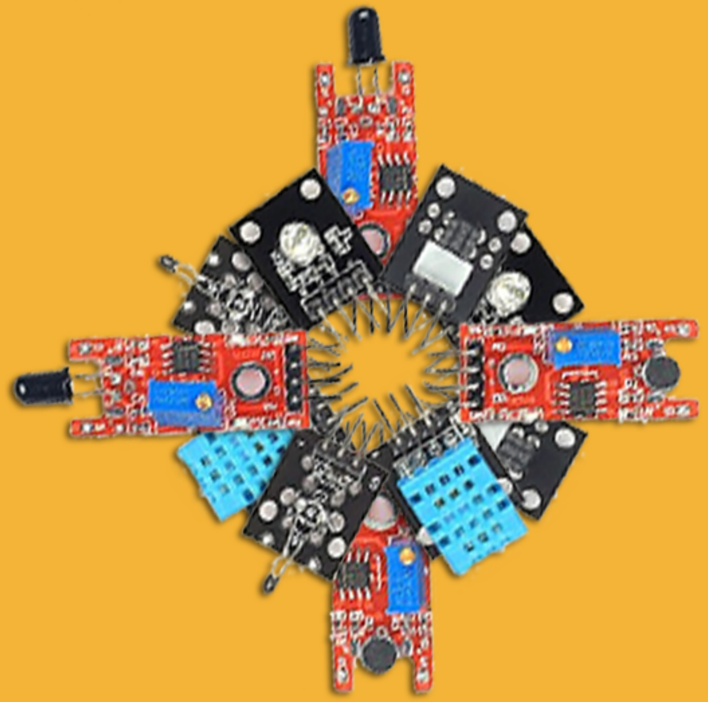
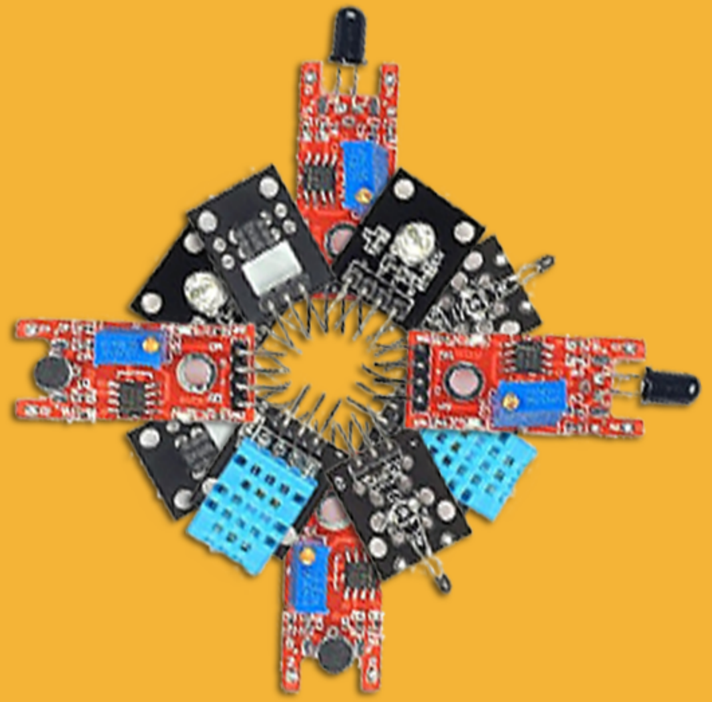


Modern Alchemists

IOT SESSIONS #04

NODEMCU | SENSORS



DIRTY GUIDE TO SENSORS

IN TERMINAL:

input:

sudo aptitude search apache2-utils apache2-doc

output:

i apache2-doc -apache HTTP Server (on-site documentation)

i A apache2 utils -apache HTTP Server (utility programs for web servers)

PREREQUISITES

I'M ASSUMING:

1. you've got Arduino IDE working with your NodeMCU and you've played around a little.
2. You know nothing about sensors.
3. You know nothing about sampling data.
4. You know nothing about binary.
5. You're wearing a seatbelt.

WHAT IS A SENSOR?

An electronic device that detects a change in its environment and communicates this change

FLAVOURS

- Sounds
- Sight
- Smell
- Touch
- Taste (Yes, really)

WHY USE A SENSOR?

1. Want IRL/"real world" information
2. Gives situational awareness to project
3. Gives context for actions
4. Gives decision-making capability
    - "If <Event> occurs, do <Action>"
    - Most important.

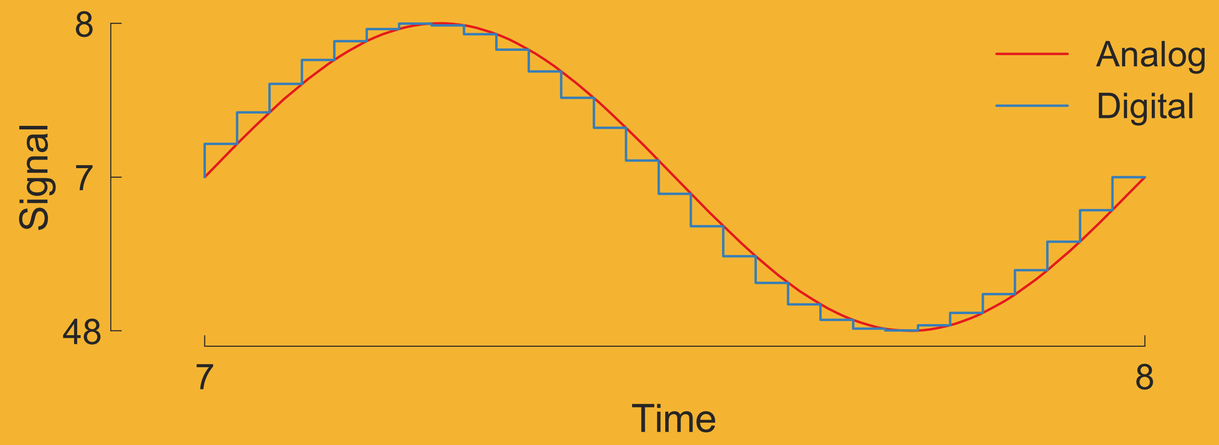
TWO TYPES OF SENSORS

ANALOG

- Signal: Continuous -> Discrete / Digital (process is called Sampling)
- Always a voltage is that is sampled and interpreted.

DIGITAL

- Signal: Already discrete
- No sampling needed by MCU.



ANALOG SENSORS

PROS

- Cheap
- Super easy to use
- Easy hardware configuration
  (passive components)
- Don’t need libraries
- Can make your own!
- Resolution is set by ADC of MCU

CONS

- Need more I/O for large numbers
  (there are workarounds)
- Need ADC pins if not binary
- Less accurate (usually)
- Noise
- Resolution is set by ADC of MCU

DIGITAL SENSORS

PROS

- More accurate
- Repeatability (position)
- Less I/O for multiple devices (same bus)
- More features
- Less sensitive to noise + interference

CONS

- More expensive
- More difficult to use
- Will need to read datasheet
  (not that scary)
- Will need to search for a library
- Worst case: write your own library
  (hard for beginners)

TIME TO GO SHOPPING!

SENSORS IN SOUTH AFRICA

UNDERGROUND.CO.ZA

Indexes most of RSAs online hobbyist stores
Compare prices

RS COMPONENTS

Biggest range (individual components, eg. capacitors, resistors, voltage regulators, etc)
Note minimum quantities
Use the filters
Website is slow, and does suck

ALTERNATIVES

Mantech, etc.
Suck even more -> only if you're desperate, or BIG quantities.

I'VE FOUND A SENSOR, NOW WHAT?

1. You'll need to check if it matches your needs.
2. You'll need to check if it's compatible.

    In order:

        1. Use-case requirements
        2. Power input requirements
        3. Output signal voltage (analog sensors)
        4. Logic levels (digital sensors)

3. This means looking at the datasheet

DATASHEETS

CRASH COURSE IN DATASHEETS

- Everything has one
- Not all datasheets are created equal(hello chinese components
- Dont panic. Only scary the first time you go through them
- Can find 99% of what you need on the first page

DATASHEETS

MEETING REQUIREMENTS

- Check accuracy
- Check operational range
- Check sampling rate / response time
- Check packaging (does it fit breadboard? Is it hard to solder?)
- You’re checking if the sensor can do what you’re going to ask it to do.

DATASHEETS | ACCURACY, RANGE AND FORMS

ANALOG DEVICES

Low Voltage Temperature Sensors

TMP35/TMP36/TMP37

FEATURES

Low voltage operation (2.7 V to 5.5 V)
Calibrated directly in °C
10 mV/°C scale factor (20 mV/°C on TMP37)
±2°C accuracy over temperature (typ)
±0.5°C linearity (typ)
Stable with large capacitive loads
Specified −40°C to +125°C, operation to +150°C
Less than 50 µA quiescent current
Shutdown current 0.5 µA max
Low self-heating

data format
accuracy
range

FUNCTIONAL BLOCK DIAGRAM

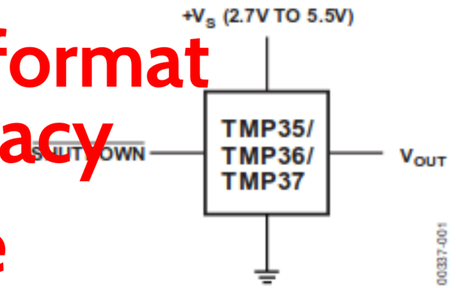


Figure 1.

PIN CONFIGURATIONS

Modern Alchemists

DATASHEETS | PACKAGING

voltage, precision centi-
ide a voltage output that
(centigrade) temperature.
require any external
ies of ±1°C at +25°C
emperature range.

P35/TMP36/TMP37 and
n simplify interfacing to
Cs. All three devices are
from 2.7 V to 5.5 V maxi-
elow 50 µA, providing
in still air. In addition, a
the supply current to less

packaging

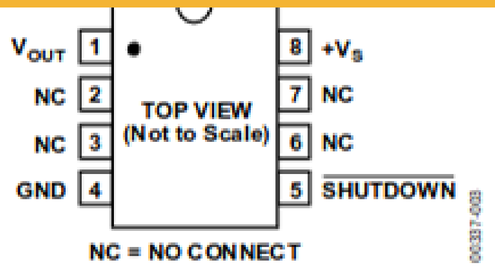


Figure 3. R-8 (SOIC_N)

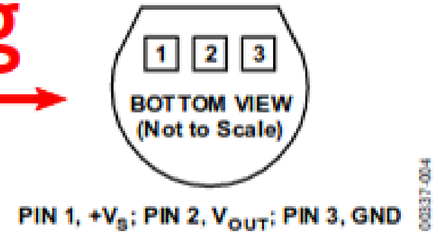


Figure 4. T-3 (TO-92)

Modern Alchemists

DATASHEETS | RESPONSE TIMES

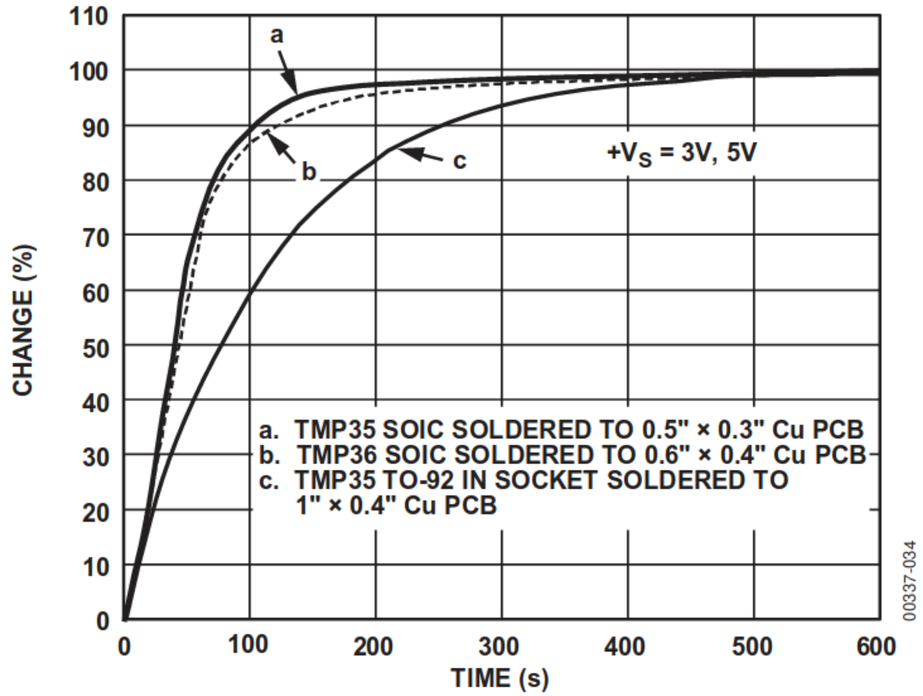


Figure 17. Thermal Response Time in Still Air

DATASHEETS | INPUT POWER

1. Check required voltage (Too high? Too low?)
2. Check input current (Can I supply enough?)
3. Questions you want to answer:
    - Can I power it from USB power? (5V, @500mA)
    - Can I power it from my battery? (1.5V, 4.2V, 9V, 12V)
    - Can I power it via my device? (3.3V, @500mA from regulator)
    - Can I power it from the wall?

DATASHEETS | INPUT POWER

ANALOG DEVICES

Low Voltage Temperature Sensors

TMP35/TMP36/TMP37

FEATURES

- Low voltage operation (2.7 V to 5.5 V) ← Voltage
- Calibrated directly in °C
- 10 mV/°C scale factor (20 mV/°C on TMP37)
- ±2°C accuracy over temperature (typ)
- ±0.5°C linearity (typ)
- Stable with large capacitive loads
- Specified −40°C to +125°C, operation to +150°C
- Less than 50 µA quiescent current ← Current
- Shutdown current 0.5 µA max
- Low self-heating

FUNCTIONAL BLOCK DIAGRAM

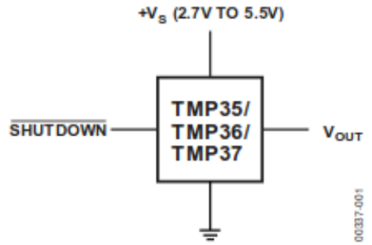


Figure 1.

PIN CONFIGURATIONS

Modern Alchemists

DATASHEETS | OUTPUT SIGNAL

- Analog sensors all output a voltage that must be sampled using an ADC (analog-to-digital converter).
- ADCs have maximum input voltages.
- Exceeding max results in clipping + damage (if unlucky).
- Must make sure max sensor output is less than max ADC input.
- Two things to check, then
    1. Max expected sensor output voltage
    2. Max ADC input voltage

DATASHEETS | OUTPUT SIGNAL

Table 4. TMP3x Output Characteristics

Sensor	Offset Voltage (V)	Output Voltage Scaling (mV/°C)	Output Voltage @ 25°C (mV)
TMP35	0	10	250
TMP36	0.5	10	750
TMP37	0	20	500

Output signal

DATASHEETS | OUTPUT SIGNAL

CAN OUR NODEMCU HANDLE THIS?

3.9. ADC (Analog-to-digital Converter)

ESP8266EX is embedded with a 10-bit precision SARADC. Currently, TOUT (Pin6) is defined as ADC interface, the definition of which is described below:

Pin Name	Pin Num	Function Name
TOUT	6	ADC Interface

Table 16 Pin Definition of ADC

Hardware Design: The input voltage range is 0 to 1.0 V when TOUT is connected to external circuit.

Max input = 1V

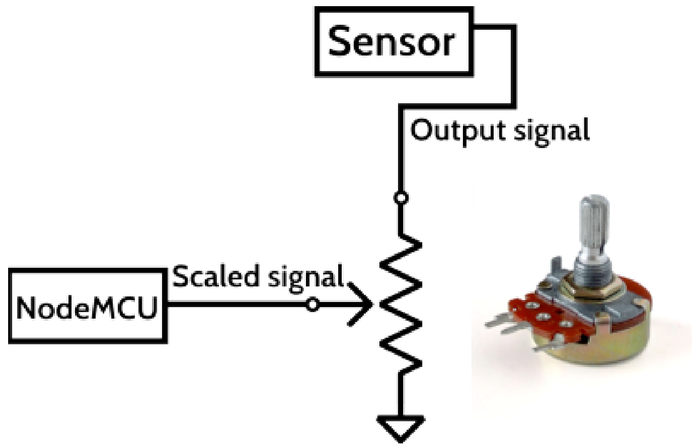
DATASHEETS | OUTPUT SIGNAL

CAN OUR NODEMCU HANDLE THIS?

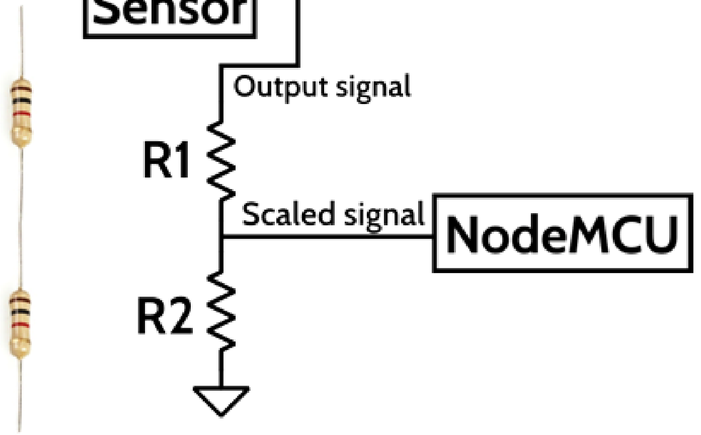
- Do some back-of-the-napkin math to check
- Sensor: 0˚C = 0.5V
- Sensor: 10mV per ˚C
- NodeMCU: 1V maximum input
- (1-0.5)/0.01 = 50˚C maximum we can measure before clipping.
- Does this meet requirements?

OUTPUT SIGNAL

WHAT HAPPENS IF OUR OUTPUT SIGNAL WAS TOO LARGE?



OR



$$V_{\text{scaled}} = V_{\text{output}} \cdot \frac{R2}{R1+R2}$$

Modern Alchemists

DATASHEETS | LOGIC LEVELS

- This is an extra step that only applies to digital sensors.
- Devices communicate at different logic level voltages
    - 5V can be seen as a binary ‘1’ (5V logic)
    - 3.3V can be seen as a binary ‘1’ (3.3V logic)
    - 1.8V can be seen as a binary ‘1’ (1.8V logic)
- Devices with different logic levels cannot communicate!
- Use a logic-level converter if there is a logic level difference.
    - Cheap.
    - Found everywhere.
    - Can even make your own with transistors.


BREAK TIME?
HOW ABOUT NO!!!

WIRE IT UP

TIME TO COLLECT SENSOR DATA

- Analog sensors
    - Sample using ADC if continuous value.
    - Interpret as logic-level input if binary signal (assuming correct voltage level)
    - Conversion to correct value after sampling.
- Digital sensors
    - Communication on some kind of data bus with specific protocol
    - Three big ones:
        - SPI
        - I2C
        - 1-wire communication

READING SENSORS | ANALOG

SAMPLING USING ADC

- Use analogRead() function.
- 10-bit resolution on the NodeMCU
- This means max sample value = 2^10 = 1024
- Can be converted to a voltage knowing that
  1024 corresponds to the max ADC voltage (1.0V).
    - Voltage of signal = 1.0V/1024
    - Once you have voltage, use sensor datasheet for conversion.

Table 4. TMP3x Output Characteristics

Sensor	Offset Voltage (V)	Output Voltage Scaling (mV/°C)	Output Voltage @ 25°C (mV)
TMP35	0	10	250
TMP36	0.5	10	750
TMP37	0	20	500

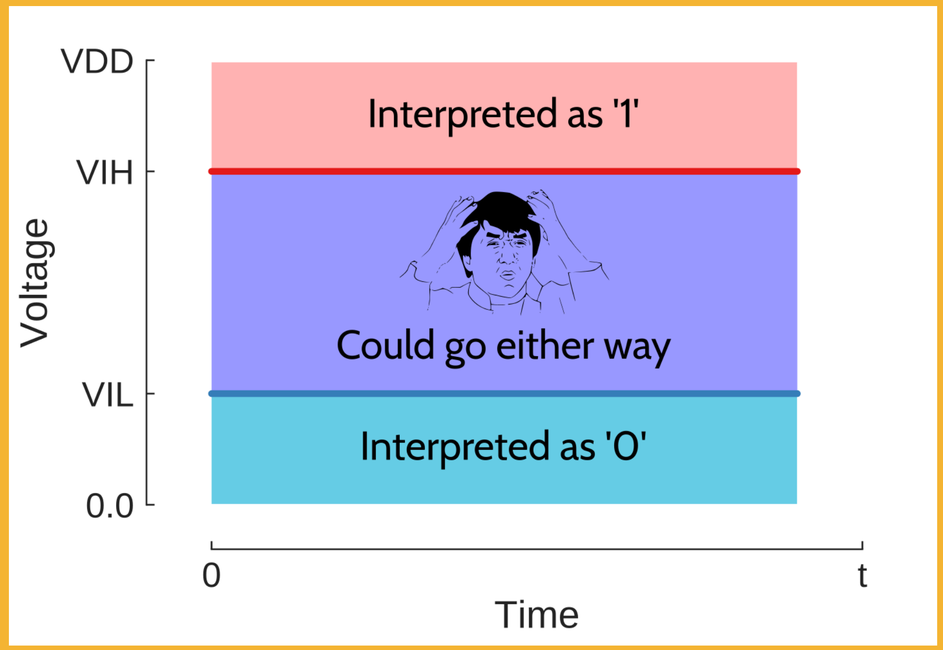
Output signal

READING SENSORS | ANALOG

- Sample binary using digitalRead()
- Assumes your logic level is matched!
- Simple example: button / switch
- Datasheet for NodeMCU:

    VIH = 0.75 VDD

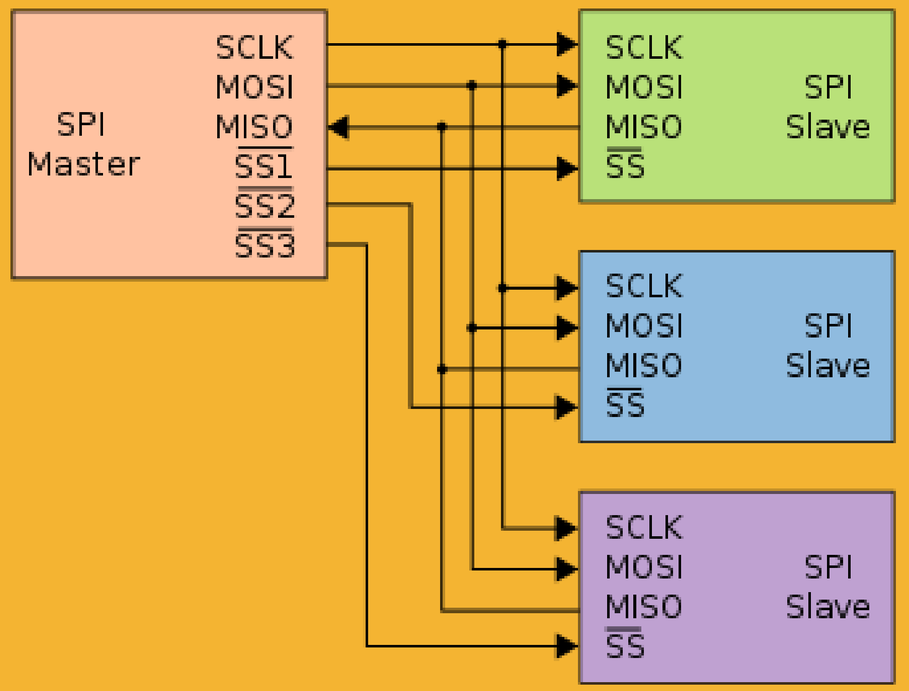
    VIL = 0.25 VDD



READING SENSORS | DIGITAL

SPI (SERIAL PERIPHERAL INTERFACE BUS)

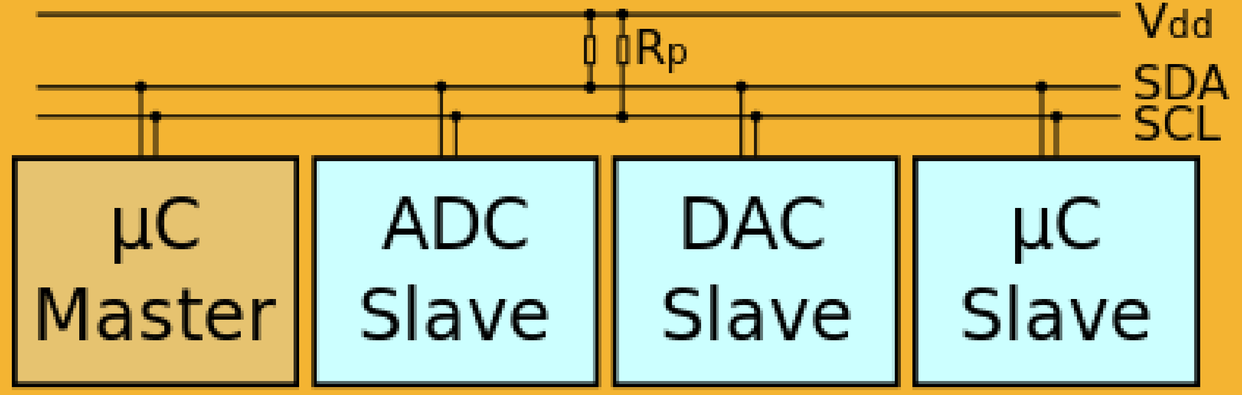
- Single master
- Well supported
- 3 or 4- data lines
- + extra CS/SS line for each additional device
- Fast (I've seen 20Mhz+)
- Better for fast devices



READING SENSORS | DIGITAL

TIME TO COLLECT SENSOR DATA

- More complex
    - Addressing of devices
    - Lots of error conditions that must be handled
- Slower (400Khz standard, 1Mhz high-speed)
- Only two data lines though!
- Multiple masters!



READING SENSORS | DIGITAL

1 WIRE

- Low speed
- Long range
- Only two lines total – DATA and GND
    - Is able to power itself from the DATA line using a tiny capacitor
- Devices can share the same bus
    - Each device has unique 64-bit serial number

READING SENSORS | TIMING

TIME TO COLLECT SENSOR DATA

- In many cases you want to sample at very precise intervals.
- Or you want to do other work inbetween waiting for samples.
- Timer Interrupts are what you want!
    Ticker.h a good place to start for NodeMCU (millisecond accuracy)
    Microseconds are a challenge on ESP8266 boards
    (a unique weakness due to the WiFi chip).
- Even if you use millis() or micros() for less-accurate timers,
  avoid polling (i.e. delay()) as much as possible*
- *ESP8266 uses this differently than Arduino.

CONVERTING / RECONSTRUCTING DATA

SAMPLED ANALOG SIGNALS

- ADC has a resolution, specified in “bits”
- x-bits = 2^x values that subdivide ADC range.
- Max value will be sampled as 2^x
- Min value will be sampled as 0
- Convert ADC sample to voltage
- Convert voltage to data signal (datasheet will specify)

CONVERTING / RECONSTRUCTING DATA

DIGITAL SENSORS

- Typically, resolution > 8-bits (eg. 10-bit or 12-bit)
- But you get your data as 8-bit packets – now what?
- Binary reconstruction!
- Take 8-bit packets and stick them together in the right order to get data.
- We'll need to know some bit-wise operators (AND, OR, Shift)

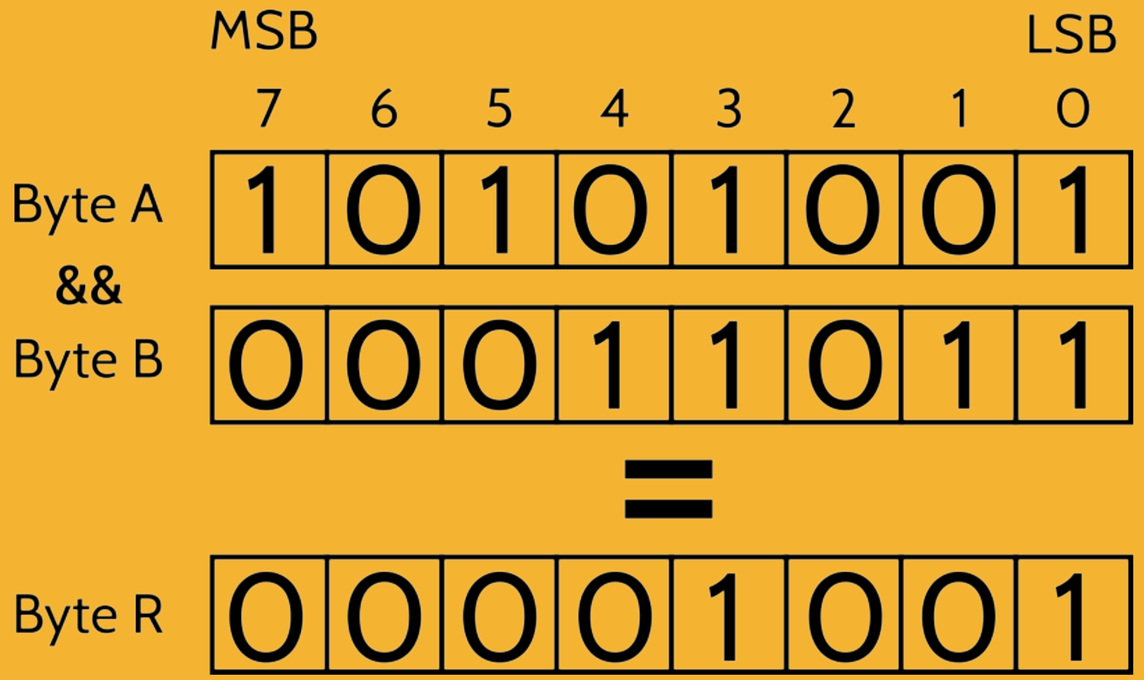
BINARY RECONSTRUCTION

FIRST THINGS FIRST

- How many bits is your MCU?
  - Arduino = 8-bit
  - NodeMCU = 32-bit (!)
- These tells you how big integers can be on your platform
  - This is simplified – there are workarounds and caveats.
- This limits the number of packets you can reconstruct per sample, so always double-check this.
- 1 Byte = 8 bits

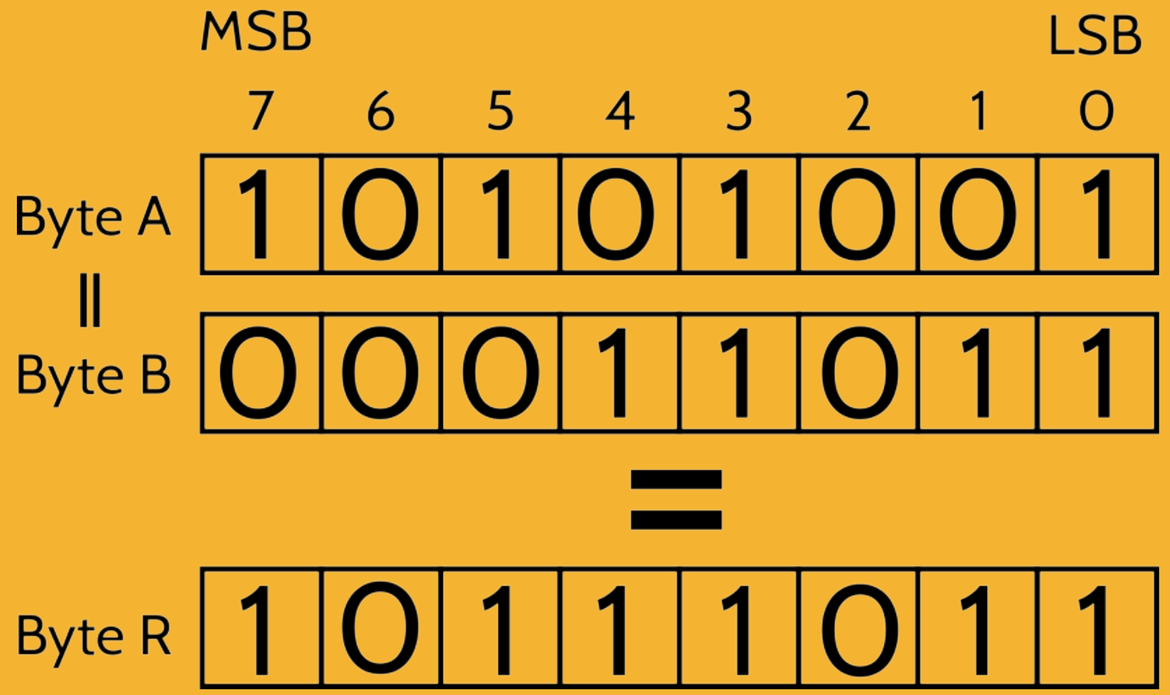
BINARY RECONSTRUCTION | AND OPERATOR

char ByteR = ByteA && ByteB



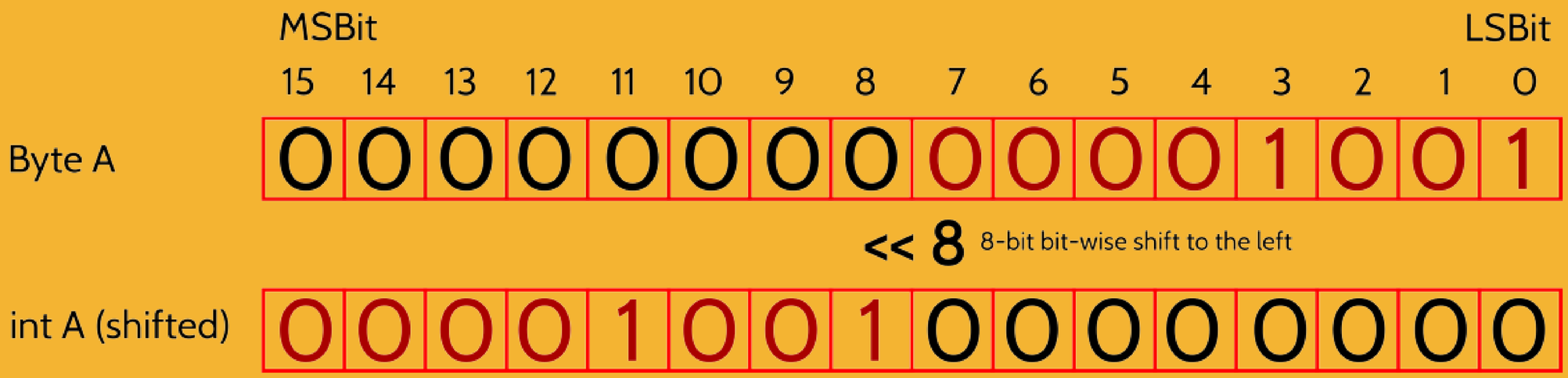
BINARY RECONSTRUCTION | OR OPERATOR

char ByteR = ByteA || ByteB



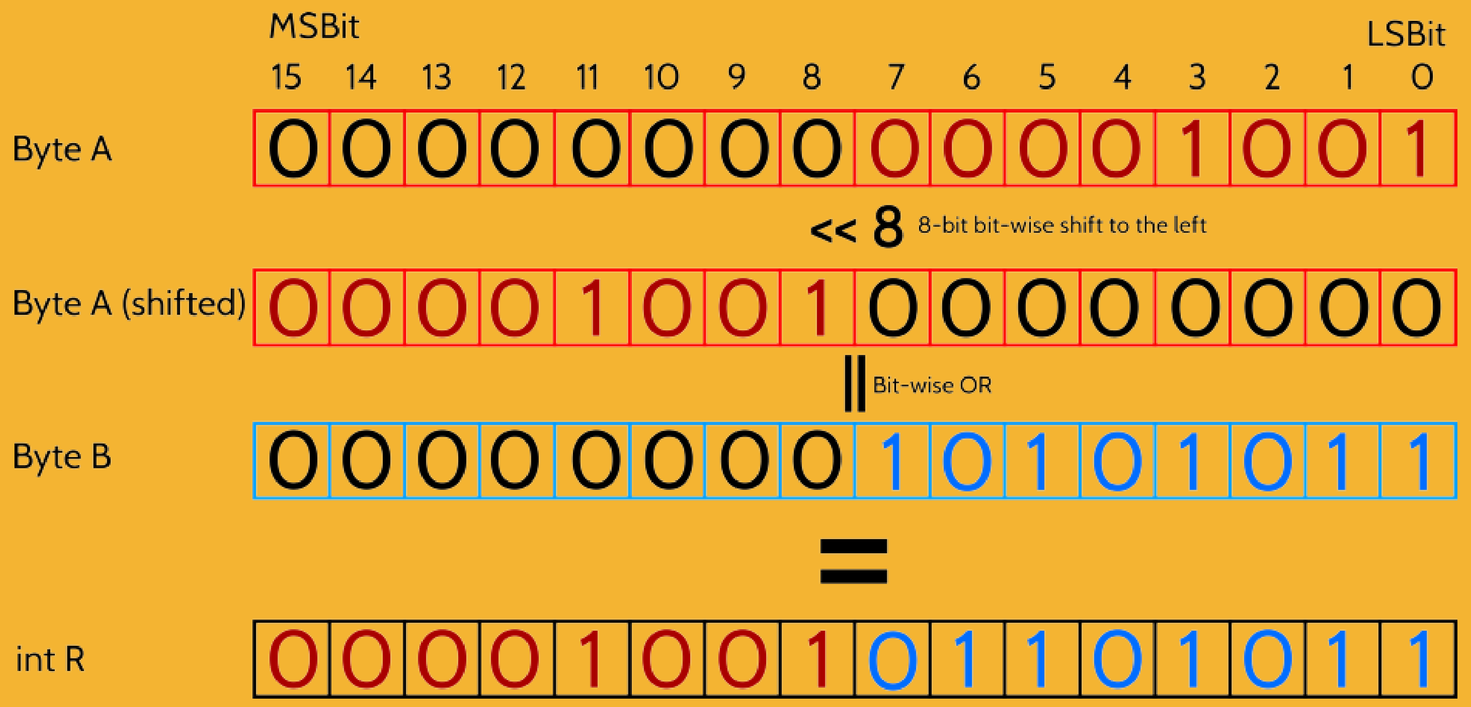
BINARY RECONSTRUCTION | BIT SHIFITING

int A = ByteA << 8



BINARY RECONSTRUCTION | STICKING BYTES TOGETHER

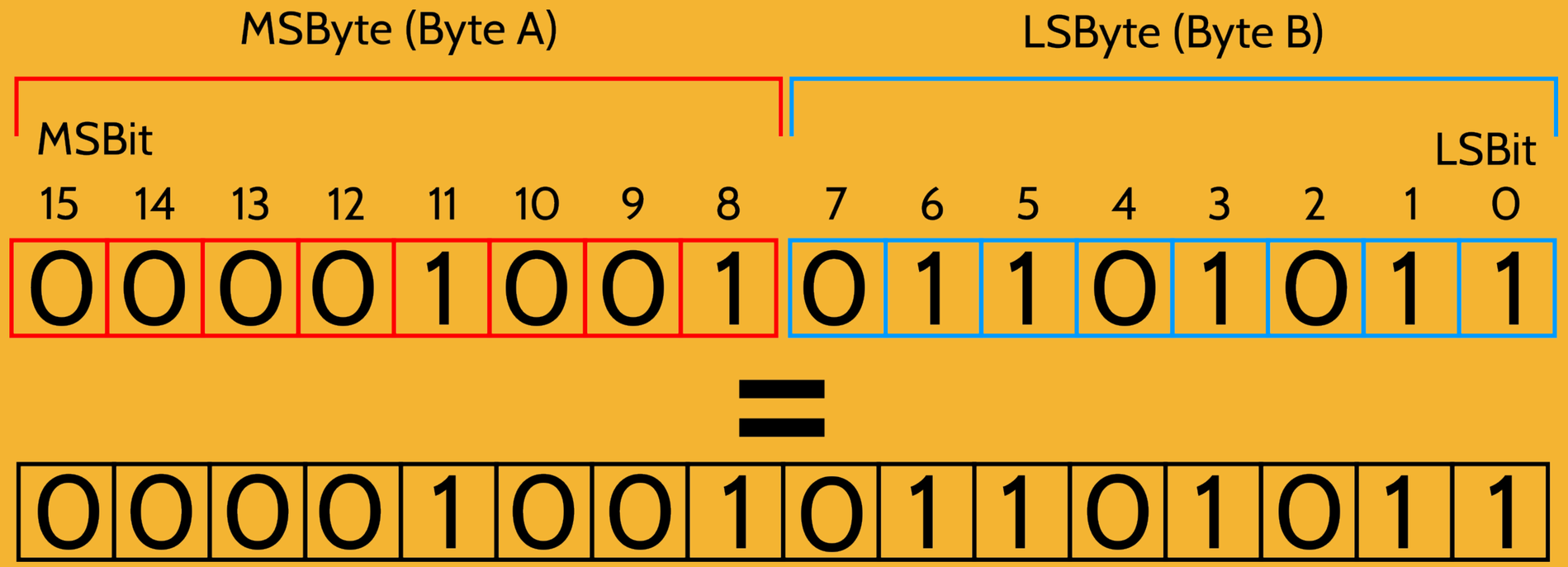
int R = (ByteA << 8) || ByteB



BINARY RECONSTRUCTION | OVERVIEW

You combine bytes in the right order, and you get a value out at the end.

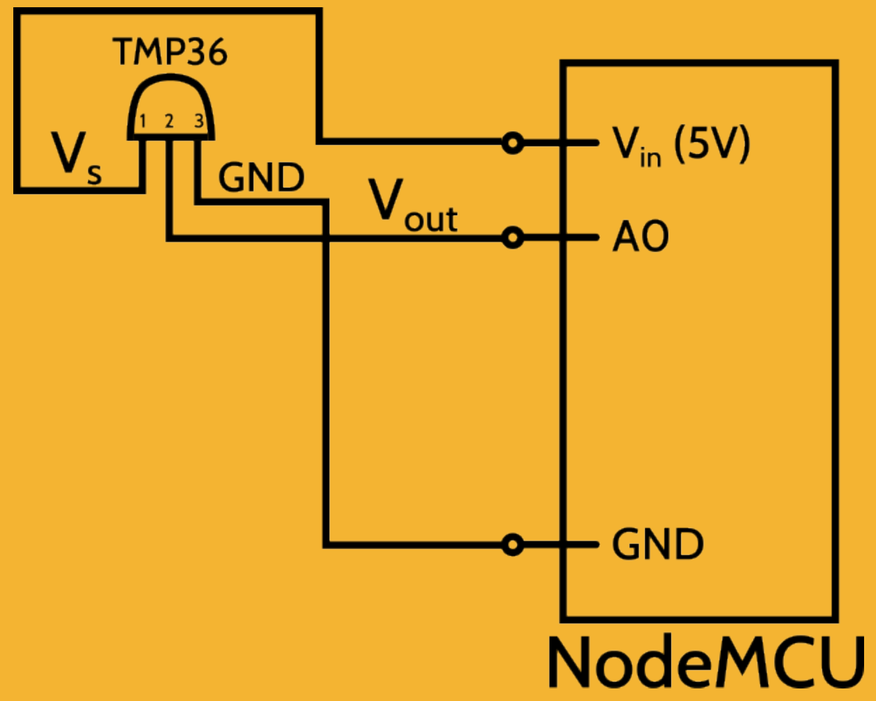
Datasheet will explain the bytes and their order.



Modern Alchemists

WORKSHOP EXAMPLE | TMP36

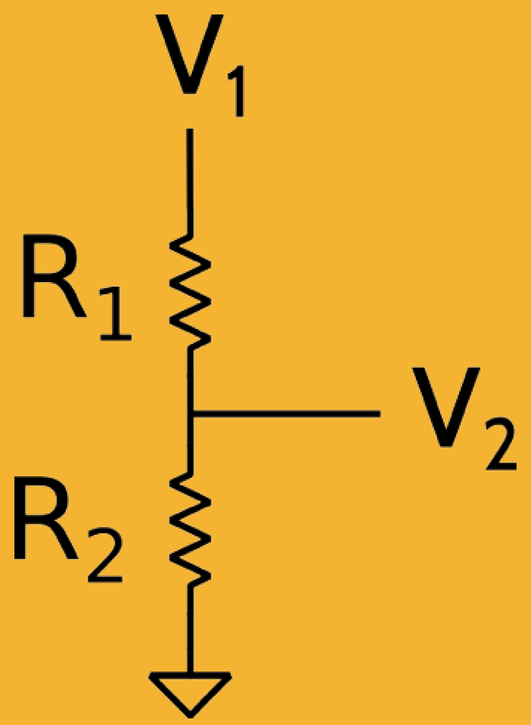
Basically covered already (you've been looking at its datasheet this whole time).



Modern Alchemists

WORKSHOP EXAMPLE | LDR

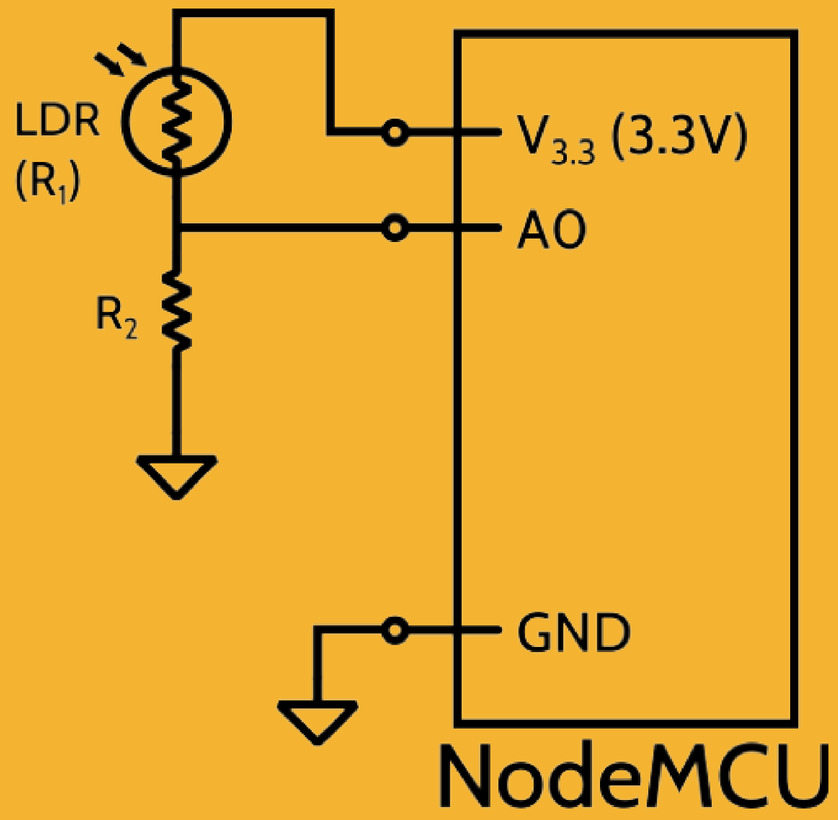
- Light-dependent resistor changes its resistance depending on the amount of light it receives (more light = lower resistance).
- Easiest implementation is as a resistor in a voltage-divider circuit.



$$V_2 = V_1 \frac{R_2}{R_1 + R_2}$$

WORKSHOP EXAMPLE | LDR DIAGRAM

- Must choose R1 so that input to A0 is less than 1.0V
- From equation:
    - We can see that V2 will be biggest when R1 is smallest (most light).
    - Find minimum value of LDR
    - Calculate R2



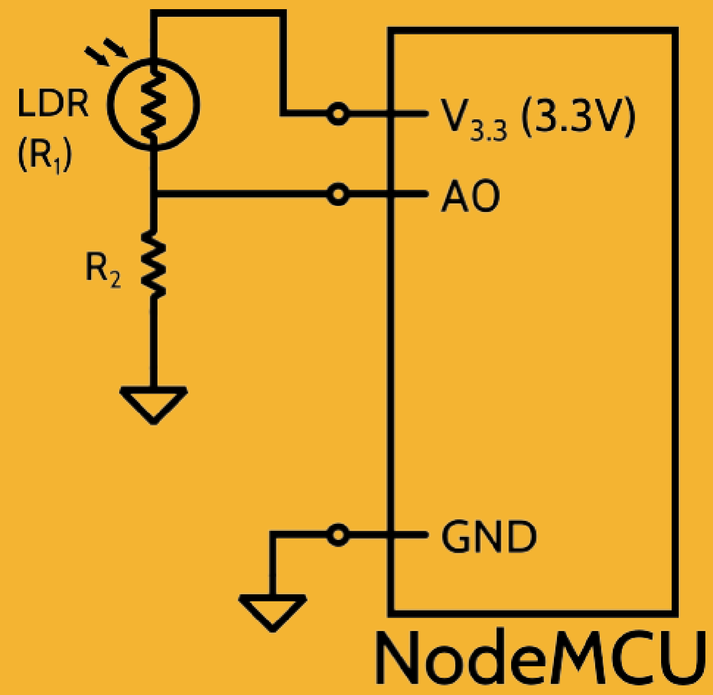
$$V_2 = V_1 \frac{R_2}{R_1 + R_2}$$

WORKSHOP EXAMPLE | LDR - FINDING R2

$$V_2 = V_1 \frac{R_2}{R_1 + R_2}$$

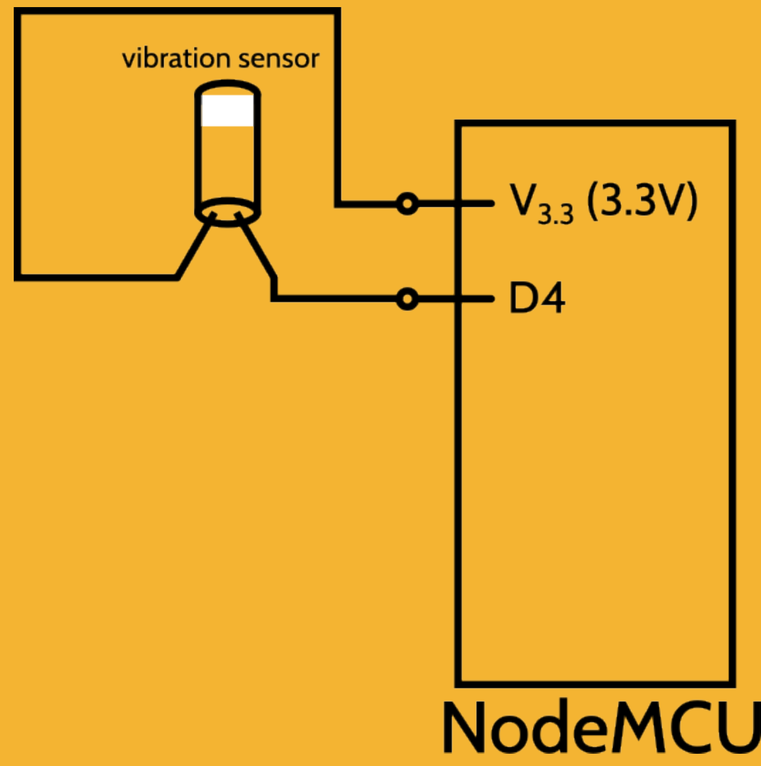
$$R_2 = \frac{V_2 R_1}{V_1 - V_2}$$

$$R_2 = \frac{1.0 \times \mathrm{LDR}_{\mathrm{min}}}{3.3 - 1.0}$$



WORKSHOP EXAMPLE | VIBRATION SENSOR

- Different than previous examples
  - we are not sampling a value.
- We want to set-up a trigger.
- To do this we use what's known as an interrupt
  - There are different types (eg. timer)
  - We're using a GPIO / pin interrupt



WORKSHOP EXAMPLE | VIBRATION SENSOR INTERUPTS

- When event occurs
    -> execute a particular piece of code.
- Interrupt -> Interrupt subroutine.
- Subroutine essentials:
    Variables given “volatile” declaration.
    Keep them short
    Keep them fast.
- When vibration detected
    - Set motion flag to True