Porting of drivers/i2c/busses/i2c-bcm2835.c

May 26, 2017

1 Introduction

The driver drivers/i2c/busses/i2c-bcm2835.c was introduced in 2013 in the commit f3b54b9. It implements a very basic I2C host driver for the BCM2835 SoC. This driver was accompanied by changes in Kconfig and a makefile, as well as a new documentation file. Gcc produces 7 errors and 3 warnings that reduce to 6 errors.

2 reinit_completion

Gcc reports that the function `reinit_completion` is not known. We try the following patch query (step1.cocci):

```
@bad depends on before@
@@

  reinit_completion
  (...)

@depends on !bad@
@@

+ reinit_completion
  (...)
```

A variety of changes are illustrated, including adding the call from nothing, adding the call in place of a call to init_completion, and adding the call in place of call to INIT_COMPLETION. The calls to init_completion are indeed presented as being bugs. It is not clear what was there before. This is a failure for our approach.

=== failure for approach

3 struct i2c_adapter_quirks

Gcc reports that the type `struct i2c_adapter_quirks` is not known. We try the following patch query (step2.cocci):

```
@bad depends on before@
@@

  struct i2c_adapter_quirks

@depends on !bad@
```

```
@@

+ struct i2c_adapter_quirks
```

The first commit is a94d306 at 20%. This shows that the i2c_adapter_quirks structure is added from nothing, and that an initialization of a quirks field to this structure under a conditional is also added from nothing. Our code has the structure but not the conditional. We can guess that if the assignment is there with no conditional around it, then we can drop it as well. Looking further, this is indeed the change made by the next commit, 3aaa34b at 16%.

Note that in this case a commit with a lower score is a better match. The lower score comes from the number of fields in the quirks structure, but that is not relevant to the change.

```
@r@
identifier i;
@@

-struct i2c_adapter_quirks i = { ... };

@@
struct i2c_adapter *adap;
identifier r.i;
@@

- adap->quirks = &i;
```

=== success 2/15, 3aaa34b

4 I2C_AQ_NO_CLK_STRETCH

Gcc reports that the constant I2C_AQ_NO_CLK_STRETCH is not known. We try the following patch query (step3.cocci):

```
@bad depends on before@
symbol I2C_AQ_NO_CLK_STRETCH;
@@

  I2C_AQ_NO_CLK_STRETCH

@depends on !bad@
@@

+ I2C_AQ_NO_CLK_STRETCH
```

The constant is only used in the structure that is removed by the previous rule, so there is no need to even look at the commit history in this case.

=== success 1/1, a94d306

5 devm_ioremap_resource

Gcc reports that the function devm_ioremap_resource is not known. We try the following patch query (step4.cocci):

```
@bad depends on before@
@@

  devm_ioremap_resource
  (...)

@depends on !bad@
@@

+ devm_ioremap_resource
  (...)
```

The first four commits, 0859ffc, cd5d581, 10f0577, and c829f95, all at 50%, just replace `devm_request_and_ioremap` by `devm_ioremap_resource`, but three of the four are presented as bug fixes, because the code is testing the wrong return balue. We thus move on to the commits with a lower percentage, where more changes are made.

The next commit is 3140747 at 25%. This commit suggests backporting `devm_ioremap_resource` to `ioremap`. This is not the target function that was used in the 50% commits, and independent knowledge suggests that this is not the right change because there is no `request_mem_region`.

The next commit is 32766ff at 25%. This commit again suggests backporting `devm_ioremap_resource` to `devm_request_and_ioremap`, but the context is not satisfactory. This commit illustrates the NULL test, as mentioned by the 50% commits, but there is no informationabout what to do if the test is true, because the commit code doesn't fail in that case, and our code does.

The final 25% commit is 62ffac1, and this one has no error checking or handling in the original code. We thus move on to the 16% commits.

The next commit is 6d4294d at 16%. This one contains many examples that illustrate the NULL test followed by a return. The returns all return `-EADDRNOTAVAIL`. This seems promising. We look at another example for confirmation.

The next commit is f560fabat 16%. This illustrates the same as the previous one, but uses `-EBUSY` as the return value.

The next commit is 7c390a7 at 16%. This time the return value on failure is `-ENOMEM`.

This is a partial failure for Prequel, because we don't get a definite answer about what the return value should be. Looking back at the original code, it seems that it should be `-EBUSY`.

The patches also vary by whether the NULL test is implemented with `!` or `== NULL`. We choose `!`, which independent knowledge suggests is more common in the kernel.

The original version of our driver also contained a `dev_err` call in case of failure. Only the `-ENOMEM` patch among the 16% ones that we looked at contained that, and there was no particular motivation for including it. On the other hand, the presence of such a debugging statement is not very important either, as illustrated by its lacking most other cases.

```
@@
expression e;
@@

  e =
-  devm_ioremap_resource
+  devm_request_and_ioremap
  (...);
  if (
-      ERR_PTR(e)
+      !e
     )
      return
```

```
-           PTR_ERR(e)
+           -EBUSY
      ;
```

=== partial failure for the approach 11/28, 7c390a7

6 I2C_CLASS_DEPRECATED

Gcc reports that the constant `I2C_CLASS_DEPRECATED` is not known. We try the following patch query (step5.cocci):

```
@bad depends on before@
symbol I2C_CLASS_DEPRECATED;
@@

  I2C_CLASS_DEPRECATED

@depends on !bad@
@@

+ I2C_CLASS_DEPRECATED
```

The first commit is 878f00b abd 50%. The change and the commit message suggest that this value is specific to the `class` field of an`i2c_adapter` structure and should just be dropped when backporting.

In the commit theconstant is used with other constants in a structure initialization. In our driver, it is used alone in an assignment. We thus look at someother commits to see if we could just drop the assignment.

There is no example, however, that has only `I2C_CLASS_DEPRECATED` as the field value. Since we are working with a bit or operation, presumably, nothing is represented as 0. We could either hope that the containing structure is kzalloced, or explicitly set the field to 0.

Looking back at the old code, however, shows that the `class` field was originally set to `I2C_CLASS_HWMON`. The problem is that the code has been subject to two patches: an earlier patch, 37888f71e2c, creates `I2C_CLASS_HWMON | I2C_CLASS_DEPRECATED` and then a later patch, 37e4f91aa391, replaces this by just `I2C_CLASS_DEPRECATED`. From the endpoint, we have noway of recovering the initial value, `I2C_CLASS_HWMON`. This is again a failure for Prequel.

== failure for approach

7 struct i2c_adapter.quirks

Gcc reports that the field `quirks` of the structure `struct struct i2c_adapter` is not known. We try the following patch query (step6.cocci):

```
@bad1 depends on before@
struct i2c_adapter *e;
@@

 e->quirks

@bad2 depends on before@
struct i2c_adapter e;
@@

 e.quirks
```

```
@bad3 depends on before@
identifier i;
expression e;
@@

struct i2c_adapter i = {
  .quirks
    = e,
};

@depends on !bad1 && !bad2 && !bad3@
struct i2c_adapter *e;
@@

+ e->quirks

@depends on !bad1 && !bad2 && !bad3@
struct i2c_adapter e;
@@

+ e.quirks

@depends on !bad1 && !bad2 && !bad3@
identifier i;
expression e;
@@

struct i2c_adapter i = {
+  .quirks
    = e,
};
```

The only occurrence is removed by change described in Section 3. So there is nothing to do here.
=== success 3/15, 3aaa34b