QUICK REFERENCE · TK28

Knock Sensor

lonely binary
open-source is love

A spring around a rod: every knock is a short burst of HIGH pulses.

VCC sets the signal

Each time the spring touches its rod, SIGNAL is joined straight to VCC. **On an ESP32, ESP32-S3 or Pico, feed VCC from 3V3, never 5V**: whatever is on VCC reaches your input several times per knock.

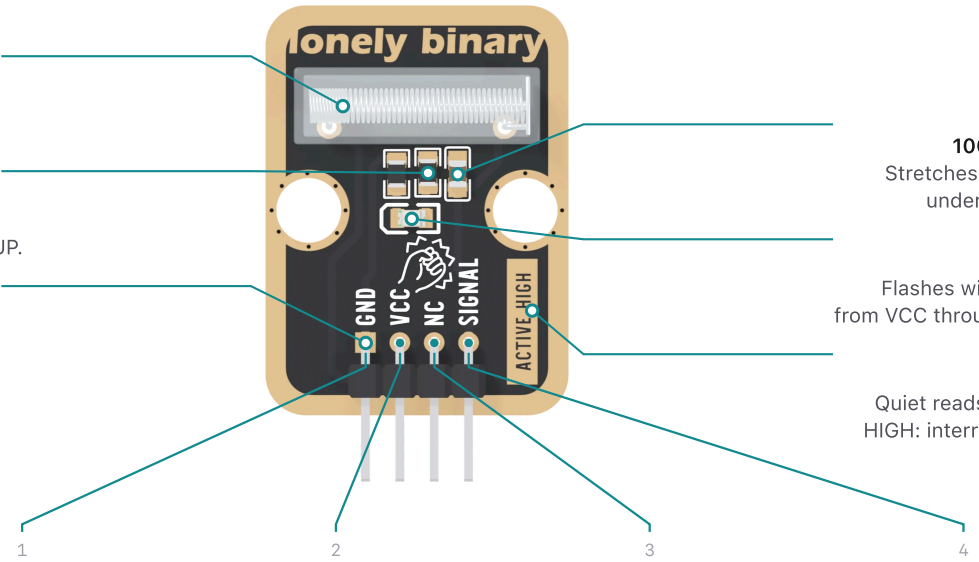
Pins, and which to wire

Switch at the top, header at the bottom. Wire GND, VCC and SIGNAL; NC is connected to nothing on the board.

SW-280 spring switch
A spring round a rod. It rings: several pulses a knock.

10 kΩ pull-down, fitted
Holds SIGNAL LOW at rest. Use INPUT, not INPUT_PULLUP.

GND is the square pad
Count from it. The back prints no pin names.



100 nF capacitor
Stretches each touch by under a millisecond.

Red LED, 1 kΩ
Flashes with each touch, from VCC through the switch.

ACTIVE HIGH
Quiet reads LOW, a touch HIGH: interrupt on RISING.

Front	GND	VCC	NC	SIGNAL
Read it	GND	VCC	●	SIGNAL

One knock, several pulses

The spring bounces on the rod, so a knock is a burst of HIGH pulses, each about a millisecond. Read without delay() or on a RISING interrupt, and ignore 100 ms after each knock.

A knock in time

After each touch

+1.1 ms to LOW, 3.3 V board

The capacitor holds SIGNAL up a little after the spring lets go: about +0.5 ms on an Uno.

Loop with delay(100)

1 in 30 knocks caught

SIGNAL is HIGH for only a few ms per knock. Read it continuously, or with an interrupt.

Hold-off

100 ms ignored after a knock

Longer than the spring rings, shorter than the gap between two knocks.

Count the first change to HIGH and ignore the rest for a moment. A hard knock and a soft one read the same, and how firmly the block is fixed to what is knocked decides whether a soft one registers at all.

The ringing is illustrative, not measured. The capacitor's timing is worked out from 10 kΩ, 1 kΩ, 100 nF and a red LED's typical 2.0 V drop; the LED takes about 3.0 mA from 5 V during a touch, 3.5 mA with the pull-down.

Count knocks, once each

Arduino IDE · Board: your board · ESP32-S3: USB CDC On Boot: Enabled

```
// SIGNAL's pin. Uno: 2. ESP32: 25. ESP32-S3: 4. Pico: 15.
const int KNOCK_PIN = 4;
const unsigned long HOLD_OFF_MS = 100;  // spring still ringing

#ifndef IRAM_ATTR
#define IRAM_ATTR            // only the ESP32 cores need it
#endif

volatile unsigned long knocks = 0, lastKnock = 0;

void IRAM_ATTR onKnock() {   // runs at every rising edge
  unsigned long t = millis();
  if (t - lastKnock >= HOLD_OFF_MS) { knocks++; lastKnock = t; }
}

void setup() {
  Serial.begin(115200);
  pinMode(KNOCK_PIN, INPUT);   // the board has its own pull-down
  attachInterrupt(digitalPinToInterrupt(KNOCK_PIN), onKnock,
                  RISING);
}

void loop() {
  static unsigned long shown = 0;
  noInterrupts(); unsigned long n = knocks; interrupts();
  if (n != shown) { shown = n; Serial.println(n); }
  delay(100);                // busy elsewhere: the count survives
}
```

```
1
2
3
```

Serial Monitor at 115200: one number per knock, however many pulses it made.

Wiring

	FRONT	UNO	ESP32	S3	PICO
1	GND	GND	GND	GND	GND
2	VCC	5V	3V3	3V3	3V3
3	NC				
4	SIGNAL	D2	GPIO 25	GPIO 4	GP15

Hole 1 is the square pad. VCC is 3V3 on every 3.3 V board. On an Uno only D2 and D3 can raise an interrupt, so SIGNAL goes on D2.

A secret knock

Keep the time of each knock in the interrupt. After 1.5 s of silence, divide every gap by the longest one and compare with the rhythm you want, allowing 0.1 either way: the speed drops out and only the rhythm counts. Check the number of knocks first, so an extra knock fails the try.

When it does not work

No flicker, nothing read

VCC or GND is not connected, or the cable is one pin over. Count from the square pad: GND, VCC, NC, SIGNAL.

One knock counts as several

The spring ringing. Ignore every pulse for 100 ms after a counted knock.

Soft knocks never register

Fix the block firmly to what is knocked, through its mounting holes. There is no sensitivity adjustment.

LED flickers, sketch sees nothing

A delay() in loop() sleeps through the pulses. Read continuously or use an interrupt; then check the pin number.

Reads a knock all the time

A sketch testing for LOW, or INPUT_PULLUP. If the LED flickers too, something nearby is shaking the table.

Interrupt fires at the wrong time

Use RISING: a knock is a change from LOW to HIGH on this board. FALLING fires as each pulse ends.

Specifications

Switch	SW-280 spring vibration switch	**Body**	18.2 × 5.7 × 5.6 mm, clear, lying flat
Output	Active high: a burst of pulses per knock	**Pull-down**	10 kΩ, SIGNAL to GND
Capacitor	100 nF, SIGNAL to GND	**Supply**	3.3 V or 5 V, the same as your board
Header	4 pins, right-angle, 2.54 mm	**Size**	22.4 × 30.4 mm