CC2511 Week 6 Lecture 1

Assignment 2

- It’s time to start on Assignment 2.
- You’ll work on the schematic in this week’s lab.
- **For this week only, attend the same lab as your team members.**

Week number	Note
6	We are here. Work on schematic in the lab.
7	Schematics due on Friday. Worth 10% of assignment mark.
8	Receive feedback and revise schematic if needed.
9	PCB layout due on Monday.

Assignment 2: What not to do

- In the past some groups decided to split the tasks as:
  - One person does the schematic.
  - One person does the PCB layout.
  - One person writes the code.
- Do not do this! These are not equal tasks.
- The person who does the code will have a big advantage on the final exam.

This lecture

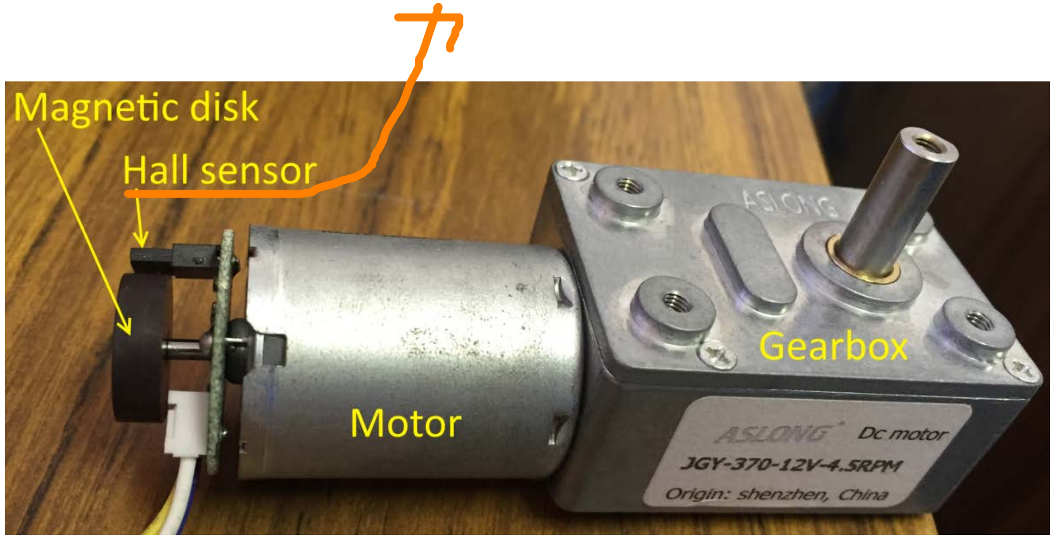
- Controlling and monitoring DC motors:
  - Hall Effect sensors
  - H bridge driver logic
- Discrete logic chips (e.g. 7400 series)
- Primer on digital logic

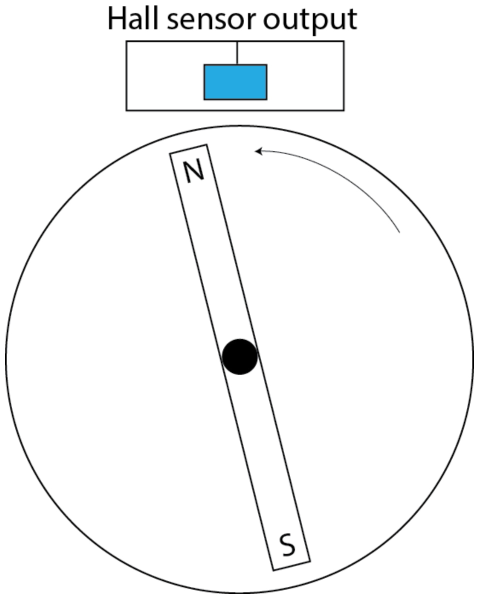
DC motors

- DC motors generate a **torque proportional to the current** that flows.
- Adjusting the average voltage using PWM controls the motor torque.

Monitoring a DC motor

- Hall Effect sensors are commonly used to measure the angular velocity of an axle.
- The sensor measures magnetic field strength.



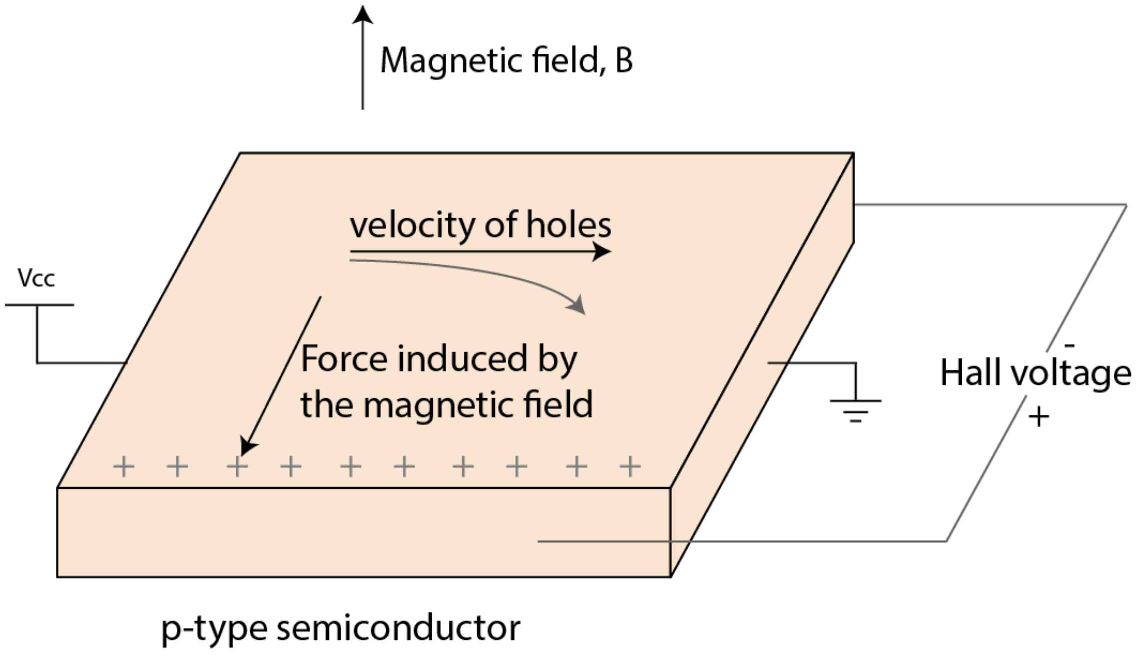


Fundamentals: The Hall Effect

- A particle carrying charge q moving with velocity **v** in a magnetic field **B** experiences a force

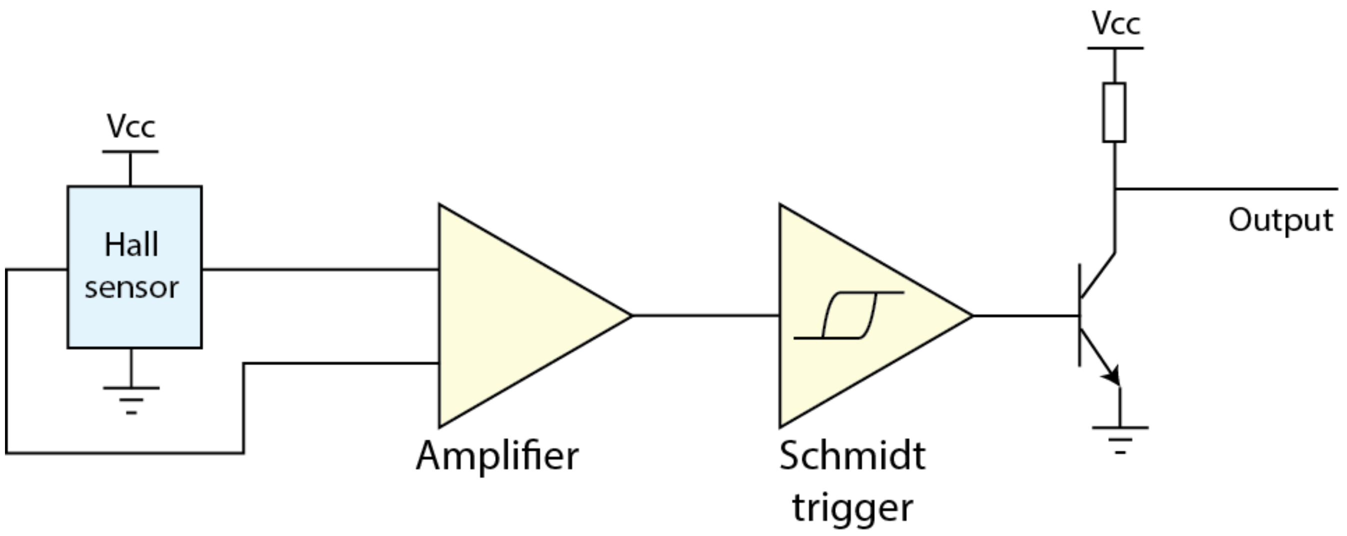
$$\boldsymbol{F} = q\,\boldsymbol{v} \times \boldsymbol{B}.$$

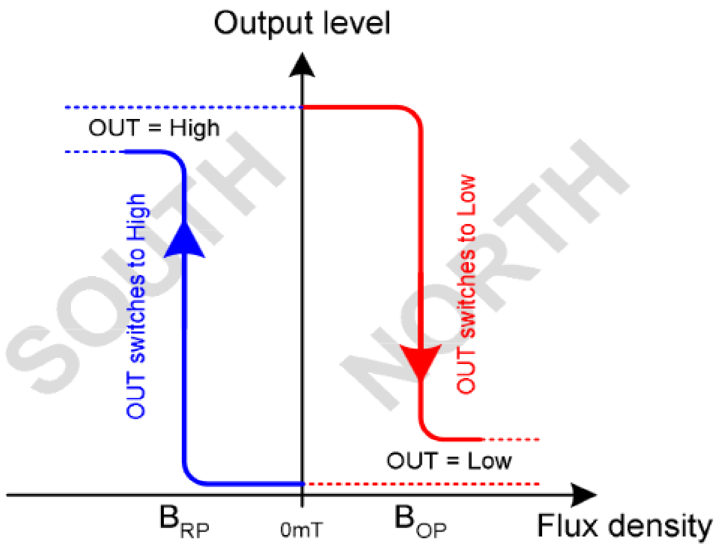
- A voltage develops across a conductor perpendicular to the velocity of charge carriers and the magnetic field.



A rotational Hall Effect sensor

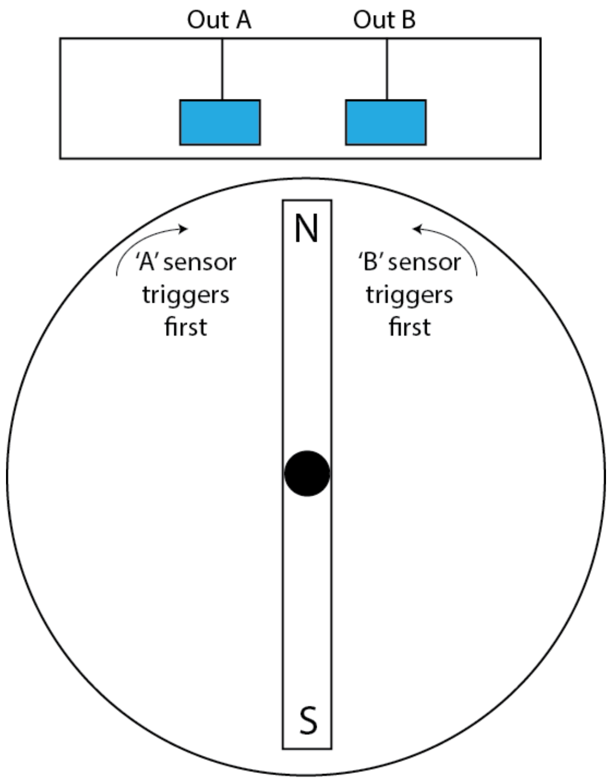
- Output (for example)
  - A high voltage once magnetic North has passed the sensor.
  - A low voltage once magnetic South has passed the sensor.
  - Hold the voltage until a strong field in the other direction is seen.



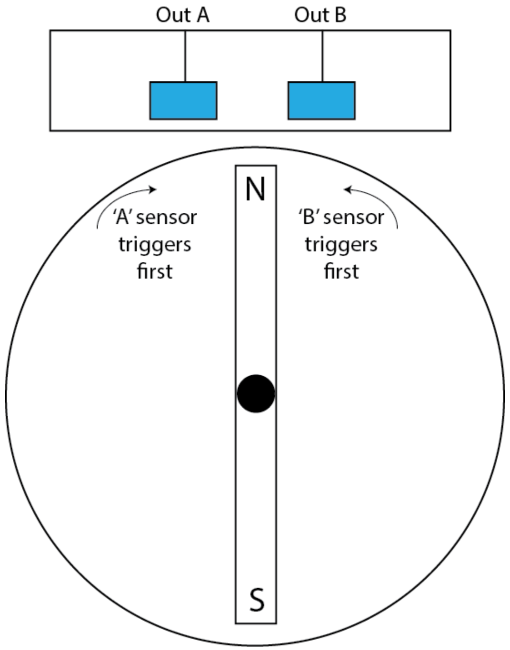


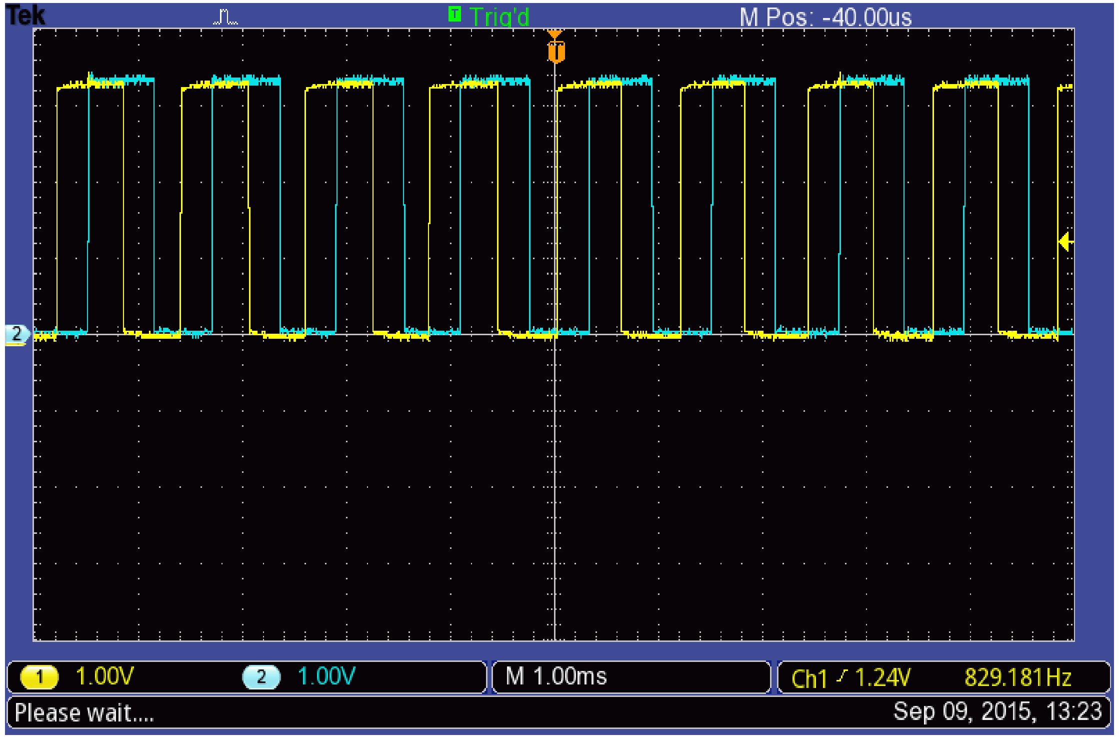
Measuring the direction of rotation

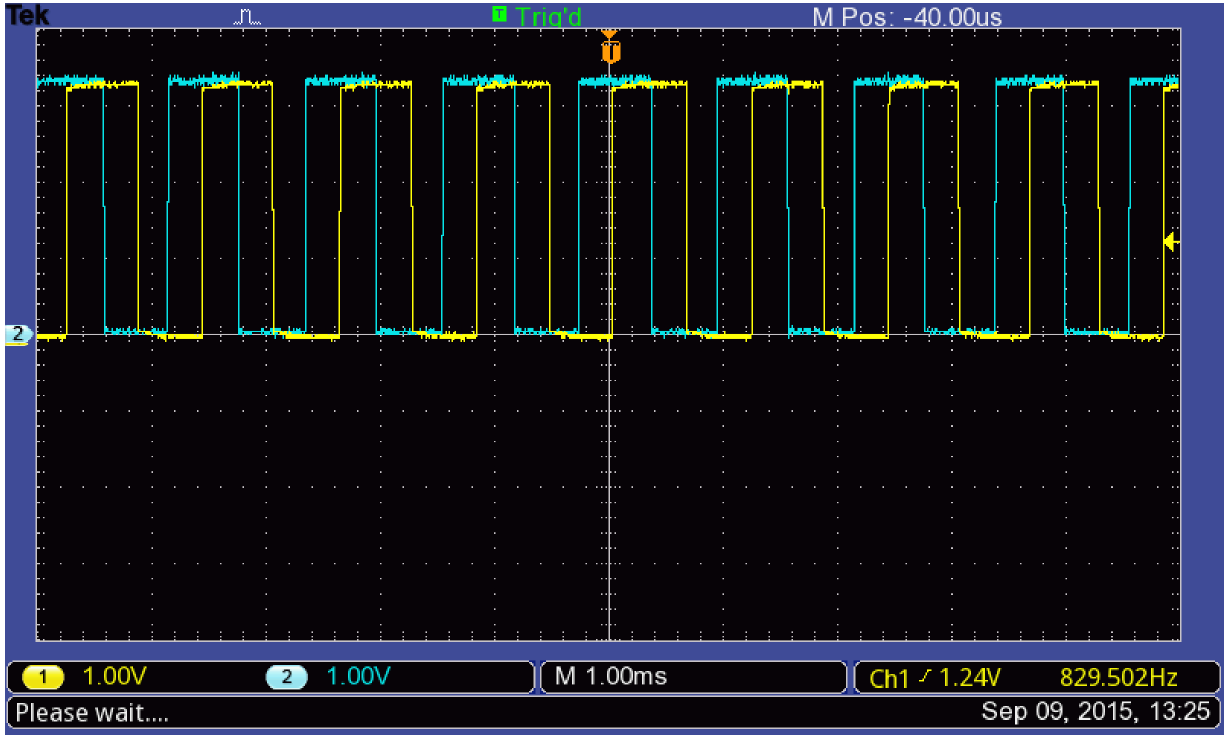
- A single sensor cannot distinguish forward vs reverse motion.
- To measure the direction of rotation, use two sensors.
- The direction is revealed by which sensor triggers first.



Measurements of two directions of rotation





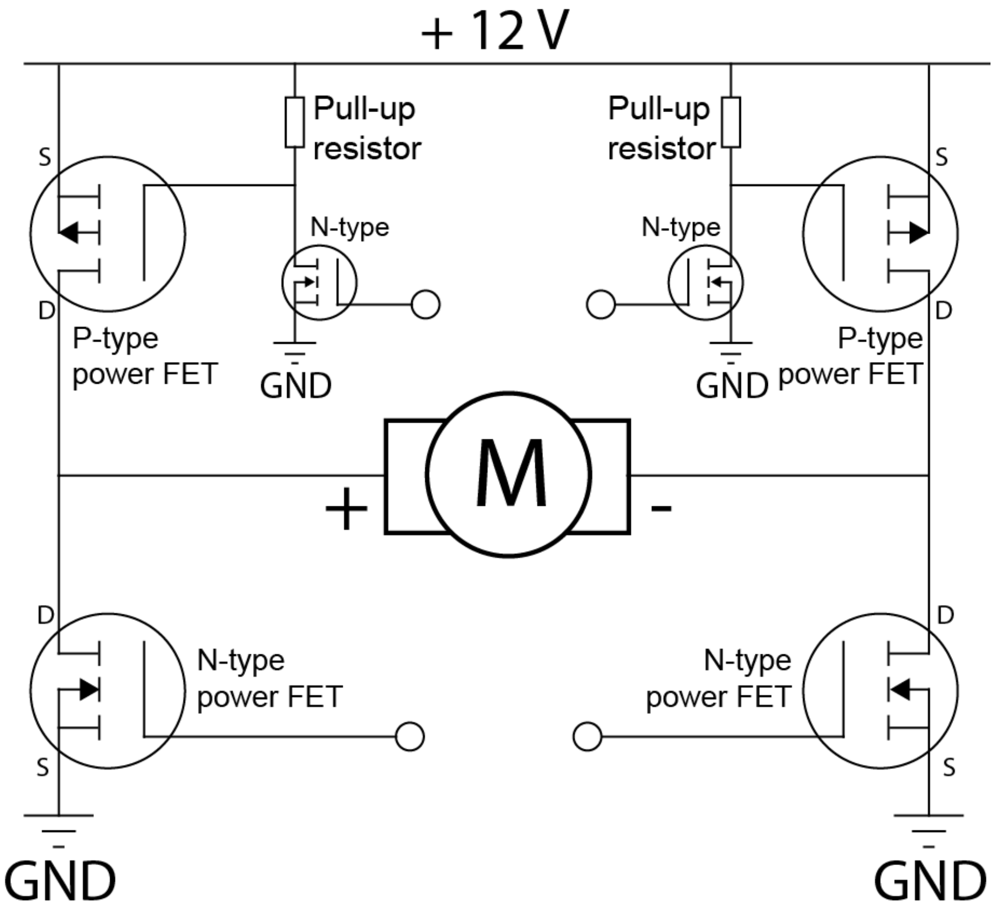


Using a Hall Effect sensor in Kinetis

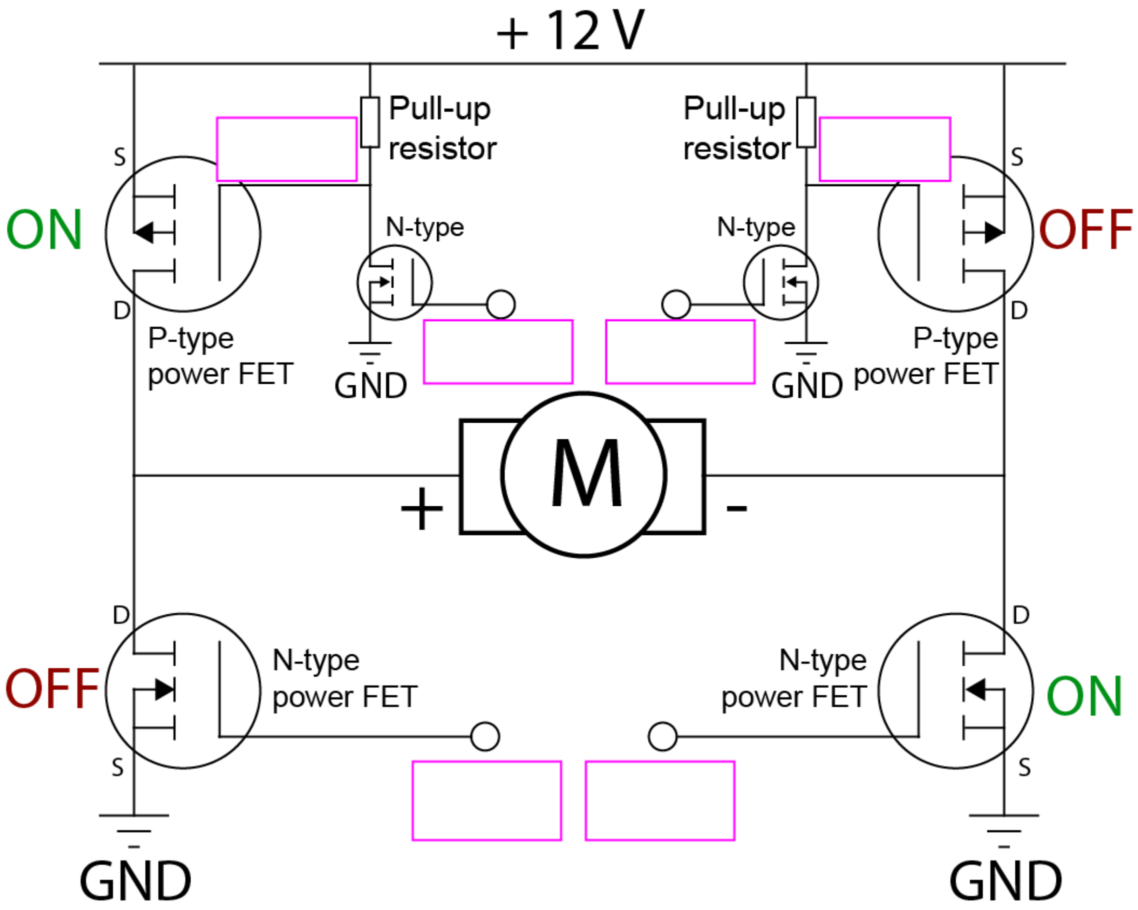
- The **Capture** component measures the time of a rising or falling edge.
- The **time between edges** gives the angular velocity.
- Count the **number of edges** to keep track of angular position.

- Alternatively: the FlexTimer module contains hardware support for quadrature phase decoding.
  - It will increment/decrement the FlexTimer counter according to the relative phase of the A and B inputs.
  - This functionality is not exposed in Processor Expert.

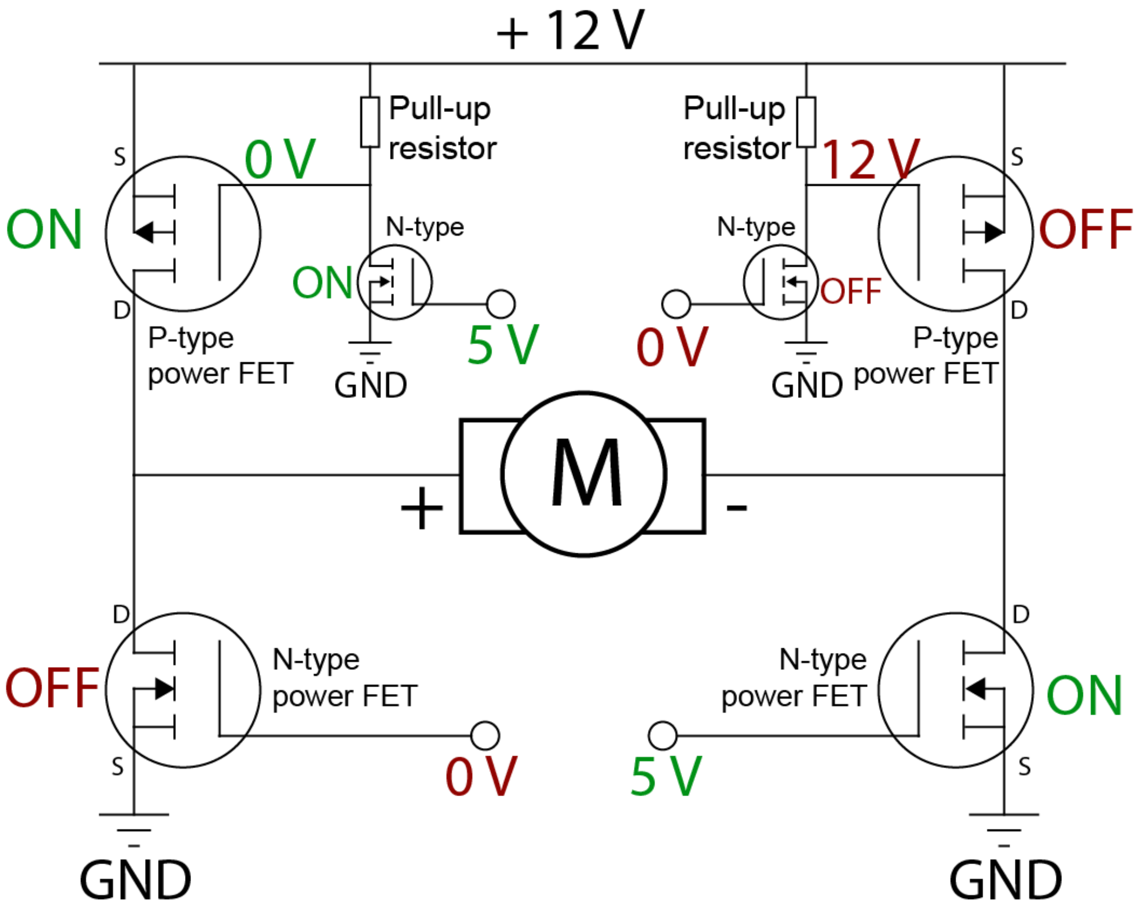
Review: H bridge control circuit for DC motors



Voltage levels to control the H bridge



Voltage levels to control the H bridge



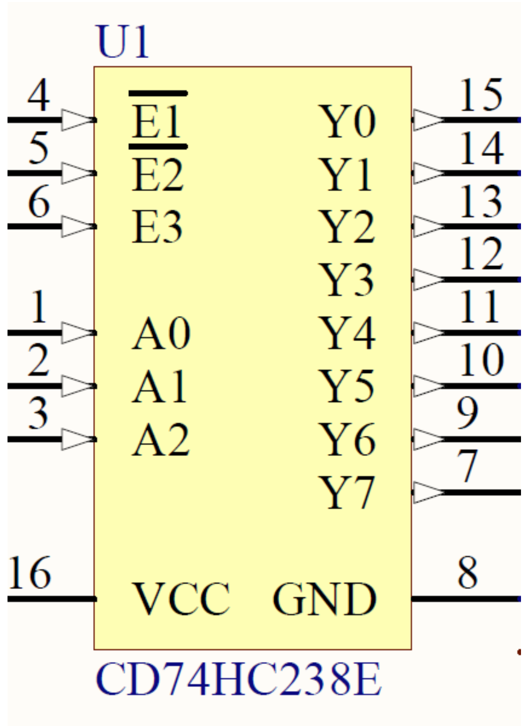
Logic levels

- The FRDM board generates 3.3 V signals.
- What if we needed 5 V logic levels?

- **Answer**: introduce discrete logic chips.

The 7400 series

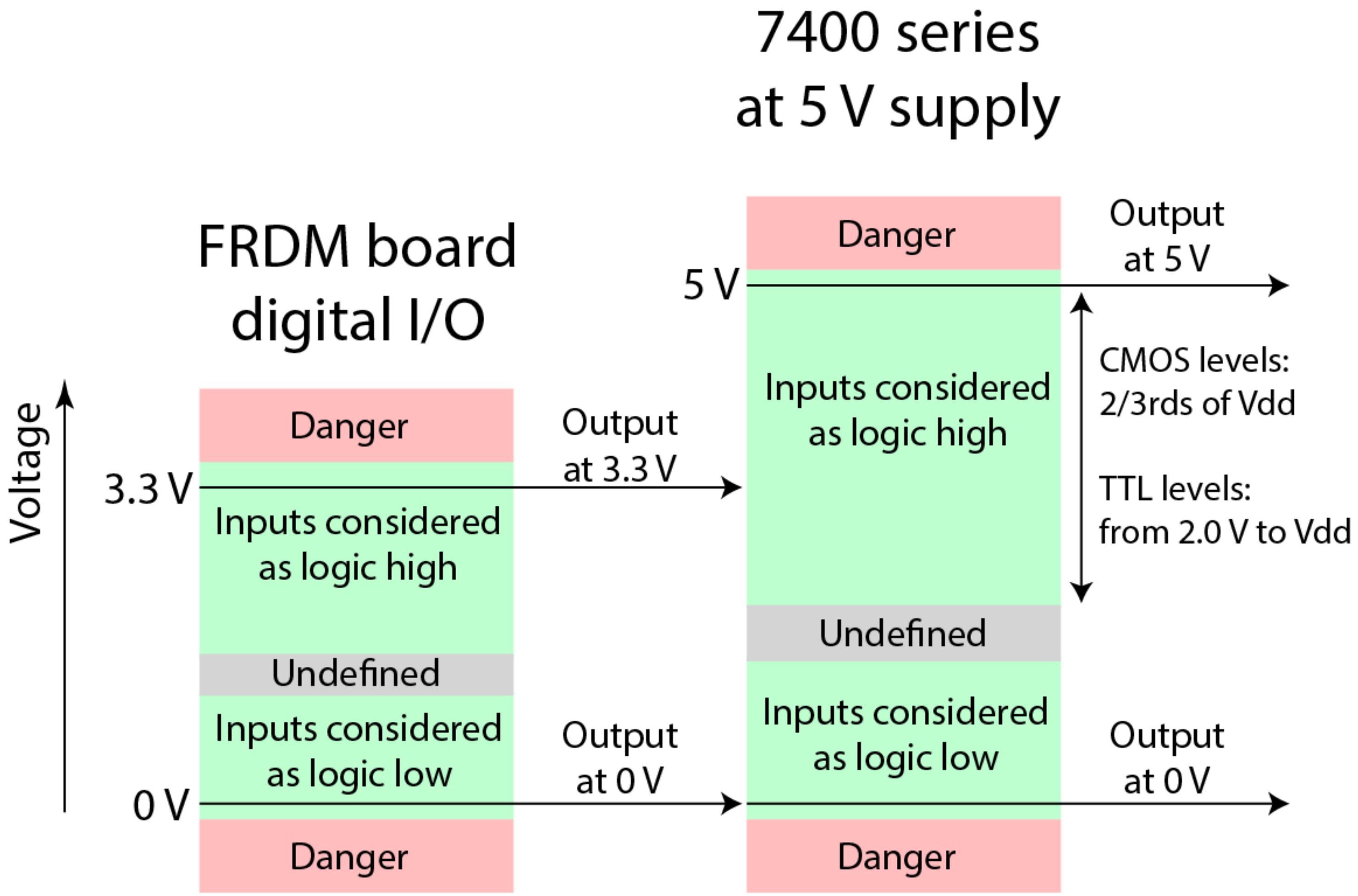
- The “7400 series” are a family of digital logic chips.
- Examples:
  - 7400 = 4 x NAND gates
  - 7402 = 4 x NOR gates
  - 7404 = 6 x inverters (“hex inverter”)
  - 74238 = 3-to-8 line decoder (used in Assign. 1)
- Originally made by Texas Instruments, since then many other manufacturers copied the pin layouts.
- Now a “de-facto” standard for discrete logic.



7400 technologies

- The original 7400 series chips (from the 1960s and 1970s) used transistor-transistor logic (TTL) technology made from BJTs.
- Now, two or three letters after the “74” indicate the technology:
  - 74HCxx = high speed CMOS
  - 74HCTxx = high speed CMOS compatible with TTL voltages
- A letter suffix indicates the package:
  - **74HCxxN = through hole dual inline package**
  - 74HCxxD = surface mount small outline package

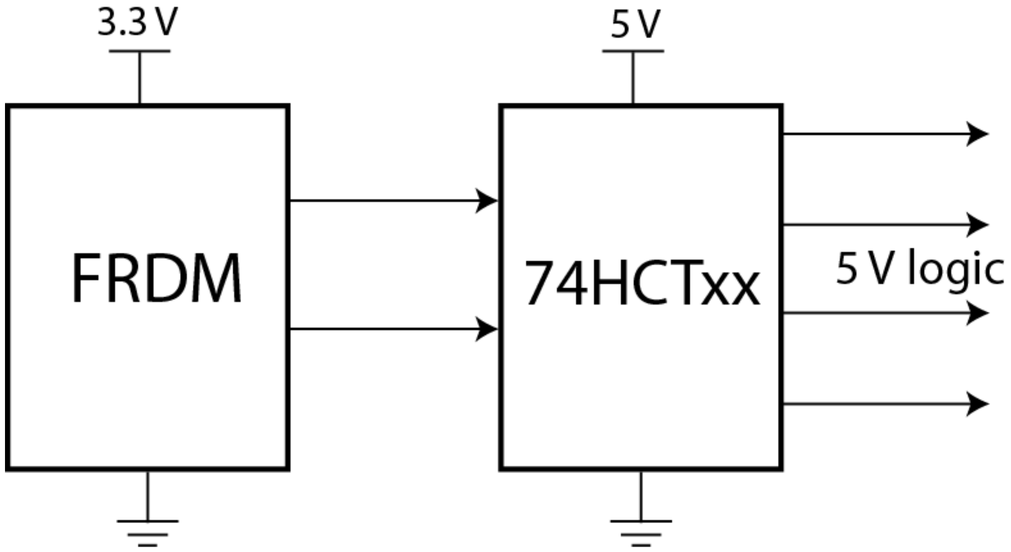
Voltage levels



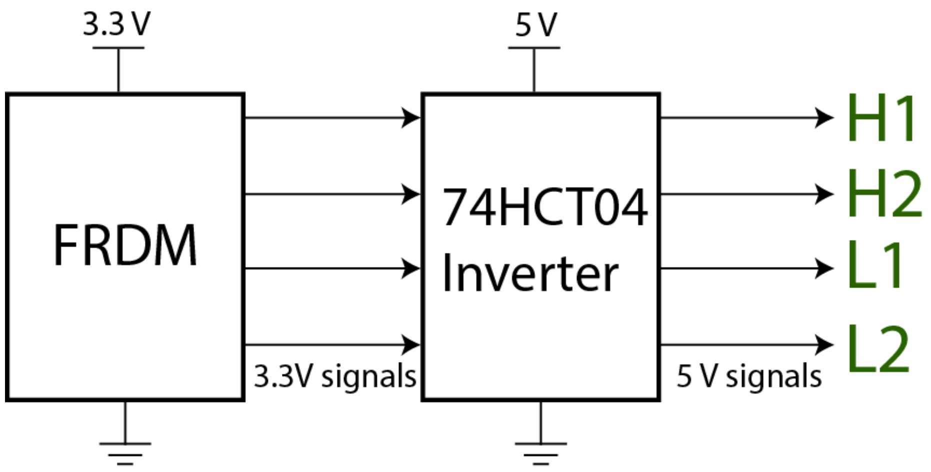
Always check the datasheet for tolerated voltage levels!

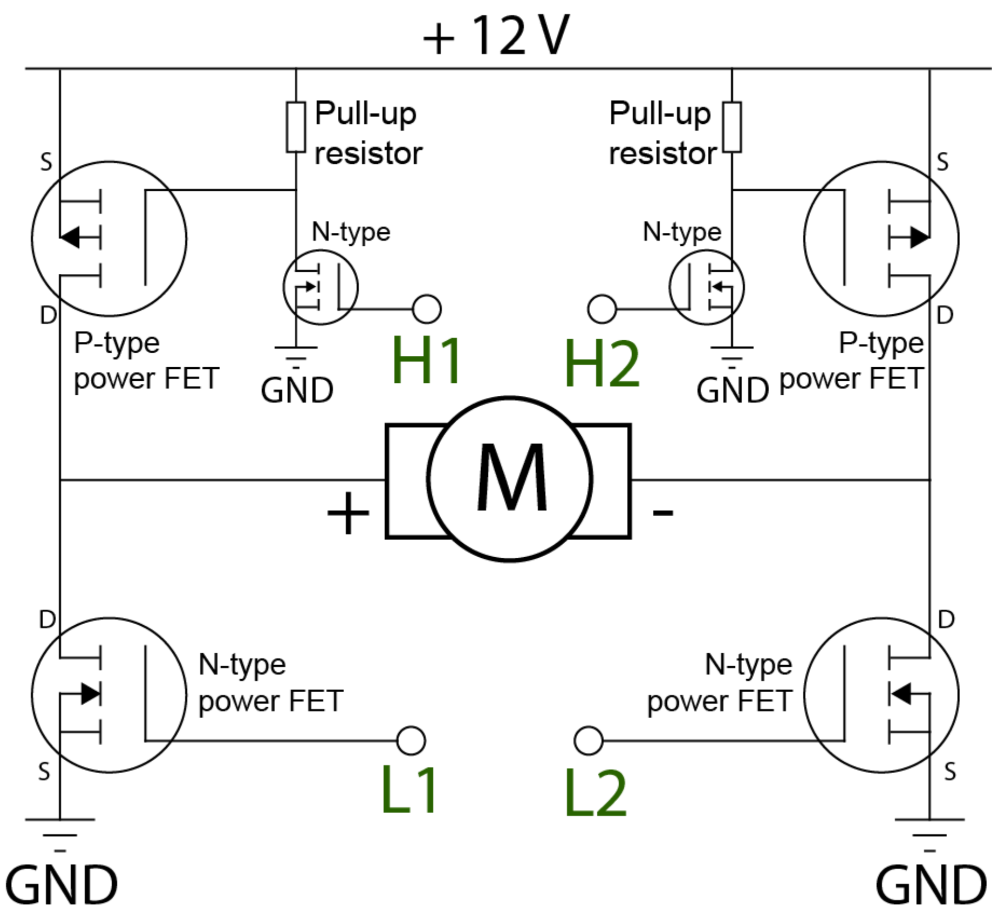
Generating 5 V logic to drive the H bridge

- 74**HC**xx: CMOS voltage levels
  - High level typically 2/3rds of supply but this varies (check the datasheet)
  - (2/3)*5V = 3.3 V.
- 74**HCT**xx: TTL voltage levels
  - High level = 2.0 V
- Use **HCT** components (to guarantee that 3.3 V is considered to be logic high)



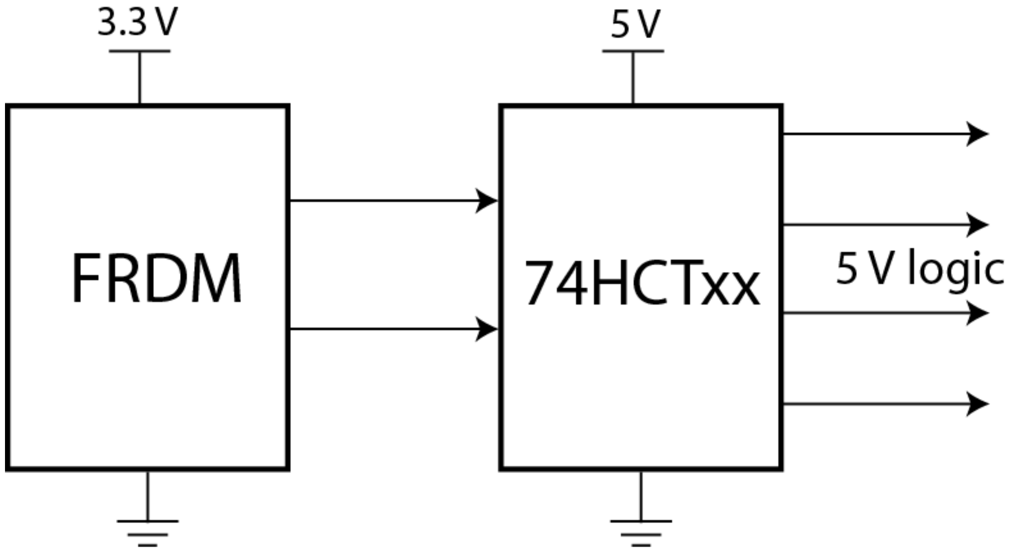
First attempt





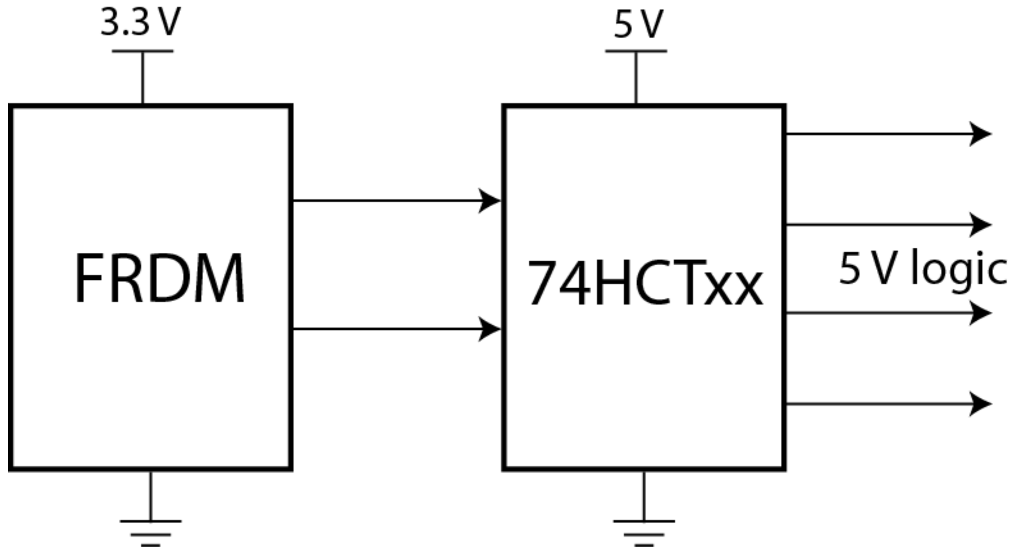
Advantages of discrete components

- Useful for guaranteeing electrical safety, e.g. interlock/fault signals.
- Discrete logic is simple enough that you can **prove** that every state is safe.
- It’s impractical or impossible to prove that software is completely correct.



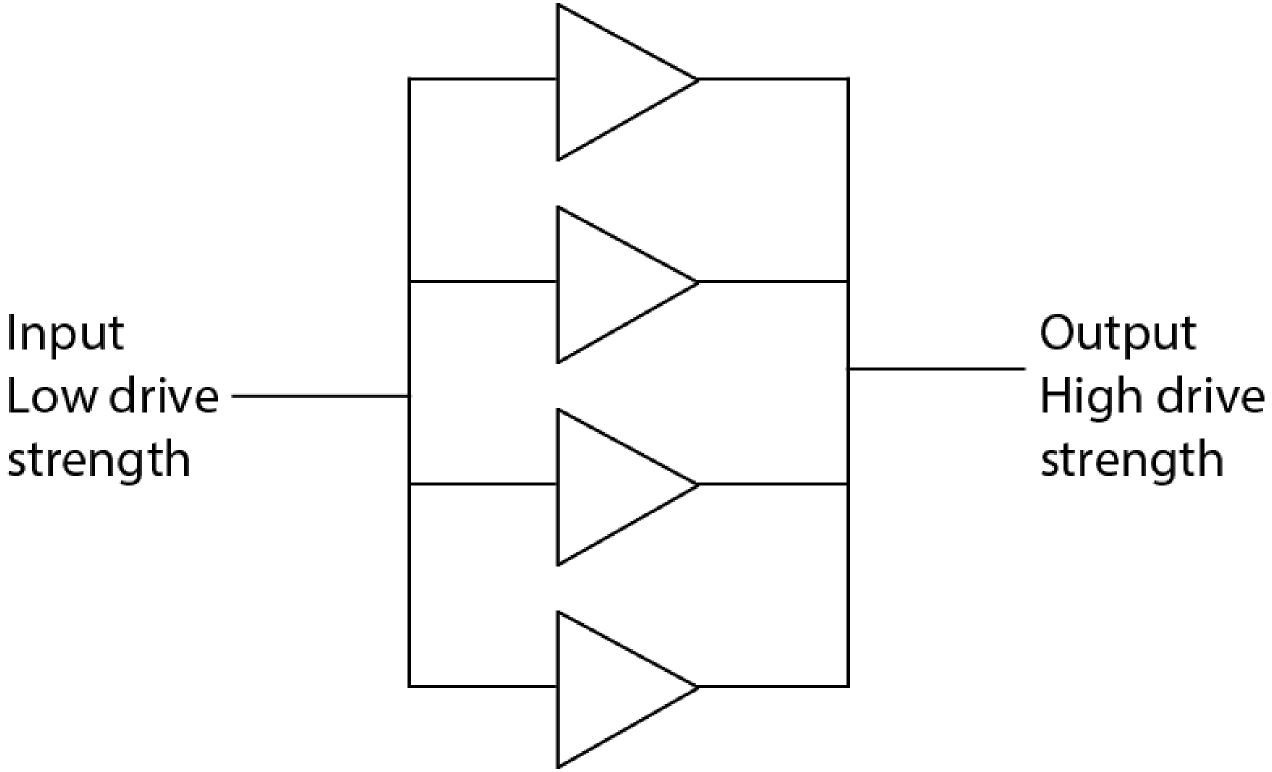
Advantages of discrete components

- Isolates expensive or harder to replace electronics behind a CMOS buffer.
- A short circuit or overvoltage will probably destroy the 7400 chip instead of the microprocessor.



Advantages of discrete components

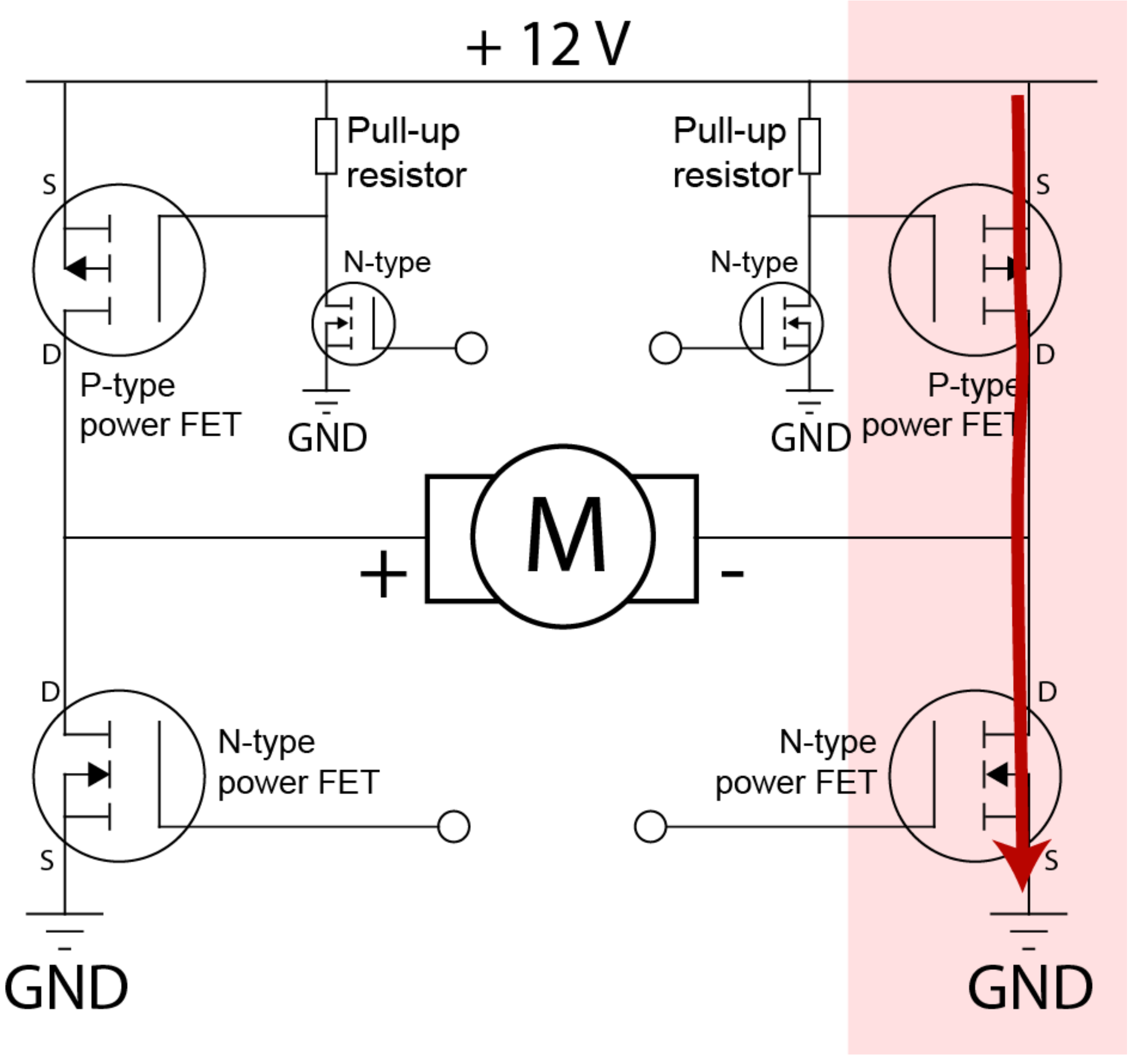
- Voltage levels can be changed.
- It's possible to "gang together" multiple outputs to increase the drive strength.
  - 4 x 25 mA = 100 mA drive.



H bridge logic wishlist

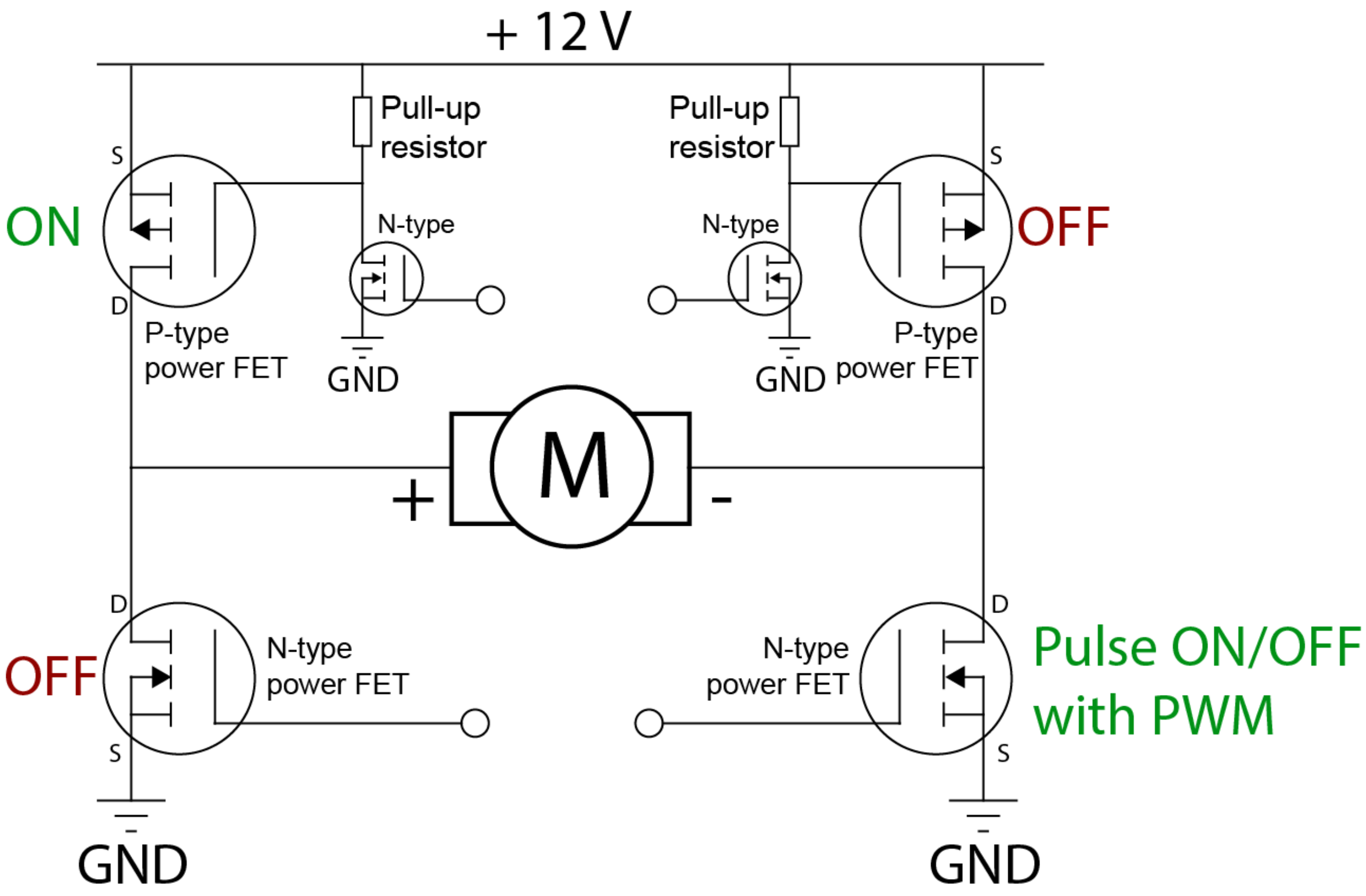
Wishlist:

- **Electrical safety!** Make it impossible to short the supply.
- Can you promise that your software will never fail? Of course not.
- You should implement protection in hardware.



H bridge logic wishlist

- Optional: only drive the low side with PWM.
- Advantage: faster switching because the low side gate is actively driven.
- Disadvantage: low and high side cannot connect to the same logic output.



Designing with discrete logic

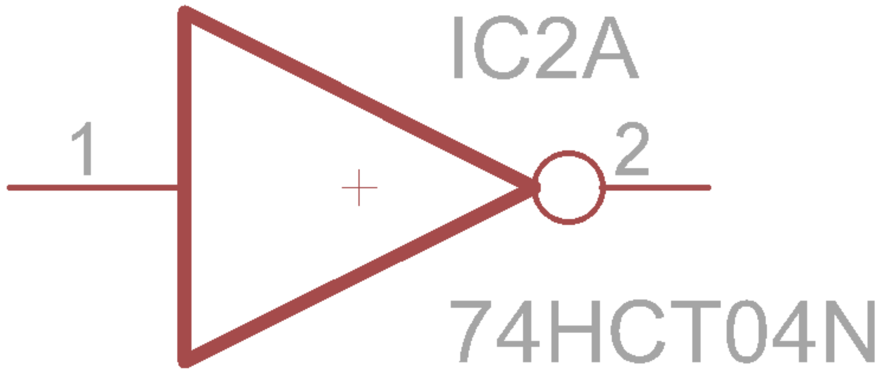
Typical components:

- Inverter
- NAND (not AND)
- NOR (not OR)
- AND
- OR
- XOR (exclusive OR)

Inverter

- Written $\bar{A}$, !A, not A, ¬A
- The circle on the output pin indicates the inversion.

A	!A
0	1
1	0

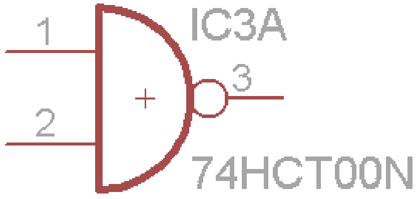


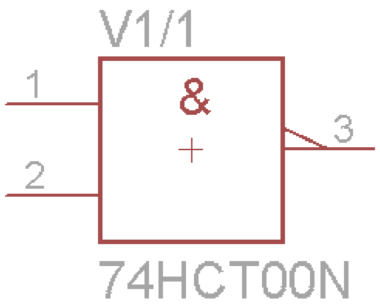
NAND (not AND)

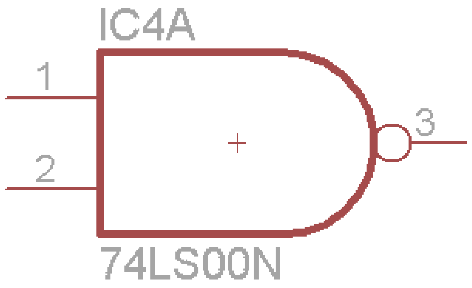
NAND:

- $\overline{AB}$
- !(A and B)

A	B	A nand B
0	0	1
0	1	1
1	0	1
1	1	0





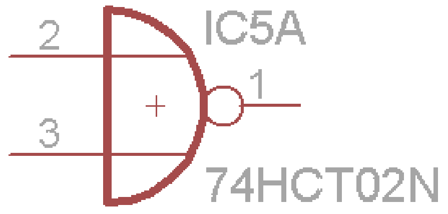


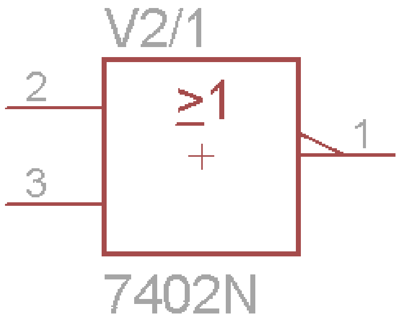
NOR (not OR)

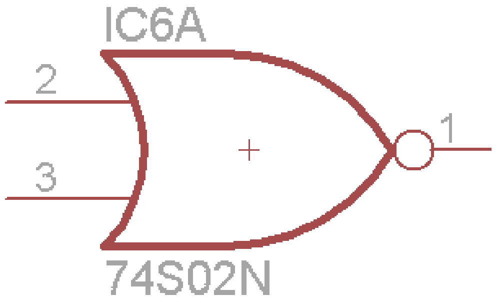
NOR:

- $\overline{A + B}$
- !(A or B)

A	B	A nor B
0	0	1
0	1	0
1	0	0
1	1	0







NAND and NOR

- NAND and NOR are universal gates, i.e. all logic can be built using either of these.
- For simple designs, often a single NAND or NOR chip can be used.

De Morgan’s Theorem

- Useful for algebraic manipulation of digital logic

$$\overline{AB} = \bar{A} + \bar{B}$$

$$\overline{A + B} = \bar{A}\bar{B}$$

In words: to bring an inversion into the brackets, swap “and” and “or”.

Boolean Algebra Primer

- See handout sheet

Example 1

- A motor is enabled when R = 1.
- The motor should run if either of two switches (S1, S2) are on (logic high).
- The motor should never run if a fault signal (F) is high.

1. Write down the truth table (use x = don’t care to abbreviate).

S1	S2	F	R

Example 1: truth table

S1	S2	F	R
x	x	1	0
1	x	0	1
x	1	0	1

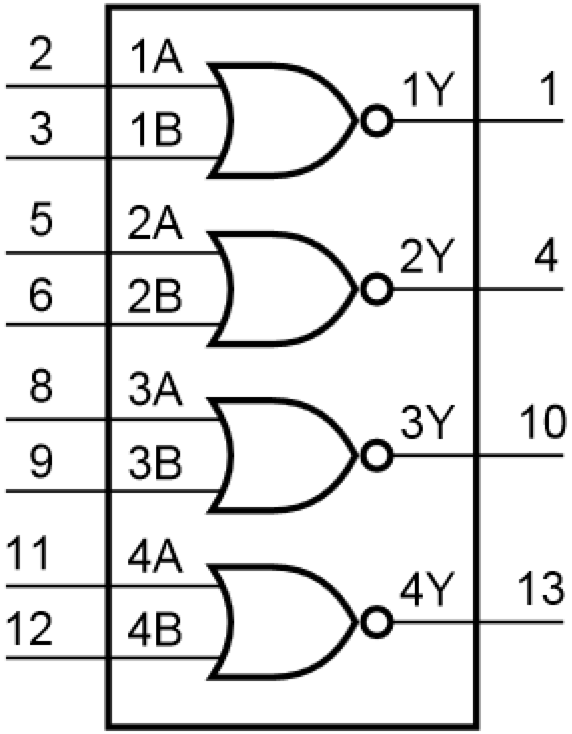
- Read off each row in which R=1, and OR them together.

$$R = S_1\bar{F} + S_2\bar{F}$$

This is the **minterm canonical form** (i.e. the sum of products).

Example 1: Algebraic rearrangement

- Implement this logic with a single 74HC02N chip.
- This is 4 x NOR gates in a single IC.



Example 2

- Consider a microprocessor controlled motor with:
  - Software generated enable signal (active high, i.e. high = run the motor)
  - Fault detection (active high, i.e. high = fault exists, don’t run the motor)
  - Manual override (active high, i.e. high = don’t run the motor)
- Design discrete logic to implement these requirements.

- Step 1: Draw the truth table (use X = don’t care to abbreviate)
- Step 2: Write down a Boolean expression to implement this logic.

Example 2: Truth table

E	F	M	R
0	x	x	0
x	1	x	0
x	x	1	0
1	0	0	1

- Place in minterm canonical form by reading off all rows where R is 1.

$$R = E\bar{F}\bar{M}$$

- Manipulate this expression to implement with 4 NOR gates.

Summary

- Discrete digital logic provides:
  - Electrical buffering/isolation.
  - Voltage gains (usually with TTL compatible parts).
  - Interlock systems to protect against software failure.
- NOR and NAND gates are commonly used because these are universal (they can implement any logic)