Introdução

Walter Fetter Lages

w.fetter@ieee.org

Universidade Federal do Rio Grande do Sul

Escola de Engenharia

Departamento de Engenharia Elétrica

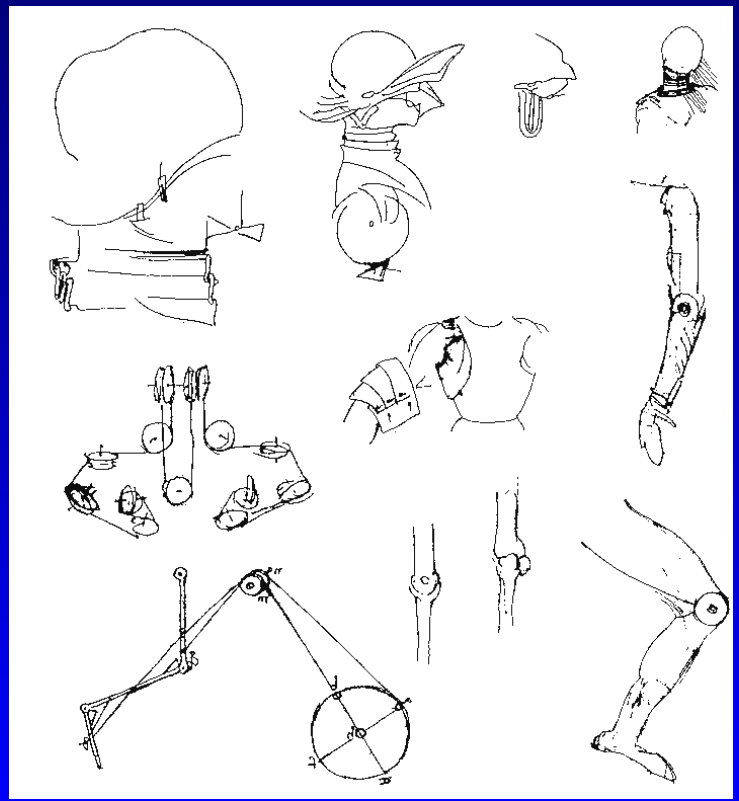
Programa de Pós-Graduação em Engenharia Elétrica

ELE00070 Tópicos Especiais em Controle e Automação I

Robôs são Idéia Antiga

270 A.C. Ctesibius da Grécia construiu relógios d'água com figuras móveis

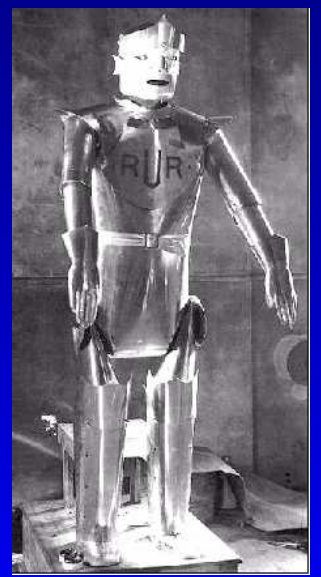
1452-1419 Leonardo Da Vinci imaginou robôs humanóides para proteger castelos



Robô

- A palavra robô vem da palavra tcheca "Robota"= trabalho escravo
- 1921 Peça teatral "Robôs Universais de Russum", de Karel Capek
- Isaac Asimov cunhou a palavra robótica


ST. MARTIN'S THEATRE
LONDON
R. U. R.
(Rossum's Universal Robots)
A Fantastic Melodrama
By KAREL CAPEK.
THE CHARACTERS IN THE ORDER OF THEIR APPEARANCE

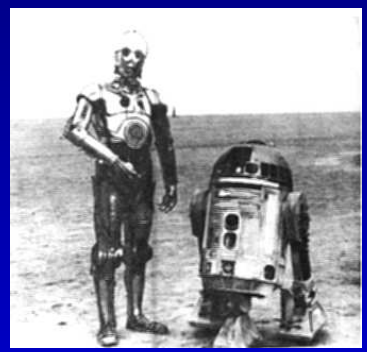


UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

Filmes e Robôs

- Os Jetson's 1962
- Perdidos no espaço 1964
- Guerra nas estrelas 1977 - R2D2 e C3P0
- Robôs em propagandas
- Animatronics 2000








UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

Robô

- Manipulador multi-funcional reprogramável projetado para movimentar materiais, peças, ferramentas ou dispositivos especiais seguindo movimentos programados variáveis, tendo por objetivo a realização de tarefas variadas
- Máquina flexível programável com a qual um objeto pode ser movido para um local definido no espaço, ou com o qual pode ser realizada uma trajetória com o objeto para realizar uma determinada tarefa

UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

Robôs Industriais

- Utilizados nas indústrias para
  - movimentação de peças
  - pintura
  - soldagem
- Normalmente constituídos por:
  - um braço articulado
  - uma unidade de controle
  - um *teaching–pad*
  - diversas interfaces com periféricos

UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

Robôs Móveis

- Robôs que podem movimentar-se autonomamente no solo ou no espaço.
- Frequentemente o termo robô móvel é utilizado para designar apenas a plataforma.

UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

Robôs de Serviço

- Robôs desenvolvidos para a execução de tarefas específicas
  - aspiração de pó
  - cuidar de pessoas idosas e deficientes
  - limpeza de navios e aviões
  - cortar grama
  - esquilar ovelhas
  - inspecionar linhas de transmissão de energia elétrica

UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

Robótica é um Campo Vasto

- Necessita conhecimentos de:
  - Eletrônica
  - Mecânica
  - Computação
  - Controle
  - Psicologia

UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

Robótica e Desemprego

- Muitas vezes é afirmado que os robôs causam desemprego
- Na verdade, os robôs mudam deslocam os postos de trabalho de lugar na linha de produção
- Com robôs, toda a linha de produção funciona mais rápido
  - são necessárias mais trabalhadores para
    - empacotar os produtos
    - vender um maior volume de produtos
    - alimentar o robô

Tipos de Acionamento

- Lagartas
- Rodas
- Perna

Principais Problemas

- *Where am I?*
  - Localização
- *Where am I going?*
  - Objetivo
- *How should I get there?*
  - Geração de trajetória
  - Desvio de obstáculos
- *How do I get there?*
  - Controle

UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

Modelagem de Robôs Móveis

- Modelo cinemático
  - Modelo cinemático de postura
  - Modelo cinemático de configuração
- Modelo dinâmico
  - Modelo dinâmico de postura
  - Modelo dinâmico de configuração
- Modelo do ambiente

Acionamento e Controle

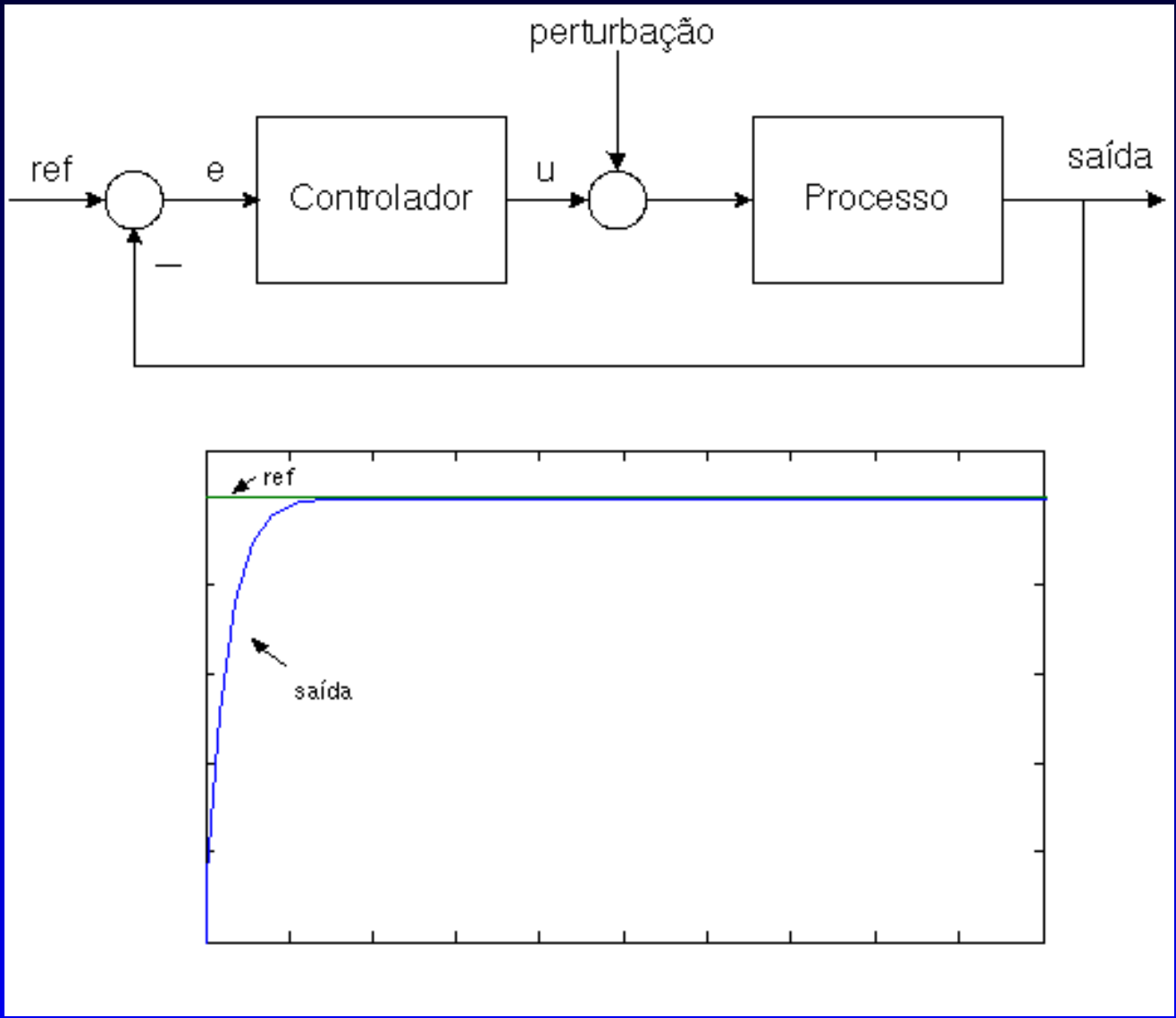
- Acionamento diz respeito aos circuitos utilizados para acionar os atuadores do robô
- Controle é a regra segundo a qual o acionamento é utilizado para atingir-se o objetivo

UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

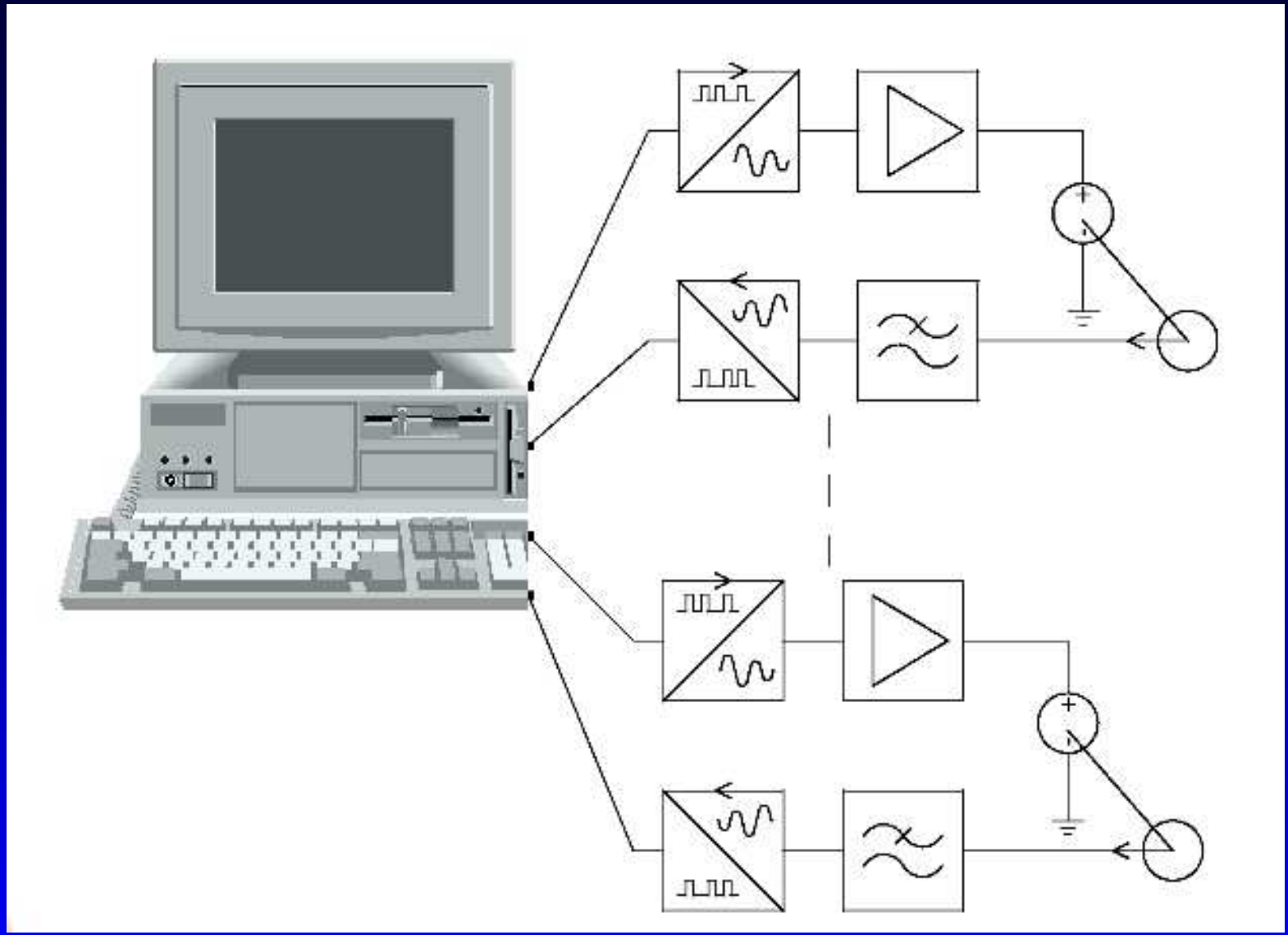
Controle

- Controle em malha-aberta
  - Não existe realimentação a partir de sensores
  - Decisões de controle em função de fatores externos, tipicamente o tempo.
- Controle em malha-fechada
  - Existe realimentação
  - O sinal de controle é determinado a partir dos sinais obtidos de sensores

Controle em Malha Fechada

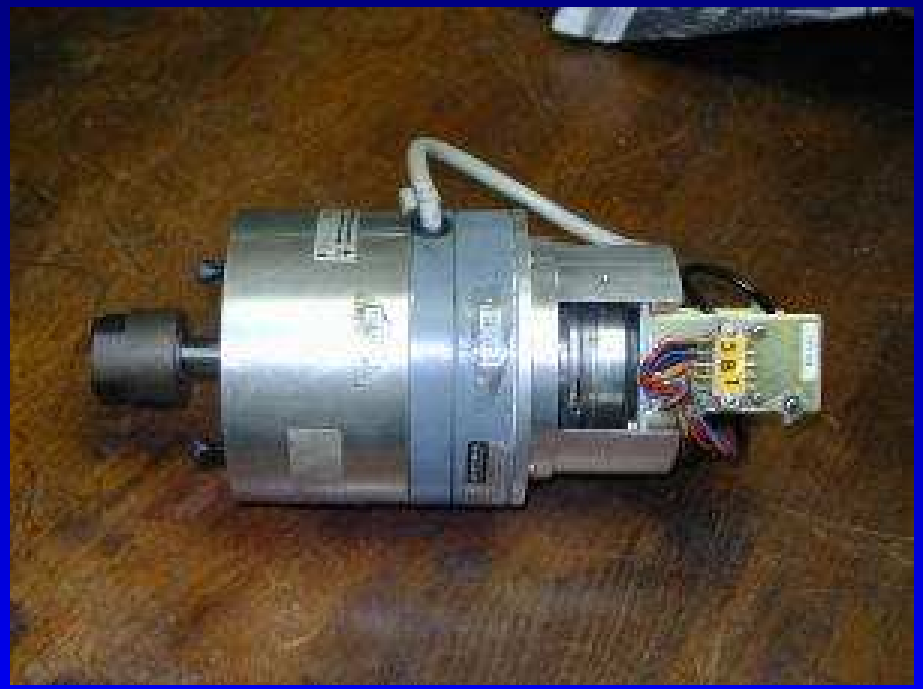


Arquitetura de Hardware Típica



Atuador

- Tipicamente o atuador é um motor D.C. com imã permanente
- Tensões típicas entre 12V e 56V
- Correntes típicas entre 500mA e 20A

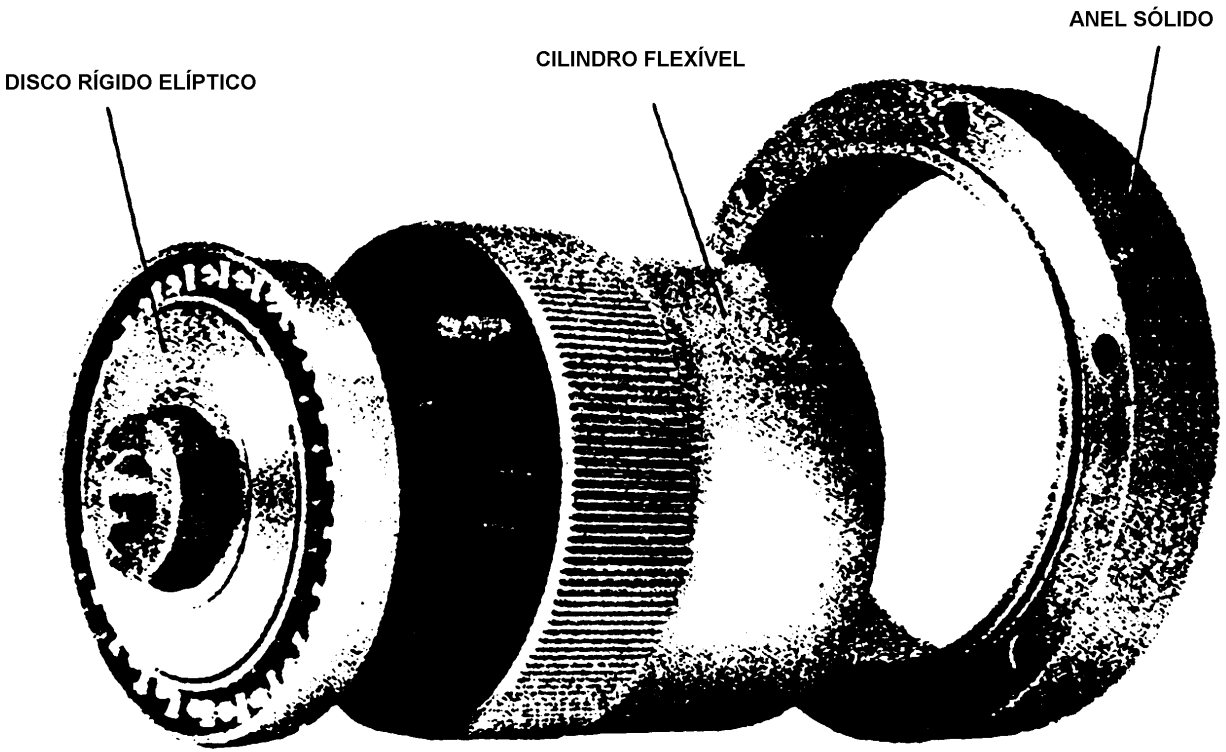


Transmissão

- Caixa de engrenagens
  - Problemas de folgas e sensíveis a vibrações
- Polias
- Alavancas
- Fusos
- Harmonic-drive

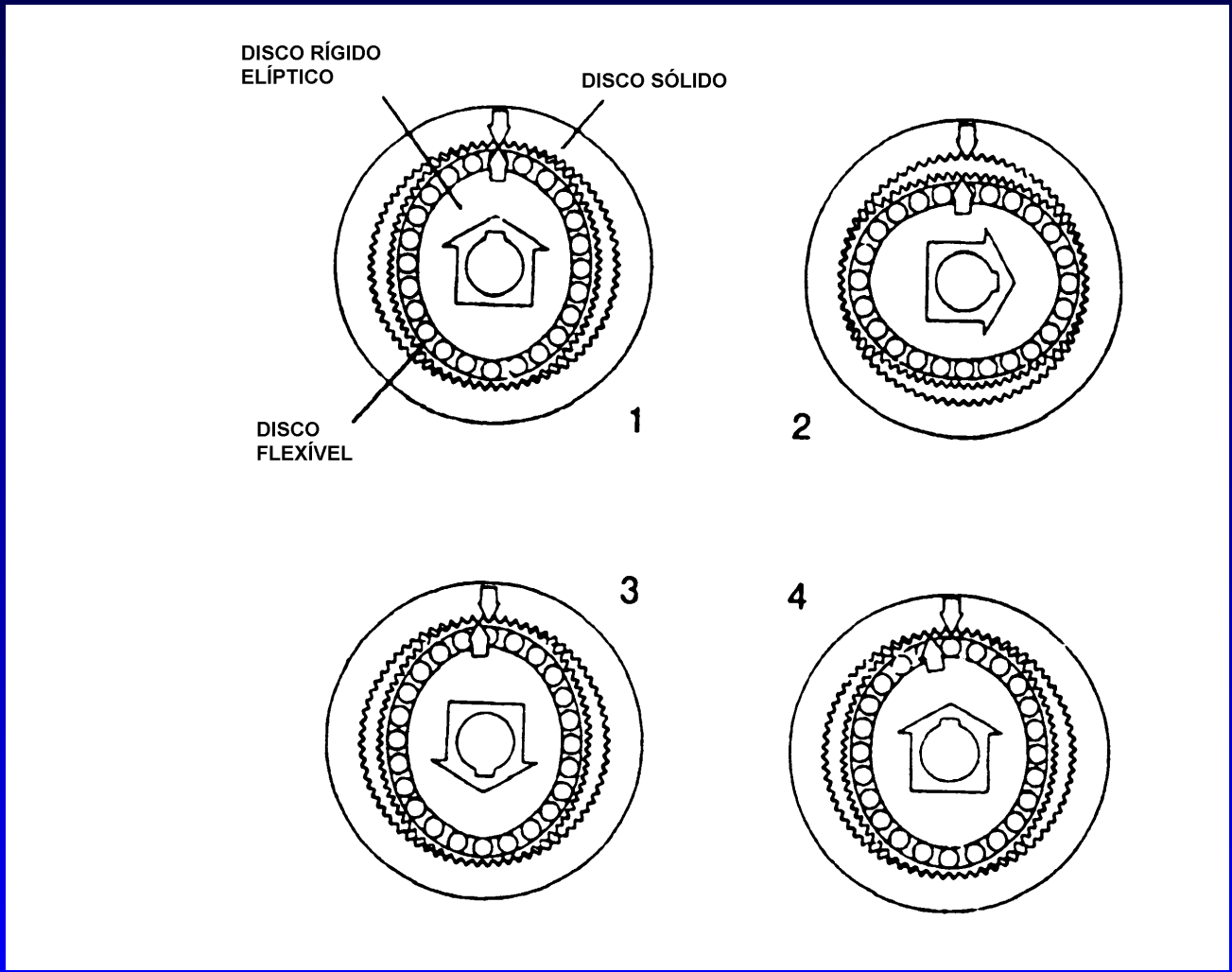
UFRGS
UNIVERSIDADE FEDERAL
DO RIO GRANDE DO SUL

Harmonic Drive



Harmonic Drive

- Redução de $2/n_f$ em uma única etapa



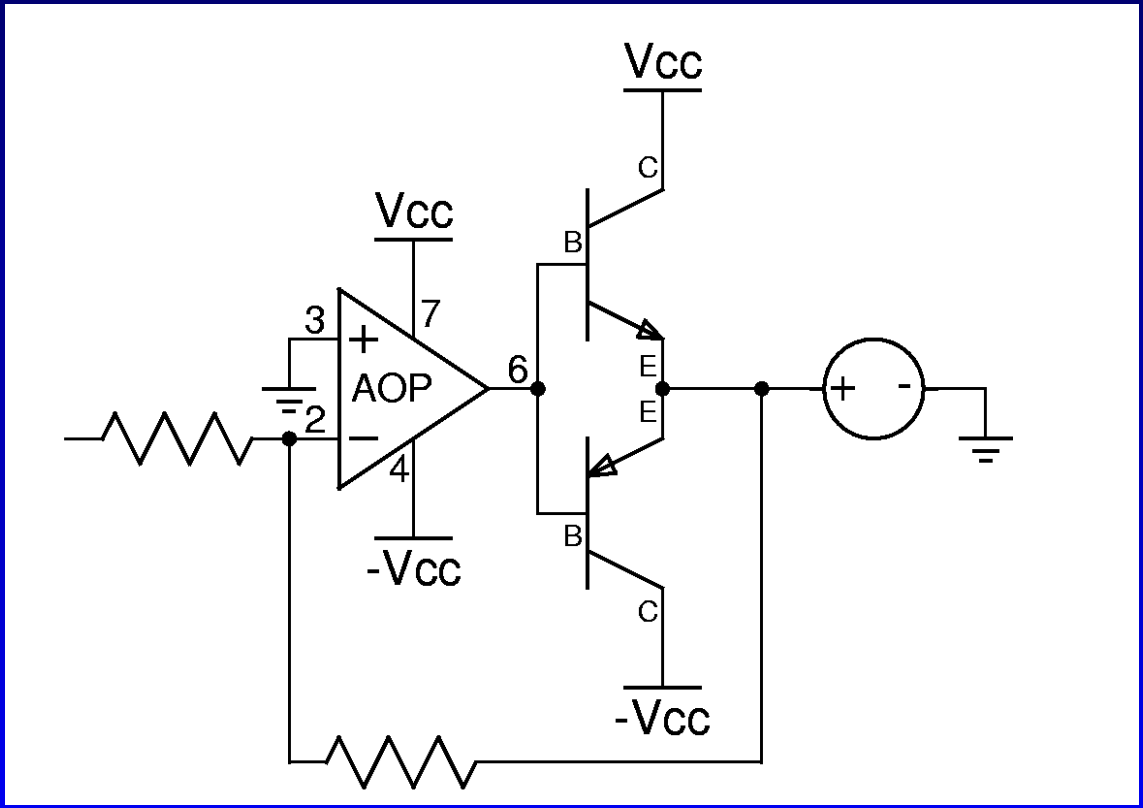
UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

Acionamento do Motor

- Conversor D/A + Amplificador Linear
  - Excelente linearidade
  - Alta dissipação de potência
- Conversor D/A + Amplificador Chaveado
  - Baixa dissipação de potência
  - Pouca imunidade à ruído
- Acionamento por PWM
  - Baixa dissipação de potência
  - Boa imunidade à ruído

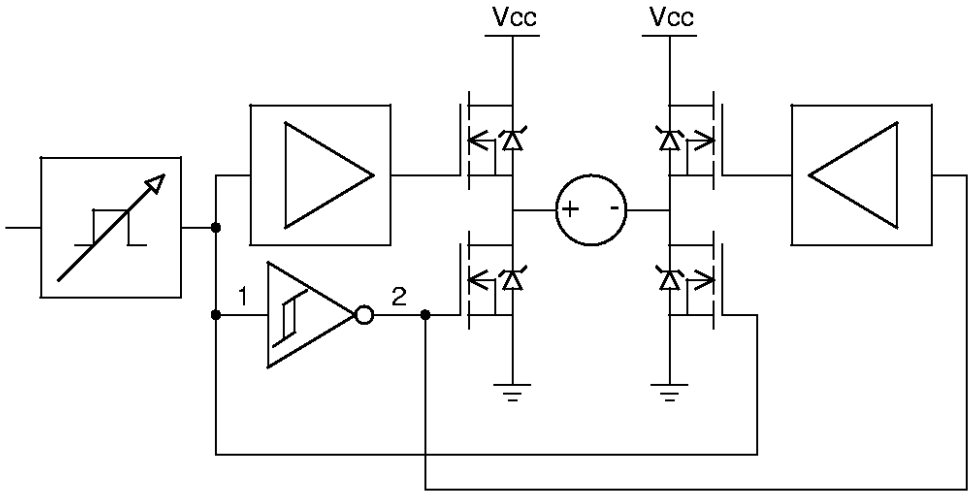
D/A + Amplificador Linear

- Arquitetura clássica
- Utiliza etapa de potência em *push-pull*
- Transistores de saída operam na região linear

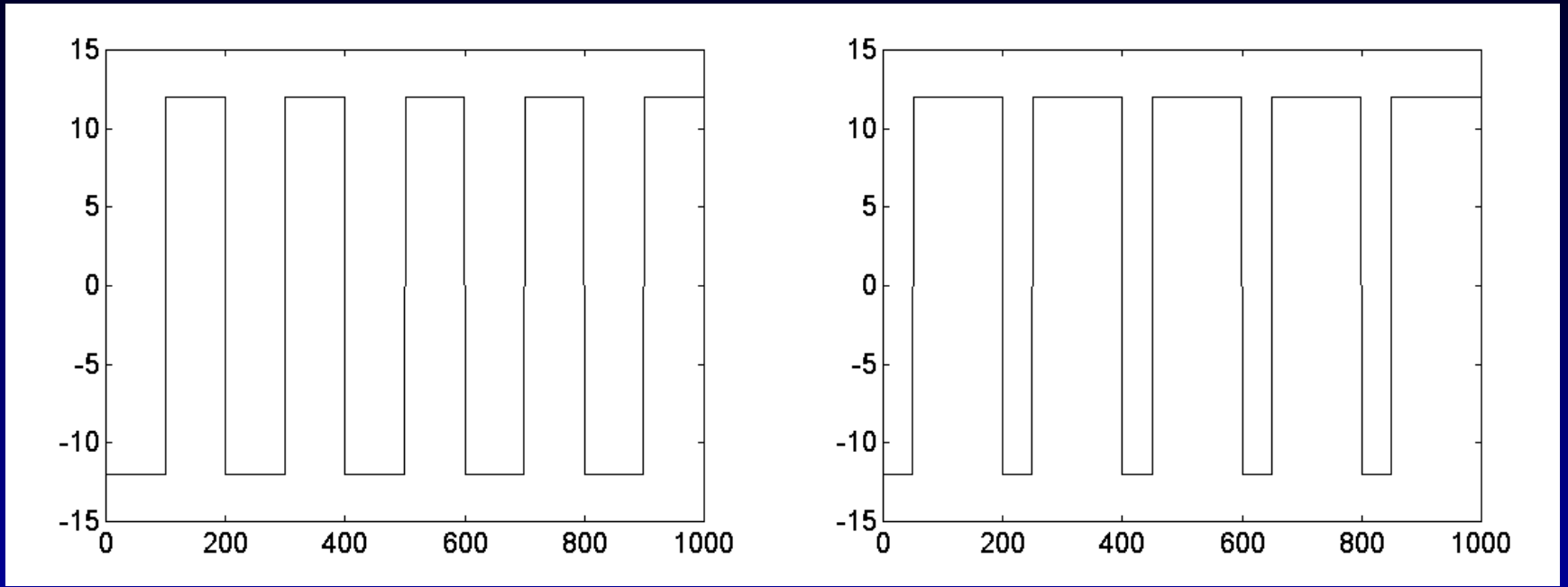


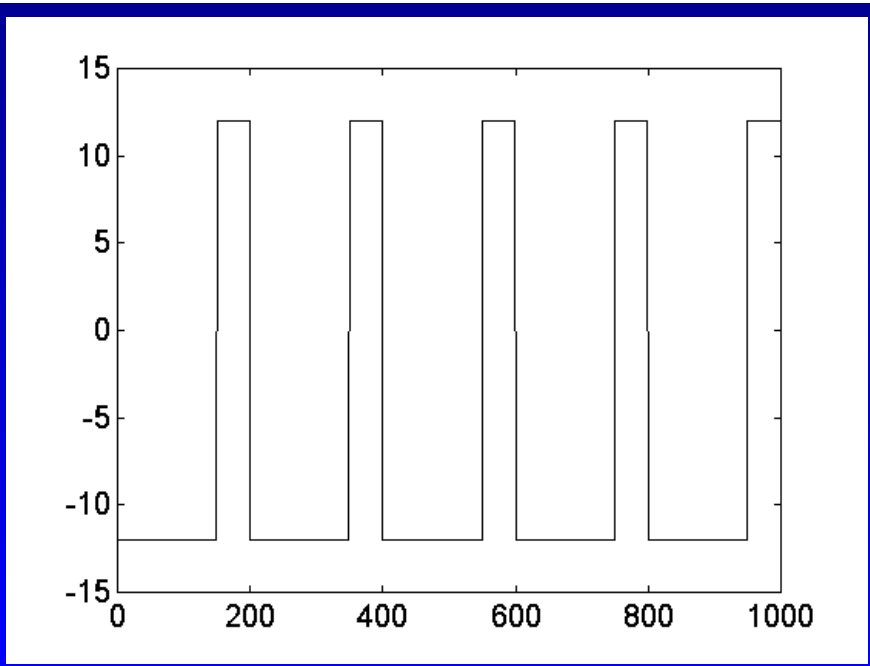
D/A + Amplificador Chaveado

- Substitui o amplificador linear por um amplificador classe G
- O sinal analógico é modulado em PWM e aplicado à ponte H

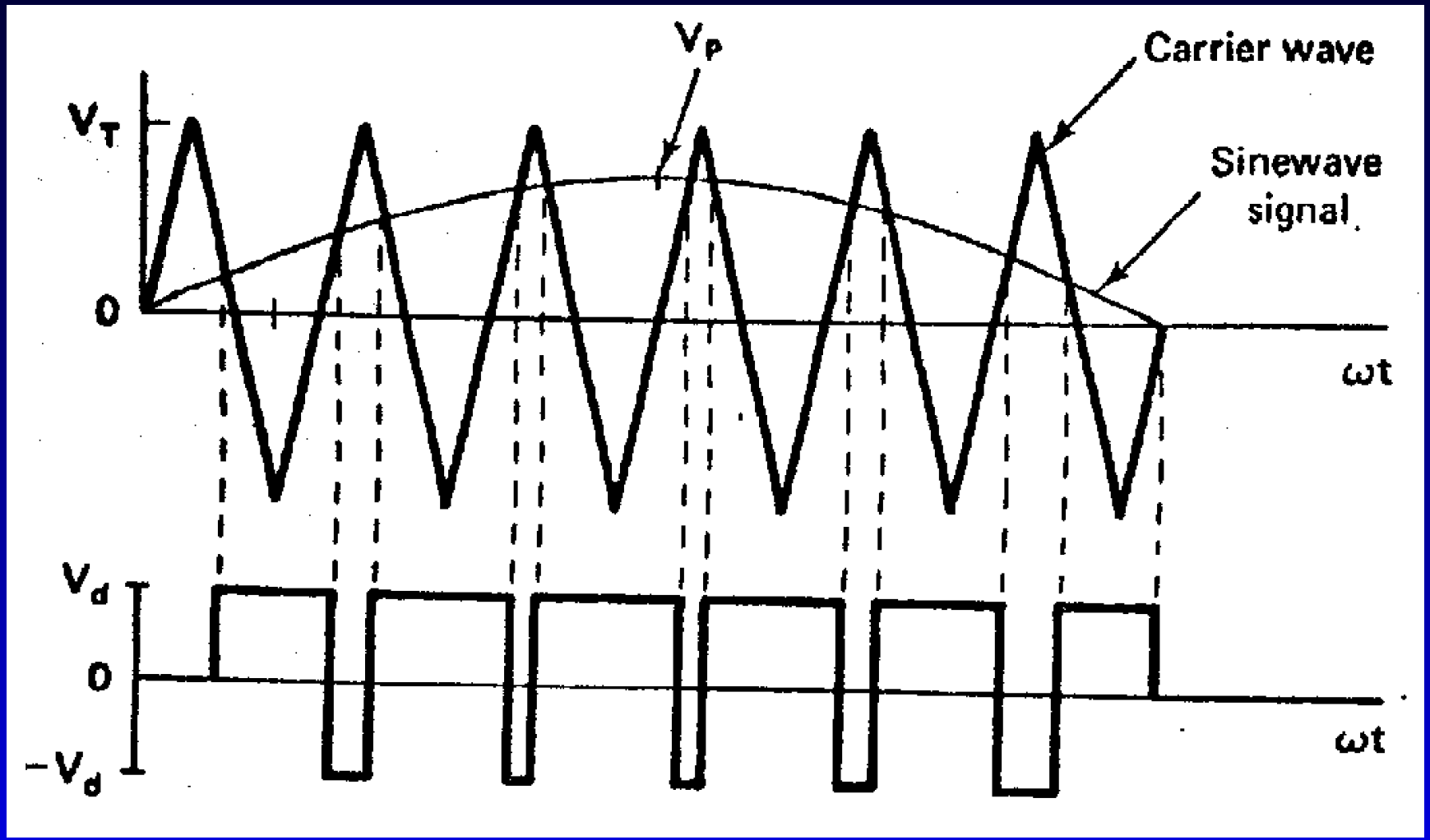


Pulse Width Modulation



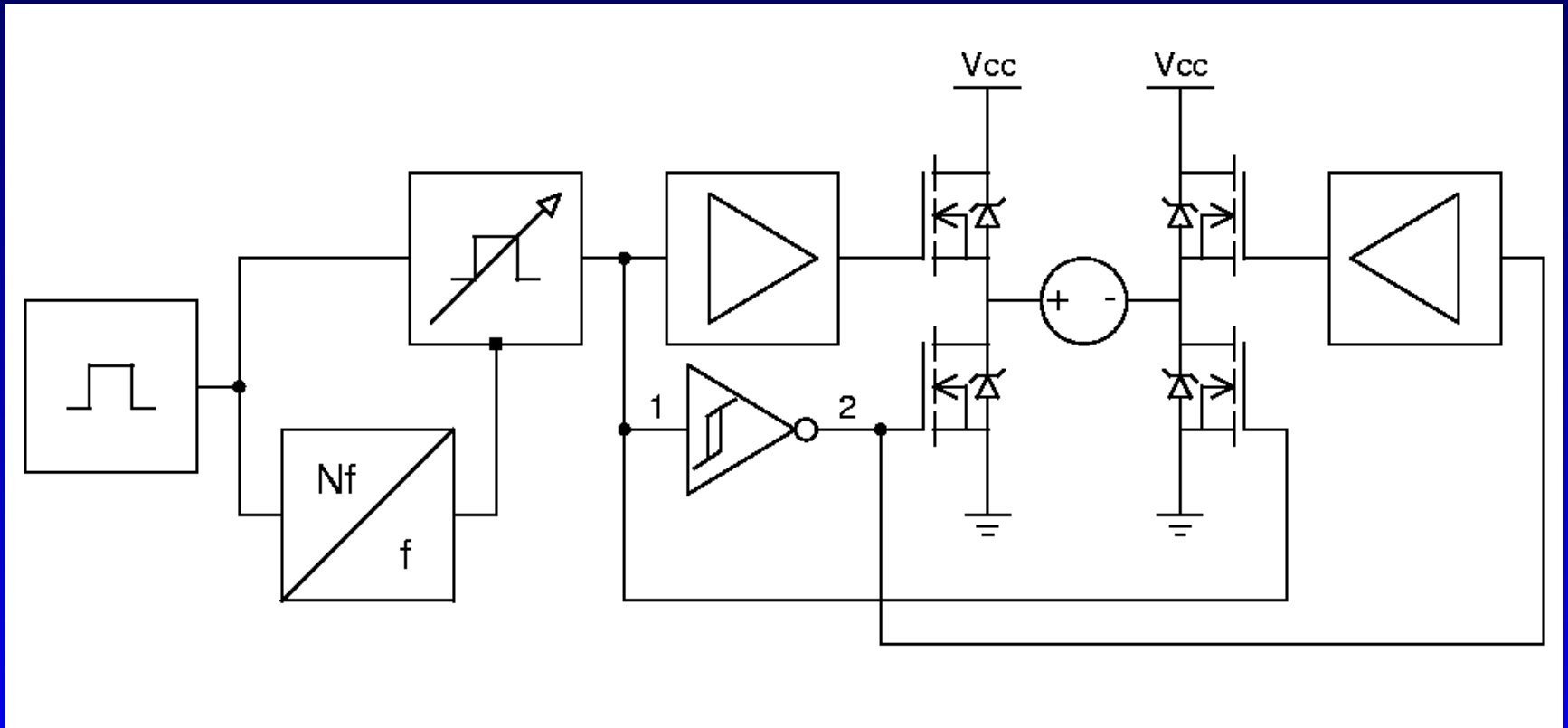


Modulador PWM Analógico

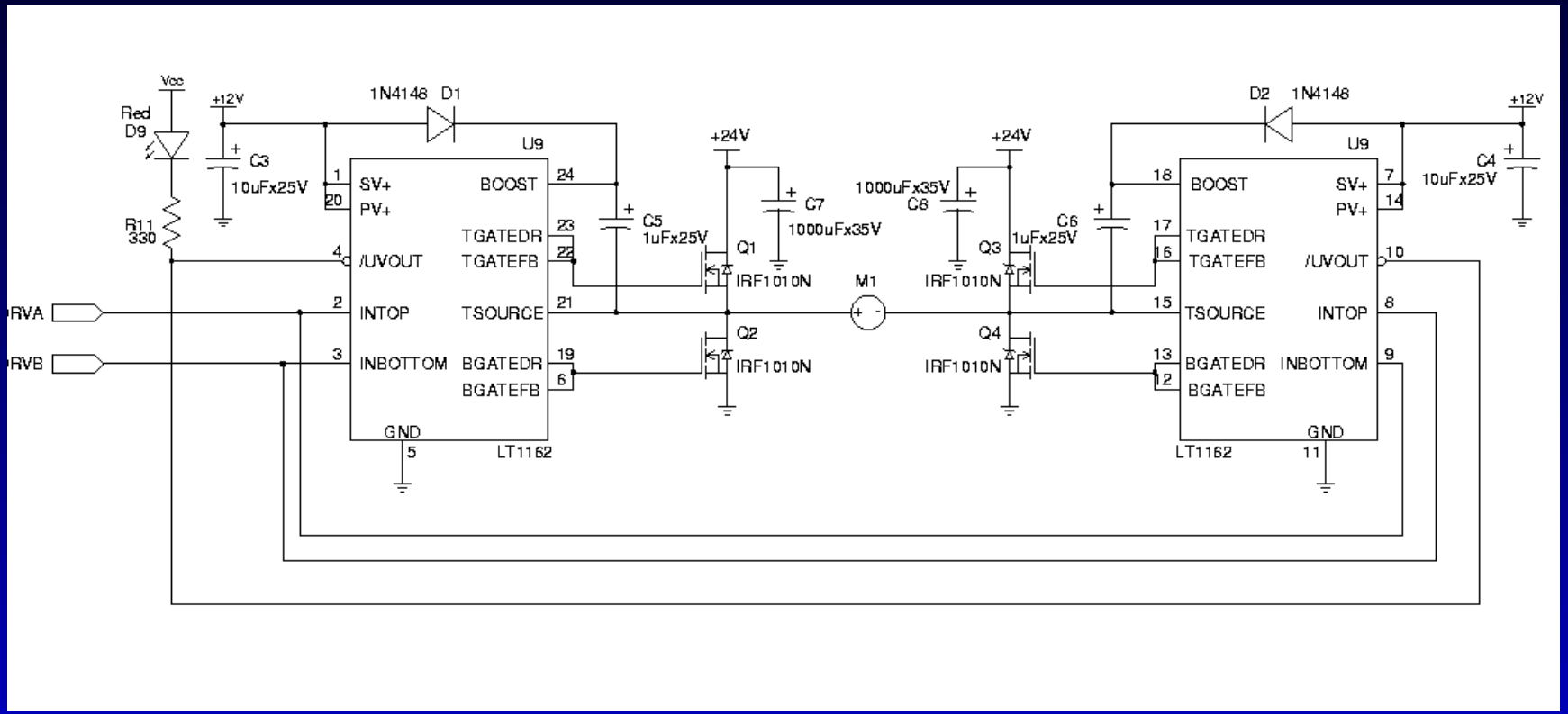


Acionamento por PWM Digital

- Acionamento totalmente digital
- Freqüência do PWM pode ser programada

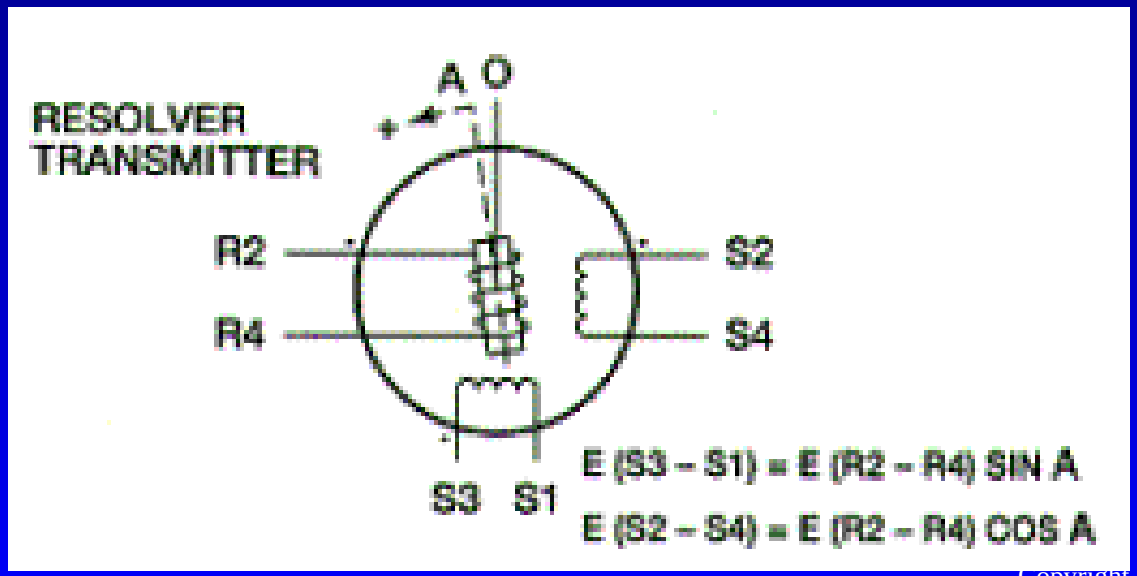


Charge Pump

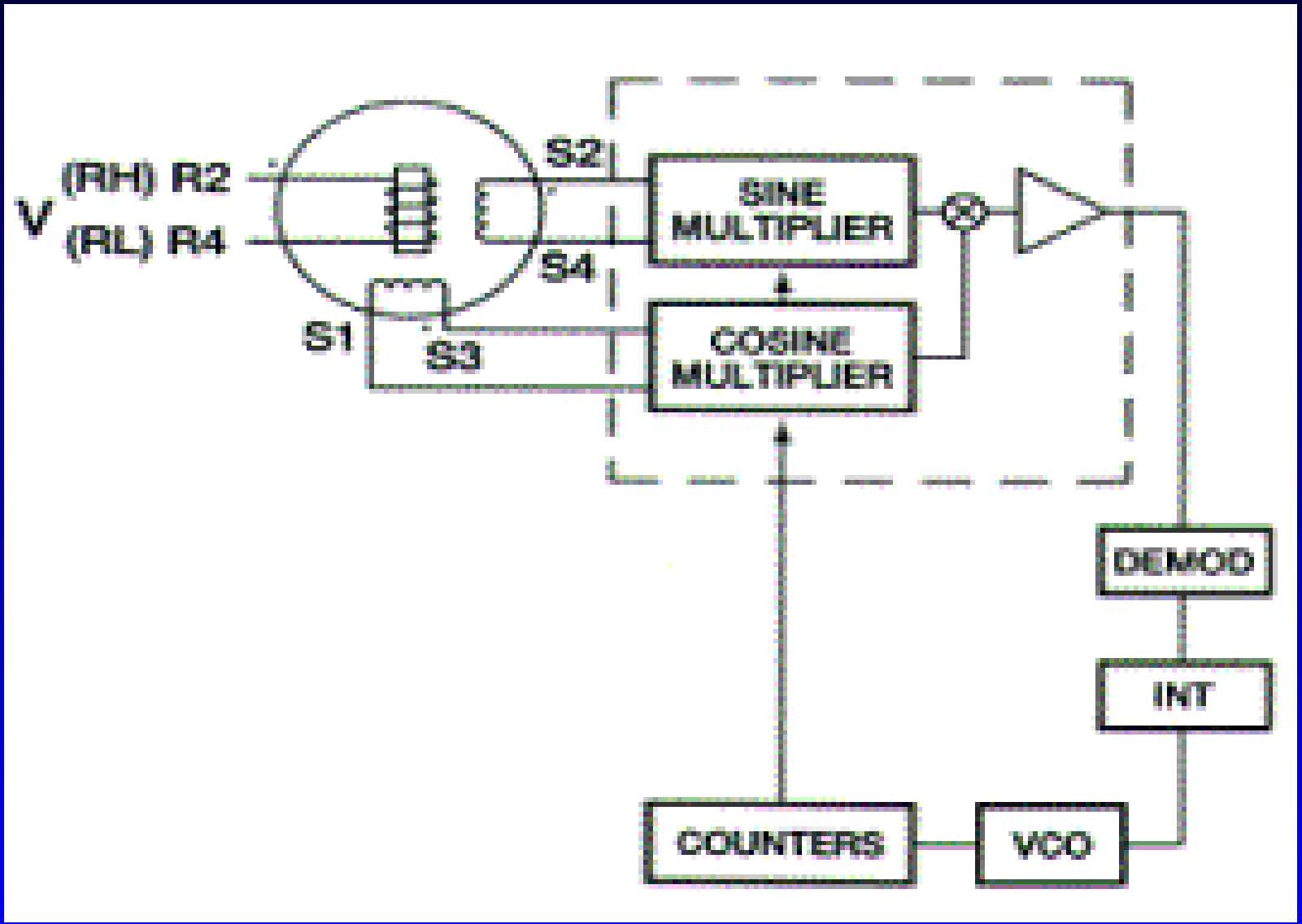


Resolvers

- Sinal de excitação de 400Hz
- $V_1 = V \sin(\omega t) \sin A$
- $V_2 = V \sin(\omega t) \sin A$
- Processamento
  - por demodulação
  - por amostragem

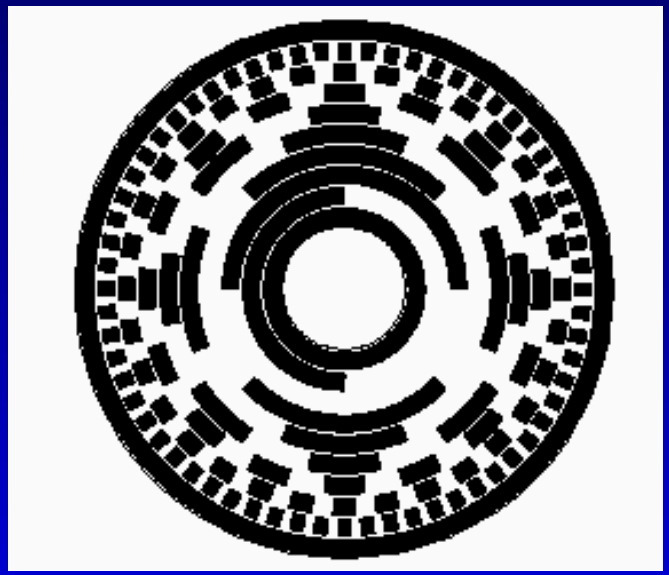


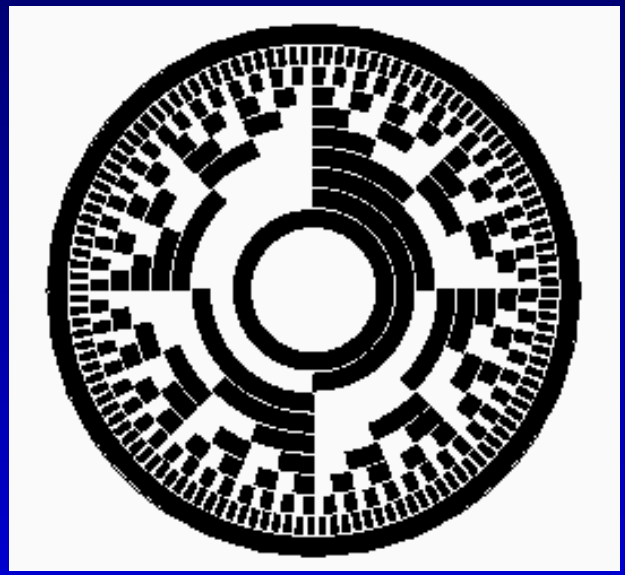
Demodulação do *Resolver*



Encoder Óptico Absoluto

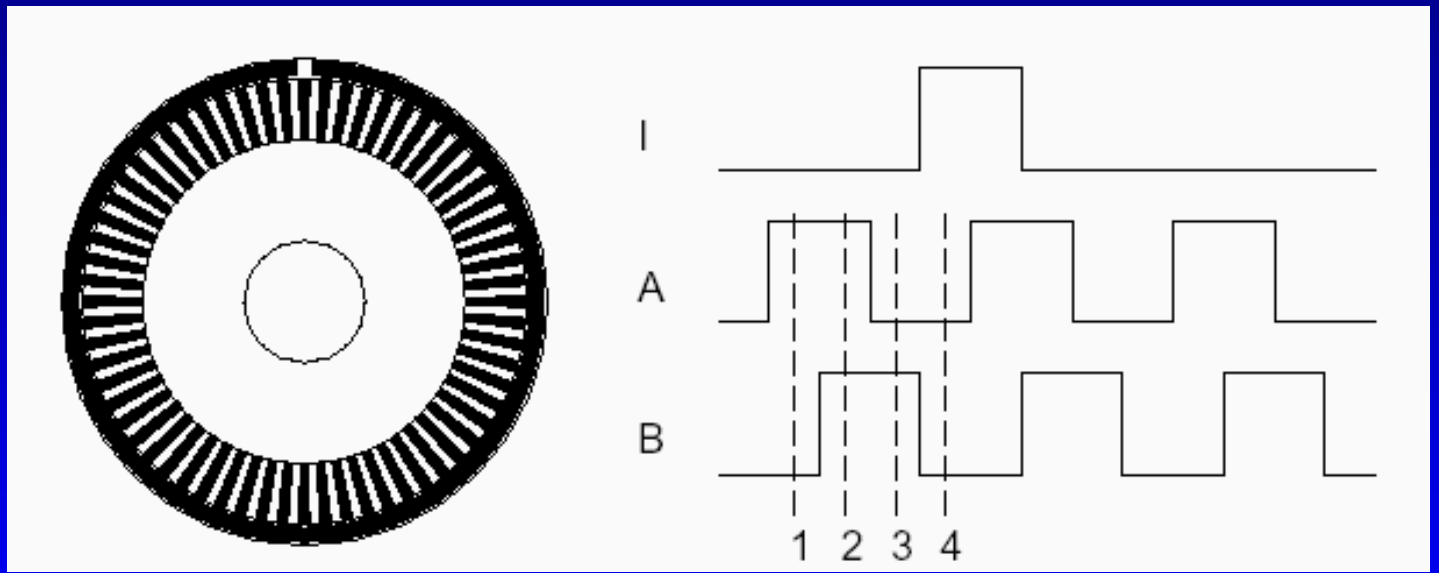
- Não é prático para resoluções elevadas
- Erros não são cumulativos
- Deve ser sempre utilizada codificação em Gray



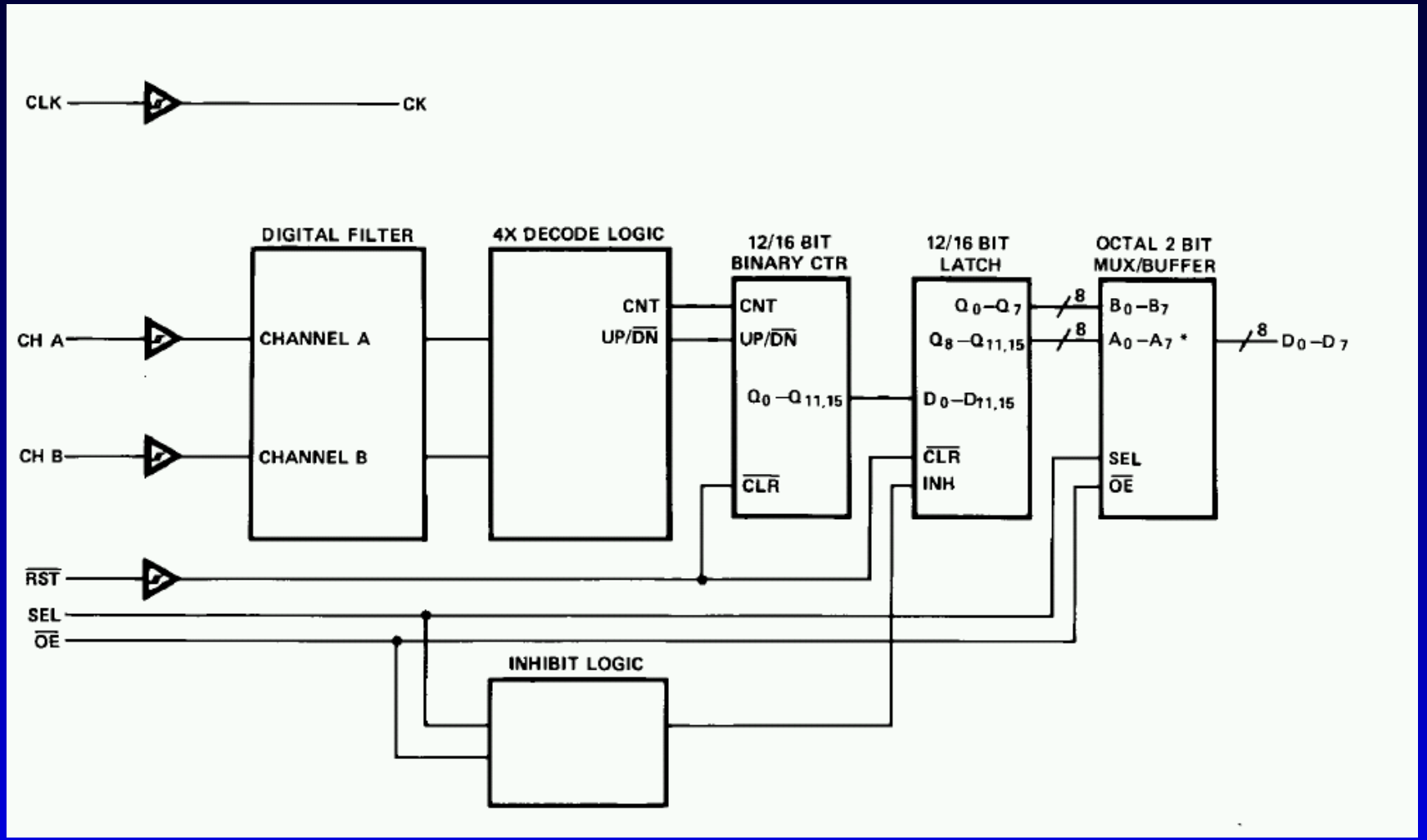


Encoder Óptico Incremental

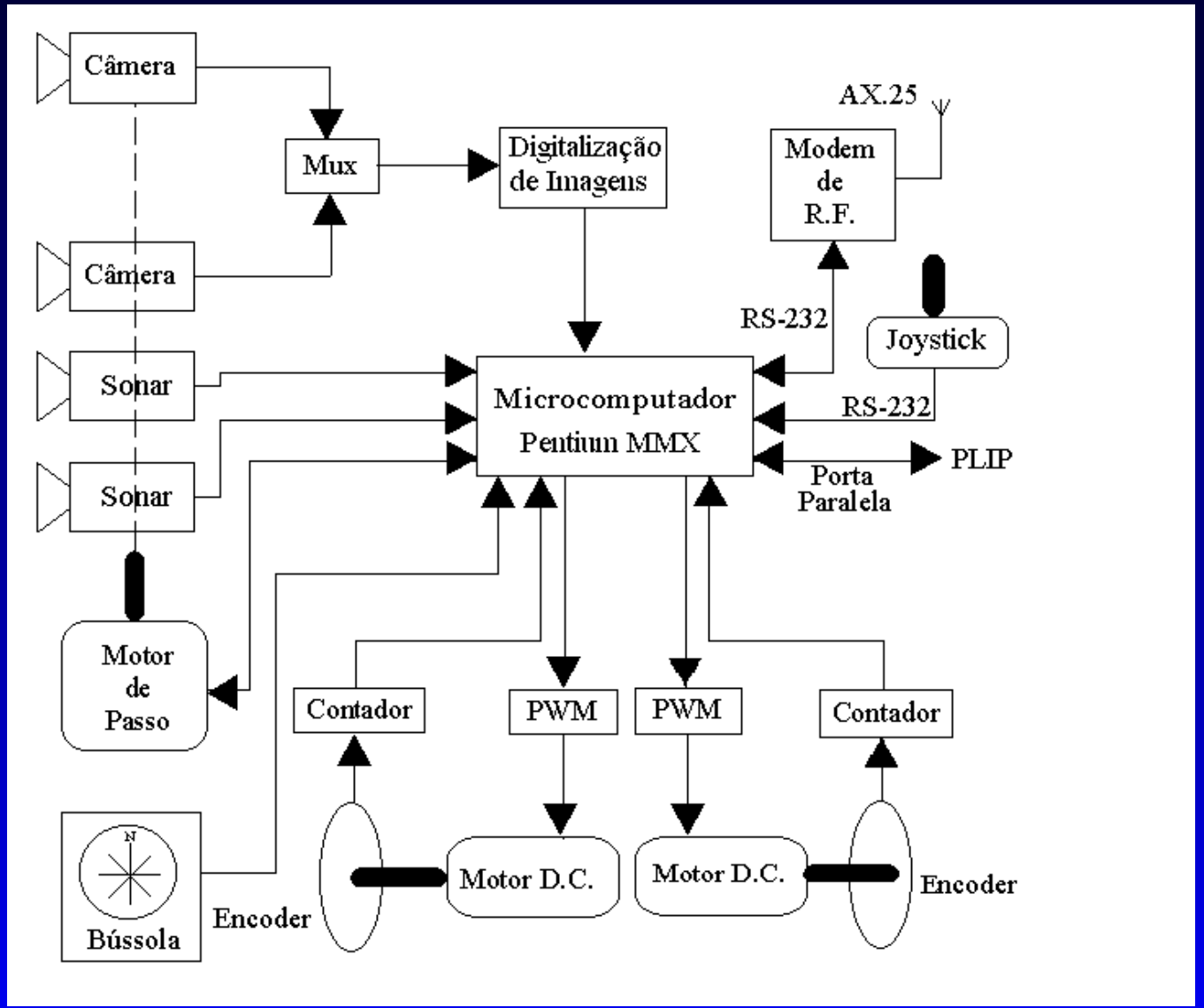
- Permite altas resoluções
- Requer sensor de índice
- Decodificação em quadratura permite multiplicar por 4 a resolução do disco
- Contagem e decodificação deve ser feita por hardware



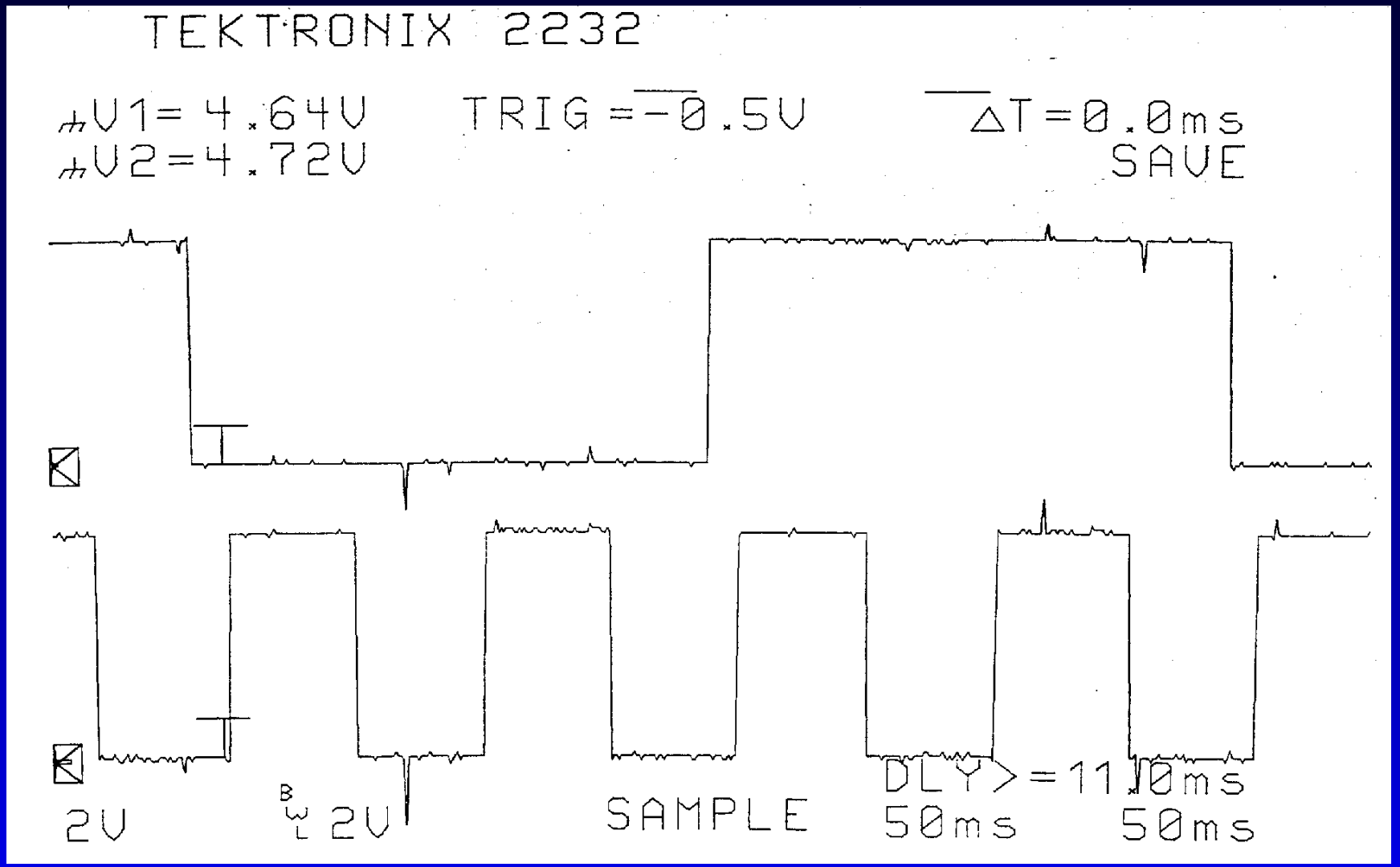
Decodificação em Quadratura



Arquitetura de *Hardware* do Twil



Amostragem de Encoder e Bússola

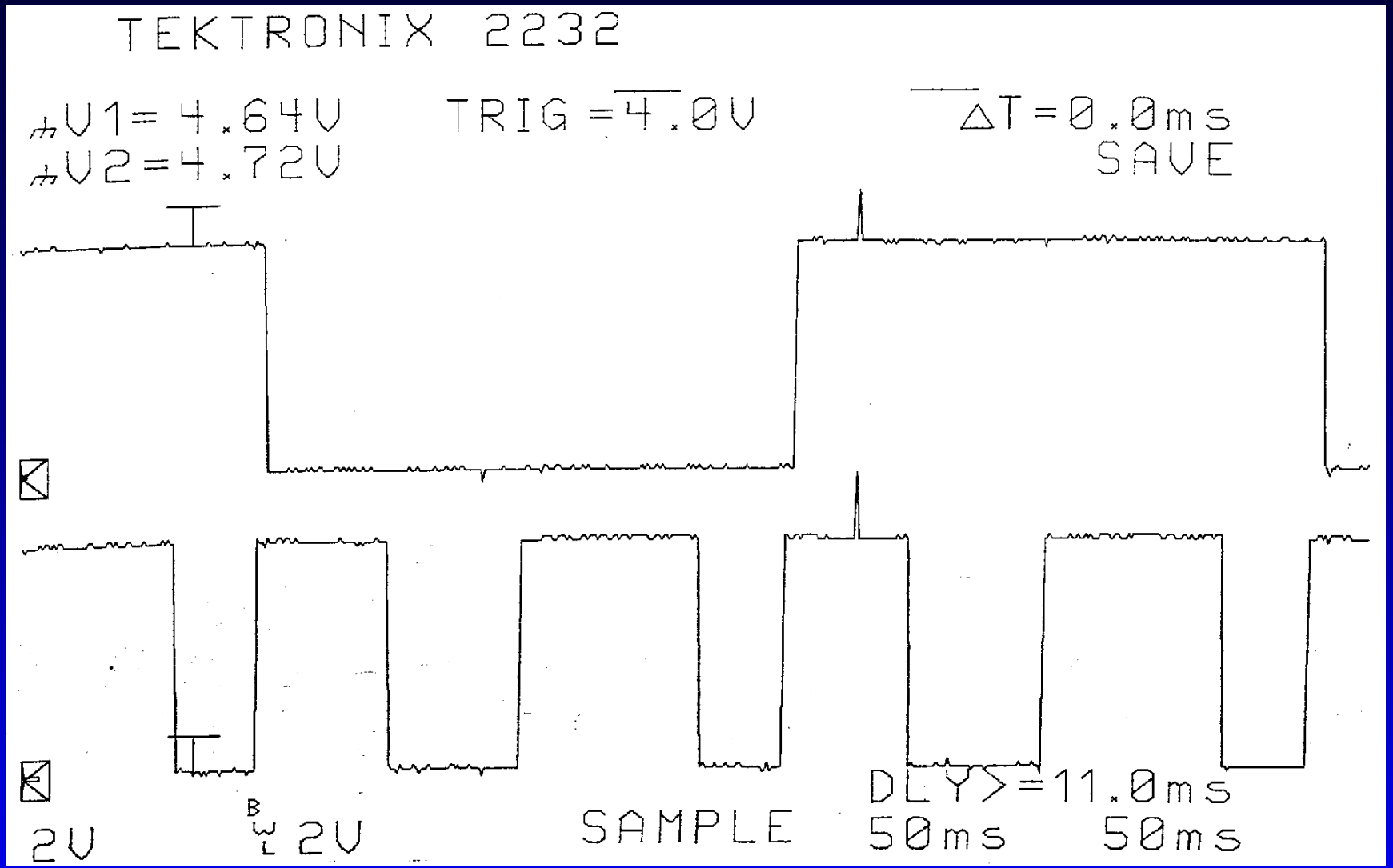


UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

Problemas de Concorrência

- Em sistemas com diversos sensores e atuadores geralmente ocorrem problemas de concorrência
- Os diversos dispositivos possuem exigências de temporização difícieis de serem abordadas de forma empírica
- Torna-se necessário o uso de um sistema operacional multitarefa e que opere em tempo-real
- Permite tratar cada dispositivo de forma independente
- A quantidade de dispositivos força a utilização de uma arquitetura adequada de software

Encoder e Bússola Sincronizados

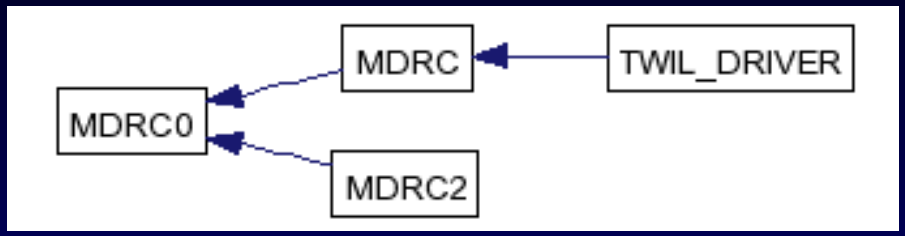


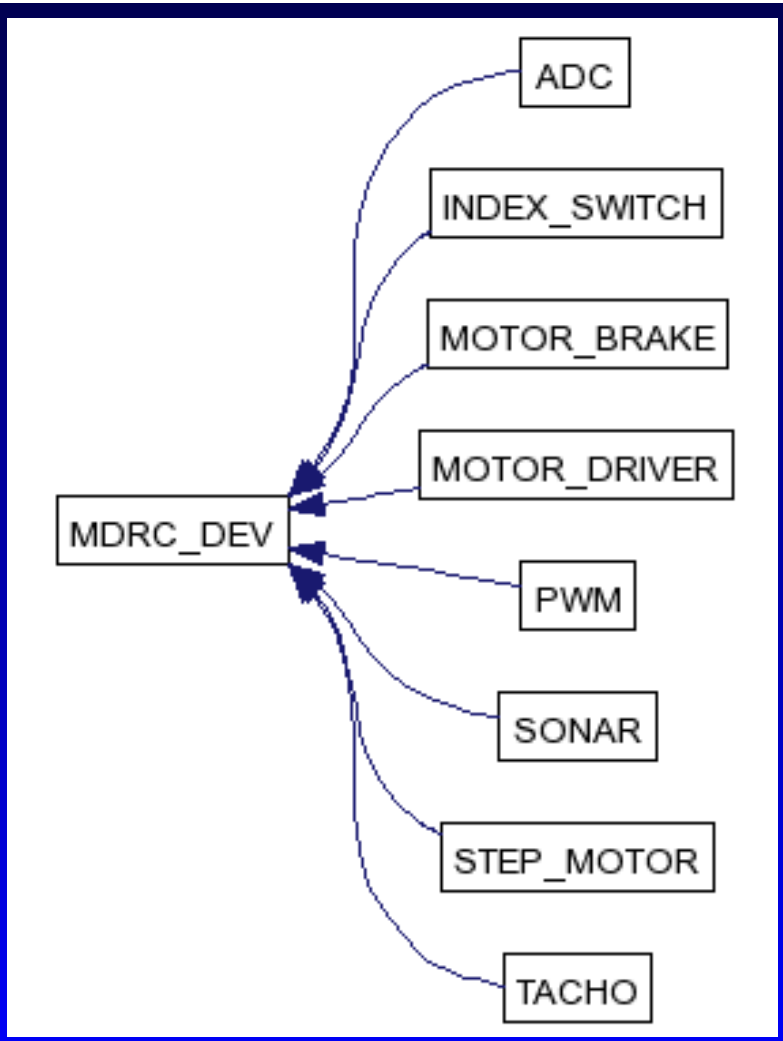
UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

Arquitetura de *Software* do Twil

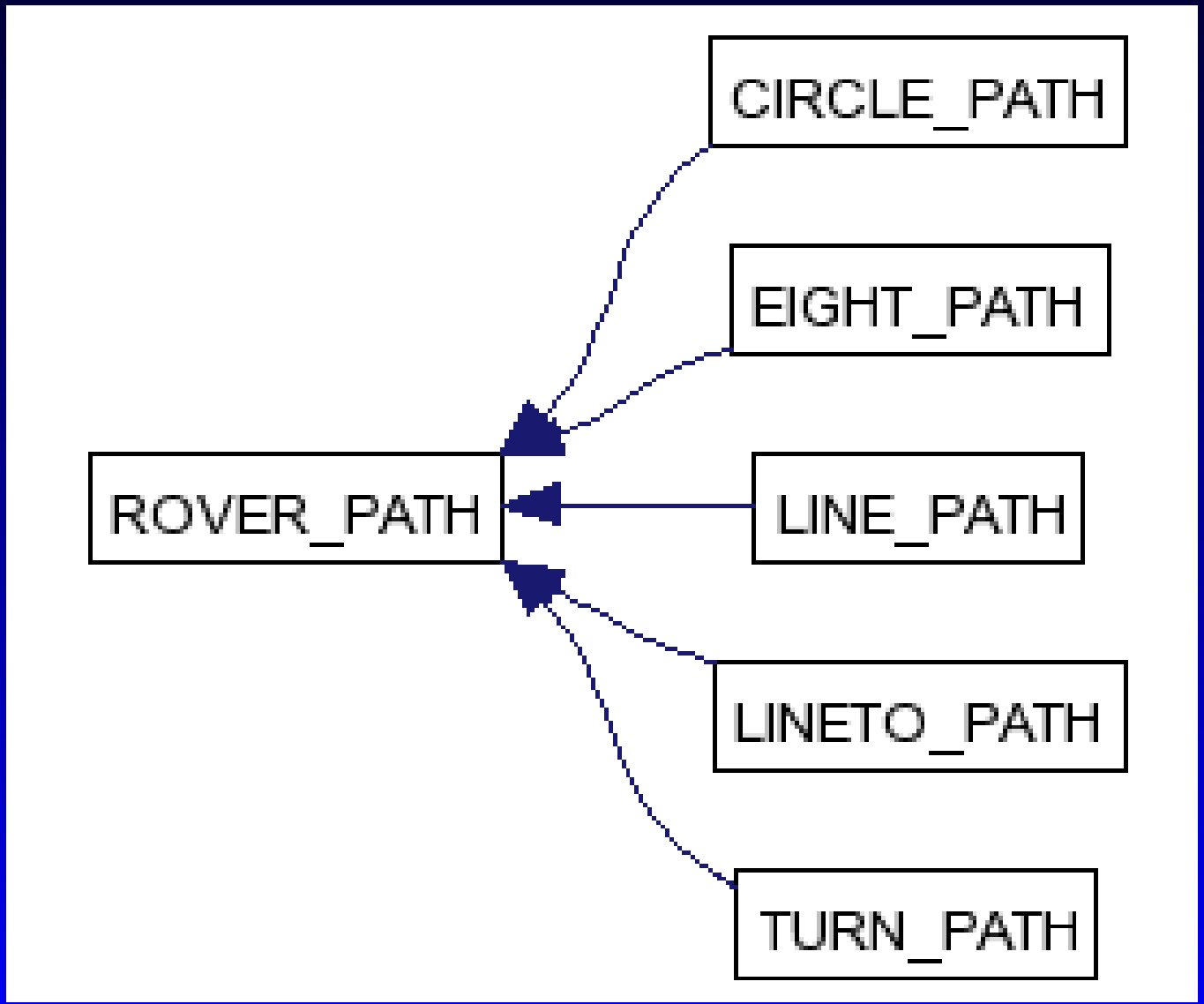
- O software executa no RTAI no modo LXRT
- Cada dispositivo é tratado por um trhead
- A biblioteca libtwil.a suporta as operações que podem ser realizadas pelo usuário.
- A biblioteca libmdrc.a suporta as operações possíveis no hardware e é utilizada pela libtwil.a
- O driver mdrc.o é utilizado pela libmdrc.a e acessa diretamente o hardware

Dispositivos

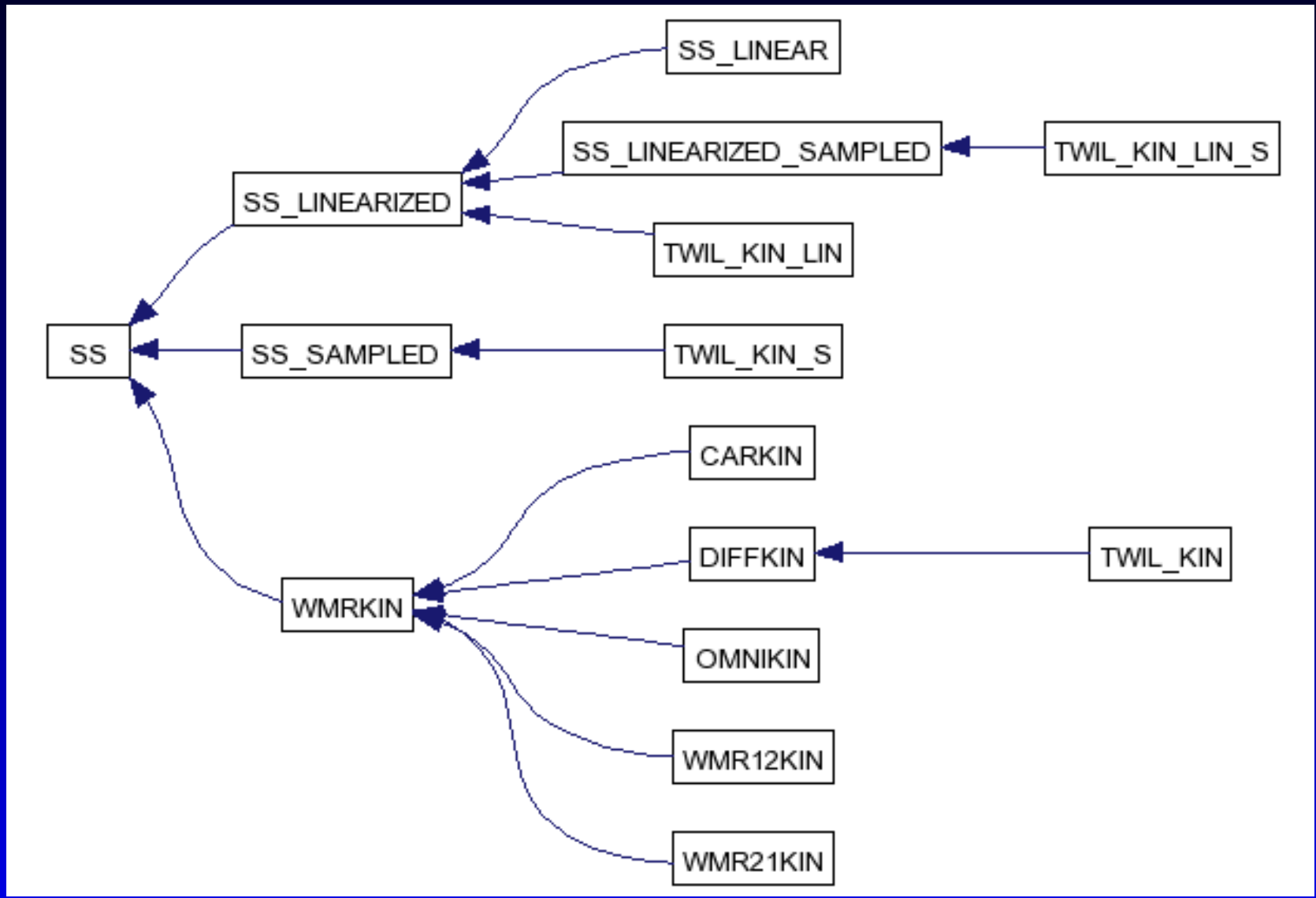


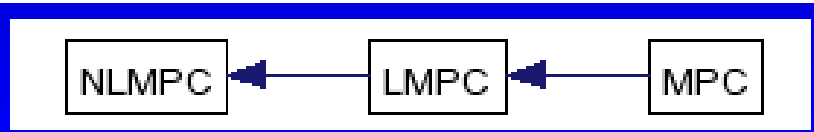


Trajetórias



Modelos e Controladores





Classe TWIL_DRIVER

```
Class TWIL_DRIVER
{
        public:
        MOTOR_DRIVER rmotor;
        MOTOR_DRIVER lmotor;
        STEP_MOTOR pmotor;
        TACHO rtacho;
        TACHO ltacho;
        SONAR tsonar;
        SONAR bsonar;
        COMPASS compass;
        TWIL_DRIVER(void);
        ~TWIL_DRIVER(void);
        void on(void);
        void off(void);
};
```

UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

Classe MOTOR_DRIVER

```
class MOTOR_DRIVER
{
        double volt;
        PWM *pwm;

        public:
        MOTOR_DRIVER(int number,double voltage,
                double sw_freq=SW_FREQ);
        ~MOTOR_DRIVER(void);
        void on(void);
        void off(void);
        double operator=(double voltage);
};
```

UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

Classe PWM

```
class PWM
{
        int dev;

        public:
        PWM(int number,double sw_freq=SW_FREQ);
        ~PWM(void);
        unsigned int operator=(double dutycicle);
        void on(void);
        void off(void);
        void freq(double frequency);
        double get_freq(void);
        class bad_pwm { };
};
```

Implementação Classe PWM

```
PWM::PWM(int number,double sw_freq)
{
    switch(number)
    {
        case 0:
        {
            dev=open("/dev/pwm0",O_WRONLY | O_SYNC);
            freq(sw_freq);
            break;
        }
        case 1:
        {
            dev=open("/dev/pwm1",O_WRONLY | O_SYNC);
            freq(sw_freq);
            break;
        }
        default: throw bad_pwm();
    }
```

Implementação Classe PWM

```
PWM::~PWM(void)
{
        close(dev);
}
void PWM::on(void)
{
        ioctl(dev,PWM_ON);
}
void PWM::off(void)
{
        ioctl(dev,PWM_OFF);
}
void PWM::freq(double f)
{
        int max=REF_FREQ/f;
        ioctl(dev,PWM_FREQ,max);
}
```

Driver mdr.o

```
static int pwm_open(struct inode *inode,
    struct file *file)
{
    int counter;
    int base=pwm_base(inode,&counter);
    outb((counter << 6) | ONE_SHOT_BIN,base+CTRL8254);
    outb((max_count/2) & 0xff,base+counter);
    outb((max_count/2) >> 8,base+counter);
    return 0;
}
```

Driver mdrc.o

```
static int pwm_release(struct inode *inode,
    struct file *file)
{
    int counter;
    int base=pwm_base(inode,&counter);
    outb((max_count/2) & 0xff,base+counter);
    outb((max_count/2) >> 8, base+counter);
    return 0;
}
```

Driver mdrc.o

```
static int pwm_ioctl(struct inode *inode,
    struct file *file,unsigned int cmd,
    unsigned long arg)
{
    int nbr=MINOR(inode-> i_rdev)-PWM0_MINOR;
    switch(cmd)
    {
        case PWM_ON:
        {
            unsigned char olddata=inpb(baseadd);
            outb(olddata & (~(0x01 << nbr)),baseadd);
            break;
        }
        default: return -EINVAL;
    }
    return 0;
}
```

UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

Conclusão

- C++ é melhor para controle avançado do que Java
  - A teoria de controle moderno e o controle de robôs em particular é baseado em álgebra matricial
  - C++ supporta sobrecarga operadores
  - Bibliotecas de manipulação de matrizes podem ser construidas de forma bastante conveniente
- É necessário o suporte de um sistema de tempo real
  - Não há suporte para execução de Java em tempo real
  - Real-time java não é tão madura quanto POSIX

UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

Conclusão

- Arquiteturas abertas para controle de robôs podem ser construídas à um custo relativamente baixo
- Espera-se que não aconteça o mesmo que aconteceu com manipuladores, onde as arquiteturas são proprietárias