

Re: [PATCH] Open Firmware device tree virtual filesystem

Source: <http://linux.derkeiler.com/Mailing-Lists/Kernel/2006-12/msg08106.html>

- *From:* David Miller <davem@xxxxxxxxxxxxxx>
 - *Date:* Sun, 31 Dec 2006 01:52:10 -0800 (PST)
-

From: Mitch Bradley <wmb@xxxxxxxxxxxxxx>
Date: Sat, 30 Dec 2006 23:36:45 -1000

The base interface function is `callofw()`, which is effectively identical to `call_prom_ret()` in `arch/powerpc/kernel/prom_init.c`. So it seems that PowerPC could use it. I suppose I could change the name of `callofw()` to `call_prom_ret()`, thus making the base interface identical to PowerPC's. All it does is argument marshalling, translating between C varargs argument lists and the OFW argarray format.

Please create explicit function calls for each operation, this way the caller is immune to open-firmware implementation details.

SPARC should be able to use that same base interface function directly. It is written to the standard OFW client interface.

Sparc 32-bit predates the OFW specification and does things differently.

Please use a functional interface using a C function for each device tree operation, then the implementation behind it doesn't matter.

It wouldn't work on ancient Sun machines with the `sunmon romvec` interface, but Sun stopped making such machines something like 16 years ago.

Yet we still support them in the 32-bit sparc port. And it's so easy to support this properly, please use the clean stuff we've created for this.

I did consider first creating a memory data structure identical to the `powerpc/sparc` one, but that looked like it was going to be essentially twice as much code for no extra capability. The code to traverse the

Re: [PATCH] Open Firmware device tree virtual filesystem

device tree and create the memory data structure is roughly the same as the code to create the filesystem structure. I just didn't see the value of an intermediate representation for systems that don't otherwise need it.

Since we put it in memory, we have zero reason to call into the firmware for device tree access and this simplifies things a lot.

But all of that really doesn't matter, if you use a functional C interface for each device tree access operation, it doesn't matter what's behind it right?

—

To unsubscribe from this list: send the line "unsubscribe linux-kernel" in the body of a message to majordomo@xxxxxxxxxxxxxxxxx

More majordomo info at <http://vger.kernel.org/majordomo-info.html>

Please read the FAQ at <http://www.tux.org/lkml/>