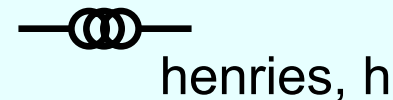
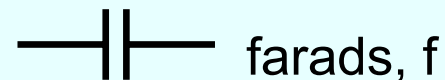
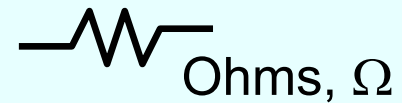


# Electronics Essentials for 2.017/2.019

# Reviewing Basics

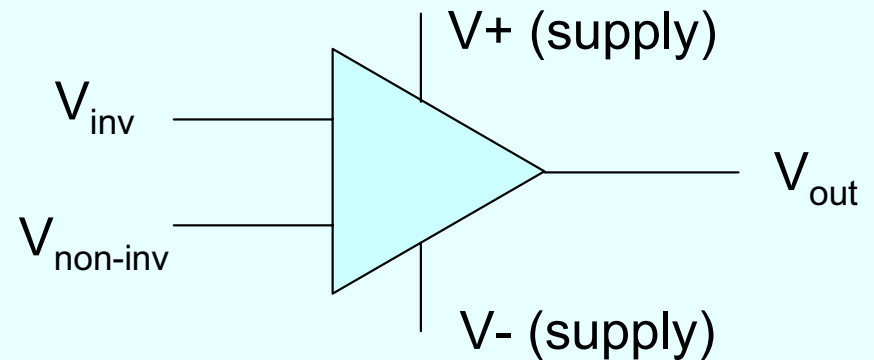
- *Kirchoff's Voltage rule*: voltages  $V$  at a node are the same.
- *Kirchoff's Current rule*: sum of currents  $i$  flowing into and out of a node is zero.
- *Analogy*: Voltage is like fluid pressure, current is like fluid volumetric flow rate. The wire is like a pipe.
- Resistor  $R$ :  $V = IR$ ,
  - Dissipation: Resistive Power  $P = I^2R = V^2/R$
  - Analogy: viscous losses in pipe flow
- Capacitor  $C$ :  $i = C dV/dt$ 
  - Analogy: a hydraulic accumulator
- Inductor  $H$ :  $V = L di/dt$ 
  - Analogy: inertia of water in a pipe



# The Op-Amp

Two inputs (called inverting and non-inverting); one output.

The output voltage is a HUGE gain multiplied by the difference between the inputs.



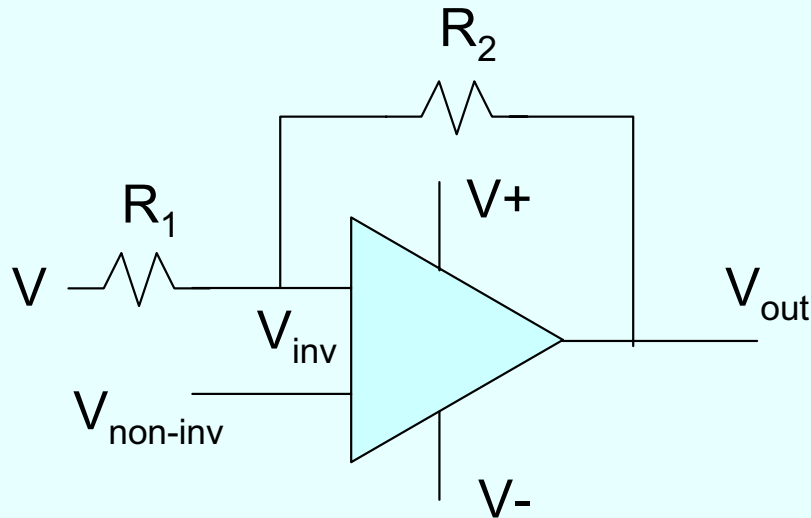
Horiwitz's & Hill's golden rules:

a. *The op-amp enforces (in proper use)*

$$V_{inv} = V_{non-inv}$$

b. *No current flows into the device at either input*

# Example Op-Amp: Adding a Voltage Bias



Voltage bias useful for bringing signal levels into the range of sensors.

The op-amp is discussed in detail by Horowitz and Hill, covering integrators, filters, etc.

$$(V - V_{\text{inv}})/R_1 = (V_{\text{inv}} - V_{\text{out}})/R_2 \quad \text{and} \\ V_{\text{inv}} = V_{\text{non-inv}} \rightarrow$$

$$VR_2 = V_{\text{inv}}(R_1 + R_2) - V_{\text{out}}R_1 \rightarrow$$

$$V_{\text{out}} = V_{\text{non-inv}}(R_1 + R_2)/R_1 - VR_2/R_1$$

Letting  $R_1 = R_2$ , then

$$V_{\text{out}} = 2V_{\text{non-inv}} - V$$

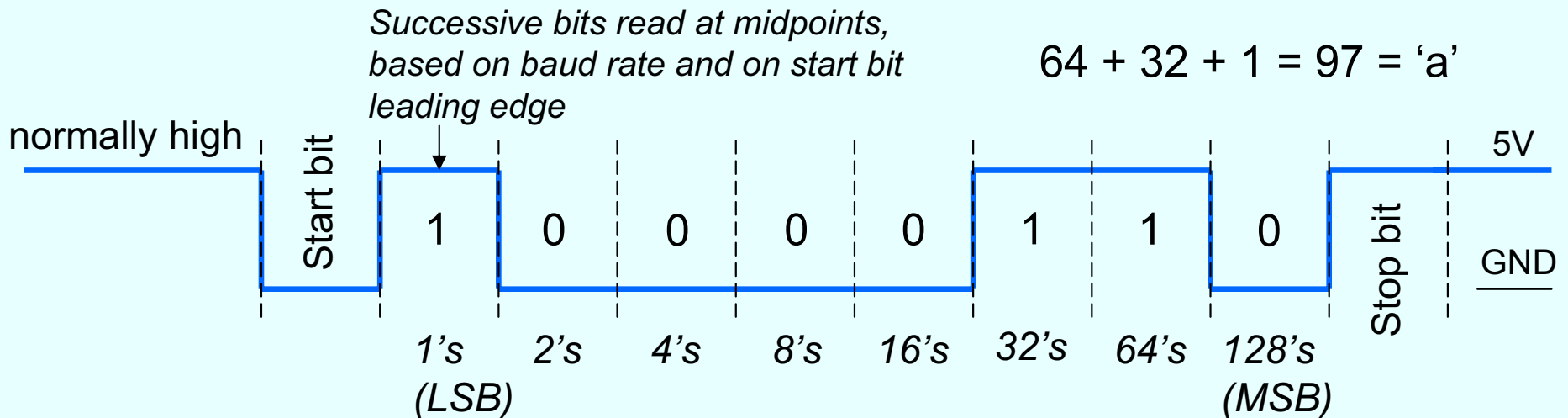
*The circuit inverts the input  $V$  and adds on  $2V_{\text{non-inv}}$*

*IF  $V_{\text{non-inv}}$  is ground, then  $V_{\text{out}}$  is just  $-V$ . This is just an inverting amplifier.*

# Serial Communications

- *How to transmit digital information fast and reliably over a few wires?*
- Examples: RS-232, RS-485, etc. refer to pins & wires
- A minimal case of RS-232 (DB25 connector is full case):
  - Asynchronous operation; both sides agree on BAUD rate
  - Three wires: send (TX), receive (RX), ground
  - No error checking! No flow control!

## EXAMPLE using CMOS components:



# EXAMPLE: A GPS String

- Garmin GPS25 series – Smart embedded device!
- Similar to TT8's interface with you – I/O strings are passed through a serial port
- Reconfigurable through special commands
- Output at 1Hz
- String maintains exactly the same syntax: e.g.,

```
$GPRMC,hhmmss,V,  
ddmm.mmmm,N,dddmm.mmmm,E,  
000.0,000.0,ddmmyy,000.0,E,N,*XX<CR><LF>
```

**→** *73 chars appear as one line:*

```
$GPRMC,hhmmss,V,ddmm.mmmm,N,dddmm.mmmm,E,000.0,000.0,ddmmyy,000.0,E,N,*XX
```

# How the TT8 Reads a TPU Serial Line

```
#define SERCHAN    0           // TPU channel for serial input comms
#define TSBUFFSIZ 128        // buffer size for TPU serial channel, a factor of 2!
.....
void main( ) {
    int i ;
    char c[TSBUFSIZ] ;
    .....
    printf("Serial input port opened:  %d ( check: should be %d ) \n",
           TSerOpen( SERCHAN,           // channel
                    MiddlePrior,       // "middle priority" (advanced!)
                    INP,                // configured as input
                    malloc(TSBUFSIZ+TSER_MIN_MEM), // allocate the buffer
                    TSBUFFSIZ,         // size of the buffer
                    9600, 'N', 8, 1    ), // comm. protocol
           tsOK );                   // (second item printed: "OK")

    for (i=0;i<TSBUFSIZ;i++) // get the chars from buffer as fast as you can!
        c[i] = TSerGetByte(SERCHAN) ;

    for(i=0;i<TSBUFSIZ;i++) // now print them out
        printf("%c",c[i]) ;
    printf("\n") ;
}
```

**Buffer:** A data space where the port loads incoming data, or accumulates outgoing data.

Serial devices communicate using characters encoded into bits. This includes upper- and lowercase letters, carriage returns and linefeeds, punctuation, etc.

Characters are not numbers! E.g.,

```
char c = '7' ;  
char d[2] = '92' ;  
int n ;
```

The numerical value of **c** is [0110111] (binary) or 55 (decimal).

But because the ASCII characters '0','1','2','3','4','5','6','7','8', and '9' occur in order, making simple conversions is easy:

```
n = c - '0' ;
```

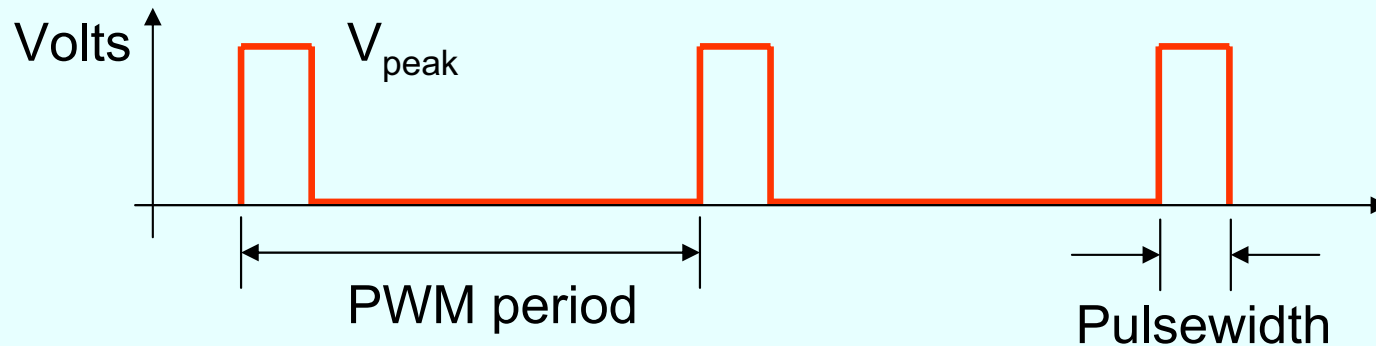
assigns to **n** the actual number 7. The ASCII character that goes with 7 is known as BEL – on many machines this will ring a bell if it is sent to the screen as a character! – **printf(“%c”,n) ;**

How to turn d[2] into a number?

```
n = 10*( d[0] - '0' ) + ( d[1] - '0' ) ;
```

# Pulse Width Modulation

- A Regular Waveform



- PWM frequency (Hz) =  $1 / \text{PWM period}$
- Duty cycle =  $\text{Pulsewidth} / \text{PWM period}$
- PWM frequencies typically range from 100Hz into MHz
- Duty cycles can be used from 0 – 100%, although some systems use much smaller ranges, e.g. 5-10% for hobby remote servos.
- The waveform has two pieces of information: Period and Pulsewidth, although they are usually not changed simultaneously.

# Some PWM Uses

- The Allure: very fast, cheap switches and clocks to approximate continuous processes. Also, two-state signal resists noise corruption.
- Sensors: PWM period is naturally related to *rotation or update rate*: Hall effect, anemometers, incremental encoders, tachometers, etc.
- Communication: PWM duty cycle is *continuously variable* → like an D/A and an A/D.
- Actuation: At very high frequencies, physical systems filter out all but the mean; i.e.,

$$V_{\text{effective}} = \text{duty\_cycle} * V_{\text{peak}}$$

High frequency switching is the dominant mode for powering large motors!

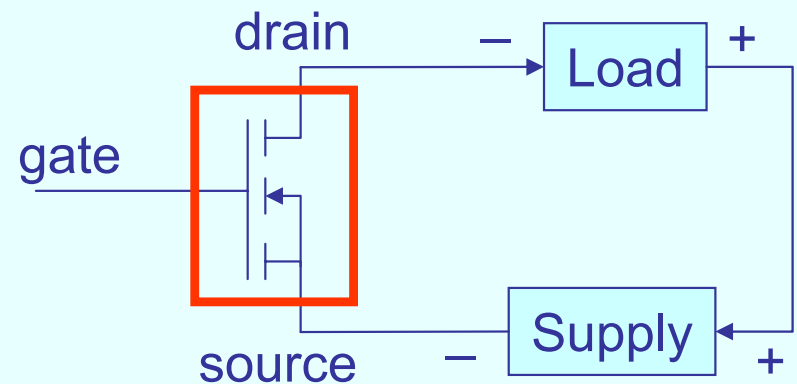
Image removed for copyright reasons.  
Propellers on a large ship.

# Field Effect Transistor (FET)

- Like a “valve”, that is very easy to open or close. When FET is open, resistance is low (milli-Ohms); when FET is closed, resistance is high (mega-Ohms or higher)

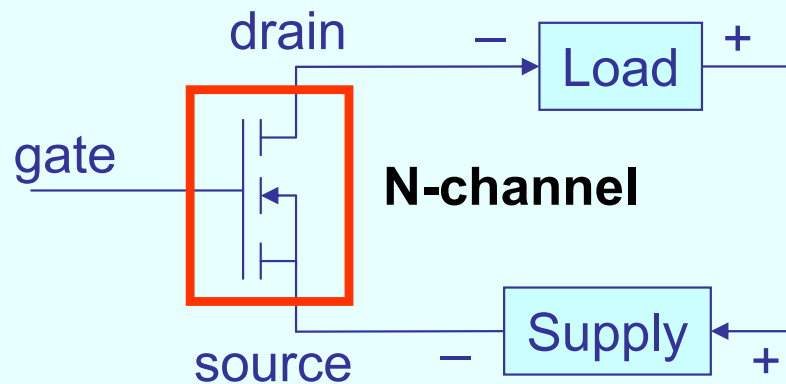
- Typically three connections:

- Gate: the signal; low current
- Source: power in
- Drain: power out

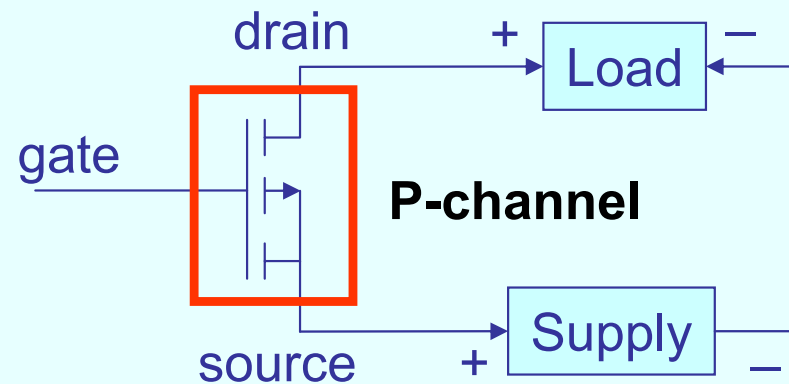


- *N*- and *P*-type junctions are common, and involve the polarity of the device. (*N* is shown)
- Extremely sensitive to static discharge! *Handle with care.*
- MOSFET: modern FET's capable of handling higher power levels.

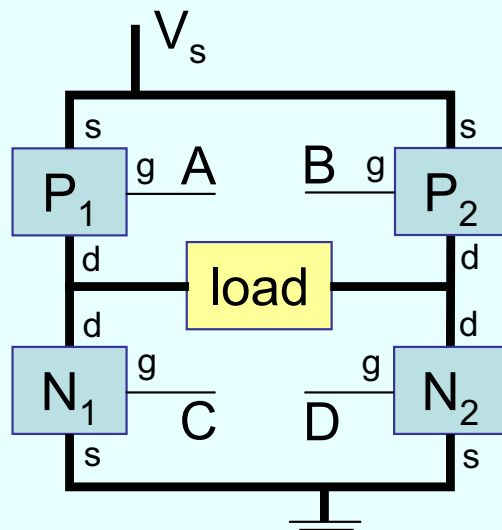
# Bipolar Control with a MOSFET H-Bridge



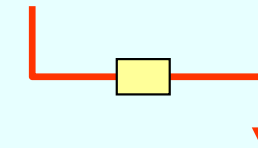
MOSFET turns on when  $V_{gate} > V_{source}$



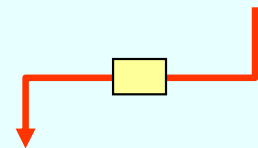
MOSFET turns on when  $V_{gate} < V_{source}$



To make flow UL to LR,  
set A = GND and D =  $V_s$



To make flow UR to LL,  
Set B = GND and C =  $V_s$



*Connect A and B to  $V_s$  with pull-up resistors;  
Connect C and D to GND with pull-down resistors;  
Control all four gates explicitly*

# The Basic DC Brush Motor

Torque  $\tau \leftrightarrow$  (coils)(flux density)(current  $i$ ),  
or, in a given motor,

$$\tau = k_t * i \quad \text{where } k_t \text{ is the torque constant}$$

But the motion of the coils also induces a voltage in the coil, the back-EMF:

$$e_b = k_t * \omega \quad (\text{YES, that's the same } k_t!)$$

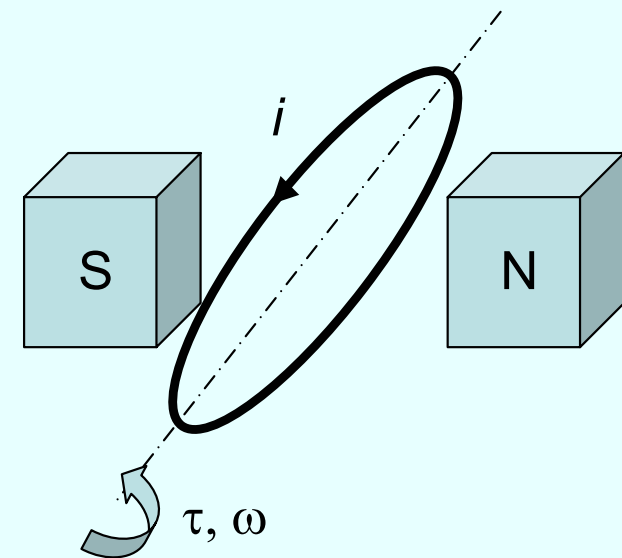
And the windings have a resistance  $R$ :

$$e_R = R * i$$

Summing voltages around the loop,

$$V_{\text{supply}} = e_b + e_R$$

*Vector relations:*  
*force = current x flux*  
*field = velocity x flux*



# Properties of the DC Brush Motor

- No-load speed:

$$\tau = 0 \rightarrow i = 0 \rightarrow$$

$$\omega = V / k_t$$

- Zero-speed torque (*BURNS UP MOTOR IF SUSTAINED*):

$$\omega = 0 \rightarrow e_b = 0 \rightarrow i = V / R \rightarrow$$

$$\tau = k_t V / R$$

- Power output:

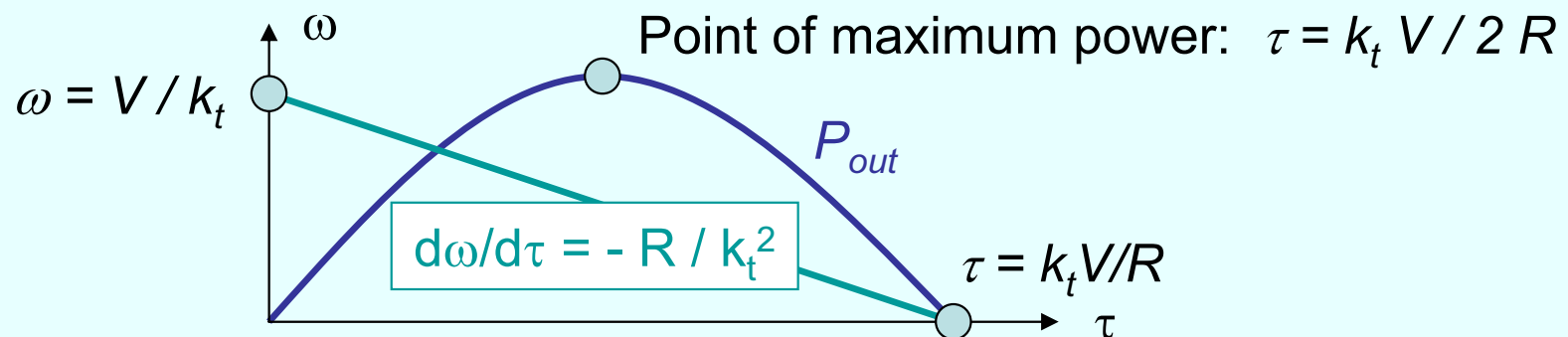
$$P_{out} = \tau \omega = i e_b \rightarrow$$

$$P_{out} = i (V - Ri)$$

- Efficiency:

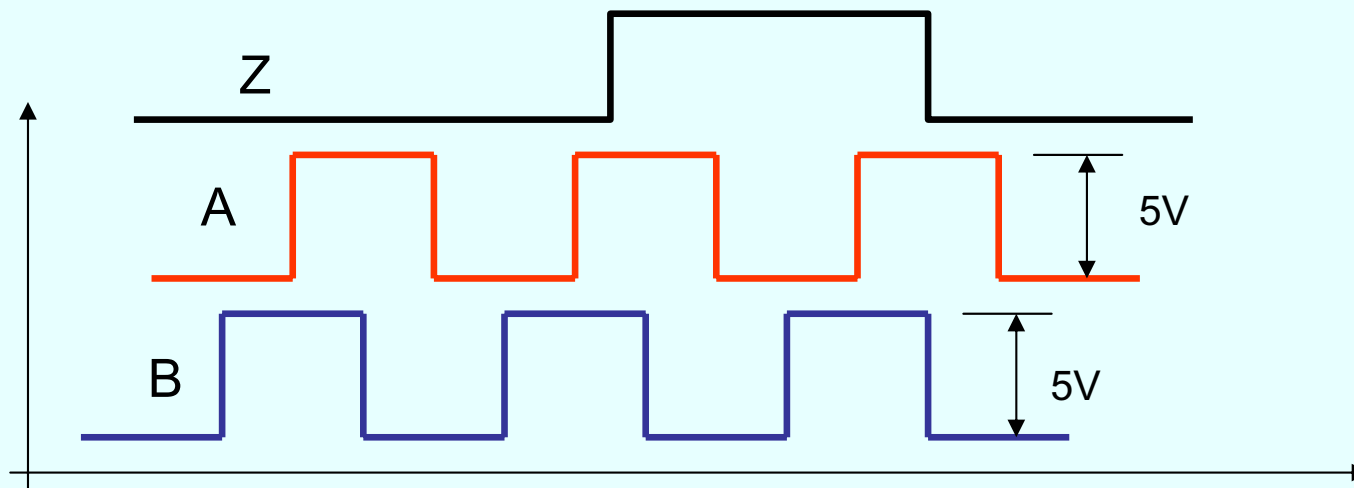
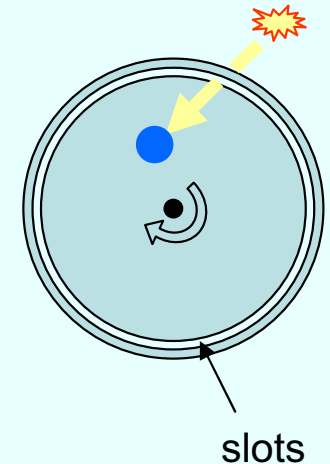
$$\eta = P_{out} / P_{in} = \tau \omega / i V \rightarrow$$

$$\eta = 1 - i R / V$$



# Incremental Encoders for Control

- *What is the position of the motor?*
- Take advantage of cheap, fast counters → make a large number of pulses per revolution, and count them!
- Advantages of the incremental encoder:
  - High resilience to noise because it is a digital signal
  - Counting chip can keep track of multiple motor turns
  - Easy to make – phototransistor, light source, slotted disk
- Two pulse trains required to discern direction: *quadrature*



# Stepper Motors

Switched coils at fixed positions on the stator attract permanent magnets at fixed positions on the rotor.

Smooth variation of switching leads to half-stepping and micro-stepping

Encoder still recommended!

