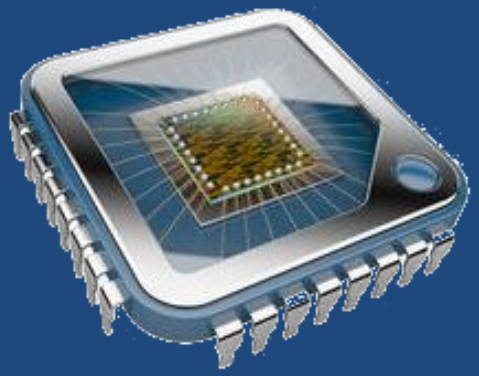
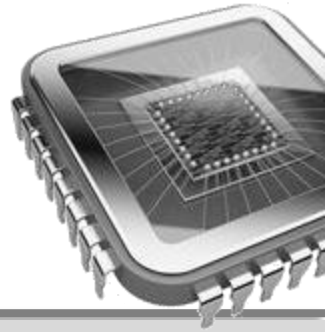


Introdução à Programação
Aula 14

Prof. Max Santana Rolemberg Farias
max.santana@univasf.edu.br
Colegiado de Engenharia de Computação

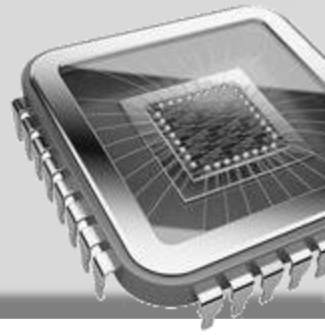
GESE
Grupo de Engenharia de Sistemas Embarcados

CEComp
Engenharia da Computação
UNIVASF



O QUE É UM PONTEIRO?

Ponteiros



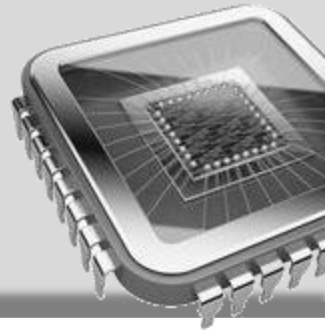
- Variáveis que guardam endereços de memória.

- Para declarar um ponteiro temos a seguinte forma:

<tipo_ponteiro> *<nome_variável>

Quando um ponteiro é declarado ele aponta para um lugar de memória indefinido. Por isso, devemos sempre inicializar um ponteiro antes de utilizá-lo.

Ponteiros

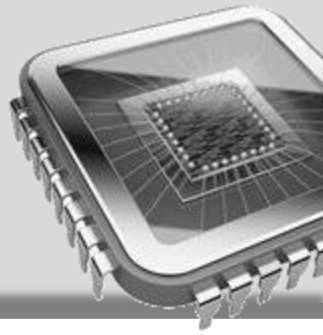


Para atribuir um valor válido a um ponteiro podemos atribuir um endereço de memória de uma variável declarada.

- Para saber a posição de memória (endereço) de uma variável basta usar o operador &.

```
int c = 10;
int *p;
p = &c; // agora o ponteiro p pode ser utilizado
```

Ponteiros

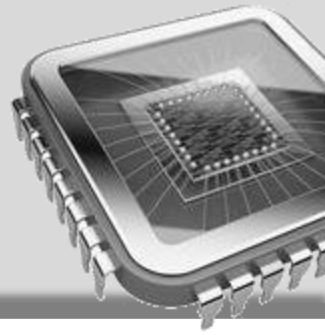


```
int c = 10;
int *p;
p = &c; // agora o ponteiro p pode ser utilizado
```

- Para alterar o valor de c usando o ponteiro p basta usar o operador *.

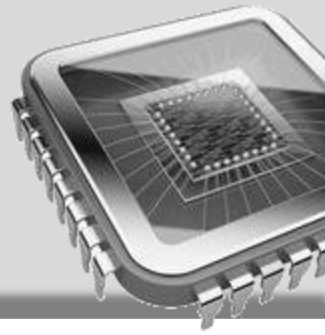
```
*p = 12; // *p é equivalente a variável c
```

Ponteiros
Exemplos



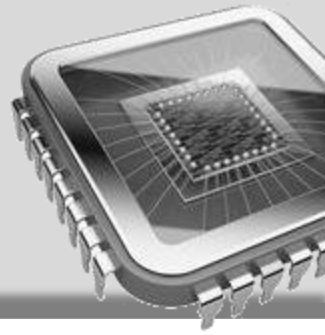
```
#include <stdio.h>
main(){
  int n, v;
  int *p;
  n = 5;
  p = &n;
  v = *p;
  printf("O endereço para onde o ponteiro aponta: %p\n", p);
  printf("O valor da variável apontada : %d\n",*p);
}
```

Ponteiros
Exemplos



```
#include <stdio.h>
main(){
  int n, *p;
  n = 50;
  p = &n;
  printf("O valor inicial: %d\n", n);
  *p = 100;
  printf("O valor final: %d\n", n);
}
```

Ponteiros
Operações



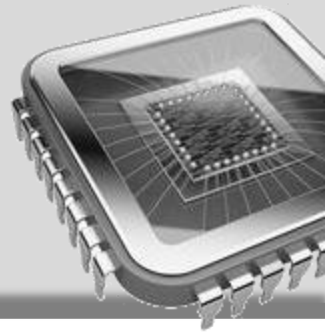
- Atribuição
  - Se temos dois ponteiros, p1 e p2, e quisermos que p1 aponte para o mesmo endereço de p2, basta fazer:

$$P1 = p2$$

  - Para a variável apontada por p1 ter o mesmo conteúdo da variável apontada por p2, basta fazer:

$$*p1 = *p2$$

Ponteiros
Operações



- Incremento e decremento
  - Quando incrementamos um ponteiro ele passa a pontar para o próximo endereço do tipo o qual o ponteiro aponta.

p++

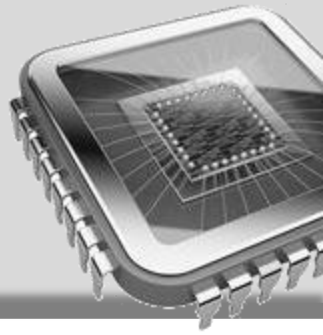
  - O decremento funciona de forma semelhante.

p--

Para incrementar e decrementar o conteúdo das variáveis para as quais os ponteiros apontam basta faz:

(*p)++

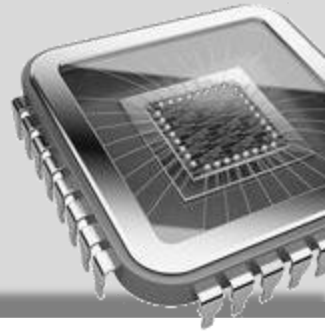
Ponteiros
Operações



- Comparação
  - Para saber se dois ponteiros são iguais ou diferentes basta usar == e !=
  - Os operadores do tipo >, <, >= e <= são usados para comparar qual ponteiro aponta para uma posição mais alta na memória.

```
p1 == p2;
p1 > p2;
```

Vetores de Ponteiros



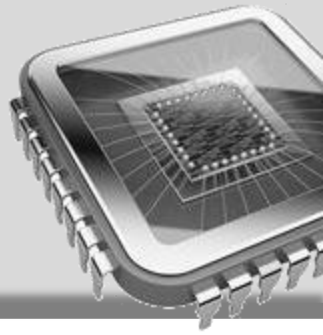
Podemos construir vetores de ponteiros como declaramos vetores de qualquer tipo.

- Um exemplo de declaração de um vetor de ponteiro é:

```
int *pvetor[10];
```

pvetor é um vetor que armazena 10 ponteiros para inteiros.

Ponteiro para Ponteiro



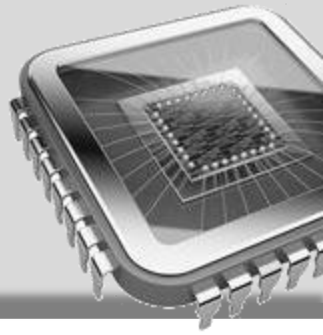
- Para declarar um ponteiro para um ponteiro usa a seguinte notação:

<tipo da variável> **<nome da variável>;

****<nome da variável>** é o conteúdo final da variável apontada.

***<nome da variável>** é o conteúdo do ponteiro intermediário.

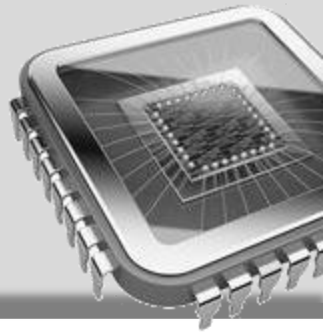
Ponteiro para Ponteiro



- Na linguagem C é possível declarar ponteiros para ponteiros para ponteiros...

- Para isso basta aumentar o número de asteriscos na declaração.

Ponteiro para Ponteiro Exemplo



```
#include <stdio.h>

void main() {
  float pi = 3.1415, *pf, **ppf;
  pf = &pi;
  ppf = &pf;
  printf("\n%.4f",**ppf);
  printf("\n%.4f",*pf);
}
```