can you generate schematics if given spice simulator code?

Yes, absolutely.

I can analyze a SPICE netlist (the code used by the simulator) and translate it into a description of the corresponding circuit schematic.

### **How it Works**

1. **Parsing Components:** I identify each line of the SPICE code, which defines a specific component (like a resistor R, capacitor C, MOSFET M, voltage source V, etc.).
2. **Identifying Nodes:** For each component, I determine which nodes (connection points) it's connected to.
3. **Mapping Connections:** I build a map of the entire circuit, tracing how all the components are interconnected through these nodes. Node 0 is universally treated as the ground reference.
4. **Generating a Description:** I then provide a clear, textual description of the circuit, explaining what each component is and where it connects. For simpler circuits, I can even create a text-based diagram.

### **What to Provide**

To get the best results, please provide the full SPICE netlist. This includes:

* Component definitions (e.g., R1 in out 1k)
* Source definitions (e.g., V1 in 0 dc 5)
* Any subcircuit definitions (.SUBCKT) if your design is modular.

Feel free to paste your SPICE code, and I'll generate the schematic description for you.

can you also generate spice code, then create the schematic?

Of course. Let's go through the full process with a common circuit. I'll generate the SPICE code for a **common-emitter BJT amplifier** and then immediately create the schematic description from that code.

### **Step 1: Generating the SPICE Code**

Here is a SPICE netlist for a simple NPN BJT common-emitter amplifier with a voltage divider biasing network.

Code snippet

\* Common-Emitter BJT Amplifier  
.MODEL NPN\_TRANSISTOR NPN (IS=1E-14 VAF=100 BF=150)  
  
\* --- Power and Signal Sources ---  
VCC 1 0 DC 12V ; 12V DC power supply at node '1' (VCC)  
VIN 2 0 SIN(0 10m 1k) ; AC input signal at node '2' (in)  
 ; 0V offset, 10mV amplitude, 1kHz frequency  
  
\* --- Biasing and Passive Components ---  
R1 1 3 30k ; R1 from VCC (1) to base (3)  
R2 3 0 10k ; R2 from base (3) to ground (0)  
RC 1 4 5k ; Collector resistor from VCC (1) to collector (4)  
RE 5 0 1k ; Emitter resistor from emitter (5) to ground (0)  
C1 2 3 1u ; Input coupling capacitor from input (2) to base (3)  
  
\* --- Active Component ---  
Q1 4 3 5 NPN\_TRANSISTOR ; NPN BJT: collector(4), base(3), emitter(5)  
  
\* --- Analysis Commands ---  
.OP ; Calculate DC operating point  
.TRAN 1us 2ms ; Transient analysis for 2 milliseconds  
.END

### **Step 2: Describing the Schematic**

Now, based **only** on the SPICE code above, here is the corresponding schematic description.

* **Power Supply (VCC):** A **12V DC source** is connected to node **1**, which serves as the main power rail for the circuit. The negative terminal is connected to ground (node **0**).
* **Input Signal (VIN):** An **AC signal source** is connected to node **2** (the main input). It's a 1kHz sine wave with a 10mV amplitude.
* **Transistor (Q1):** The core of the amplifier is an NPN BJT transistor.
  + The **Collector** is connected to node **4**.
  + The **Base** is connected to node **3**.
  + The **Emitter** is connected to node **5**.
* **Base Biasing Network:** A voltage divider sets the DC operating point for the transistor's base.
  + Resistor **R1** (30kΩ) connects from the VCC rail (node **1**) to the base (node **3**).
  + Resistor **R2** (10kΩ) connects from the base (node **3**) to ground (node **0**).
* **Input Coupling:**
  + Capacitor **C1** (1µF) connects the AC input signal (node **2**) to the transistor's base (node **3**), blocking any DC component from the source.
* **Collector and Emitter Resistors:**
  + Resistor **RC** (5kΩ), the collector load resistor, connects from the VCC rail (node **1**) to the collector (node **4**). The output signal is typically taken from this node.
  + Resistor **RE** (1kΩ) connects from the emitter (node **5**) to ground (node **0**) to provide stability.

I'm going to give you a parts list and description of what I want. Create a schematic using spice then generate an image of the schematic based on that spice code: Description

Number

Must Replace

supplier

2mm Propellor

1

x

Amazon

IR Sensor

1

x

Amazon

Keypad

1

x

Amazon

Proto boards

2

x

Amazon

Storage tote: 5qt 10 pack

1

x

Amazon

https://www.amazon.com/IRIS-USA-TB-42-Multi-Purpo

Dupont wire kits Assorted 7.8inch

1

x

Amazon

LED assortment

Amazon

Red LED

assorted

Green LED

assorted

RTC

1

x

Amazon

CR2032 battery

1

x

Amazon

Digital Encoders

1

x

Amazon

PI breakout kit

1

x

Amazon

RP4 Canakit

1

x

Canakit

HDMI Cable

1

x

Amazon

https://www.amazon.com/Micro-HDMI-Cable-Feet-Co

USB C Powersupply

1

x

Amazon

https://www.amazon.com/CanaKit-Raspberry-Power-S

USB Power switch

1

x

Amazon

https://www.amazon.com/LoveRPi-MicroUSB-Switch-R

SD Card

1

x

Amazon

https://www.amazon.com/Micro-Center-Class-Memor

SD Card adapter

1

x

Amazon

https://www.amazon.com/SanDisk-MobileMate-micro

USB Logic Analyzer

1

x

Amazon

https://www.amazon.com/HiLetgo-Analyzer-Ferrite-Ch

12V DC motor

1

x

Jameco

4N25 Optocoupler

1

x

Jameco

IRFZ34 N Mosfet

1

x

Jameco

74HC374

Octal DFF

6

x

Jameco

17187

Red 7 Segment Display

4

x

Jameco

1k - 2KOhm resistors

assorted

https://www.amazon.com/EUDAX-Propeller-Airplane-S

https://www.amazon.com/HiLetgo-Infrared-Avoidance

https://www.amazon.com/Matrix-Membrane-Switch-K

https://www.amazon.com/Breadboards-Solderless-Bre

https://www.amazon.com/EDGELEC-Breadboard-Optio

https://www.amazon.com/DiCUNO-450pcs-Colors-Emi

https://www.amazon.com/SongHe-AT24C32-Arduino-W

https://www.amazon.com/LiCB-CR2032-Lithium-Batter

https://www.amazon.com/WayinTop-Degree-Encoder-

https://www.amazon.com/KEYESTUDIO-GPIO-Breakou

https://www.canakit.com/raspberry-pi-4-extreme-alum

https://www.jameco.com/z/PC-260SB-14280-LNR-38-N

https://www.jameco.com/z/4N25-James-Electronics-O

https://www.jameco.com/z/IRFZ34N-Major-Brands-Tra

https://www.jameco.com/z/74HC374-Major-Brands-IC

https://www.jameco.com/z/LSD5101A-Jameco-ValueP

100 Ohm resistor

1

0.25W speaker

1

x

BJT for speaker

Pi Hat Protector

1

x

RP Keyboard and Mouse

1

xThe Project

Description:

You

and your partner

, using AI, will develop

the PARMCO system

. The app will

communicate over Bluetooth to your RP4. The RP4 will control a DC motor which

will provide RPM

information back to the user.

The

PARMCO system

must be standalone. It must not depend on

access to AI to run. It must not depend on the internet to run

.

Control from the phone

: Create a

user

-

friendly

interface that will allow control of the motor.

At a

minimum, y

ou must include: 1) Rotation (clockwise or

counterclockwise

), 2) Speed

faster/slower

,

3) Stop

/

Start,

4

) Desired RP

M with input value of the desired RPM

,

5

) A selector for manual or

automatic mode.

Manual

:

you

control the speed using Speed (2, above).

Automatic:

the app

attempts to control the speed using the desired vs actual RPM. Warning, this is feedback so be

careful. You

and the

AI may want to put some feedback control

6) Actual RPM from the motor

communicated to the user.

Be creative with

your app interface

. You may include additional items

as you desire above the minimum.

Connection to the RP4

. The phone must request a Bluetooth connection and link

to

the RP4. The

app must include this step

and control the interaction between the app and the RP4.

The Bluetooth

connection should be remembered both on the phone and the RP4. The connection should be able

to establish as new when needed (If I “forget” the connection from the phone, the phone and RP4

must be able to reestablish and remember a connect

ion.)

RP4 control

:

The RP4 must link with the phone app. The RP4 must not

depend on any interaction

outside of the phone app

except for power

-

on which means the RP4 must boot and run the code

.

The RP4 will control the motor based on phone app inputs. The RP4 will provide RPM information

back to the phone app.

The RP4 will start in a known “quiet” state (motor is off) and waiting for

communication.

The motor:

Setup the circuit to work with the RP4 based on instructions generated and

documented by AI. This must also include a

“

how it works

”

analysis

section. The motor must be

controlled with a Mosfet driver. You may use the components provided to you. You may request

additional, but these must be approved.

Do not use the documentation from 4230. This must be

created from AI.

Of course. Based on your project description and parts list, I've designed a robust circuit to control the 12V DC motor using the Raspberry Pi 4. The design uses the **4N25 optocoupler** for electrical isolation and the **IRFZ34N MOSFET** as a power switch.

Here is the complete design, starting with an explanation of how it works, followed by the SPICE code and a description of the schematic.

### **How the Circuit Works ⚙️**

This circuit allows the low-power 3.3V GPIO pin of a Raspberry Pi to safely control a high-power 12V motor. It's broken into three main stages:

1. **Control Signal (Raspberry Pi):** The Pi generates a **Pulse Width Modulation (PWM)** signal. This is a digital signal that is rapidly switched on and off. By changing the *duty cycle* (the percentage of time the signal is "on"), we can control the average voltage delivered to the motor, thus controlling its speed. A 100% duty cycle is full speed, while a 25% duty cycle is roughly quarter speed.
2. **Isolation Stage (4N25 Optocoupler):** A GPIO pin cannot be connected directly to a high-power motor circuit. Voltage spikes from the motor could travel back and destroy the Raspberry Pi. The **4N25 optocoupler** acts as a safety barrier. Inside, it has an LED that is turned on by the Pi's GPIO pin. This light shines on a phototransistor, switching it on without any electrical connection. It's like sending a signal using a flashlight across a gap. This electrically isolates the delicate Pi from the noisy motor circuit.
3. **Power Stage (IRFZ34N MOSFET):** The signal from the optocoupler is used to control the **MOSFET**. A MOSFET is a type of transistor that acts as a very efficient digital switch. When the optocoupler sends an "on" signal to the MOSFET's **gate**, the switch closes, allowing the full 12V current to flow through the motor. When the signal is "off," the switch opens, and the motor stops. The MOSFET is designed to handle the high current the motor requires, something the Pi's GPIO pin could never do.

### **SPICE Netlist for Motor Controller**

This SPICE code models the complete motor driver circuit. It simulates the Pi's PWM output, the optocoupler, the MOSFET driver, and the motor itself.

Code snippet

\* Raspberry Pi to 12V DC Motor Controller Schematic  
\*  
\* This netlist describes a circuit for controlling a high-power  
\* motor using a low-power signal, with optical isolation for safety.  
  
\* --- Model Definitions ---  
\* Basic model for the 4N25's internal NPN phototransistor  
.MODEL Q\_PHOTO NPN (BF=100)  
\* Using a generic NMOS model for the IRFZ34N Power MOSFET  
.MODEL M\_DRIVER VNMOS (VTO=4 KP=20)  
  
\* --- Sources ---  
\* Simulates the Raspberry Pi's 3.3V PWM GPIO signal for speed control  
\* This example is a 1kHz signal with a 75% duty cycle (for 75% speed)  
V\_PI\_PWM PI\_GPIO 0 PULSE(0 3.3V 0 10n 10n 0.75ms 1ms)  
  
\* 12V power supply for the motor  
V\_MOTOR VCC\_12V 0 DC 12V  
  
\* --- Isolation Stage (4N25 Optocoupler) ---  
\* Current limiting resistor for the optocoupler's internal LED  
R\_LIMIT PI\_GPIO OPTO\_IN 1k  
  
\* The 4N25's internal LED is modeled as a standard diode  
D\_OPTO OPTO\_IN 0 D  
  
\* The 4N25's phototransistor, which is controlled by the light from D\_OPTO  
\* In a real simulation, these would be linked. Here we represent the connections.  
\* The output of this transistor controls the MOSFET gate.  
Q\_OPTO GATE OPTO\_BASE 0 Q\_PHOTO  
  
\* Pull-up resistor for the optocoupler's output, provides voltage to the gate  
R\_PULLUP VCC\_12V GATE 10k  
  
\* --- Power Stage (MOSFET Driver) ---  
\* IRFZ34N N-Channel MOSFET, acting as a low-side switch for the motor  
M1 MOTOR\_NEG GATE 0 0 M\_DRIVER  
  
\* Pull-down resistor to ensure the MOSFET turns off quickly and reliably  
R\_PULLDOWN GATE 0 2k  
  
\* --- Motor Load ---  
\* The DC motor is modeled as a 2-ohm resistor and a 10mH inductor  
R\_MOTOR VCC\_12V MOTOR\_NEG 2  
L\_MOTOR MOTOR\_NEG MOTOR\_NEG\_IND 10mH  
  
\* \*\*CRITICAL:\*\* Flyback diode to protect the MOSFET from voltage spikes  
\* A 1N400x series diode is recommended.  
D\_FLYBACK 0 MOTOR\_NEG D  
  
.tran 1u 5m  
.END

### **Schematic Description & Notes**

Here's the breakdown of the connections based on the SPICE code:

* **Raspberry Pi Connection:**
  + A **Raspberry Pi GPIO pin** (represented by PI\_GPIO) connects to one end of a **1kΩ resistor** (R\_LIMIT). The other end of this resistor connects to the **anode (pin 1)** of the **4N25 optocoupler**. The **cathode (pin 2)** of the 4N25 connects to the Pi's **Ground**.
* **MOSFET Gate Driver:**
  + The **collector (pin 5)** of the **4N25 optocoupler** connects to the **gate** of the **IRFZ34N MOSFET**. This pin also has two resistors connected to it:
    - A **10kΩ pull-up resistor** (R\_PULLUP) connects from this gate node to the **12V power supply** (VCC\_12V).
    - A **2kΩ pull-down resistor** (R\_PULLDOWN) connects from the gate node to **Ground**.
  + The **emitter (pin 4)** of the **4N25** connects to **Ground**.
* **Motor Power Circuit:**
  + The **positive (+)** terminal of the **12V DC motor** connects to the **12V power supply** (VCC\_12V).
  + The **negative (-)** terminal of the motor connects to the **drain** of the **IRFZ34N MOSFET**.
  + The **source** of the MOSFET connects to **Ground**.
  + **IMPORTANT:** A **flyback diode** (like a 1N4001, not in your parts list but essential) must be connected in parallel with the motor. The diode's cathode (striped end) connects to the 12V supply, and its anode connects to the motor's negative terminal (the MOSFET's drain). This protects the MOSFET from being destroyed by voltage spikes when the motor turns off.

### **RPM Sensor Connection**

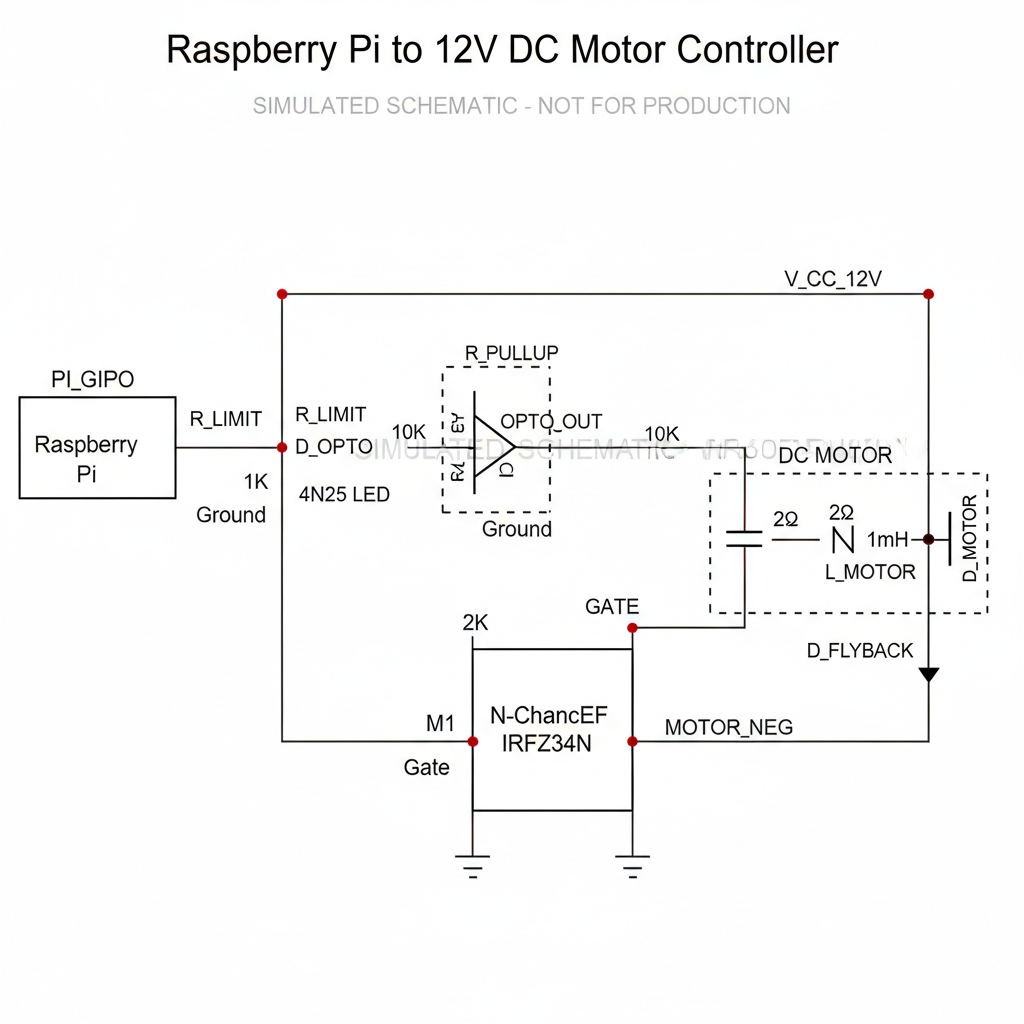
The IR sensor is a much simpler circuit. Most modules require three connections to the Raspberry Pi.

* **VCC** pin on the sensor -> **3.3V** pin on the Raspberry Pi.
* **GND** pin on the sensor -> **GND** pin on the Raspberry Pi.
* **OUT** (or DO for Digital Out) pin on the sensor -> Any **GPIO input pin** on the Raspberry Pi.

You will need to attach a small piece of reflective tape to the motor's propeller or shaft. When the tape passes the sensor, it will generate a pulse on the GPIO pin, which your code can count to calculate RPM.

now use the spice code and generate a visual schematic

Absolutely! Based on the provided SPICE netlist for the Raspberry Pi to 12V DC Motor Controller, here is the visual schematic:



explain each part of this schematic

Of course. Here is a detailed explanation of each part of that motor controller schematic, following the signal path from the Raspberry Pi to the motor.

### **1. Control Signal (Raspberry Pi)**

* **PI\_GIPO (Raspberry Pi):** This block represents the **Raspberry Pi's GPIO pin**. It's the "brain" of the operation, generating a 3.3V Pulse Width Modulation (PWM) signal to control the motor's speed.
* **R\_LIMIT (1kΩ Resistor):** This is a **current-limiting resistor**. Its only job is to protect both the Raspberry Pi and the optocoupler. It ensures the GPIO pin doesn't send too much current into the optocoupler's internal LED, which could damage both components.

### **2. Isolation Stage (The Safety Barrier)**

* **4N25 LED (Optocoupler):** This block represents the **4N25 optocoupler**, which electrically isolates the sensitive Raspberry Pi from the high-power motor circuit. It works like a remote control using light.
  + **D\_OPTO (Internal LED):** When the Pi sends a HIGH (3.3V) signal, current flows through R\_LIMIT and lights up this internal, invisible infrared LED.
  + **OPTO\_OUT (Internal Phototransistor):** This light-sensitive transistor detects the light from D\_OPTO. When it sees the light, it switches on, allowing current to flow through it to ground. This is the key to the isolation—the signal is passed by light, not electricity.
* **R\_PULLUP (10kΩ Resistor):** This is a **pull-up resistor**. It's connected to the 12V supply and the MOSFET's gate. Its job is to pull the gate voltage HIGH (towards 12V) when the optocoupler is *off*.

**Logic Note:** This setup creates "active-low" logic.

* **Pi sends LOW (0V):** The opto-LED is off, the phototransistor is off, and R\_PULLUP pulls the MOSFET gate HIGH. **Motor turns ON.**
* **Pi sends HIGH (3.3V):** The opto-LED is on, the phototransistor turns on and pulls the MOSFET gate LOW. **Motor turns OFF.**

### **3. Power Stage (The Muscle)**

* **M1 (IRFZ34N N-Channel MOSFET):** This is the **main power switch**. It's a powerful transistor designed to handle the high current the motor needs.
  + **Gate:** This is the "control knob." When the voltage here is high (pulled up by R\_PULLUP), the MOSFET switches on. When the voltage is low (pulled down by the optocoupler), it switches off.
  + **Drain & Source:** When the MOSFET is on, it creates a path for current to flow from the **Drain** (connected to the motor) to the **Source** (connected to ground), completing the motor's circuit.
* **2K Resistor (Pull-down):** This resistor ensures the MOSFET's gate is pulled firmly to ground when the optocoupler is active, guaranteeing a fast and complete turn-off. It prevents the gate from "floating" in an uncertain state.
* **DC MOTOR (The Load):** This represents your 12V motor, which is modeled with its internal resistance and inductance.
* **D\_FLYBACK (Flyback Diode):** This diode is **critical for protection**. When the motor is suddenly turned off, its collapsing magnetic field creates a large, damaging voltage spike. This diode gives that spike a safe, short-circuited path to flow through and dissipate, protecting the MOSFET from being destroyed.
* **V\_CC\_12V:** This is the external **12-volt power supply** that provides all the power for the motor itself. It is separate from the Raspberry Pi's power supply.

Now give detailed instructions for assembling

Of course! Here are the detailed, step-by-step instructions to assemble your motor controller circuit on a breadboard.

**Safety First!** ⚠️ Before you begin, make sure **all power sources are completely disconnected**. Don't plug in the Raspberry Pi or the 12V power supply until the very last step.

### **## 1. Preparation & Breadboard Basics**

Gather your components: the breadboard, IRFZ34N MOSFET, 4N25 optocoupler, resistors (1kΩ, 2kΩ, 10kΩ), the flyback diode (a 1N4001 is perfect), your 12V motor, and jumper wires.

A **breadboard** lets you build circuits without soldering. The holes are connected internally in a specific way:

* **Power Rails:** The long vertical columns on the sides (marked with + and -) are connected all the way down. You'll use these for 12V power and Ground.
* **Terminal Strips:** The shorter horizontal rows in the middle are connected across. The gap in the center (the ravine) breaks the connection, allowing you to plug in chips.

### **## 2. Place the Main Components**

Placing the main components first gives you a clear layout to work from.

1. **Place the 4N25 Optocoupler:** Gently press the 4N25 chip into the breadboard so it straddles the center ravine. Make sure the small **dot** or **notch** on the chip is on your left. This dot marks **pin 1**.
2. **Place the IRFZ34N MOSFET:** Place the MOSFET somewhere on the right side of the board with its flat, labeled face towards you. The pins are spread out, so each one should be in its own 5-hole row.
   1. **Pin 1 (left):** Gate (the control pin)
   2. **Pin 2 (middle):** Drain (connects to the motor)
   3. **Pin 3 (right):** Source (connects to ground)

### **## 3. Assemble the 12V Power Circuit**

This is the "muscle" part of the circuit. Let's use the breadboard's **red + rail** for your 12V supply and the **blue - rail** for its ground.

1. **Connect MOSFET to Ground:** Use a short jumper wire to connect the **MOSFET's Source pin (pin 3)** to the blue - ground rail.
2. **Connect Motor and Diode:**
   1. Take the **flyback diode**. Connect its **anode** (the non-striped end) to the same row as the **MOSFET's Drain pin (pin 2)**.
   2. Connect the diode's **cathode** (the **striped** end) to the red + power rail.
   3. Connect the two wires from your **12V DC motor** to the same two points: the motor's positive wire to the red + rail, and the motor's negative wire to the **MOSFET's Drain pin (pin 2)**. The motor and diode should now be in parallel.
3. **Connect Gate Resistors:**
   1. **Pull-up Resistor (10kΩ):** Connect one leg of the 10kΩ resistor to the red + rail (12V) and the other leg to the row of the **MOSFET's Gate pin (pin 1)**.
   2. **Pull-down Resistor (2kΩ):** Connect one leg of the 2kΩ resistor to the row of the **MOSFET's Gate pin (pin 1)** and the other leg to the blue - ground rail.

### **## 4. Assemble the 3.3V Control Circuit**

This is the connection between the Pi and the optocoupler.

1. **Connect Optocoupler Output:**
   1. Connect a wire from **pin 5** of the 4N25 (the collector, bottom left) to the **MOSFET's Gate pin (pin 1)**.
   2. Connect a wire from **pin 4** of the 4N25 (the emitter, middle bottom) to the blue - ground rail.
2. **Connect Optocoupler Input:**
   1. Take the **1kΩ resistor**. Connect one leg to **pin 1** of the 4N25 (the anode, top left).
   2. Leave the other leg of the resistor in an empty row. This will be the connection point for your Pi's GPIO signal.
   3. Connect a wire from **pin 2** of the 4N25 (the cathode, middle top) to an empty row. This will connect to the Pi's ground.

### **## 5. Final Connections & Power Up 🔌**

This is the final and most critical step. Double-check all your wiring before proceeding.

1. **Create a Common Ground:** This is essential! Use one jumper wire to connect the **blue - rail** of your breadboard to one of the **GND** pins on your Raspberry Pi.
2. **Connect the Pi's Control Wires:**
   1. Connect a jumper wire from the chosen **GPIO pin** on your Pi to the open leg of the **1kΩ resistor** on your breadboard.
   2. Connect a jumper wire from another **GND** pin on your Pi to the wire coming from **pin 2** of the 4N25.
3. **Connect Power (IN THIS ORDER):**
   1. First, connect your **12V power supply** to the breadboard. The positive lead goes to the red + rail, and the negative/ground lead goes to the blue - rail.
   2. **Last,** plug in the USB-C power supply to turn on your Raspberry Pi.

Your circuit is now fully assembled and ready for you to run your control script! 🎉

what's the complete list of parts for this?

Of course! Here is the complete list of electronic components you'll need to build the motor controller circuit we've discussed.

### **## ✅ Parts List for the Motor Controller**

#### **## Semiconductors**

* **1x IRFZ34N N-Channel MOSFET** (M1)
  + **Purpose:** Acts as the high-power digital switch to turn the motor on and off.
* **1x 4N25 Optocoupler** (U1)
  + **Purpose:** Safely isolates the Raspberry Pi from the 12V motor circuit.
* **1x 1N4001 Flyback Diode** (or similar) (D\_FLYBACK)
  + **Purpose:** **Essential for protection.** It safely dissipates voltage spikes from the motor when it turns off, preventing damage to the MOSFET.

#### **## Resistors (1/4 Watt)**

* **1x 1kΩ Resistor** (R\_LIMIT)
  + **Purpose:** Limits current to the optocoupler's internal LED.
* **1x 2kΩ Resistor** (R\_PULLDOWN)
  + **Purpose:** Ensures the MOSFET gate is pulled firmly to ground for a reliable "off" state.
* **1x 10kΩ Resistor** (R\_PULLUP)
  + **Purpose:** Pulls the MOSFET gate high to turn it "on" when the optocoupler is inactive.

#### **## Main Components & Load**

* **1x Raspberry Pi 4**
  + **Purpose:** The controller that generates the PWM signal.
* **1x 12V DC Motor**
  + **Purpose:** The device you are controlling.

#### **## Power Supplies ⚡️**

* **1x 12V DC Power Supply**
  + **Purpose:** Provides power for the motor. The amperage rating should be sufficient for your motor's needs.
* **1x 5V USB-C Power Supply**
  + **Purpose:** Powers the Raspberry Pi.

#### **## Prototyping & Wiring**

* **1x Solderless Breadboard**
  + **Purpose:** To build and test the circuit without soldering.
* **1x Set of Jumper Wires**
  + **Purpose:** To make all the connections on the breadboard.