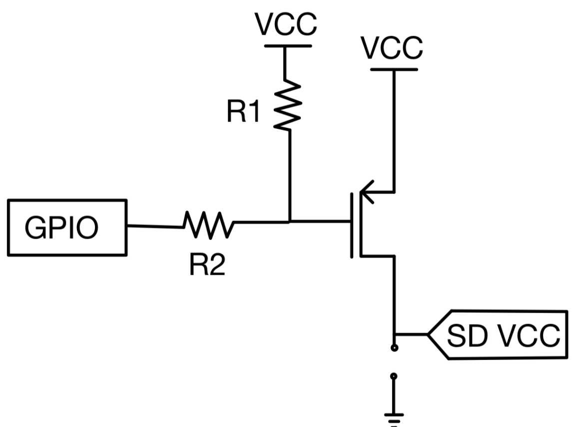
# Low-power Datalogging System

## 1) Circuit design

One solution is to use a PMOS as a high-side switch, which a single GPIO can control to supply power to the SD card. I would recommend adding two resistors if space allows, even if using 0201 SMD sizes.

- A gate pull-up resistor (R1) is used to ensure the PMOS is off and not floating during MCU start-up or reset, such that the SD card will not be unintentionally turned on
- A series gate resistor (R2) should be added, if space allows, to limit inrush current during GPIO transitions and protects the pin from brief current spikes.




## 2) MCU Code

I choose the MKRZero for a balance of low sleep current (~2-5μA), real-time clock (RTC) on board (for 10-min interval counting), fast wake-up from sleep (~3μs), SD card integration on board, and robust Arduino library support. The microcontroller chip on board, SAMD21G18A, makes many of the desirable features above available, and would be a great option if we were to consider designing a custom PCB, instead of using a development board like the MKRZero. Below is a comparison of microcontroller options I considered. The MKRZero is chosen because given that it consumes very low power, it's development convenience edges over the Arduino Pro Mini of having an onboard RTC and the Teensy LC for having an SD slot on board. If Bluetooth may be desired for the project, then the nRF52832 would be my choice, although it is harder to develop using the Arduino IDE.

| Board / MCU | Deep-Sleep Current | Active Current* | Onboard RTC? | SD-Slot on Board? | Flash / RAM | Notes |
|---|---|---|---|---|---|---|
| **Pro Mini 3.3 V @ 8 MHz** | 0.3–1 µA (power-down mode) | 5–10 mA (5 V-to-3.3 V reg) | No (WDT only) | No | 32 KB / 2 KB | Lowest deep-sleep. WDT limits you to ≈8 s intervals → you'll need to count to 10 min or add a DS3231 RTC module for precise timing. Very well-tuned for sub-µA sleep. |
| **Teensy LC (MKL26)** | ≈2 µA (VLPR + VLLS) | 5–8 mA | Yes | No | 128 KB / 8 KB | Has a built-in RTC that can wake you at 10 min intervals. Very low-power core, small form factor; needs an SD breakout. |
| **Feather M0 / MKR Zero (SAMD21)** | 2–5 µA (standby + regulator) | 10–15 mA | Yes | MKR Zero: Yes<br><br>Feather: No | 256 KB / 32 KB | Both share ATSAMD21G18A: true RTC, fast flash writes (48 MHz), integrated µSD on MKR Zero. Feather M0 will need an SD "wing." |

| | | | | | | |
|---|---|---|---|---|---|---|
| **nRF52832 Feather** | ≈1 µA (system OFF) | 4–6 mA | Yes | No | 512 KB / 64 KB | BLE radio can be disabled; has RTC; very low deep sleep. SD-card via SPI breakout. |

## 3) Current Draw

There are 3 phases of operation based on my implementation. Assuming a 3.3V supply: Note that based on the datasheet for Arduino MKRZero, the 3.3V pin can supply a maximum of 600mA, and GPIOs can supply 7mA.

1. Setup / Sleep Entry
    a. The SAMD21G18A core draws 10-12mA at 48Hz, which occurs either during setup before any SD or USB activity, or while preparing to enter sleep after writing to and powering off the SD card.
    b. **Current Draw: 12mA**
    c. Time: Setup (2ms) + wake handshake (6ms) + sleep entry (1ms) = **9 ms**
2. SD Power-on and Write
    a. 12mA from the MCU core continues to be drawn
    b. About 30-40mA is drawn for SPI-mode initialization of microSD
    c. PMOS Pull-up bleed: When the gate is held low and the SD card is powered, the 499kΩ resistor bleeds: I = 3.3V / 499kΩ = 6.6µA
    d. **Subtotal Current Draw**: 12 mA + 40 mA + 6.6µA ≈ **42mA**
    e. Time: PMOS stabilization (10ms) + SPI SD initialization (100ms) + Data Write (5ms) = **115ms**
3. Sleep
    a. MCU Standby: 2-5µA (when all non-RTC peripherals are off)
    b. RTC Peripheral: 5µA
    c. **Subtotal Current Draw**: **10µA**
    d. Time: **600 s**

With a 100mAh battery (e.g. rechargeable LiPo), this system yields the following runtime:
Charge per cycle: 9ms * 12mA + 115ms • 42mA + 600s • 10µA = 0.00994 mA•s
Average Current: 0.00994 mA•s / 600.124s = 16.6 µA

100mAh Battery = 360000 mA•s
Estimated Runtime: 360000 mA•s / 16.6 μA = 21,686,747 s = **8.25 months**

Please refer to the README.md for details on usage of this system.