

## GPIO

- `INPUT PULL-UP` : pin HIGH par défaut → interrupteur vers GND.
- `INPUT PULL-DOWN` : pin LOW par défaut → interrupteur vers VCC.
- `ANALOG INPUT` : lecture de tension (0 - Vref).
- `OUTPUT PUSH-PULL` : sortie numérique.

## UART

- Câblage : `TX1 <-> RX2` , `TX2 <-> RX1` , `GND <-> GND` .
- Trame : `START(0)` → 8 bits (LSB first) → `STOP(1)`.
- Vérifier : nécessite la même configuration (baud, parité, stop bits).

## C

- `x--` : évalue puis décrémente (`while(x--)` affiche 4..0 si `x=5`).
- `--x` : décrémente puis évalue.
- `#define NAME value` (PAS de `;`).
- `static` local conserve valeur entre appels.
- Globaux : déclarer dans `.h` avec `extern` redéclarée dans un seul `.c` pour l'allocation mémoire.
- `printf` sur STM32 : rediriger via `int __io_putchar(int ch)` ; flottants non inclus par défaut.
- Utiliser `"\n\r"` si affichage série requis.

## Pointeurs / Structs

- `&` adresse, `*` déréférencement.
- Déclarer et initialiser : `int *p = NULL;` .
- `.` pour variable, `->` pour pointeur vers structure.

## HAL (rapide)

- Lire pin : `HAL_GPIO_ReadPin(GPIOx, PIN)` → `GPIO_PIN_SET` (1) / `GPIO_PIN_RESET` (0).
- Écrire pin : `HAL_GPIO_WritePin(GPIOx, PIN, GPIO_PIN_SET/RESET)` .
- Attention : `HAL_Delay()` est bloquant (le programme s'arrête pendant ce temps).

## Exemples courts

- Blink 500ms :

```
HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, GPIO_PIN_SET);
HAL_Delay(250);
HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, GPIO_PIN_RESET);
HAL_Delay(250);
```

## Chaîne de compilation

- `*.c` + `*.h` → préprocesseur → compilation → `.s` → assemblage → `.o` → linkage → `.elf` / `.hex` .