**Per-board hardware RESET (Pi pulls each Arduino’s RESET pin low)**

* Resets only the bad board; fast; no power rail glitches; preserves I²C pull-ups.
* Needs 1 GPIO per board (or a GPIO expander).

Replace reset-arduino() with resetSwitchModule(addr)

Short answer: **those two (relay XC4419 / MOSFET XC4526) are for cutting the 5 V power rail** (global/group power-cycle).  
For **per-board hardware RESET**, you don’t need them. You just need a tiny transistor per Arduino to pull its **RESET** pin low.

**Per-board RESET (what to use)**

* **1× GPIO per Arduino** (or a GPIO expander like MCP23017 if you’re short on pins)
* **1× small NPN transistor** (e.g., 2N3904/BC547) *or* **small N-MOSFET** (e.g., 2N7000/BSS138)
* **Resistors:**
  + NPN: ~1 kΩ from Pi GPIO → base; emitter → GND; collector → Arduino **RESET**
  + (The Arduino already has a ~10 k pull-up on RESET, so you don’t need to add one.)
  + N-MOSFET option: gate from Pi GPIO (100 Ω in series optional), source → GND, drain → RESET; add ~100 kΩ gate pulldown to GND.

**Wiring (NPN example):**

* Pi GPIO → **1 kΩ** → NPN **base**
* NPN **emitter** → **GND** (Pi/Arduino grounds common)
* NPN **collector** → Arduino **RESET** pin
* When GPIO = **HIGH**, transistor conducts and pulls RESET **LOW** → board resets.
* Pulse low for **10–50 ms** to reset, then release.

**How this “plays” with your code**

* This is **not** reset\_arduino() (that one power-cycles everything).
* Add a new helper like reset\_board(addr) that toggles the specific GPIO tied to that board’s RESET for ~20 ms.
* It’s **more robust**: resets only the bad board, avoids dropping I²C pull-ups and keeps other boards running.

**Keep your XC4486**

* Continue using the **XC4486 level shifter** for **I²C (SDA/SCL)** signals.
* It’s unrelated to the RESET line pulldown.