MA 511: Computer Programming

Lecture 11

http://www.iitg.ernet.in/psm/indexing_ma511/y08/index.html

Partha Sarathi Mandal

psm@iitg.ernet.ac.in

Dept. of Mathematics, IIT Guwahati

Semester 1, 2008-09
Mon 10:00-10:55 Tue 11:00-11:55 Fri 9:00-9:55 Class: 1G2
MA512 Lab : Wed 14:00-16:55

typedef

```
int i, j;        equivalent to        typedef int mydef;
                                      mydef i, j;

typedef struct {
        int acct_no;
        char acct_type;
        char name[80];
        float balance;
} account ;
account oldcustomer, newcustomer;
```

Member of a struct may be a struct

```
typedef struct {
        int day;
        int month;
        int year;
        } date;
typedef struct {
        int acct_no;
        char acct_type;
        char name[80];
        float balance;
        date update;
} customer[100] ;
```

Equivalent to:

```
struct date {
                int day;
                int month;
                int year;
                };
struct  account {
                int acct_no;
                char acct_type;
                char name[80];
                float balance;
                struct date update;
};
struct  account  customer[100];
```

customer[i].acct_no: variable of the structure account

customer[i].*update*.month: variable of the structure date

Union

```
Union tag {
      member 1;
           …
      member m;
};
```

```
Union account {
      int acct_no;
      char acct_type;
      char name[80];
      float balance;
};
```

- Like structures, contain members whose individual data types may differ from one another.
- union allocates the memory equal to the maximum memory required by the member of the union but structure allocates the memory equal to the total memory required by the members.
- In union, one block is used by all the member of the union but in case of structure, each member have their own memory space
- Union is useful for application where values need not be assigned to all of the members simultaneously.

pointers

- Is a ***variable*** that represents the ***location*** (address) of a data item.
- Each data item occupies one or more contiguous memory cells in computer memory.
- No of memory cells depends on the type of data item.
  - A single character needs 1 byte (8bits)
  - An integer usually needs 2 contiguous bytes
  - A floating point no needs 4 contiguous bytes
  - Double-precision quantity may needs 8 contiguous bytes

pointers

- Let v is a variable of some data item.
  float v;
- Then data item can then be assessed if we know the location of *first* memory cell.
- &v = address of v's memory location.

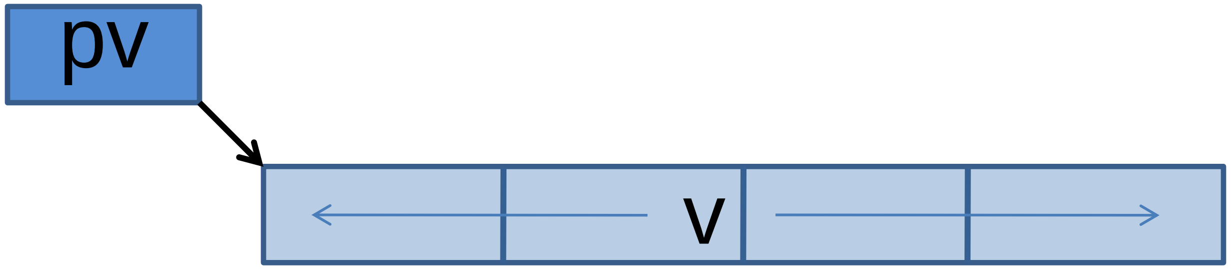
pv = &v; // & unary operator, *address operator*

pv = pointer of the variable v

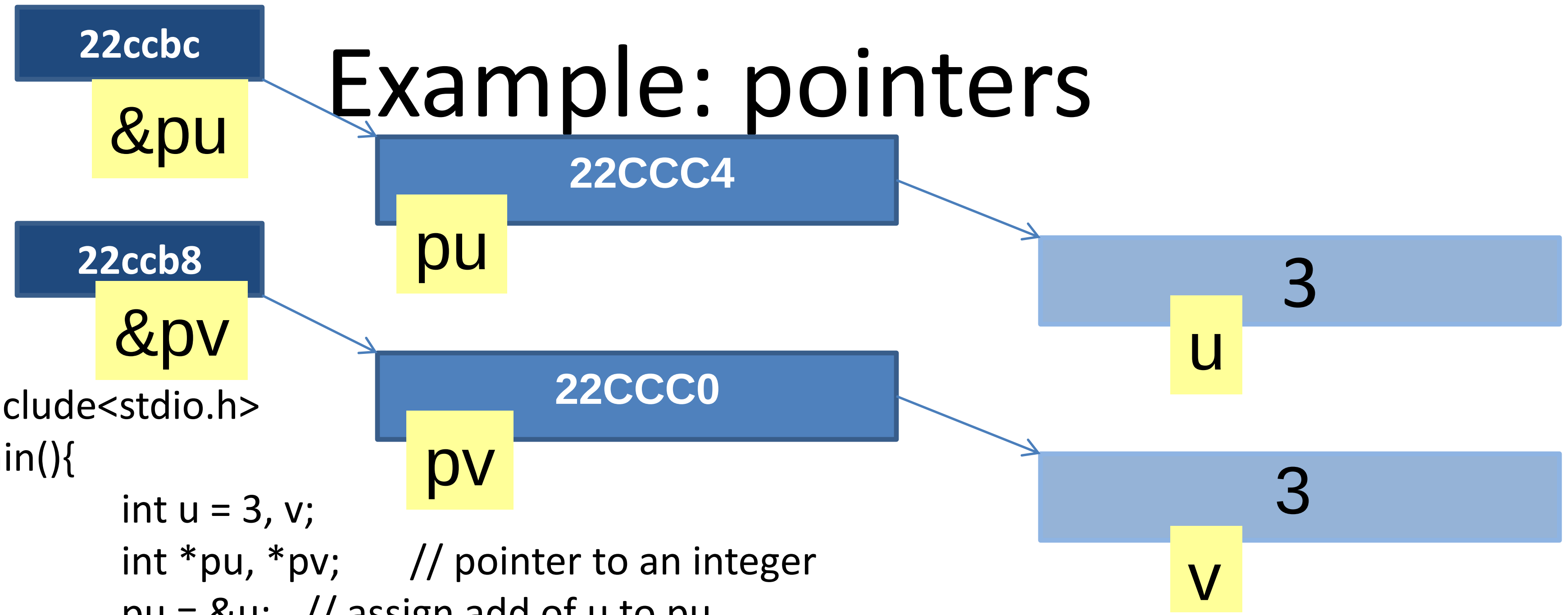
v = *pv;// * unary operator, *indirection operator*

v, *pv = represent same data type.

Now if pv = &v and u = *pv then u and v represent the same value.



Example: pointers

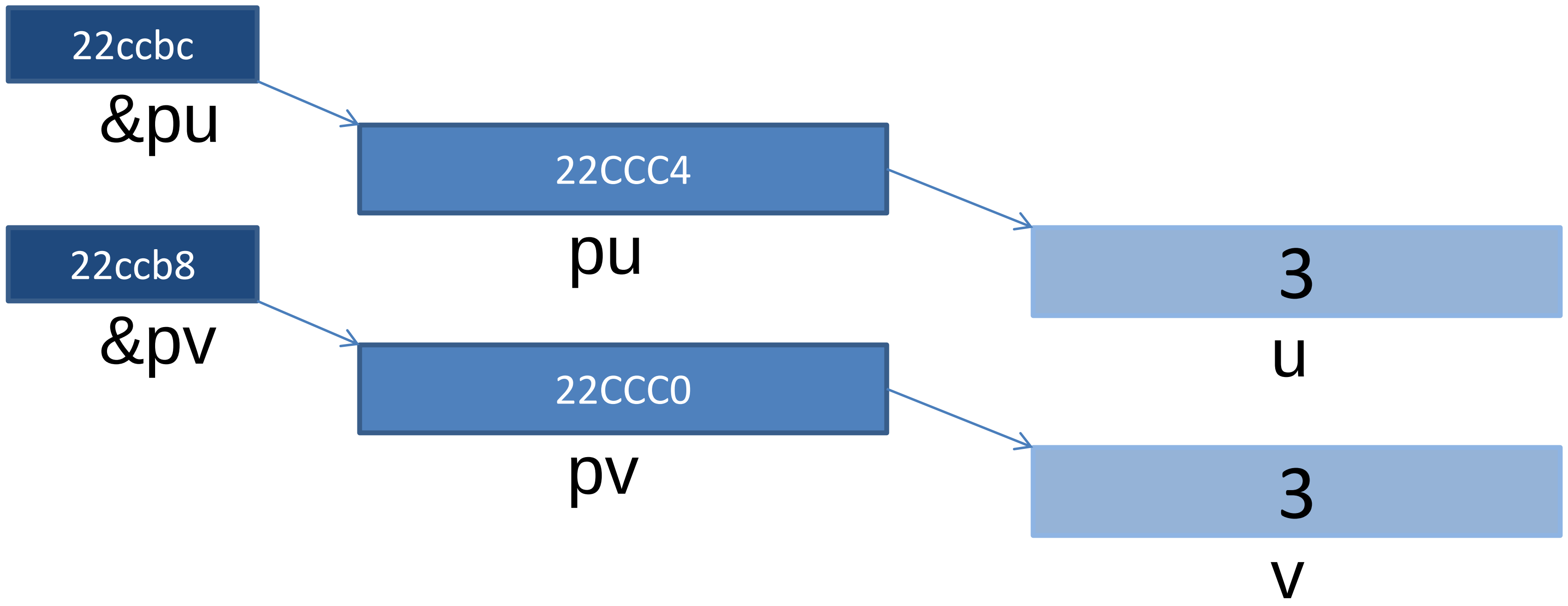


```
#include<stdio.h>
main(){
        int u = 3, v;
        int *pu, *pv;      // pointer to an integer
        pu = &u;   // assign add of u to pu
        v = *pu;                  // assign value of u to v
        pv = &v;   // assign add of v to pv
printf("u=%d  &u = %X pu = %x *pu =%d\n", u, &u, pu, *pu);
printf("v=%d  &v = %X  pv = %x *pv =%d\n",  v, &v,  pv,  *pv);
}
```

Output:

```
u=3  &u = 22CCC4 pu = 22ccc4 *pu =3
v=3  &v = 22CCC0  pv = 22ccc0 *pv =3
```

Example: pointers


22ccbc
&pu
22CCC4
pu
3
u
22ccb8
&pv
22CCC0
pv
3
v