



CPAL Link Protokolü: v1.0 (sözleşme (3))

Axion Robot *CircuitPal V2*

Özet—Bu belge CircuitPal V2 eğitim robotunun teknik sözleşmelerinden birini IEEE dergi biçiminde sunar. İçerik depodaki Markdown kaynağından üretilmiştir; kaynak ile bu çıktı arasında bir fark görülürse depo kazanır.

Index Terms—CircuitPal, ESP32-S3, eğitim robotu, gömülü yazılım, Axion Robot

Sahibi: WS-C · Faz 4 · Değiştirmek için ADR + STATE .md'de duyuru (AGENTS K2). Bu dokümanı okuyan biri **kodu görmeden** istemci yazabilmeli: sayısal tablolar burada, C başlıklarına bakmak gerekmez. Üç implementasyon aynı vektörleri geçer: Python (tools/protocol/vectors_gen.py, üreteç), C (cpal_link), JS (tools/protocol_js). Taşımadan bağımsız: USB-CDC (Faz 4), BLE-NUS (Faz 7), host sahte taşıyıcı (test). **Kapsam notu (sözleşmeyi değiştirmez):** kartın WiFi HTTP yüzeyi (apps/studio/src/link/wifi-transport.mjs ↔ cpal_wifi_ota; düz POST /program, POST /vm, POST /ota) bu sözleşmenin **dışındadır**, COBS/CRC çerçevelemesi yoktur, telemetri ve değişken aboneliği yoktur. "Tek yol seri" sanılmasın diye yazıldı.

I. ÇERÇEVELEME

Bayt akışı üzerinde her çerçeve **COBS** ile kodlanır ve 0x00 ile biter.

Alıcı durum makinesi: 0x00 görene kadar biriktir. Biriken bayt sayısı MAX_FRAME (1040) 0x00 hariç sınırını **aşarsa** biriktirmeyi bırak, bir sonraki 0x00'a kadar her şeyi at (**1 atma sayılır**). 0x00 gelince: biriken **boş** ise (ardışık 0x00) sessizce yut, **sayma**; değilse çöz. Çözüm hatası (COBS/len/CRC) = **1 atma**, yanıt yok. Bu sayede kablo çekilip takılınca ya da çöp gelince akış kendini toparlar.

COBS: standart. **Sınır varyantı sabitlenir:** girdi tam 254'lük sıfırsız bir blokla bitiyorsa kodlayıcı sonda 0x01 kod baytı **yazar** (üreteç böyle yapar); çözücü her iki biçimi de (sondaki 0x01 var/yok) kabul eder.

COBS çözülmüş **paket** (little-endian):

Çözücü önce len'i okur: 4 + len + 2 paket boyuyla tutmuyorsa ya da len > MAX_PAYLOAD ise **BAD_LEN** (COBS çözümü tamamlanmadan da karar verilebilir); paket 6 bayttan kısaysa **TOO_SHORT**; CRC tutmazsa **BAD_CRC**. Payload'da çok baytlı sayılar little-endian; dizeler [u8 len][bayt...] (UTF-8, \0 yok). **Şekil ihlali** (payload beklenen boyda değil, alan aralık dışı) her zaman ERROR BAD_LEN, klemp yapılmaz.

TABLO 1

Ofset	Boyut	Alan
0	2	len, payload bayt sayısı (0..MAX_PAYLOAD = 1024)
2	1	seq, 0..255 sarmalı
3	1	type, §3
4	len	payload
4+len	2	crc16, CRC-16/CCITT-FALSE (poly 0x1021, init 0xFFFF, refin/refout yok, xorout 0; kontrol: "123456789" → 0x29B1), seq, type, payload üstünden (len hariç)

II. OTURUM VE SIRA NUMARASI

- **Tip uzayı:** host istekleri 0x01--0x5F; yanıt type | 0x80 (0x81--0xDF); cihaz-kaynaklı 0xE0--0xEF; ERROR 0xFF. 0x60--0x7F **rezerve** (yanıtları cihaz-kaynaklı aralığa düşer, bu yüzden kullanılmaz).
- **İstek/yanıt:** her isteğe aynı seq ile tam bir yanıt ya da ERROR. Host seq'i 0'dan başlatır, her yeni istekte +1.
- **Tekrar-yanıt (idempotens):** cihaz son işlediği isteğin (seq, type) çiftini ve yanıtını saklar. Aynı (seq, type) tekrar gelirse istek **yeniden işlenmez**, saklı yanıt yollanır. HELLO bu kuralın **dışındadır**: her zaman taze işlenir.
- **Zaman aşımı (host önerisi):** yanıt 500 ms'de gelmezse aynı (seq, type) ile en çok 3 tekrar; sonra oturum yeniden (HELLO).
- **Oturum:** HELLO ile açılır; HELLO gelmeden gelen diğer istekler ERROR BAD_STATE. **Her HELLO oturumu sıfırlar:** yarım program yükleme atılır, abonelikler ve TRACE kapanır, tekrar-yanıt önbelleği silinir, PAUSED ise program durdurulur (HALTED); koşan (RUNNING) program **durmaz**.
- **Kopma:** taşıyıcı read < 0 döndürürse rx birikimi temizlenir, oturum kapanır (HELLO beklenir), abonelik/TRACE/yarım yükleme atılır; PAUSED ise STOP uygulanır (robot yarı-durumda kalmasın); RUNNING program **durmaz**, çocuk kabloyu çekince robot koşturmaya devam eder. Sonraki read ≥ 0 yeni bir bağlantı sayılır.
- **Cihaz-kaynaklı** çerçeveler cihazın kendi seq sayacını taşır; host yanıt vermez.

III. KOMUT SETİ

board_id: 6 bayt (cihaz MAC; host'ta 00:00:00:00:00:00:01).

A. Komut × durum tablosu

vm_state değerleri (§4). Boş hücre = izinli; BAD_STATE = ERROR BAD_STATE.

TABLO 2

type	Ad	İstek payload	Yanıt payload
0x01	HELLO	[proto_major u8][proto_minor u8]	[proto_major u8][proto_minor u8][format_major u8][opcode_hash u32][board_id 6B][fw_version str][caps u16], caps: bit0 OTA, bit1 STORAGE, bit2 TRACE
0x02	PING	herhangi (≤ 64 B)	aynı payload
0x10	PROGRAM_BEGIN	[total_len u16][crc32 u32][opcode_hash u32]	[chunk_max u16][window u8] (v1: 512, 1)
0x11	PROGRAM_CHUNK	[offset u16][data...]	[next_offset u16]
0x12	PROGRAM_END	-	[0] (yüklendi, READY), her ret ERROR (§5)
0x13	RUN	-	[vm_state u8]
0x14	STOP	-	[vm_state u8]
0x15	PAUSE	-	[vm_state u8]
0x16	STEP	-	[vm_state u8][pc u16]
0x20	VAR_SUBSCRIBE	[period_ms u16][n u8][idx u8 $\times$ n] ($n \leq 16$, period ≥ 50 ; n=0 iptal)	[n u8] (kabul edilen; $idx \geq n_vars$ atılır)
0x21	TRACE	[enable u8] (0/1)	[enable u8]
0x30	STATUS	-	[vm_state u8][vm_fault u8][pc u16][loop_count u32][batt_mv u16][batt_state u8][thermal u8][ntc_c10 i16][uptime_ms u32][runaway u32][rx_dropped u16]
0x40	PROGRAM_SAVE	[slot u8][name str] (slot < MAX_SLOTS, name ≤ 24 B)	[slot u8]
0x41	PROGRAM_LOAD	[slot u8]	[0], her ret ERROR
0x42	PROGRAM_LIST	-	[n u8][(slot u8, size u16, name str) $\times$ n] (yalnız dolu slotlar)
0x50	CAL_READ	[key str] (key ≤ 15 B)	[blob...] (≤ 512 B)
0x51	CAL_WRITE	[key str][blob...] (blob ≤ 512 B)	-
0x58	OTA_BEGIN	[size u32][crc32 u32]	-
0x59	OTA_CHUNK	[offset u32][data...] (≤ 512 B)	[next_offset u32]
0x5A	OTA_END	-	, (cihaz yeniden başlar)
0xE0	EVENT	[event u8][arg i32]	-
0xE1	VAR_UPDATE	[t_ms u32][n u8][(idx u8, value i32) $\times$ n]	-
0xE2	TRACE_TICK	[t_ms u32][pc u16][loop_count u32][ops_last u16] (her LOOP_END)	-
0xFF	ERROR (istek seq'iyle)	[req_type u8][err u8][detail i32][msg str]	-

TABLO 3

İstek	IDLE	READ	RUNNING/SLEEPING	PAUSED	HALTED	FAULT
RUN	BAD_STATE	BAD_STATE	no-op (RUNNING)	sürdür	sıfırdan yeniden başlat (imaj korunur)	sıfırdan yeniden başlat
STOP	BAD_STATE	BAD_STATE	durdur $\rightarrow$ HALTED, motorlar kesilir, bloklayıcı çağrı kesilir	$\rightarrow$ HALTED	no-op	no-op
PAUSE	BAD_STATE	BAD_STATE	PAUSED, motorlar kesilir; sürmekte olan bloklayıcı çağrı kesilmez (en çok o çağrının süresi kadar gecikir, ≤ 60 s)	no-op	BAD_STATE	BAD_STATE
STEP	BAD_STATE	BAD_STATE	BAD_STATE	tek tick koşar, PAUSED kalır	BAD_STATE	BAD_STATE
PROGRAM_BEGIN			önce STOP etkisi	önce STOP etkisi		
PROGRAM_SAVE	BAD_STATE					
PROGRAM_LOAD			önce STOP etkisi	önce STOP etkisi		
VAR_SUBSCRIBE/TRACE/STATUS/PING/CAL_*						

IV. SAYISAL TABLOLAR (DONMUŞ)

vm_state: 0 IDLE · 1 READY · 2 RUNNING · 3 SLEEPING (bloklayıcı çağrıda) · 4 HALTED · 5 FAULT · 6 PAUSED (link katmanı). vm_fault: 0 NONE

· 1 STACK_OVERFLOW · 2 STACK_UNDERFLOW · 3 DIV_ZERO · 4 BAD_VAR · 5 BAD_POOL_INDEX. load_err (ERROR VM_LOAD.detail): 0 OK · 1 BAD_MAGIC · 2 BAD_FORMAT · 3 BAD_HASH ·

4 BAD_CRC · 5 TOO_BIG · 6 BAD_LAYOUT · 7 BAD_OPCODE · 8 BAD_JUMP · 9 BAD_ARG · 10 BUSY.
 batt_state: 0 NORMAL · 1 LOW · 2 CRITICAL · 3 SHUTTING_DOWN. thermal: 0 NORMAL · 1 FAN_ON · 2 CRITICAL. EVENT.event: 1 LOW_BATTERY (arg mV) · 2 CRITICAL_BATTERY · 3 SHUTTING_DOWN · 4 PICKED_UP · 5 VM_FAULT (arg vm_fault) · 6 VM_STALL · 7 THERMAL (arg c10) · 8 PROGRAM_HALTED · 9 PROGRAM_LOADED (arg slot; link yüklemesi -1). ERROR.err: 1 UNKNOWN_TYPE · 2 BAD_LEN (şekil/aralık) · 3 BAD_STATE · 4 HASH_MISMATCH (detail = cihaz hash'i → "robotun yazılımı editörden eski") · 5 CRC_MISMATCH · 6 TOO_BIG · 7 UNSUPPORTED · 8 STORAGE · 9 VM_LOAD (detail = load_err) · 10 BAD_OFFSET (detail = beklenen offset) · 11 BUSY.

Sessiz başarısızlık yoktur: her istek yanıt ya da ERROR alır. Çerçeve düzeyinde atılanlar STATUS.rx_dropped'da sayılır.

V. PROGRAM YÜKLEME

crc32 = **imajın tamamı** (total_len bayt, gömülü bytecode CRC'si dahil) üstünden IEEE CRC-32 (taşıma bütünlüğü). VM'in gömülü CRC'si ayrıca cpal_vm_load içinde doğrulanır.

Gerekçe (window=1): ESP32-S3 CDC tamponu + çocuğun kablo çekmesi; her chunk tek başına tekrar edilebilir, kayıp yanıt (seq, type) idempotensiyle tekrar alınır. window yanıtta taşınır, ileride büyütülebilir.

VAR_UPDATE: period_ms'de bir, **son gönderilen** değere göre değişenler (ilkinde hepsi). Taşıyıcı write 0 dönerse (şimdi sığmaz) o periyot atlanır, değişiklik bir sonraki periyotta gider; rx_dropped sayılmaz. TRACE_TICK her LOOP_END'de; sığmazsa atlanır.

```

PROGRAM_BEGIN(total_len, crc32, hash)
    hash ≠ cihaz          → ERROR HASH_MISMATCH (detail = cihaz hash'i)
    total_len < 28 ya da > MAX_IMAGE → ERROR TOO_BIG
    yükleme sürüyorsa      → öncekini at, yeniden başla (BUSY değil)
PROGRAM_CHUNK(offset, data) dur-ve-bekle (window=1): yanıt gelmeden sonraki gönderilmez
BEGIN yok                → ERROR BAD_STATE
len(data) > chunk_max    → ERROR BAD_LEN
offset ≠ beklenen        → ERROR BAD_OFFSET (detail = beklenen; host oradan sürer)
offset + len > total_len → ERROR TOO_BIG
PROGRAM_END
BEGIN yok                → ERROR BAD_STATE
eksik bayt              → ERROR BAD_OFFSET (detail = beklenen offset)
crc32 ≠                  → ERROR CRC_MISMATCH
cpal_vm_load ≠ OK        → ERROR VM_LOAD (detail = load_err)
→ yanıt [0], durum READY, EVENT PROGRAM_LOADED(-1)

```

TABLO 4

Sabit	Değer
PROTO_	1 / 0
MAJOR/MINOR	
MAX_PAYLOAD	1024
MAX_PACKET	1030 (4 + 1024 + 2)
MAX_FRAME	1040 (COBS'lu, 0x00 hariç; 1040 kabul, 1041 atılır)
CHUNK_MAX	512
MAX_SUBS	16
MIN_VAR_	50
PERIOD_MS	
MAX_IMAGE	10308 (64 + 2048 + 8192 + 4, header_len büyümesine pay)
MAX_SLOTS	4
MAX_NAME	24 B · MAX_KEY 15 B · MAX_BLOB 512 B

VI. SINIRLAR VE SABİTLER

Kalıcı kapasite garantisi: cihazdaki dört slotun her biri aynı anda MAX_IMAGE bayt tutabilir. Ürün adaptörü ayrılmış storage LittleFS bölümünü kullanır; NVS bu sözleşme için yetersizdir. Her değer geçici dosyaya yazılıp atomik rename ile yayınlanır. Kalibrasyon anahtarları dosya yoluna ham konmaz, hex kodlanır.

VII. ENJEKTE ARAYÜZLER (CIHAZ/HOST FARKI)

transport.read(buf, cap): >0 bayt, 0 yok, <0 **kopuk** (§2). transport.write(buf, len): len yazıldı, 0 **şimdi sığmaz** (çerçeve atlanır, kopuk değil), <0 kopuk. Kısmi yazım yoktur: adaptör ya bütün çerçeveyi tamponlar ya 0 döner. **Cihaz görev kuralı (Faz 6/7):** link istekleri VM task'ının bağlamında işlenir (kuyruk); Faz 4 host'ta cpal_link_poll runtime kuantumundan çağrılır, kilit yok.

Ürün kuantumu sırası: cpal_runtime_tick → cpal_link_poll → cpal_link_vm_step → 10 ms gecikme → heartbeat → taşıyıcı pump. USB-CDC yalnız ikili protokole ayrılır; log/konsol UART0'da kalır.

VIII. TEST VEKTÖRLERİ

tools/protocol/vectors.json, Python üretici üretir; C(link_vectors.inc) ve JS aynı dosyayı geçer; bayatsa ctest kırmızı. Çerçeve vektörleri: geçerli (HELLO, PING, PROGRAM_*, OTA_*, VAR_SUBSCRIBE, STATUS, ERROR, boş, 1024 B, 0x00'lı payload, 254/255 blok sınırı, 254-blokla biten paket + 0x01'siz çözüm-only biçimi) + negatif (bozuk CRC, len alanı yanlış, len > MAX,

çok kısa, COBS bozuk). Akış vektörleri: ardışık iki çerçeve, çöp + çerçeve, iki read'e bölünmüş çerçeve, bozuk + iyi, MAX_FRAME = 1040 / 1041 kenarı, öndeki 0x00'lar. **Oturum senaryoları** (C test_link_session, JS istemci simülasyonu): HELLO'suz istek → BAD_STATE; aynı (seq,type) tekrar → aynı yanıt, işlem tekrarlanmaz; HELLO tekrarı taze işlenir; kopma + yeniden HELLO; BEGIN'siz CHUNK; offset taşması; hash uyuşmazlığı; tam yükleme → RUN → VAR_UPDATE akışı → PAUSE/STEP → STOP → RUN (yeniden başlar).

IX. SÜRÜMLEME

proto_major uyuşmazsa HELLO'ya ERROR UNSUPPORTED; minor geriye uyumlu ekleme (yeni tip / yanıt sonuna alan). opcode_hash uyuşmazlığı HELLO'da **hata değildir** (host bilgilendirir), PROGRAM_BEGIN'de hatadır.