MODULE 03 FIELD EXERCISE

Tamper Tripwire: Accelerometer to Pi to Relay to Buzzer

TASK: Build and arm a self-powered tamper node: the accelerometer feeds the Pi over I2C, the Pi decides, and the relay sounds the isolated buzzer when the device is handled.

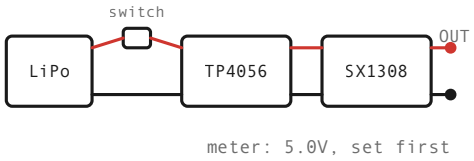
CONDITION: Given the proven isolated switch, the enrolled controller, the ADXL345, the boost pack with its output set to 5.0 volts before it touches the Pi, a breadboard and jumpers, the buzzer with its own supply, and a multimeter.

STANDARD: The node runs from the pack; i2cdetect shows the sensor at 0x53; armed, a bump on the bench sounds the buzzer for the set time and the node re-arms; continuity between the Pi's ground and the buzzer supply's negative reads OL; every connection matches your drawing; and the finished breadboard is photographed next to its drawing for your build log.

> **SAFETY FIRST**
>
> This exercise never touches mains. The switched load stays a low-voltage buzzer on its own supply; keep it that way. When the pack powers the system, apply the cell rules from Unit 2: switch off before rewiring, keep anything metal away from the cell terminals, and stop if the cell gets warm.

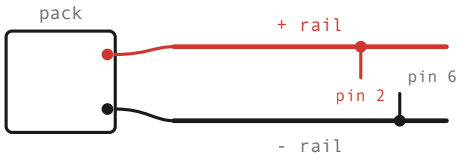
Task 1 · Build and set the pack

1. Wire the cell's positive lead through the slide switch to the TP4056's `B+` pad, and the negative lead to `B−`.
2. Feed the SX1308's input from the same pads: `B+` to `IN+`, `B−` to `IN−`.
3. Switch the pack on with nothing on its output. Set the meter to DC volts across `OUT+` and `OUT−`.
4. Turn the SX1308's trimmer until the meter reads 5.0V. The module is set first and loaded second; whatever it is set to is what the Pi gets.



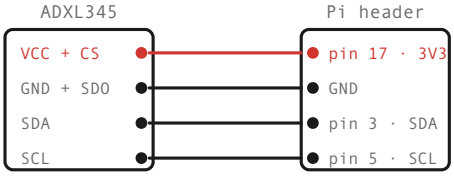
Task 2 · Rails and the Pi

1. Switch the pack off. Wire its output to the breadboard: `OUT+` to the + rail, `OUT−` to the - rail.
2. Jumper Pi pin `2` (5V) to the + rail and pin `6` (GND) to the - rail.
3. Switch the pack on. The Pi boots from the cell; reach it from the laptop over SSH as in Unit 3.
4. Meter across the rails while the board runs: the pack holds 5.0V under load.



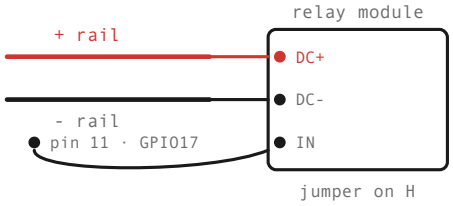
Task 3 · Wire and prove the sensor

1. Power down. Seat the ADXL345 and run its jumpers: `VCC` and `CS` to 3.3V (pin `17`), `GND` and `SDO` to ground, `SDA` to pin `3`, `SCL` to pin `5`.
2. Power up. Enable the bus once: `sudo raspi-config`, Interface Options, I2C.
3. Install the tools: `sudo apt install -y python3-smbus i2c-tools`.
4. Scan the bus: `i2cdetect -y 1`. A `53` in the grid is the ADXL345; SDO tied low sets that address.



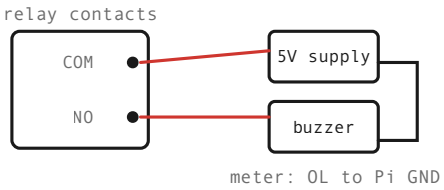
Task 4 · Wire and prove the relay

1. Power down. Wire the relay module: `DC+` to the + rail, `DC−` to the - rail, `IN` to Pi pin `11` (GPIO17).
2. Set the board's trigger jumper to `H` (high level trigger).
3. Power up. Drive the pin from the SSH session: `pinctrl set 17 op dh` to close, `pinctrl set 17 op dl` to release.
4. The relay clicks on high and releases on low, and the board's indicator LED follows.



Task 5 · Wire the isolated buzzer and prove the isolation

1. Power everything down, buzzer supply included.
2. Wire the buzzer supply's positive to the relay's `COM` screw terminal.
3. Wire the relay's `NO` terminal to the buzzer's + lead, matching the + printed on the buzzer, and the buzzer's - lead back to its supply's negative. `NC` stays empty.
4. Set the meter to continuity. Probe from the buzzer supply's negative to the Pi's - rail: the meter reads OL, no tone, because the two circuits share nothing.



Task 6 · Arm it

1. Copy `tamper.py` from the module page into the SSH session.
2. Run it: `python3 tamper.py`. It starts the sensor measuring, records a baseline, and prints `armed` with the baseline vector.
3. Bump the bench. The script prints `ALARM`, the relay closes, and the buzzer sounds for five seconds; then the node re-arms on a fresh baseline.
4. Leave it armed and hands off for a full minute: a still node stays silent. If it false-alarms, raise `THRESH` in the script and re-run.

Task 7 · Field notes

Log the threshold you settled on, any connection that did not match your drawing on the first try, and the rail voltage while the buzzer fires. Photograph the finished breadboard next to its drawing for your build log.

#	Task	Meter Setting	Expected	Your Reading	Pass / Fail
1	Pack output	DC Volts	5.0V unloaded, before the Pi touches it		
2	Rails under load	DC Volts	~5.0V across + and - while the Pi runs		
3	I2C scan	`i2cdetect -y 1`	Device answers at 0x53		
4	Relay coil	Listen / LED	Clicks on high Releases on low		
5	Isolation	Continuity	OL / no tone between buzzer circuit and Pi GND		
6	Armed node	Observe	Bump sounds buzzer 5 s; still node stays silent		