Porting of drivers/iio/adc/nau7802.c

May 26, 2017

1 Introduction

The driver drivers/iio/adc/nau7802.c was introduced in 2013 in the commit 8b20be8. It is a Nuvoton NAU7802 ADC driver. This driver was accompanied by changesin Kconfig and a make file, as well as a new documentation file. Gcc produces 2 errors and 1 warning that reduce to 2 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
  (...)
```

We have seen this issue in step1 of f3b54b9:2013-02-12/drivers_i2c_busses_i2c-bcm2835.c_f3b54b9. We obtain the same results. As noted there, the results don'tpermit to decide what to do, and this is a failure for our approach.

=== failure for approach

3 devm_iio_device_alloc

Gcc reports that the function `devm_iio_device_alloc` is not known. We try the following patch query (step2.cocci):

```
@bad depends on before@
@@

  devm_iio_device_alloc
  (...)

@depends on !bad@
@@
```

```
+ devm_iio_device_alloc
  (...)
```

The first commit is b909459 at 15%. This only shows what to do in the probe function, but there should be a corresponding action for the remove function as well.

The next commit is 3d0ccba at 14%. This illusrates how to change a remove function as well. This commit illustrates the changes required when there is no resource-release requiring call before the call to devm_iio_device_alloc (independent knowledge required). That is also the case for our driver. We make the changes as follows:

```
@r@
identifier i,pfn,rfn;
@@

struct i2c_driver i = {
        .probe = pfn,
        .remove = rfn,
};

@s@
identifier r.pfn,l1,l2;
expression e,e1,e2,ret;
fresh identifier out;
@@

pfn(...) {
  ... when any
  e =
-     devm_iio_device_alloc(e1,
+     iio_device_alloc(
        e2);
(
  <...
  if (...)
-    return ret;
+    goto out;
  ...>
&
  ... when any
(
  return \(0\|ret\);
|
 l1: ... when != l2:
+    out:
+    iio_device_free(e);
     return ret;
)
)
}

@depends on s@
identifier r.rfn,i;
@@
```

```
rfn(...)
{
        ... when any
        struct iio_dev *i = i2c_get_clientdata(...);
        ... when any
+       iio_device_free(i);
        return ...;
}
```

=== success 2/51, 3d0ccba