

3. Driver Patch

```
--- a/cc3100-spi-kernelmodul/cc3100-spi.c
+++ b/cc3100-spi-kernelmodul/cc3100-spi.c
@@ -21,6 +21,8 @@
#include <linux/spi/spi.h>
#include "cc3100-spi.h"

struct cc3100_spi { struct
device *dev; struct
spi_device *spidev;
@@ -42,7 +44,9 @@ ssize_t cc3100_spi_read(struct file *fildes, char __user * buff, size_t
count,
/* size check of given length */ if (count >
sizeof(fp->buffer)) return -EFAULT;
```

```

err = spi_read(fp->spidev, fp->buffer,
count);
if(copy_to_user(buff, fp->buffer, count)) return -
EFAULT;

@@ -58,6 +62,7 @@ ssize_t cc3100_spi_write(struct file *fildes, const char __user *
buff, if (count > sizeof(fp->buffer)) return -EFAULT;
if(copy_from_user(fp->buffer, buff, count )) return -
EFAULT;
return spi_write(fp->spidev, fp->buffer,
count);
}
@@ -71,10 +76,12 @@ static long cc3100_spi_ioctl(struct file *file, unsigned int cmd,
switch(cmd)
{
    case CC3100_nHIB_HIGH:
-
-                gpio_set_value(fp->pdata->gpio_nHIB, 1);
+
+                if (gpio_is_valid(fp->pdata->gpio_nHIB))
+
+                gpio_set_value(fp->pdata-
>gpio_nHIB, 1); break; case
CC3100_nHIB_LOW:

-                gpio_set_value(fp->pdata->gpio_nHIB, 0);
+
+                if (gpio_is_valid(fp->pdata->gpio_nHIB))
+
+                gpio_set_value(fp->pdata-
>gpio_nHIB, 0); break; default:
dev_err(fp->dev, "unkown ioctl\n");

```

```

@@ -95,7 +102,8 @@ int cc3100_spi_release(struct inode *inode, struct file
*filp)          container_of(filp->private_data, struct cc3100_spi, mdev);
/* set nHIB low */
-   gpio_set_value(fp->pdata->gpio_nHIB, 0);
+ if (gpio_is_valid(fp->pdata->gpio_nHIB))
+   gpio_set_value(fp->pdata->gpio_nHIB, 0);
   return
0;
}
@@ -150,14 +158,16 @@ static struct cc3100_spi_platform_data *cc3100_spi_of_init(
   return ERR_PTR(-ENOMEM);
}

-   if(of_property_read_u32(node, "gpio_irq", &pdata->gpio_irq) ||
-   of_property_read_u32(node, "gpio_nHIB", &pdata->gpio_nHIB)) - {
+ if(of_property_read_u32(node, "gpio_irq", &pdata->gpio_irq)) {

```

```

   dev_err(&spi->dev, "missing device tree property\n");
   kfree(pdata);   return ERR_PTR(-ENODEV);
}

+ if(of_property_read_u32(node, "gpio_nHIB", &pdata->gpio_nHIB)) {
+   pdata->gpio_nHIB = -1;
+ } +   return
pdata;
}
#else
@@ -221,15 +231,17 @@ static int cc3100_spi_probe(struct spi_device *spi)
}

/* register pin for nHIB */
-   err=gpio_request(fp->pdata->gpio_nHIB, "cCHIB");
-   if(err) {
-   dev_err(&spi->dev, "failed to register GPIO\n");
-   goto out_nHIB_request_failed;
+ if (gpio_is_valid(fp->pdata->gpio_nHIB)) {
+   err=gpio_request(fp->pdata->gpio_nHIB, "cCHIB");

+   if(err) {
+
+       dev_err(&spi->dev, "failed to register GPIO\n");
+
+       goto out_nHIB_request_failed;

```

```

+     }
+     gpio_direction_output(fp->pdata->gpio_nHIB, 0);
+ }
-     gpio_direction_output(fp->pdata->gpio_nHIB, 0);
/* register pin for IRQ */
-     err=gpio_request(fp->pdata->gpio_irq, "ccIRQ"); +
err=gpio_request(fp->pdata->gpio_irq, "ccIRQ"); if(err){
    dev_err(&spi->dev, "failed to register GPIO\n");
    goto out_irq_request_failed;
@@ -254,7 +266,8 @@ static int cc3100_spi_probe(struct spi_device *spi)
out_irq_interrupt_request_failed: gpio_free(fp->pdata->gpio_irq);
out_irq_request_failed:
-     gpio_free(fp->pdata->gpio_nHIB);
+ if (gpio_is_valid(fp->pdata->gpio_nHIB)) +
    gpio_free(fp->pdata->gpio_nHIB);
out_nHIB_request_failed:
    misc_deregister(&fp->mdev);

out_misc_register_failed:
@@ -273,7 +286,8 @@ static int cc3100_spi_remove(struct spi_device *dev)
    free_irq(fp->irq, fp); gpio_free(fp-
>pdata->gpio_irq); - gpio_free(fp-
>pdata->gpio_nHIB);
+ if (gpio_is_valid(fp->pdata->gpio_nHIB))
+     gpio_free(fp->pdata->gpio_nHIB);

    misc_deregister(&fp->mdev);

```

Note: remove gpio **nHIB**, because it's not controlled by cpu.