Subject: Re: Structure field concatenation
Posted by davidf on Tue, 05 Sep 2000 14:08:21 GMT
View Forum Message <> Reply to Message

Amara Graps (Amara.Graps@mpi-hd.removethis.mpg.de) writes:

> Can anyone give me a hint about why the second pointer is overwriting
> the definition of the first pointer?

The problem here, Arara, is not that the second pointer is
overwriting the definition of the first pointer. It is that
the second pointer *IS* the first pointer! And you have
re-defined what it points to.

The problem comes in how you define the original structure:

```
   thisstruc = {orbit:'',freq:ptr_new(/allocate_heap)}
```

By allocating heap memory to the pointer, you make it
a valid pointer (to an undefined variable). When you
replicate the structure, each pointer in each structure
points to the very same undefined variable. Another
way of saying this is that each pointer is pointing to
the same area of heap memory. Obviously, if what you
store there changes, then all the pointers point to the
same changed thing.

What you want to do is define your structure like this:

```
   thisstruc = {orbit:'',freq:ptr_new()}
```

Then, when you fill them up:

```
   new[0].freq = Ptr_New(DINDGEN(100))
   new[1].freq = Ptr_New(DINDGEN(50))
```

Each freq field is a *separate* pointer to a *different*
area of memory on the heap. This is what you had in
mind.

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting
Phone: 970-221-0438 E-Mail: davidf@dfanning.com