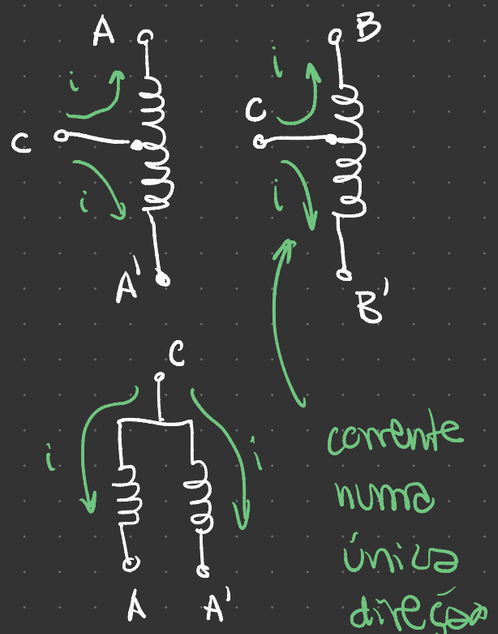
Motor de passo

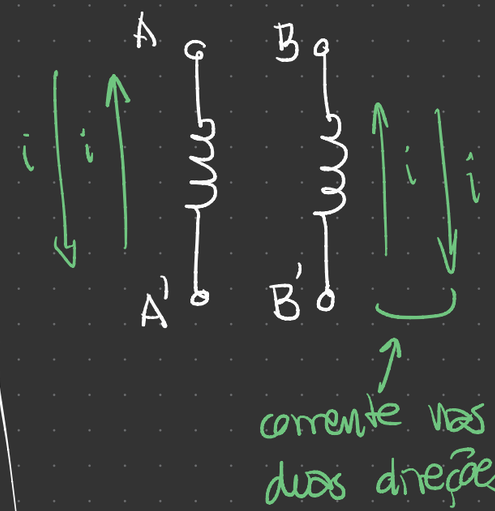
Tipos
- imagneto permanente (rotor é imã)
- relutância variável (rotor é de ferro dentado)
- híbrido (rotor dentado e imã)

Tipos de enrolamento

Unipolar



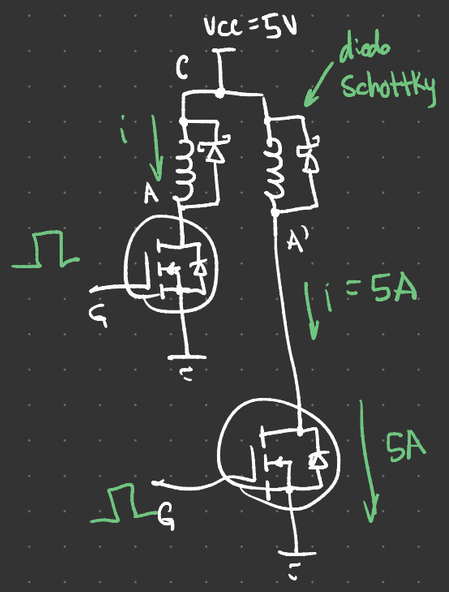
Bipolar



Características

- nº pulsos → ângulo
  200 pulsos/volta → 1,8°
- velocidade do motor → frequência
- holding torque
- malha aberta
- controle digital

Acionamento unipolar



V_{DS} max = 60 V

I_D max = 20A

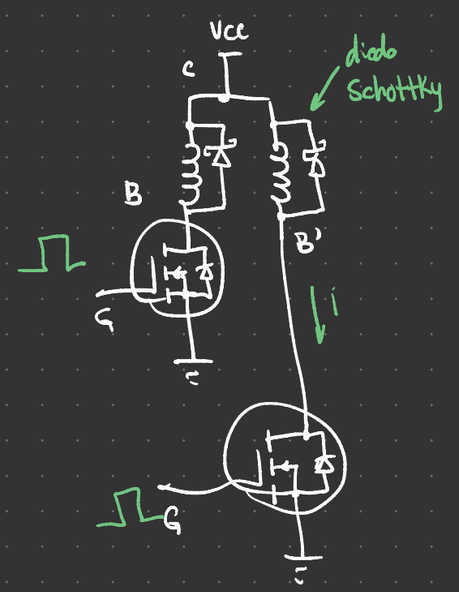
V_{GS} threshold = 0,8V – 2,0V

R_{DS} (on) = 1Ω → 2,5W

0,1Ω → 2,5V

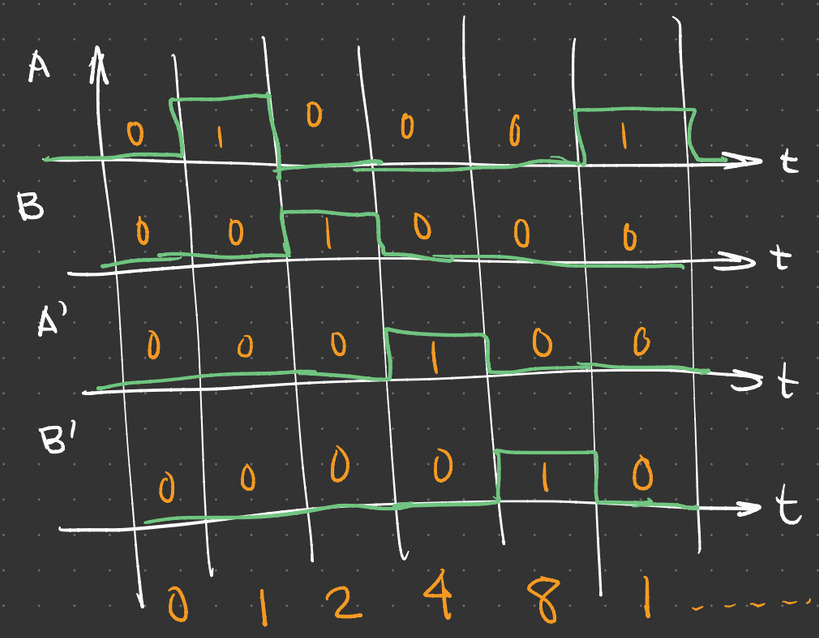
↓

mΩ

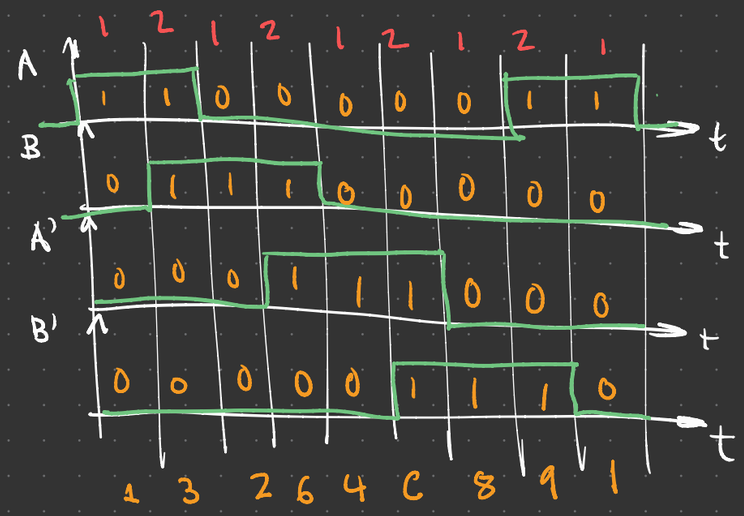


Full Step

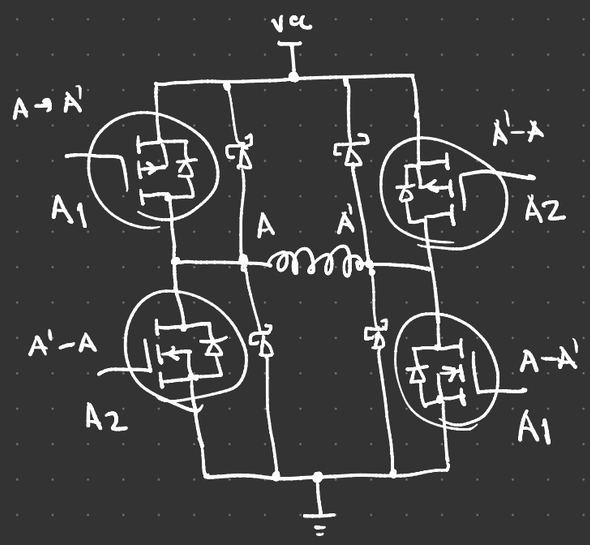
C→A , C→B, C→A', C→B'

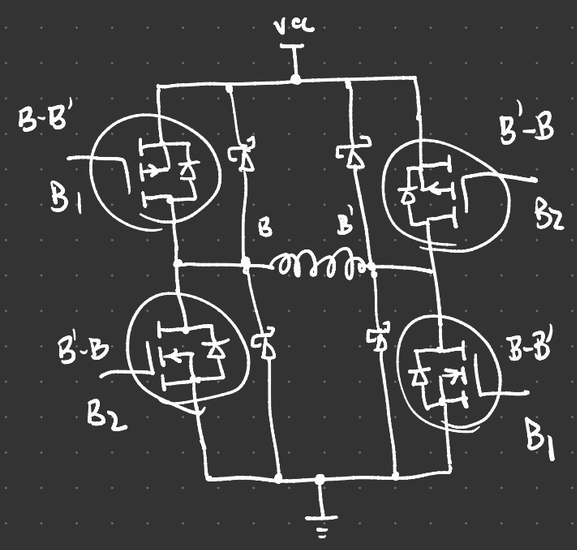


Half Step



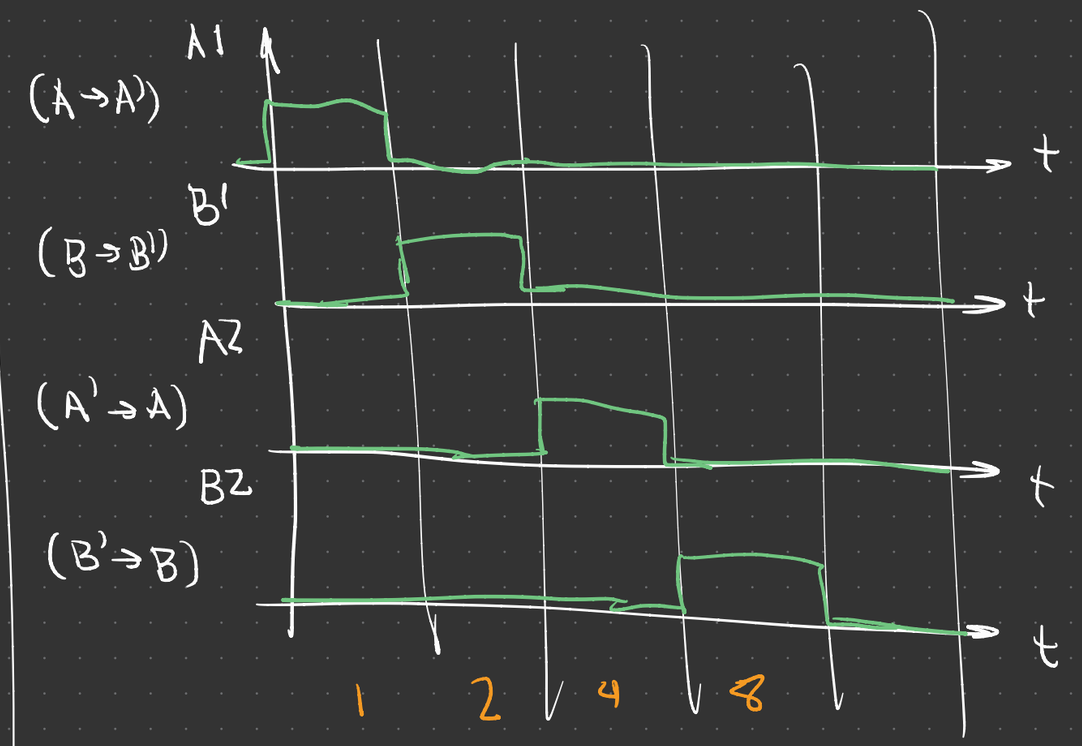
Acionamento Bipolar



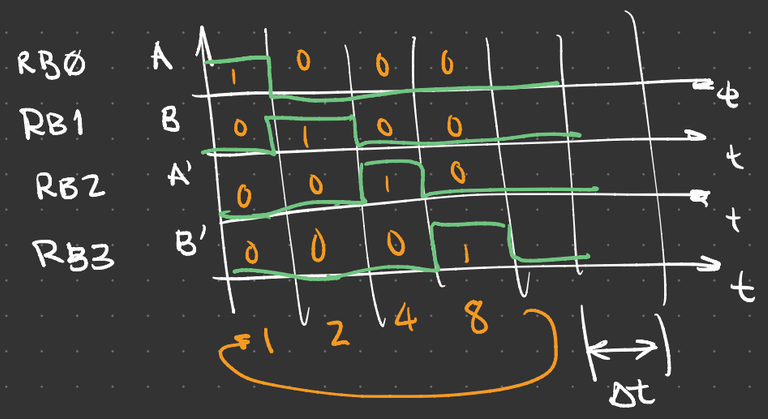


Full Step

A→A'	A'→A	A→A'	A'→A
B→B'	B→B'	B'→B	B'→B



Exemplo: Acionamento unipolar full-step



2 funções { inicialização / acionamento

RB0 / AN12
RB1 / AN10
RB2 / AN8
RB3 / AN9


```
typedef struct {
    char low:4;
    char high:4;
} twoNibbles_t
twoNibbles_t PORTBnibbles @ 0x006; //endereço do PORTB
```

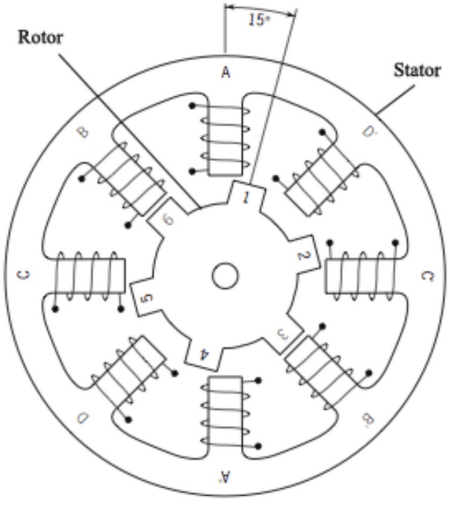
Inicialização

```
void stepper_init(void) {
    TRISB &= 0b11110000; // bits 0-3 saída
    ANSELH &= 0b11101000; // AN8,9,10,12 digital
    PORTB &= 0b11110000; // zera saída do motor de passo
}
```

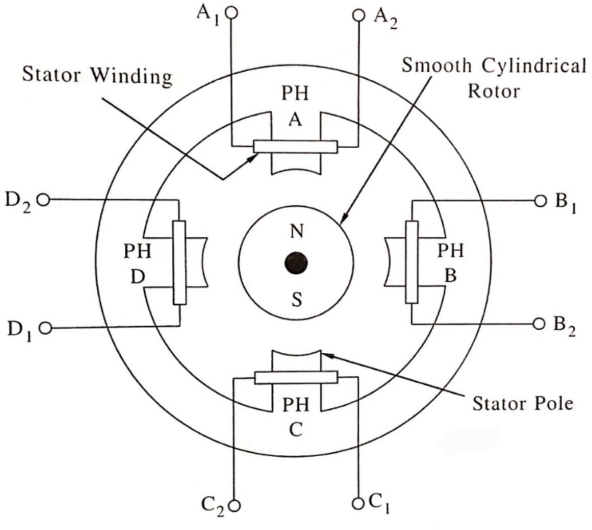
Acionamento

```
void stepper_set(int steps, unsigned int step_time) {
    char states[] = {1,2,4,8}; // full step
    static int i=0; // inicializa estático
    while(steps != 0) {
        PORTBnibbles.low = states[(steps>=0)? i++ : i--];
        i = (i>3) ? 0 : (i<0) ? 3 : i; // corrige indice circular
        steps += (steps>0) ? -1 : 1; // acerta nº passos
        for(int j=0; j<step_time; j++) delay_ms(1); Δt = n×1ms
    }
}
```

MOTORES DE PASSO

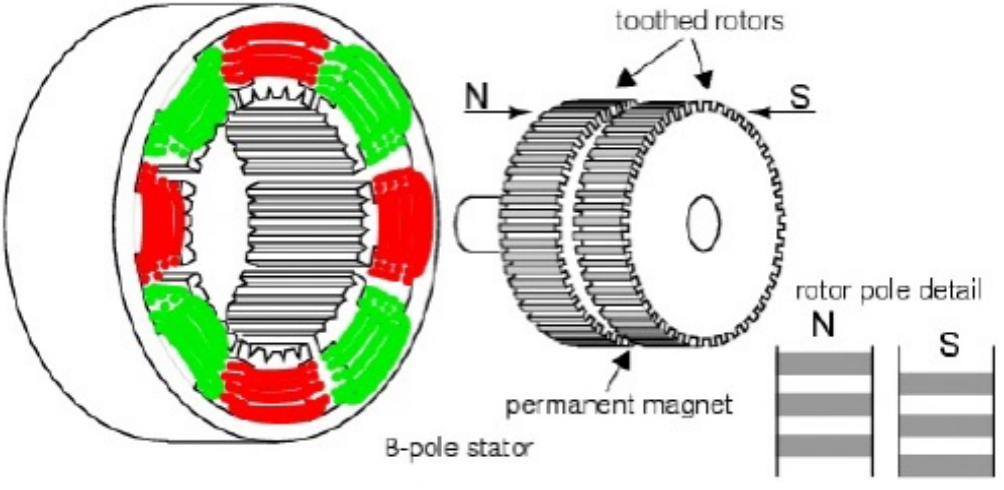


Relutância variável



Magneto permanente

MOTORES DE PASSO



Híbrido