# Robot Arm Remote + Firmware (README)

This repo contains two Arduino sketches that work together:

1. **Remote Control** – a handheld controller that sends single-letter commands over an **HC-05** Bluetooth module. It supports:
    - **Gesture mode (FLEX):** one flex sensor to open/close the gripper.
    - **Manual mode (MANUAL):** control via **push-buttons** *or* **potentiometers**(buttons are default at the moment).
2. **Robot Arm Firmware** – listens for those single-letter commands and moves the arm using a **PCA9685** servo driver (HCPCA9685 library) and a **stepper** base (DIR/STEP).

Both sketches use **4800 baud** over Bluetooth/Serial.

## Contents

- remote_control/remote_control.ino
  One flex sensor + HC-05 + mode switch, with manual control via either buttons or pots.
- robot_arm/robot_arm.ino
  PCA9685-driven joints + optional base stepper. Parses single-letter commands from the remote.

## Command Protocol (one character per action)

The remote transmits these letters; the robot firmware responds accordingly:

| Letter | Meaning (Robot Action) |
|---|---|
| **L** | Base rotate **Left** (stepper) |
| **R** | Base rotate **Right** (stepper) |
| **G** | **Shoulder Up** (backward) |
| **U** | **Shoulder Down** (forward) |
| **C** | **Elbow Up** (backward) |
| **c** | **Elbow Down** (forward) |
| **F** | **Gripper Open** |

**f** **Gripper Close**

The arm firmware also accepts some **legacy aliases** that were in previous drafts of code:
S/O for base L/R, C/c for shoulder up/down, P/p for elbow up/down, R/L wrist2 CW/CCW, G/U wrist1 down/up.
With the new remote, R/L are used for **base**, so wrist2 won't be reachable from the remote unless new letters are mapped later on.

# Remote Control Sketch

## Features

- **Two modes via a toggle on pin 2**
  - **FLEX mode (LOW):** one flex sensor on A0 auto-calibrates once, then sends:
    - Above high threshold → **F** (open)
    - Below low threshold → **f** (close)
  - **MANUAL mode (HIGH):** choose *buttons* (default) or *pots*:
    - **Buttons** (recommended first): press/hold to send jog commands at a fixed rate.
    - **Pots**: push past a deadband to send positive/negative jogs; center = idle.
- **Rate limiting** so holding a button doesn't spam too quickly (set by **response_time**).

## Wiring (default)

- **Mode switch** → pin **2** (LOW = FLEX, HIGH = MANUAL). Use a pull-down or set the pin to **INPUT_PULLUP** and invert.
- **HC-05**: TX→Arduino RX0, RX→Arduino TX0 (divider to 3.3V on HC-05 RX). Baud **4800**.
- **Flex sensor**: A0 with a voltage divider (e.g., 22k–47k to GND).

**If using BUTTONS** (default manual input):

**Base L -> D4 sends 'L'**
**Base R -> D5 sends 'R'**
**Shoulder Up -> D6 sends 'G'**
**Shoulder Down -> D7 sends 'U'**
**Elbow CW -> D8 sends 'C'**
**Elbow CCW -> D9 sends 'c'**
**Grip Open -> D10 sends 'F'**
**Grip Close -> D11 sends 'f'**

Wire buttons to **5V** and use **INPUT_PULLDOWN** (or wire to GND and use **`INPUT_PULLUP`** then invert logic).

**If using POTS** (enable in code):

**Base -> A1 (>HI='R', <LO='L')**
**Shoulder-> A2 (>HI='G', <LO='U')**
**Elbow -> A3 (>HI='C', <LO='c')**
**Gripper -> A4 (>HI='F', <LO='f')**

Adjust **POT_LO/POT_HI** deadband in code.

### Customizers

- **response_time** sets how often a held input transmits (default ~100 ms).
- Flex thresholds auto-set once; tune multipliers inside **flexModeLoop()** if needed.

# Robot Arm Firmware Sketch

### Hardware

**PCA9685** (HCPCA9685 library) controlling servos. Default channel mapping:

**CH 0: Base Left Servo (if using opposing servos – not active in the stepper base setup)**
**CH 1: Base Right Servo (if using opposing servos – not active in the stepper base setup)**
**CH 2: Shoulder**
**CH 3: Wrist1 (legacy demo)**
**CH 4: Wrist2 (legacy demo)**
**CH 5: Gripper**

- **Base Stepper** (enabled in this sketch):
  **dirPin = D4, stepPin = D5** (adjustable).
  Each **L/R** command performs a short “jog” rotation (steps set in **stepsPerRevolution** & delays).

### Motion Functions (Kept from previous draft)

- Shoulder, elbow, wrist, and gripper motion are **incremental** jogs with min/max clamps.
- Position variables end in **_parking_pos_i** and are updated each jog.
- A **rate limiter** (~25 Hz) prevents excessive Serial spam when a button is held.

### Serial / Bluetooth

- **4800 baud** (**Serial.begin(4800)**).

- The main loop reads the latest incoming byte and maps it via the table above.

### Safety & Limits

- Min/max bounds per joint are kept from previous draft.
    - Shoulder up to ~400 units (library scale)
    - Elbow/Wrist up to ~380
    - Gripper up to ~120
- **Tune increments** (**servo_joint_*_pos_increment**) for smoothness (e.g., 4–10) and adjust **stepDelay** for base speed.

## Tuning Cheatsheet

- **Jog Rate:**
  Remote: **response_time** (how often a held input transmits).
  Arm: **actionIntervalMs** (how often actions are applied).
- **Jog Size (smoothness):**
  **servo_joint_*_pos_increment** (lower = smoother, more presses).
- **Base Speed (stepper):**
  **stepDelay** (µs per half-step). Lower = faster; look out for torque/driver limits.
- **Gripper End-Stops:**
  **servo_joint_4_min_pos / servo_joint_4_max_pos** to prevent binding.

## Troubleshooting

- **If Nothing moves:** check HC-05 pairing and that both sketches use **4800 baud**.
- **If Wrong joint moves:** verify PCA9685 channel wiring matches the sketch.
- **If Jitter or overshoot:** reduce increments and/or increase rate limiting.
- **If Wrist not responding from remote:** with the new mapping, **R/L** are used for **base**. Wrist functions remain in firmware (legacy letters), or assign new wrist letters if needed.

## Quick Reference

**Remote = Arm Commands**

**L = Base Left**
**R = Base Right**
**G = Shoulder Up**
**U = Shoulder Down**
**C = Elbow Up**
**c = Elbow Down**
**F = Gripper Open**
**f = Gripper Close**

**Key Pins**

**Remote:**
- **HC-05: TX→RX0, RX→TX0 (voltage-divide to 3.3V), 4800 baud**
- **Mode Switch: D2 (LOW=FLEX, HIGH=MANUAL)**
- **Flex: A0 (with divider)**
- **Buttons: D4..D11 (default) OR Pots: A1..A4 (optional)**

**Robot Arm:**
- **PCA9685 I2C addr: 0x40**
- **Channels: Base L(0), Base R(1), Shoulder(2), Wrist1(3), Wrist2(4), Gripper(5)**
- **Base Stepper: DIR=D4, STEP=D5**