F998 – Documentación general (v1.0 revisada)

Estado: base estable · nomenclatura corregida · referencia normativa

---

1. Filosofía del proyecto

El **F998** es un **panel programable con inteligencia distribuida**:

- El **firmware (Arduino)** controla hardware, lecturas, LEDs y animaciones simples.
- El **host (Python)** define la lógica, los modos de trabajo y las aplicaciones.
- La comunicación se realiza **exclusivamente por puerto serie**, usando comandos ASCII.

El dispositivo **no es tonto**, pero tampoco intenta ser una aplicación completa.

---

2. Convenciones de nomenclatura (CRÍTICO)

Toda la API —firmware, protocolo y Python— utiliza **español técnico coherente**.

❌ Nombres incorrectos (NO usar)

- `battery()`
- `get()`
- `in_key()`

✅ Nombres correctos (OBLIGATORIOS)

- `bateria()`
- `estado()`
- `tecla()`

> ⚠️ La documentación es normativa: el código debe adaptarse a ella.

---

3. Rangos de valores

Símbolo	Rango	Descripción
`n`	0..39	número de botón
`a`	1..9	potenciómetro
`v`	0..3	vúmetro
`c`	0..8	columna
`i`	0..6	segmento
`j`	0..9	grid

Símbolo	Rango	Descripción
`p`	0..100	porcentaje
`mask`	0..15	máscara de animación

---

4. API del firmware (Arduino)

4.1 Inicialización

```
void F998_setup();
```

4.2 Lectura de entradas

```
int tecla();            // última tecla pulsada (0 = ninguna)
int pot(int a);         // potenciómetro a (0..100)
int digPot(int a);      // potenciómetro discreto (0..6)
```

4.3 Matriz / vúmetros

```
void vumetro(int v, int p);
void vumetroR(int v, int p);
void horizontalPos(int v, int c);
void verticalBar(int c, int p);
void zoom(int c);
void vumetroClear();
void vumetroFull();
```

4.4 Batería

```
void bateria(int p);
void bateriaR(int p);
void bateriaPos(int b, bool s);
void bateriaBlink(int b, bool s);
void bateriaClear();
```

4.5 LEDs de botones

```
void ledButton(int n, bool s);
void ledBlink(int n, bool s);
void ledInvert(int n);
```

4.6 Bajo nivel (matriz)

```
void ledAt(int i, int j, bool s);
void ledAtBlink(int i, int j, bool s);
```

4.7 Animaciones

```
void animModo(uint8_t mask);
```

---

5. Protocolo de comunicación serie

- Texto ASCII
- Un comando por línea
- Parámetros separados por espacios

Ejemplos

```
IN
GET
LB 27 1
LBB 27 1
BAT 50
BATR 25
ZM 4
VC
```

Respuestas

- `K <n>` → tecla pulsada
- `S P=.. D=..` → estado de potenciómetros
- `OK` → comando aceptado

---

6. Principios de diseño

• El firmware **no mantiene estado de aplicación**.
• Python es responsable de:
• modos
• seguridad
• semántica
• Los LEDs **comunican estado**, no decoran.

---

7. Nota importante sobre modos

Antes de entrar en un modo:

• Todos los `pot()` deben estar a **0**
• Todos los `digPot()` deben estar **centrados (3)**

Si no se cumple: - El modo **no se activa** - Se indica el error en la **matriz 4×9** (columna = control incorrecto)

Fin del documento – F998 Documentación v1.0 revisada