1 packet)

The VFO block contains two VFO banks (A and B), each 64 bytes, using a layout nearly identical to the channel record. Differences from the channel format:

- Byte 27 is Step Frequency (0-7 index) rather than Learn FHSS
- Bytes 32-33 hold Contact index (little-endian 16-bit) instead of at bytes 44-45
- No ScanAdd, no ChPttId field (byte 13 unused)
- No channel name field

VFO A occupies bytes 0-63, VFO B occupies bytes 64-127. If the frequency bytes are all 0xFF or all 0x00, the CPS substitutes a default frequency.

6.10 Optional Functions Block (64 bytes, 1 packet)

This block is split into two 32-byte sub-regions. Part 1 (bytes 0-31):

Offset	Field	Values / Range
0	Analog Squelch	Low nibble, 0-9
1	Power Save	Low nibble, 0-4
2	VOX Level	Low nibble, 0-9 (0=Off)
3	Auto Backlight	Low nibble, 0-8
4	TDR (Dual Watch)	Low nibble, 0=Off, 1=On
5	TOT (Tx Time Out)	Low nibble, 0-5
6	Beep	Low nibble, 0=Off, 1=On
7	Voice Prompt	Low nibble, 0=Off, 1=On
8	Language	Low nibble, 0=English, 1=Chinese
9	Side Tone	Low nibble, 0-3
10	Scan Mode	Low nibble, 0=Time, 1=Carrier, 2=Search
11	PTT ID (global)	Low nibble, 0-3
12	ID Delay Time	Low nibble, 0-6
13	Display Mode A	Low nibble, 0=Freq, 1=Channel, 2=Name
14	Display Mode B	Low nibble, 0=Freq, 1=Channel, 2=Name
15	(reserved)	0xFF
16	Auto Lock	Low nibble, 0-3
17	SOS Mode	Low nibble, 0=Off, 1=Alarm, 2=TX

Offset	Field	Values / Range
18	Alarm Sound	Low nibble, 0=Off, 1=On
19	TDR Tx Priority	Low nibble, 0=Off, 1=Ch A, 2=Ch B
20	Tail Clear	Low nibble, 0=Off, 1=On
21	Repeater Tail Clear	Low nibble, 0-10 (value in 100ms units)
22	Repeater Detect Tail	Low nibble, 0-10
23	Tx-Over Sound	Low nibble, 0=Off, 1=On
24	Current Work Mode	Low nibble, 0=A, 1=B
25	FM Radio	Low nibble, 0=Off, 1=On
26	Work Mode A/B	Low nibble=A (0=VFO, 1=Channel), High nibble=B
27	Key Lock	Low nibble, 0=Off, 1=On
28	Boot Screen	Low nibble, 0=Off, 1=On
29	(reserved)	0xFF
30	R-Tone	Low nibble, 0-3
31	Tx Start Sound	Low nibble, 0=Off, 1=On

Part 2 (bytes 32-63):

Offset (within part 2)	Field	Values / Range
0	VOX Delay Time	Low nibble, 0-15
1	Menu Auto Quit	Low nibble, 0-10 (0=Off, others = seconds)
2	Digital Squelch	Low nibble, 0-9
3-5	(reserved)	0xFF
6	STE Frequency	Low nibble, 0=55Hz, 1=259Hz
7	Weather Channel	Low nibble, 0-9
8-9	(reserved)	0xFF
10	Top Key Short Press	Low nibble, 0-6 (function index)
11	Side Key 2 Short Press	Low nibble, 0-7
12	Side Key 2 Long Press	Low nibble, 0-6
13	Side Key 3 Short Press	Low nibble, 0-6
14	Side Key 3 Long Press	Low nibble, 0-6
15-16	(reserved)	0xFF
17	Keep Call Time	Bits 4:0, 0-19 (seconds)
18	(reserved)	0xFF
19	TDR Recovery Time	Low nibble, 0-10
20-31	(reserved)	0xFF

6.11 Basic Info Block (64 bytes, 1 packet)

The basic info block carries model identification data. Only a portion of the 64-byte block is used:

Offset	Size	Field	Encoding
0-7	8	(reserved/padding)	0xFF
8-19	12	Model Name	GB2312 encoded ASCII string, null/0xFF terminated
20-27	8	Model ID	ASCII decimal string, zero-padded to 8 characters
28-63	36	(reserved/padding)	0xFF

7. Total Data Volume Summary

For a full read or write session, the total payload bytes transferred (excluding frame overhead) are:

Block	Packets	Bytes/Packet	Total Bytes
Handshake	1	15	15
Password	1	6	6
Version	1	128	128
DTMF	1	272	272
Encryption Keys	1	264	264
Address Book	80	800	64,000
Rx Group List	4	1,024	4,096
Channel Mode	64	1,024	65,536
VFO Mode	1	128	128
Optional Functions	1	64	64
Basic Info	1	64	64
End/Over	1	2	2
TOTAL	157	-	134,575

At 115200 baud (approximately 11,520 bytes/second effective throughput), a full programming session transfers roughly 134 KB of payload data, which would take approximately 12-15 seconds at wire speed, plus round-trip latency for each of the 157 packet exchanges.

8. String / Text Encoding

Text strings in this protocol use two different encodings depending on context:

- Channel names, contact names, group names, model name: GB2312 (Guojia Biaozhun, Chinese national standard encoding). This allows both ASCII and Chinese characters to be stored. The CPS uses Encoding.GetEncoding("GB2312") for these fields.
- DTMF codes, FHSS codes, encryption keys: Custom nibble/index encoding where each byte represents a single character's position in a fixed character alphabet (e.g., "0123456789ABCD*#" for DTMF, "0123456789ABCDEF" for hex).
- Model ID: Plain ASCII decimal digits, zero-padded, 8 characters.
- Handshake string: Plain ASCII.

String termination: Strings are terminated by either a null byte (0x00) or 0xFF (unused padding). The CPS scans for either condition when reading strings back.

9. Radio Capacity Limits

The following limits are enforced by the CPS (derived from the Helper class constants):

Feature	Maximum Count / Value	Notes
Channels (total)	Determined by area x channel config	64 packets x 16 channels = 1024 channel slots
Areas	Multiple (exact from ChModeHelper)	Channels divided across areas
Contacts (Address Book)	4,000 (80 packets x 50/packet)	DMR IDs 1 to 16,776,415
Rx Groups	32 (4 packets x 8/packet)	Up to 32 members per group
Encryption Keys	8	Algorithms: None, Basic, Enhanced
DTMF Code Groups	15	Up to 6 digits per code
VFO Banks	2 (A and B)	Each with full channel-equivalent settings
Frequency Range (VHF)	18-300 MHz	Minimum 18 MHz, maximum below 300 MHz
Frequency Range (UHF)	300-1000 MHz	300 MHz to below 1000 MHz
DMR Color Codes	0-15	16 possible values
DMR Time Slots	0-1	TS1 and TS2

10. Implementation Notes for Protocol Emulation

For anyone wishing to implement a compatible programming tool or radio emulator, the following observations are important:

10.1 Byte Ordering

Frequencies and most integer values use little-endian byte order within payload data. The packet framing fields (sequence number, payload length) are big-endian. CRC bytes are also big-endian (high byte first).

10.2 Padding Conventions

Unused bytes in every block are filled with 0xFF before data is overlaid. This means that a byte value of 0xFF reliably indicates "not programmed" or "end of data" in list structures. Tools should treat 0xFF as a sentinel and stop parsing list entries when encountered.

10.3 Session Must Be Sequential

The protocol does not support random access to data blocks. The radio expects the exact command sequence defined in Section 5. Skipping steps or sending commands out of order is likely to cause the radio to NAK or hang, requiring a power cycle.

10.4 The NAK Response

If the radio responds with command byte 0xEE in any response frame, the CPS ignores the response data and waits for the retry timer. This means the CPS will retransmit the last request up to 3 times before giving up. An emulated radio should respond with 0xEE to indicate it cannot process a request.

10.5 End Session Command

The End/Over command (0x01) is critical. It signals the radio to exit programming mode and return to normal operation. The payload is 2 zero bytes. Failing to send this command (e.g., if the CPS crashes) will leave the radio in programming mode until it times out internally.

10.6 Address Book Write Optimization

During write operations, the address book data is sent in 80 consecutive packets. The CPS does not send a batch end marker - the transition to the next phase simply happens when packet counter 79 is reached. Each packet is independently acknowledged.

11. Annotated Example Frames

11.1 Handshake Request

Complete on-wire bytes for the handshake packet (23 bytes total):

```
A5 02 00 00 00 0F 50 52 4F 47 52 41 4D 4A 43 38 38 31 30 44 55 [CRC_HI] [CRC_LO]
```

Breakdown:

- A5 = Frame header (SOF)
- 02 = Command: CmdHandshake
- 00 00 = Sequence number 0
- 00 0F = Payload length 15
- 50 52 4F 47 52 41 4D 4A 43 38 38 31 30 44 55 = "PROGRAMJC8810DU" ASCII
- [CRC_HI] [CRC_LO] = CRC-16/CCITT over bytes 1-20

11.2 Password Request

Complete on-wire bytes for the password packet (14 bytes total):

```
A5 05 00 00 00 06 FF FF FF FF FF FF [CRC_HI] [CRC_LO]
```

Breakdown:

- A5 = Frame header
- 05 = Command: CmdCheckPwd
- 00 00 = Sequence 0
- 00 06 = Payload length 6
- FF FF FF FF FF FF = Default blank password
- [CRC_HI] [CRC_LO] = CRC over bytes 1-11

11.3 Channel Write Packet (first packet)

The first channel write packet (for packet index 0) begins with:

```
A5 30 00 00 04 00 [1024 bytes of channel data] [CRC_HI] [CRC_LO]
```

Breakdown:

- A5 = Frame header
- 30 = Command: CmdWriteChMode (0x30 = 48)
- 00 00 = Sequence/packet 0 (first of 64)
- 04 00 = Payload length 1024 (0x0400 big-endian)
- [1024 bytes] = Channel data for channels 0-15
- [CRC_HI] [CRC_LO] = CRC over bytes 1 through 1029

Appendix: File Save Format

The CPS saves and loads radio configuration to .cpf files using .NET BinaryFormatter serialization of the RadioData class. This is a proprietary binary format tied to the C# class structure and is not directly portable. The on-disk format is NOT the same as the on-wire protocol format.

The RadioData object contains instances of: BasicInfoData, ChModeData, VfoModeData, AddrBookData, RxGroupData, EncKeyData, OptionalFunData, and DtmfData. The file is a serialized object graph of these classes.

End of Report