

## Re: [PATCH] Generic platform device IDE driver

---

*Source:* <http://linux.derkeiler.com/Mailing-Lists/Kernel/2006-10/msg04769.html>

---

- *From:* Jeff Garzik <jgarzik@xxxxxxxx>
  - *Date:* Fri, 13 Oct 2006 08:46:49 -0400
- 

Paul Mundt wrote:

On Fri, Oct 13, 2006 at 08:52:19AM +0100, Russell King wrote:

On Thu, Oct 12, 2006 at 03:13:48PM +0900, Paul Mundt wrote:

Yes, that's one thing I was thinking of as well.. Here's a patch that makes an attempt at that, can you give it a try and see if it works for you? This applies on top of the earlier patch. None of the ARM, SH, or H8300 cases need to do the remapping at least.

It's likely that ARM will switch over to using the MMIO resources if this patch makes it in. There's certain ARM platforms which would benefit from this change (since `inb()` and friends are more complex than they necessarily need to be.)

However, one issue needs to be solved before we could do that – how do we handle the case where the IDE registers are on a 4 byte spacing interval instead of the usual 1 byte?

We could solve this in the driver, but it sounds like this is something that libata should have some visibility of directly.

I notice that this driver is calling `ata_std_ports()` which handles the standard setup. Maybe that needs to become a little more intelligent?

If we go this route, I suppose the easiest option will be simply to have a private structure with a few function pointers for these sorts of things, or we can simply have an element for the spacing interval if you don't foresee needing to change the `ioaddr`s in any fashion beyond the register spacing.. Which approach would you be more comfortable with? Are there any other items that you're concerned with in the current driver?

Re: [PATCH] Generic platform device IDE driver

Here's the decision matrix for libata: Will the hardware use the standard taskfile push/pull functions like `ata_tf_load()`, `ata_tf_read()`, `ata_exec_command()`, etc.? If yes, simply replace `ata_std_ports()` call with a call to your own function, written similarly to `pd_c_ata_setup_port()` in `sata_promise.c`.

If the hardware requires non-standard I/O accessors, or the register sizes themselves changed, then you must implement your own taskfile I/O functions, similar to `k2_sata_tf_load()`, `k2_sata_tf_read()`, and `k2_bmdma_setup_mmio()` in `sata_svw.c`. For this case, data in `struct ata_ioports` is largely up to you to manage, or ignore.

If there are special command setup or teardown operations, there are other standard hooks to override as well.

Jeff

—

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/>