F2018 - Resultater

Forsøg 2 ud af Ubegrænset

Skriftlig 10. jun. 2025 17:13 - 10. jun. 2025 17:15

Your quiz has been submitted successfully.

Forsøgsscore **70 / 100 - 70 %**
Samlet karakter (Højeste forsøg) **70 / 100 - 70 %**

Spørgsmål 1 (Obligatorisk) 5 / 5 point

Hvordan uploader man den seneste version af sin kode til et (remote) GIT repository?

- ○ git clone
- ○ git commit
- ✓ ◉ git push
- ○ git add

Spørgsmål 2 (Obligatorisk) 5 / 5 point

Hvordan opretter man en character device node fra en shell

- ✓ ◉ mknod /dev/abc c 10 0
- ○ insmod abc
- ○ echo 1 > /dev/node/export
- ○ open("/dev/abc", O_RDWR)

Spørgsmål 3 (Obligatorisk) 5 / 5 point

Hvilken fil oprettes i /dev med følgende kald: device_create(my_class, NULL, devno, 1, NULL, "led%d", 0);

- ✓ ◉ led0
- ○ led1
- ○ led2
- ○ Ingen, filpointeren er NULL

Spørgsmål 4 (Obligatorisk) 5 / 5 point

Hvad må du ikke gøre i en timer funktion?

- ○ gpio_get_value()
- ✓ ◉ spi_sync()
- ○ int a=0;
- ○ wake_up_interruptible()

Spørgsmål 5 (Obligatorisk) 5 / 5 point

Du har indlæst en gpio platform driver med insmod og det tilhørende overlay med dtoverlay. Hvad er rækkefølgen af funktioner kaldt i din driver?

- ✓ __2__ platform_driver_register()
- ✓ __3__ of_gpio_count()
- ✓ __4__ device_create()
- ✓ __1__ alloc_chardev_region()

Spørgsmål 6 (Obligatorisk) 5 / 5 point

I hvilken character driver fops metode foretrækkes det at requeste en interrupt linje?

- ○ init()
- ○ probe()
- ✓ ◉ open()
- ○ release()

Spørgsmål 7 (Obligatorisk) 5 / 5 point

Det kræver omhu når man skal deklarere attributter. For A/D konverteren MCP3202 har vi lavet en attribute funktion, mcp3202_mode_store(), til opsætning af indgangsmode, men hvilket af følgende kodeudsnit deklarerer attributen korrekt?

✓ ◉
```
DEVICE_ATTR_WO(mcp3202_mode);
static struct attribute *mcp3202_attrs[] = {
   &dev_attr_mcp3202_mode.attr,
   NULL,
 };
ATTRIBUTE_GROUPS(mcp3202);
```

○
```
DEVICE_ATTR_RO(mcp3202_mode);
static struct attribute *mcp3202_attr[] = {
   &dev_attr_mcp3202_mode.attr,
   NULL,
 };
ATTRIBUTE_GROUPS(mcp3202);
```

○
```
DEVICE_ATTR_WO(mcp3202_mode);
 static struct attribute *mcp3202_attr[] = {
   &mcp3202_mode.attr,
   NULL,
 };
 ATTRIBUTE_GROUPS(mcp3202);
```

○
```
DEVICE_ATTR_WO(dev_attr_mcp3202_mode);
 static struct attribute *mcp3202_attrs[] = {
   &dev_attr_mcp3202_mode.attr,
   NULL,
 };
 ATTRIBUTE_GROUPS(mcp3202_attrs);
```

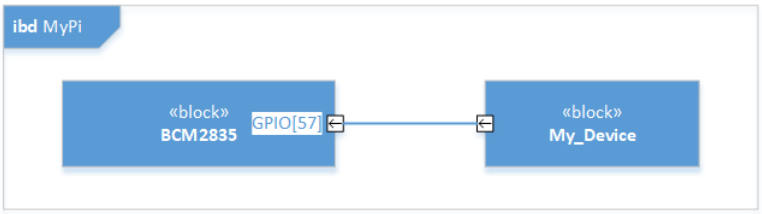
Spørgsmål 8 (Obligatorisk) 5 / 5 point

Hvilket udsagn om forskellene på en device driver og en applikation er korrekt?

- ○ En applikation og en device driver kode har de samme rettigheder til systemadgang.
- ○ Applikationskode afvikles ikke direkte på CPU'en i modsætning til device driver kode
- ○ En device driver kører ikke i virtuel hukommelse i modsætning til en applikation
- ✓ ◉ En device driver er en udvidelse af funktionaliteten i operativ systemet, mens en applikation er et brugerprogram som kan anvende funktionaliteten til f.eks. at overføre data mellem bruger og eksterne enheder

Spørgsmål 9 (Obligatorisk) 5 / 5 point

Hvilken device tree fragment kode til BCM2835 initialiserer interfacet til den viste blok korrekt



○
```
fragment@0 {
        target-path = "/";
          __overlay__ {
```

```
                mygpio: mygpio@0 {
                  compatible = "mygpio";
                  gpios = <gpio 57 0>;
                  };
                };
              };
```

○
```
fragment@0 {
          target-path = "/";
            __overlay__ {
              mygpio: mygpio@0 {
                compatible = "mygpio";
                gpios = <gpio 57 1>;
                };
              };
            };
```

○
```
fragment@0 {
          target-path = "/";
            __overlay__ {
              mygpio: mygpio@0 {
                compatible = "mygpio";
                gpios = <&gpio 57 1>;
                };
              };
            };
```

✓●
```
fragment@0 {
          target-path = "/";
            __overlay__ {
              mygpio: mygpio@0 {
                compatible = "mygpio";
                gpios = <&gpio 57 0>;
                };
              };
            };
```

Spørgsmål 10 (Obligatorisk) 5 / 5 point

Med hvilket register i BCM2835 kan man vælge funktionen af GPIO 49?

- ✓● GPFSEL4
- ○ GPFSEL1
- ○ GPFSEL2
- ○ GPFSEL3

Spørgsmål 11 (Obligatorisk) 5 / 5 point

Hvordan anvendes device_create() til at oprette et device, som med major = 243 og minor = 1?

- ○ `device_create(its_class, NULL, 243, 1,  "its_dev%d", 1);`
- ✓● `device_create(its_class, NULL, MKDEV(243, 1), &its_data, "its_dev%d", its_nbr);`
- ○ `device_create(its_class, NULL, MKDEV(MAJOR(243), i), NULL, "its_dev%d", 1);`
- ○ `device_create(its_class, MAJOR(243), MINOR(1), NULL, "its_dev%d", i);`

Spørgsmål 12 (Obligatorisk) 5 / 5 point

Hvor finder jeg headerfilerne som jeg inkluderer i min device driver?

- ○ Device tree
- ✓● Source tree
- ○ Std-C library
- ○ Root fs

Spørgsmål 13 (Obligatorisk) 5 / 5 point

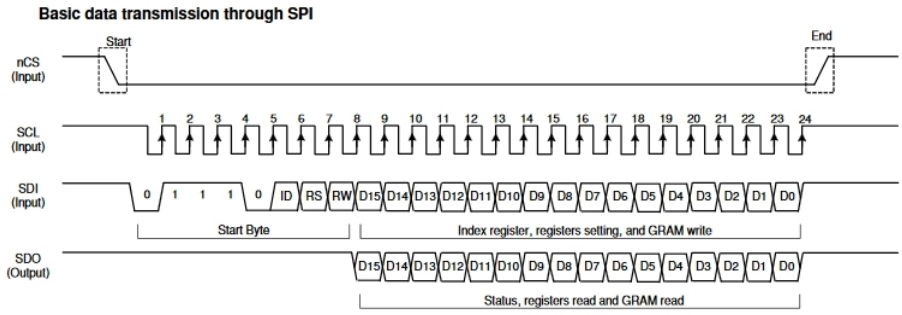
Par enheder og device typer

✓ __4__ USB Device	**1**. Character Device
✓ __1__ Accerometer	**2**. Block Device
✓ __3__ WLAN Adapter	**3**. Network Interface
✓ __2__ Memory Stick	

4. Composite (Kan indeholde flere forskellige device typer)

Spørgsmål 14 (Obligatorisk) 5 / 5 point

Hviken clock mode anvender det viste SPI device?



- ○ CPOL = 0 CPHA = 0
- ○ CPOL = 0 CPHA = 1
- ✓ ◉ CPOL = 1 CPHA = 0
- ○ CPOL = 1 CPHA = 1

Spørgsmål 15 (Obligatorisk) 0 / 9 point

Forklar i klartekst hvornår og hvorfor vi anvender funtionen copy_from_user(), samt hvilket arbejde funktionen udfører. Besvarelsen kan laves på dansk eller engelsk.

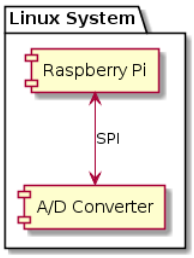
Dette spørgsmål er ikke blevet bedømt.

Det rigtige svar vises ikke for spørgsmål, der kræver skriftlige svar.

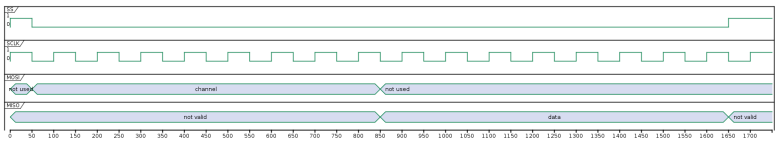
Spørgsmål 16 (Obligatorisk) 0 / 21 point

Du skal I denne opgave skrive read funktionen til en character driver som anvender et SPI interface.

Systemets Linux device (Raspberry Pi) er forbundet til en 8-kanals A/D konverter ved hjælp af en SPI forbindelse.



Protokollen imellem de to enheder er vist i figuren



En dataoverførsel starter ved, at Raspberry Pi sender en 8-bit værdi med kanal nummeret, herefter returnerer A/D konverteren den målte 8-bit værdi for den pågældende kanal.

De enkelte kanaler skal kunne tilgås vha. 8 noder i /dev mappen med fælles majornummer og individuelle minor numre.

Det er din opgave at implementere driverens read funktion. Opgaven opdeles i to dele:

a) Lav en hjælpe funktion til læsning fra et SPI interface, *spi_read_byte*, som står for opbygning- og udveksling af SPI beskeder

b) Lav en character driver read funktion, *my_spi_cdrv_read*, som læser fra SPI interfacet og overfører data user-space.

Du skal tage udgangspunkt i kodeudsnittet nedenfor. Skriv din besvarelse ind i tekstfeltet.

```
/********** Device Driver Code **********/

/* Global variables */

struct spi_device myspi_dev; /* Already initialized with a pointer to our spi device */

/* SPI Read byte helper function */

int spi_read_byte(struct spi_device* sdev, u8 channel, u8 *data){


/* Implement the code needed here */

/* Error handling not required */  }

/* Character driver read function */

ssize_t myspi_read(struct file *filep, char __user *ubuf,
                          size_t count, loff_t *f_pos)
{

/* Implement the code needed here */

/* Error handling not required */
}
/*************** Code End ***************/
```

Dette spørgsmål er ikke blevet bedømt.

Det rigtige svar vises ikke for spørgsmål, der kræver skriftlige svar.

Udført

F2018R - Resultater

Forsøg 3 ud af Ubegrænset

Skriftlig 10. jun. 2025 22:18 - 10. jun. 2025 22:20

Your quiz has been submitted successfully.

Forsøgsscore **70 / 100 - 70 %**
Samlet karakter (Højeste forsøg) **70 / 100 - 70 %**

Spørgsmål 1 (Obligatorisk) 5 / 5 point

Et bash script er en tekstfil med bash-kommandoer, men hvordan gør man tekstfilen eksekverbar?

- ○ mv my_script.sh /bin
- ✓ ◉ chmod a+x my_script.sh
- ○ echo my_script.sh > /proc/execute
- ○ cat my_script.sh

Spørgsmål 2 (Obligatorisk) 5 / 5 point

Hvis man logger ind på en embedded platform, som anvender Device Tree (f.eks Raspberry Pi), i hvilken folder kan man der finde information om det indlæste device tree?

- ○ /proc/devicetree/
- ○ /dev
- ○ /etc/firmware
- ✓ ◉ /sys/firmware/devicetree/

Spørgsmål 3 (Obligatorisk) 5 / 5 point

I hvilken folder oprettes attribute filer når en driver med device attributes loades?

- ✓ ◉ /sys
- ○ /boot
- ○ /dev
- ○ /lib/modules

Spørgsmål 4 (Obligatorisk) 5 / 5 point

Når man kalder device_create() i en driver, så oprettes en device node i filsystemet. Hvad hedder den service/deamon som opretter filerne i user space?

- ○ systemctl
- ○ inode
- ✓ ◉ udev
- ○ System V

Spørgsmål 5 (Obligatorisk) 5 / 5 point

I en driver som benytter interrupt, hvilke hjælpefunktioner skal kaldes i de forskellige driver funktioner (init, open, read, isr)?

✓ __1__ irqreturn_t mydrv_isr(int irq, ...)	**1**. wake_up_interruptible()
✓ __4__ int mydrv_init(void)	**2**. wait_event_interruptible()

✔ __2__ ssize_t mydrv_read(struct file *filep,...) **3**. request_irq()

✔ __3__ int mydrv_open(struct inode *inode, ..) **4**. register_chrdev_region()

Spørgsmål 6 (Obligatorisk) 5 / 5 point

I en device driver, på hvilken parameter er funktionen mdelay() bedre end funktionen msleep()?

- ◯ Den bruger færre CPU resourcer
- ◯ Den lader scheduleren arbejde videre på andre processer som kræver CPU tid
- ◯ Den er strømbesparende
- ✔ ◉ Den er mere præcis

Spørgsmål 7 (Obligatorisk) 5 / 5 point

Hvordan tilgår en user-space applikation device driverens funktioner?

- ◯ Applikation linkes dynamisk til funktionerne når den loades
- ◯ Applikation linkes statisk til funktionerne når den bygges
- ✔ ◉ Gennem systemkald i std. C biblioteket
- ◯ Vha shared memory

Spørgsmål 8 (Obligatorisk) 5 / 5 point

En timerfunktions udløbstid er sat til:

```
my_timer.expires = jiffies + 10;
```

Men hvornår udløber den?

- ✔ ◉ Kan ikke siges eksakt, det er afhængigt af konfugurationen af Linux
- ◯ 10 s
- ◯ 10 ms
- ◯ 10 us

Spørgsmål 9 (Obligatorisk) 5 / 5 point

Hvad er ikke tilladt i en linux device driver?

- ◯ `mem = kmalloc(sizof(var));`
- ✔ ◉ `temp = get_temp() * 0.2;`
- ◯ `udelay(10);`
- ◯ `printk( KERN_ERR "Panic!!!\n");`

Spørgsmål 10 (Obligatorisk) 5 / 5 point

Hvis jeg anvender *allocate_chrdev_region()* til dynamisk allokering af major numre, hvor kan jeg så se hvilket majornummer som driveren er tildelt? (Hvis jeg ikke anvender sysfs klasser og devices)

- ◯ /lib/modules
- ◯ /etc
- ◯ /dev
- ✔ ◉ /proc/devices

Spørgsmål 11 (Obligatorisk) 5 / 5 point

Struct'en platform_driver, som findes i platform_device.h og som er vist I udsnit herunder, indeholder bl.a. elementet remove. Hvilken type har dette elementet?

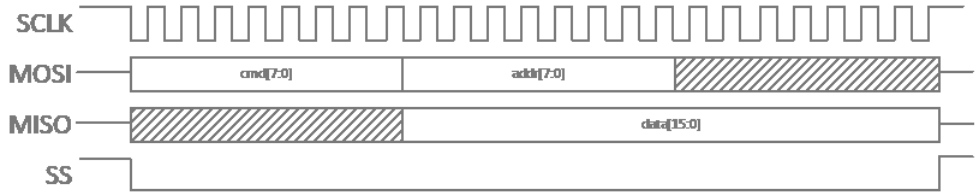
```
struct platform_driver {

 int (*probe)(struct platform_device *);
```

```
int (*remove)(struct platform_device *);
```

- ✓ function pointer
- signed long
- int
- int pointer

Spørgsmål 12 (Obligatorisk) 5 / 5 point

Hvilken af nedenstående kode udsnit implementerer SPI kommunikatioen korrekt, når det antages at bits_per_word = 8?



Kode fælles for alle udsnit:

```
/* Common */
cmd  = 0x0;     // 8-bit
addr = 0x0;     // 8-bit
data  = 0x00;   // 16-bit
data_msb = 0x0; // 8-bit
data_lsb = 0x0; // 8-bit
```

- t[0].tx_buf = (cmd <<8) | addr;
  t[0].rx_buf = data;
  t[0].len = 4;
  spi_message_add_tail(&t[0], &m);
  spi_sync(m.spi, m);
- t[0].tx_buf = &cmd;
  t[0].rx_buf = &data_msb;
  t[0].len = 2;
  spi_message_add_tail(&t[0], &m);
  t[1].tx_buf = &addr;
  t[1].rx_buf = &data_lsb;
  t[1].len = 2;
  spi_message_add_tail(&t[1], &m);
  spi_sync(m.spi, m);
  data = (data_msb <<8) | data_lsb;
- t[0].tx_buf = &cmd;
  t[0].rx_buf = NULL;
  t[0].len = 2;
  spi_message_add_tail(&t[0], &m);
  t[1].tx_buf = &addr;
  t[1].rx_buf = data;
  t[1].len = 4;
  spi_message_add_tail(&t[1], &m);
  spi_sync(m.spi, m);
- ✓ t[0].tx_buf = &cmd;
  t[0].rx_buf = NULL;
  t[0].len = 1;
  spi_message_add_tail(&t[0], &m);
  t[1].tx_buf = &addr;
  t[1].rx_buf = &data_msb;
  t[1].len = 1;
  spi_message_add_tail(&t[1], &m);
  t[2].tx_buf = NULL;

```
t[2].rx_buf = &data_lsb;

t[2].len = 1;

spi_message_add_tail(&t[2], &m);

spi_sync(m.spi, m);

data = (data_msb <<8) | data_lsb;
```

Spørgsmål 13 (Obligatorisk) 5 / 5 point

Angiv den korrekte rækkefølge som funktionerne skal kaldes i

- ✓ __3__ my_class->dev_groups = my_groups;
- ✓ __2__ class_create()
- ✓ __1__ alloc_chrdev_region()
- ✓ __4__ device_create()

Spørgsmål 14 (Obligatorisk) 5 / 5 point

Match emneområder og begreber

✓ __1__ Device tree	**1**. overlays
✓ __4__ sysfs	**2**. Linux include files
✓ __2__ Source tree	**3**. home folder
✓ __3__ rootfs	**4**. classes and devices

Spørgsmål 15 (Obligatorisk) 0 / 9 point

I forbindelse med et interrupt, skal vi have udført nogle opgaver i top- og bottom half. Når der kommer et interrupt på en gpio indgang, skal der ske følgende:

1. Check om gpio pin reelt er aktiveret
2. Lav tidsstempel
3. Læs data via SPI
4. Lav checksum
5. Overfør til user-space

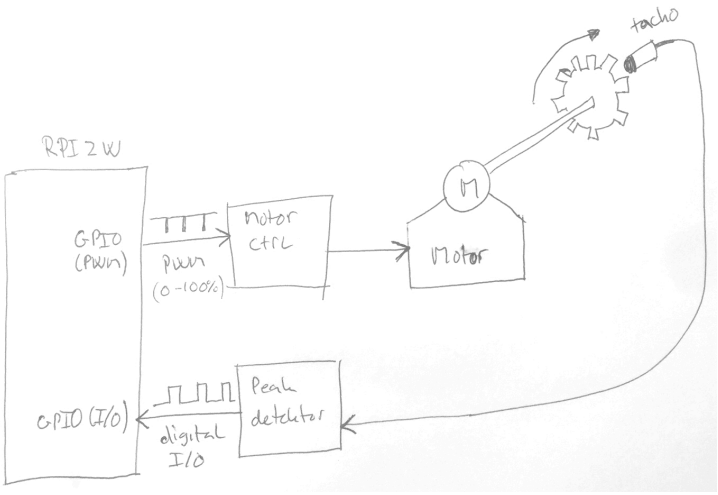
Forklar hvilke opgaver du vil udføre i top- eller bottom half og hvorfor?

Dette spørgsmål er ikke blevet bedømt.

Det rigtige svar vises ikke for spørgsmål, der kræver skriftlige svar.

Spørgsmål 16 (Obligatorisk) 0 / 21 point

Raspberry Pi (Rpi) ønskes anvendt til en motorstyring. På Rpi er en GPIO port er konfigureret som PWM output og en anden som et almindeligt GPIO input. Ved at ændre på PWM outputtets duty-cycle, justeres motorens rotationshastighed. På motorakslen er monteret en tachometerskive. En tacho sensor genererer impulser der omsættes til et digitalt signal på Rpi's GPIO indgang. Antallet af pulser/sek er et udtryk for omdrejningshastigheden.
Motorstyringssoftwaren måler rotationshastigheden og justerer duty-cycle på PWM op/ned, således at antallet af tacho pulser/sek tilnærmer sig den ønskede rotationshastighed.



På Raspberry Pi er installeret en ikke realtidsoptimeret Linux, som den vi har arbejdet med i HAL. Grundet udfordringer med tids-præcisionen i user-space, forsøger vi derfor at lave motorreguleringen i driveren.

Der er krav til motorstyringen om at den skal regulere hastigheden hver 200 ms, mens at hastigheden, målt på tacho udgangen ligger i området 0-100 impulser/sekundet. PWM udgangen har et arbejdsområde på 0-100% duty-cycle

Til løsning af problemet anvendes følgende:

- En interrupt service rutine, *tacho_isr()*, som opdaterer tælleren *unsigned double tacho_counter,* hver gang der detekteres en tacho impuls.
- En attribute funktion, *motor_speed_store()*, som bruges til at angive den ønskede motor hastighed og som sætter den globale variabel *motor_speed*.
- En timerfunktion, *motor_timer_function()*, som regulerer hastigheden, på basis af *tacho_counter og motor_speed*, ved at justere motorens pwm 1 duty-cycle procent op- eller ned (Således at motoren over tid tilnærmer sig den ønskede hastighed)

Det er din opgave at implementere koden til *motor_speed_store()* og *motor_timer_function()*.

Herunder ses funktionerne som skal kodes, samt hjælpefunktioner og globale variable:

```
//File: motorcontrol_driver.c

unsigned double tacho_counter; // Tacho impulse counter value

unsigned int motor_speed;      // Desired motor speed 0-100

void set_motor_pwm(unsigned int duty_cycle); // Helper function to set PWM 0-100

static irqreturn_t tacho_isr(int irq, void *dev_id)
{

 ++tacho_counter;

 return IRQ_HANDLED;

}



static void motor_timer_function (unsigned long arg)
 {

    /* Your Code Here */
 }


static ssize_t motor_speed_store(struct device *dev, struct device_attribute *attr,
                                 const char *buf, size_t size)
{

    /* Your Code Here */

}
```

Dette spørgsmål er ikke blevet bedømt.

Det rigtige svar vises ikke for spørgsmål, der kræver skriftlige svar.

Udført

E2018 - Resultater

Forsøg 2 ud af Ubegrænset

Skriftlig 10. jun. 2025 22:11 - 10. jun. 2025 22:14

Your quiz has been submitted successfully.

Forsøgsscore **70 / 100 - 70 %**
Samlet karakter (Højeste forsøg) **70 / 100 - 70 %**

Spørgsmål 1 (Obligatorisk) 5 / 5 point

Hvilken bash kommando kan man benytte for at se den aktuelle folder?

- ○ less
- ✓ ◉ pwd
- ○ mv
- ○ cp

Spørgsmål 2 (Obligatorisk) 5 / 5 point

På vores Raspberry Pi Linux distribution, i hvilken folder kan man se om et device tree overlay er blevet indlæst og registreret korrekt?

- ✓ ◉ /sys/firmware/devicetree
- ○ /dev
- ○ /etc/modules.d/
- ○ /proc/devicetree

Spørgsmål 3 (Obligatorisk) 5 / 5 point

På vores Raspberry Pi Linux distribution, hvor skal man kopiere et kernemodul hen, hvis det skal kunne indlæses under boot?

- ○ /tmp
- ○ /proc/devices
- ✓ ◉ /lib/modules
- ○ /dev/

Spørgsmål 4 (Obligatorisk) 5 / 5 point

Match device typer og devices

✓ __3__ Character Device	**1**. Floppy Disk
✓ __2__ Network Interface	**2**. Wimax
✓ __1__ Block Device	**3**. Serial Port
✓ __4__ Not really a device	**4**. CPU

Spørgsmål 5 (Obligatorisk) 5 / 5 point

I hvilken folder kan man finde en device drivers attributes?

- ○ /proc/devices/attribute
- ○ /dev/attribute
- ○ /etc/drivers

✓◉ /sys/class/myclass/mydevicex/

Spørgsmål 6 (Obligatorisk) 5 / 5 point

Med en Platform driver behøver man ikke oprette device nodes i shell'en med mknod.

Spørgsmålet er, hvilken funktion i platform driveren laver bindingen mellem device node og major/minor nummer?

- ◯ alloc_chrdev_region()
- ✓◉ device_create()
- ◯ cdev_add()
- ◯ class_create()

Spørgsmål 7 (Obligatorisk) 5 / 5 point

Med hvilket register i Raspberry Pi Zero W's processor, BCM2835, kan man enable/disable pull-up modstande på gpio porte?

- ◯ GPFSEL
- ◯ GPAREN
- ✓◉ GPPUD
- ◯ GPHEN

Spørgsmål 8 (Obligatorisk) 5 / 5 point

Angiv hvilke gpio funktioner som skal kaldes i hvilke device driver funktioner

✓ __4__ my_drv_remove()	**1**. gpio_request()
✓ __1__ my_drv_probe()	**2**. gpio_set_value()
✓ __2__ my_drv_attr_store()	**3**. gpio_get_value()
✓ __3__ my_drv_attr_show()	**4**. gpio_free()

Spørgsmål 9 (Obligatorisk) 5 / 5 point

Numlock led'en har en tilhørende attribute, max_brightness, som kan findes her:

```
stud@stud-virtual-machine:~$ ls -l
/sys/class/leds/input1\:\:numlock/max_brightness
-r--r--r-- 1 root root 4096 Dec 10 16:25
/sys/class/leds/input1::numlock/max_brightness
```

Ud fra ovenstående, med hvilken makro må attributen være defineret?

- ✓◉ DEVICE_ATTR_RO
- ◯ DEVICE_ATTR_WO
- ◯ DEVICE_ATTR_WR
- ◯ DEVICE_ATTR_RE

Spørgsmål 10 (Obligatorisk) 5 / 5 point

Hvad er ok at gøre i en timer funktion?

- ◯ Kalde spi_sync()
- ◯ Kalde copy_to_user()
- ◯ Kalde wait_event_interruptible()
- ✓◉ Kalde add_timer()

Spørgsmål 11 (Obligatorisk) 5 / 5 point

Hvilken device tree fragment kode initialiserer interfacet til den viste chip korrekt?

6.4.3 AC Characteristics

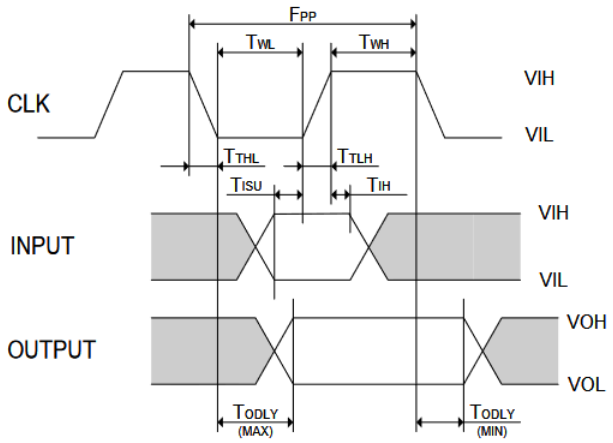


Fig 8: AC Timing Diagram

Table 8: AC Characteristics

Item	Symbol	Min.	Max.	Unit	Note
Clock Frequency (In any Sates)	F_{sty}	0	25	MHz	CL<100pF (7Cards)
Clock Frequency (Data transfer Mode)	F_{PP}	0.1	25	MHz	CL<100pF (7Cards)
Clock Frequency (Card identification Mode)	F_{OD}	100	400	kHz	CL<250pF (21Cards)
Clock Low Time	T_{WL}	10	-	ns	CL<100pF (7Cards)
Clock High Time	T_{WH}	10	-	ns	
Clock Rise Time	T_{TLH}	-	10	ns	
Clock Fall Time	T_{THL}	-	10	ns	
Clock Low Time	T_{WL}	50	-	ns	CL < 250pF (21Cards)
Clock High Time	T_{WH}	50	-	ns	
Clock Rise Time	T_{TLH}	-	50	ns	
Clock Fall Time	T_{THL}	-	50	ns	
Input Setup Time	T_{ISU}	5	-	ns	CL < 25pF (1Cards)
Input Hold Time	T_{IH}	5	-	ns	
Output Delay Time	T_{ODLY}	0	14	ns	

✔◉
```
fragment@0 {
  target = <&spi0>;
  __overlay__ {
    myspi: myspi@0 {
    compatible = "ase, myspi";
    reg = <0>;
    /* spi-cpha; */
    /* spi-cpol; */
    };
  };
};
```

○
```
fragment@1 {
  target = <&spi0>;
  __overlay__ {
    myspi: myspi@0 {
    compatible = "ase, myspi";
    reg = <0>;
    /* spi-cpha; */
    spi-cpol;
    };
  };
};
```

○
```
fragment@2 {
  target = <&spi0>;
  __overlay__ {
    myspi: myspi@0 {
    compatible = "ase, myspi";
```

```
        reg = <0>;
        spi-cpha;
        /* spi-cpol; */
        };
      };
    };
```

○
```
fragment@3 {
  target = <&spi0>;
  __overlay__ {
    myspi: myspi@0 {
    compatible = "ase, myspi";
    reg = <0>;
    spi-cpha;
    spi-cpol;
    };
  };
};
```

Spørgsmål 12 (Obligatorisk) 0 / 9 point

Der oprettes fire noder til en driver: /dev/temp0, /dev/temp1, /dev/temp2, /dev/temp3. Hver af disse noder er koblet til et specifikt, fysisk device, f. eks. en temperaturføler på en given I2C-adresse.

Forklar i tekst hvordan læsningen af en specifik node i user space er koblet til læsningen af det tilhørende fysiske device i driveren.

Besvarelsen må laves på engelsk eller dansk

Dette spørgsmål er ikke blevet bedømt.

Det rigtige svar vises ikke for spørgsmål, der kræver skriftlige svar.

Spørgsmål 13 (Obligatorisk) 5 / 5 point

En platform driver med gpios skal implementeres. Angiv den korrekte rækkefølge af funktionerne i probe():

✓ __2__ gpio_request()
✓ __1__ of_gpio_count()
✓ __4__ device_create()
✓ __3__ gpio_direction_input()

Spørgsmål 14 (Obligatorisk) 5 / 5 point

Hvad er primære opgave for processorens Memory Management Unit?

○ At styre Cache coherency
✓ ◉ At omsætte virtuelle adresser til fysiske
○ At styre ekstene memory devices
○ At styre memory endian

Spørgsmål 15 (Obligatorisk) 5 / 5 point

Linux source tree indeholder ud over kildekoden til Linux også dokumentation.

I en device tree fil kan jeg angive en gpio port's retning mm. ved at sætte værdien af "flag". Værdien og den tilhørende betydning kan være forskellig fra fabrikat til fabrikat.

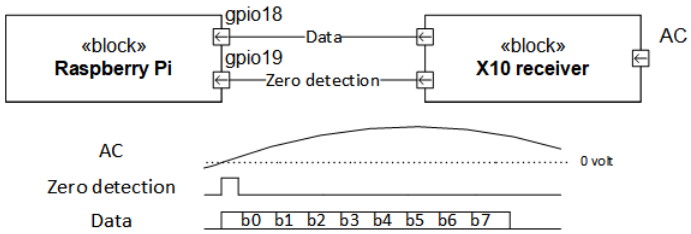
I hvilken folder, under Linux source tree, finder jeg information om hvad et flags værdi betyder (eks. '1' = output), for en given platform? (source tree på Golden Image: ~/sources/rpi-4.14/)

○ ./Documentation/devicetree/readme
○ ./Documentation/devicetree/gpio

- [x] ./Documentation/devicetree/bindings
- [ ] ./Documentation/devicetree/flags

Spørgsmål 16 (Obligatorisk) 0 / 21 point

Du skal i denne øvelse lave dele af en device driver til en X10 modtager. X10 er en data protokol til overførsel af data på et AC-elnet. Karakteristisk for X10 er at data overføres når AC signalet krydser 0 volt.



X10 modtageren er forbundet til en Raspberry Pi ved hjælp af to GPIO porte. Zero Detection signalet indikerer at AC signalet krydser 0 volt og at der nu kommer data. Data signalet indeholder de 8 serielle data bits, som sendes med en hastighed på 115200 baud.

Driveren skal afvente at AC signalet krydser 0 volt, indlæse de 8-serielle bits, bit-for-bit, sammensætte dem til en byte og overføre denne byte til user space.

Det er din opgave at implementere driverens interruptservicerutine og (fops) read funktion. Du skal selv vurdere hvilke opgaver du vil lægge i hhv. interruptservicerutine og read funktion, samt hvordan at du indlæser den serielle datastrøm.

Det antages at gpio og interrupt allerede er requested og at interruptet trigger på rising edge. Prototyperne for funktionerne er:

```
static irqreturn_t zero_detection_isr(int irq, void *dev_id) {
/* Din kode her */
}

ssize_t x10_read(struct file *filep, char __user *buf,
            size_t count, loff_t *f_pos) {
/* Din kode her */
}
```

Du kan med fordel skrive koden i en egnet editor og kopiere teksten ind i BB formen bagefter. Gem en kopi af din tekst lokalt på din computer som backup.

Dette spørgsmål er ikke blevet bedømt.

Det rigtige svar vises ikke for spørgsmål, der kræver skriftlige svar.

Udført

F2021 - Resultater

Forsøg 2 ud af Ubegrænset

Skriftlig 10. jun. 2025 22:05 - 10. jun. 2025 22:09

Your quiz has been submitted successfully.

Forsøgsscore **70 / 100 - 70 %**
Samlet karakter (Sidste forsøg) **70 / 100 - 70 %**

Spørgsmål 1 (Obligatorisk) 5 / 5 point

Med hvilken bash kommando kan jeg se alle kørende processer?

- ls /dev
- ✓ ps -A
- cat /proc/devices
- dmesg

Spørgsmål 2 (Obligatorisk) 5 / 5 point

Med hvilken bash kommando kan man kopiere fra en remote folder til en lokal folder, dvs. i Ubuntu kopiere data fra Raspberry Pi tilbage en folder i din Ubuntu?

- ssh root@10.9.8.2
- ✓ scp root@10.9.8.2:file_to_copy .
- scp file_to_copy root@10.9.8.2:
- cp /media/raspberrypi/file_to_copy /home/stud/

Spørgsmål 3 (Obligatorisk) 5 / 5 point

På et Linux system, i hvilken fil kan jeg se hvilke interrupts som er i brug?

- ✓ /proc/interrupts
- /sys/class/interrupts
- /dev/
- /sys/firmware/devicetree/interrupts

Spørgsmål 4 (Obligatorisk) 5 / 5 point

I hvilken folder skal jeg lægge en device tree blob overlay fil (.dtbo) for at den kan indlæses automatisk under boot?

- /lib/modules
- /sys/firmware/devicetree/
- ✓ /boot/overlays/
- /etc/devicetreeoverlays/

Spørgsmål 5 (Obligatorisk) 5 / 5 point

Hvornår kaldes probe() funktionen i en device driver?

- Når driveren indlæses med insmod
- Når Linux ændrer systemtilstand, f.eks. til suspend
- ✓ Når Linuxkernen har et device som er kompatibel med driveren

- ○ Når Linuxsystemet booter

Spørgsmål 6 (Obligatorisk) 5 / 5 point

Hvordan skal CPHA og CPOL konfigureres for viste SPI Device (CE = Slave Select, SCL = Serial Clock, SDI = MOSI, SDO = MISO):

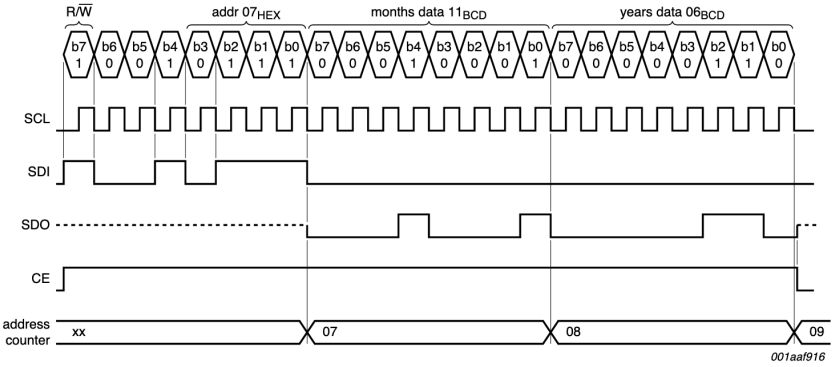


Fig 26. Serial bus read example

- ✓● CPOL = 0
  CPHA = 0
- ○ CPOL = 1
  CPHA = 0
- ○ CPOL = 0
  CPHA = 1
- ○ CPOL = 1
  CPHA = 1

Spørgsmål 7 (Obligatorisk) 5 / 5 point

Med hvilket register på BCM2835 kan man enable PWM på kanal 1? (Se evt. https://www.raspberrypi.org/app/uploads/2012/02/BCM2835-ARM-Peripherals.pdf side 141 ff.)

- ✓● CTL
- ○ DMAC
- ○ RNG1
- ○ DAT1

Spørgsmål 8 (Obligatorisk) 5 / 5 point

Hvilket kodeeksempel anvender den følgende attribute deklaration korrekt?

```
static struct device_attribute dev_attr_btn_state= {
    .attr={
        .name = "btn_state",
        .mode = 0644},
        .show = btn_state_show,
        .store = btn_state_store,
    }, };
static struct attribute *btn_attrs[]={
    &dev_attr_btn_state.attr,
    NULL,};
ATTRIBUTE_GROUPS(btn);
```

- ○
```
static int btn_init(void){
    alloc_chrdev_region(&devno,0, 255,"btn");
    btn_class->dev_groups=btn_groups;
    btn_class = class_create(THIS_MODULE,"btn");
    device_create(btn_class, NULL, devno, NULL,"btn");
    return 0;
}
```
- ○
```
static int btn_init(void){
    alloc_chrdev_region(&devno,0, 255,"btn");
    btn_class = class_create(THIS_MODULE,"btn");
    device_create(btn_class, NULL, devno, NULL,"btn");
```

```
    btn_class->dev_groups=btn_groups;
    return 0;
}
```

✔ ◉
```
static int btn_init(void){
    alloc_chrdev_region(&devno,0, 255,"btn");
    btn_class = class_create(THIS_MODULE,"btn");
    btn_class->dev_groups=btn_groups;
    device_create(btn_class, NULL, devno, NULL,"btn");
    return 0;
}
```

○
```
static int btn_init(void){
    alloc_chrdev_region(&devno,0, 255,"btn");
    btn_class = class_create(THIS_MODULE,"btn");
    btn_class->dev_groups=btn_state;
    device_create(btn_class, NULL, devno, NULL,"btn");
    return 0;
}
```

Spørgsmål 9 (Obligatorisk) 5 / 5 point

Hvad er formålet med funktionen cdev_init()?

✔ ◉ At initialisere struct cdev med fil operationerne

○ At oprette en struct cdev

○ At nulstille struct cdev

○ At tilføje den givne struct cdev til kernen

Spørgsmål 10 (Obligatorisk) 5 / 5 point

Med hvilket element i "struct spi_driver" angiver man hvilket device som driveren er kompatibel med, når man anvender et device tree?

○ .driver.name

✔ ◉ .driver.of_match_table

○ .name

○ .compatible

Spørgsmål 11 (Obligatorisk) 5 / 5 point

Følgende udsnit fra en c-applikation og tilhørende driver afvikles:

```
// Snippet from user space program
int main(int narg, char *argp[]) {
int fp = open("/dev/gpio161", O_RDWR);
char buf[] = {"apple"};
int ret = write(fp, buf, strlen(buf));
...

// Snippet from kernel driver fops write
ssize_t cdrv_write(struct file *filep, const char __user *ubuf,
                   size_t count, loff_t *f_pos)
{
  char kbuf[3];

  int len = ???;
  int ret = copy_from_user(kbuf, ubuf, len);
  ...
```

Hvad skal værdien af len sættes til i driveren?

✔ ◉ 3

○ 4

○ 5

○ 6

Spørgsmål 12 (Obligatorisk) 5 / 5 point

Hvad må du ikke gøre i en timer funktion?

- ○ gpio_get_value()
- ✓ ◉ spi_sync()
- ○ int a=0;
- ○ wake_up_interruptible()

Spørgsmål 13 (Obligatorisk) 5 / 5 point

I forbindelse med request_irq kan man sætte flag for at angive egenskaber, f.eks. rising-/falling edge, men også om der skal være et interrupt per CPU. Hvad er din primære kilde til at finde information om hvad dette flag hedder?

- ✓ ◉ <source tree>/include/linux/interrupt.h
- ○ /proc/interrupts
- ○ Google
- ○ <device tree>/interrupts

Spørgsmål 14 (Obligatorisk) 5 / 5 point

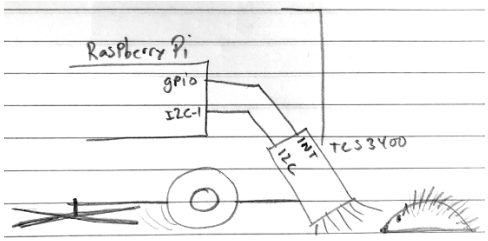
Generelt ønsker man ikke lave forward deklarationer i kode (tomme deklarationer af funktioner og variabler) af fare for at lave fejl.

Hvordan kan jeg strukturere min driver således at jeg undgår dette (I hvilken rækkefølge skal jeg have driveres funktioners og structs)?

- ✓ __2__ `static struct platform_driver my_platform__drv_spi_driver = { ...content... }`
- ✓ __1__ `static int my_platform_drv_probe(struct spi_device *sdev) { ...code... }`
- ✓ __3__ `static int __init my_platform__drv_init(void) { ...code... }`
- ✓ __4__ `module_init(... content ...)`

Spørgsmål 15 (Obligatorisk) 0 / 6 point

3. semesterprojektet SaveSonic har en ambition om at lave en robotplæneklipper som ikke dræber pindsvin. Projektgruppen har valgt farve sensor chippen TCS 3400 til måling af farver og infrarød (varme).



TCS3400 kan via I2C returnere intensiteten af farverne rød (R), grøn(G), blå (B), af baggrundsbelysningen (AL) og af infrarødt (IR) indhold.

Til kalibrering af systemet ønskes der lavet en RGB måling af en referencefarve. Ved kalibrering foretages 4 læsninger af hhv. R, G, B og AL som sammensættes til en korrektionsfaktor.

Hvor vil du foretage bestemmelsen af korrektionsfaktoren? I user-space applikationen eller i din kernel-space driver?

Forklar i tekst hvor og hvorfor.

Dette spørgsmål kan besvares på dansk eller engelsk.

Dette spørgsmål er ikke blevet bedømt.

Det rigtige svar vises ikke for spørgsmål, der kræver skriftlige svar.

Spørgsmål 16 (Obligatorisk) 0 / 24 point

Denne opgave har 3 underspørgsmål. Fed skrift markerer overskrift til det pågældende spørgsmål, efterfulgt af tilhørende information, mens selve spørgsmålet/opgaven er skrevet med kursiv.

Delspørgsmål 1: Valg af I2C funktion til kommunikation med TCS3400

Farvesensoren TCS3400 kan via I2C returnere intensiteten af farverne rød (R), grøn(G), blå (B), af baggrundsbelysningen (AL) og af infrarødt (IR) indhold.

Desuden kan TCS3400 via sit *Clear Channel Interrupt Threshold Register* konfigureres til at give et interrupt, når IR værdien overstiger en given grænse. Registrene er vist i et udsnit af databladet herunder:

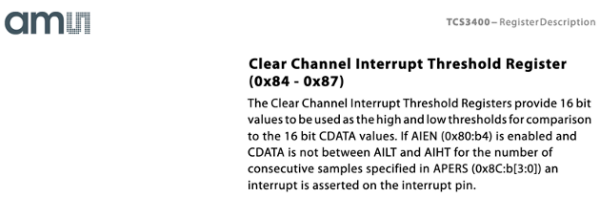
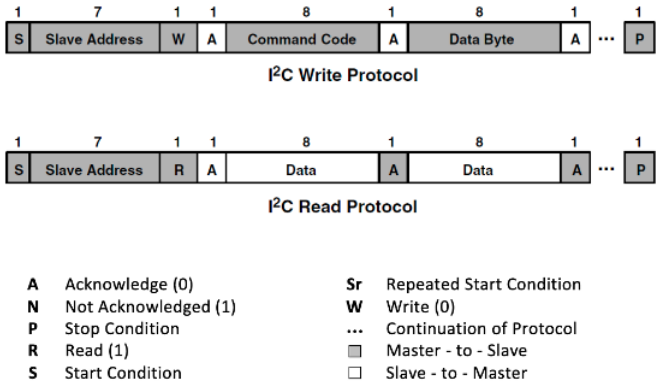


Figure 23:
Clear Channel Interrupt Threshold Registers

Register	Address	Bits	Description
AILTL	0x84	7:0	Clear Channel low threshold lower byte
AILTH	0x85	7:0	Clear Channel low threshold upper byte
AIHTL	0x86	7:0	Clear Channel high threshold lower byte
AIHTH	0x87	7:0	Clear Channel high threshold upper byte

TCS3400 anvender I2C til dataoverførsel og har slaveadressen 0x39. Når der skal kommunikeres med en specifik intern adresse på TCS3400 skal denne sendes med som en "Command Code" som vist i figuren herunder.



Dvs. man sender slave adresse, intern adresse og data for at skrive til et register på TCS3400.

Linux har et antal i2c_smbus read / write funktioner i i2c.h, kort dokumenteret i i2c-core-smbus.c og yderligere beskrevet i kernel dokumentationen: smbus-protocol

Hvilken i2c_smbus funktion vil du vælge til at skrive threshold værdien med og hvorfor?

Delspørgsmål 2: Kode til attribute funktion

Der er brug for en attribute funktion som kan sætte high threshold. Det følgende kodeudsnit viser attribute funktionens signatur og den globale variabel tcs3400_i2c_device, som er initialiseret med I2C slave adressen i driverens probe funktion.

```
static struct i2c_client *tcs3400_i2c_device;

static ssize_t tcs3400_ir_threshold_store(struct device *dev,
                                  struct device_attribute *attr,
                                  const char *buf, size_t size)
 {

   // String to value

   // i2c_smbus_XX function to transfer value to TCS3400

 }
```

Implementer tcs3400_ir_threshold's manglende kode til at sætte TCS3400's high-threshold til en 16-bit værdi.

Delspørgsmål 3: Deklarationer til attribute funktion

Attributten skal være en del af en attribute group, så der automatisk oprettes attributes, når der oprettes devices. Kodeudsnittet herunder viser strukturen af koden, hvor du skal angive deklarationen af attributten.

```
static ssize_t tcs3400_ir_threshold_store(…) {}


  // YOUR ATTRIBUTE DECLERATIONS HERE


static int __init tcs3400_init(void)
{
  …
  tcs3400_class = class_create(THIS_MODULE, "tcs3400");
  tcs3400_class->dev_groups = tcs3400_groups;
  …
```

Angiv den/de nødvendige deklaration(er) og initialisering af attribute struct(s) for at tsc3400_groups variablen kan anvendes til at angive default attributes under initialisering af device driveren.

Dette spørgsmål er ikke blevet bedømt.

Det rigtige svar vises ikke for spørgsmål, der kræver skriftlige svar.

Udført

F2022 - Resultater

Forsøg 2 ud af Ubegrænset

Skriftlig 10. jun. 2025 22:00 - 10. jun. 2025 22:03

Your quiz has been submitted successfully.

Forsøgsscore **75 / 100 - 75 %**
Samlet karakter (Sidste forsøg) **75 / 100 - 75 %**

Spørgsmål 1 5 / 5 point

I kernel space ønsker jeg dynamisk at allokere memory som ikke nødvendigvis er fysisk sammenhængende, hvordan gør jeg det?

- ✔ vmalloc()
- kmalloc()
- new()
- void *my_mem;

Spørgsmål 2 5 / 5 point

Fra en user space applikation skrives der en tekststreng til en node, hvorved my_write kaldes i kernel space. Hvilken værdi skal "len" have?

```
/* User space */
int main() {
  char buf[]="1010";
  int fp = open("/dev/mynode", O_WRONLY);
  write(fp, buf, strlen(buf)+1);
  close(fp);
}
/*--------------------------------------*/
/* Kernel space */
ssize_t my_write(struct file *filep, const char __user *ubuf,size_t count, loff_t
*f_pos)
{
  int len, val, n=0;
  char kbuf[8];

  len = ?? /* WHAT SIZE SHOULD len HAVE? */

  copy_from_user(kbuf, ubuf, len));

  while (!kbuf[n]==0) {

    val = kbuf[n]-0x30; // Ascii to int

    gpio_set_value(13, val);

    msleep(100);

    ++n; }

return len;
}
```

- 1

- ✔ ◉ 5
- ○ 4
- ○ 8

Spørgsmål 3 5 / 5 point

Når jeg første gang skal bygge og anvende et kernel module, i hvilken rækkefølge skal jeg så benytte følgende kommandoer?

- ✔ __3__ make -C <source-tree> M=. modules
- ✔ __1__ make rpizw_defconfig
- ✔ __2__ make modules_prepare
- ✔ __4__ insmod

Spørgsmål 4 5 / 5 point

Når jeg skal implementere en driver til et accelerometer med SPI, hvor finder jeg så de relevante informationer?

- ✔ __1__ Hvordan en driver for et tilsvarende device er udformet?
- ✔ __3__ Hvilket SPI Clock Mode som skal benyttes?
- ✔ __2__ Hvordan Pin Control / Pin Mux skal konfigureres?
- ✔ __4__ Hvilken SPI Slave Select som benyttes?

1. Linux Source Tree
2. Processor data sheet
3. Accelerometer data sheet
4. Electrical schematics (Diagrammer)

Spørgsmål 5 5 / 5 point

I en SPI driver kan man for en spi_transfer ud over rx_buf/tx_buf også angive et delay_usecs, som angiver en pause til den næste transfer. Med de metoder som vi har anvendt, hvilken funktion kan implementere denne funktionalitet?

- ○ mod_timer()
- ✔ ◉ udelay()
- ○ schedule_timeout()
- ○ msleep()

Spørgsmål 6 5 / 5 point

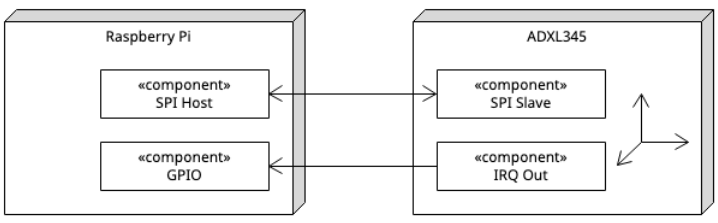
På min Raspberry Pi, hvis jeg i config.txt specificerer at et bestemt device tree overlay skal indlæses under boot, i hvilken folder, eller underfolder heraf, skal jeg så lægge det kompilerede overlay?

- ○ <root_fs>/var/lib/
- ○ <root_fs>/boot/
- ○ <root_fs>/sys/firmware/devicetree/
- ✔ ◉ <source_tree>/arch/arm/boot

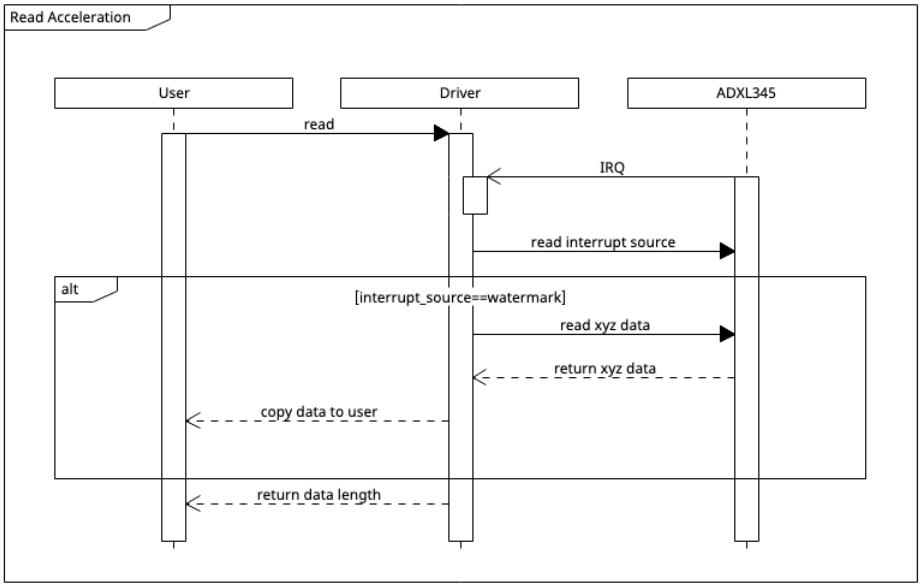
Spørgsmål 7 0 / 25 point

Bemærk at dette spørgsmål består af 3 underspørgsmål.

ADXL345 er et 3-akset accelerometer. Til et nyudviklet spilcontroller ønskes denne anvendt sammen med en Raspberry Pi Zero W:



På Raspberry Pi er der installeret Linux og der kører en user-space applikation som analyserer data fra accelerometer'et. Accelerometret sampler de tre akser X,Y,Z med en fast frekvens og lægger dem i en intern FIFO. Når FIFO'en indeholder en angivet mængde data, kaldet "Watermark level", så giver det et interrupt til Raspberry Pi om at nu skal der hentes data. Kommunikationen mellem applikation, driver og hardware ser således ud:



User-space applikation læser, driverens read funktion falder i søvn indtil at der kommer et interrupt. Så læser den interruptkilden fra ADXL345 og hvis interruptkilden er "Watermark", hvilket betyder at der er data nok i FIFOen, så læser man den tom og gemmer data i en buffer i driveren. Data returneres til sidst til user-space applikationen.

Spørgsmål 1) For at undgå SPI kommunikation efter hver sample, så gemmes accelerationsværdierne i en FIFO på 32 bytes, som overføres på en gang. Hvad skal minimum SPI frekvens være, for at der er tid til at overføre hele FIFO indholdet i mellem to accelerationssamples, når disse samples med 400 Hz?

Spørgsmål 2) Når der kommer et interrupt fra ADXL345, skal man læse interrupt source registret på denne, for at afgøre årsagen til interruptet. Registret ser således ud:

Register 0x30—INT_SOURCE (Read Only)

D7 DATA_READY	**D6** SINGLE_TAP	**D5** DOUBLE_TAP	**D4** Activity
D3 Inactivity	**D2** FREE_FALL	**D1** Watermark	**D0** Overrun

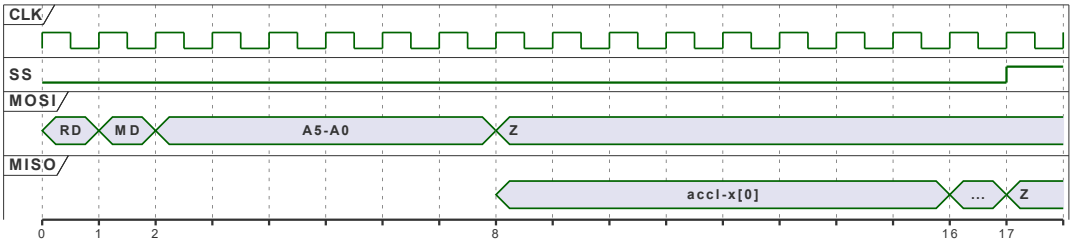
Din opgave er at lave den manglende kode som henter værdien og sætter flaget "watermark_irq_active", således at det kan bruges til at afgøre om vi skal udlæse XYZ data (som vi kigger på i sidste spørgsmål)

```
/* adxl345_read_intsource: Helper function that reads */
/* the value of the INT_SOURCE register */
/* -int_source: Value of INT Source Register */
/* Return 0 on success or negative on error */
/* USE THIS FUNCTION AS IT IS, assume implemented elsewhere */
int adxl345_read_intsource(struct spi_device *spi, u8 *int_source);
...
/* adxl345_spi_device: SPI device, assume it is initialized elsewhere */
static struct spi_device adxl345_spi_device;
...
/* Drivers read function */
ssize_t adxl345_read(struct file *filep, char __user *ubuf,
                     size_t count, loff_t *f_pos)
{
int watermark_int_active = 0;
wait_event_interruptible(..) // Wait until interrupted
/* Implement missing code */
if(watermark_int_active)
  adxl345_read_xyz_data();
```

```
...
}
```

Spørgsmål 3) Du skal her implementere koden til at lave SPI data overførslen af XYZ data fra ADXL345.

Dataoverførslen startes ved at man sender en kommando med nogle bits konfiguration og en adresse, hvorefter at ADXL345 begynder at returnere data:

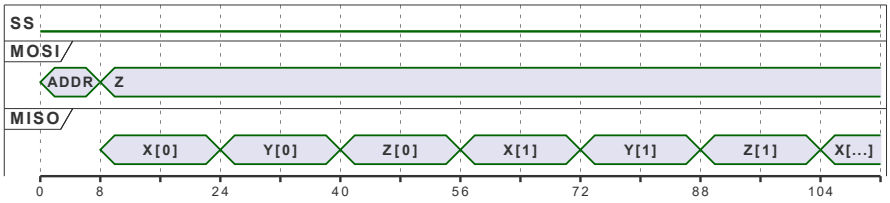


Bemærk. 'Z' betyder 'High impedance', eller ikke-gyldig/ikke-betydende værdi.

For at starte en data overførsel fra FIFO'en skal konfiguration og adresse sættes således i den første tx transfer:

Bit(s)	Værdi	
RD	1	Read transfer
MB	1	Multi-byte transfer
A[5:0]	0x32	X-Acceleration Data Register

Efterfølgende returnerer ADXL345 nu X,Y,Z data:



Det antages at vi altid tømmer FIFO'en, dvs. der overføres XYZ [0..4] af 16 bits.

Implementer den manglede kode i hjælpefunktionen herunder:

```
/* SPI read accelerations helper function */
/* Parameters: */
/* -spi: Pointer to valid SPI device */
/* -accelerations: Raw accelerometer data from adxl345 */
/* Returns negative on error */
int adxl345_read_xyz_data(struct spi_device *spi, s16 *accelerations);
{
  /* Implement missing code */
  /* Build konfiguration/address tx transfer */
  /* Build transfers to retrieve accelerometer data */
  /* Transfer and return result */
}
```

Dette spørgsmål er ikke blevet bedømt.

Det rigtige svar vises ikke for spørgsmål, der kræver skriftlige svar.

Spørgsmål 8 5 / 5 point

Parameteren filep i driverens read funktion kan bruges til at udlede hvilken major/minor nummer som er årsag til at funktionen er blevet kaldt. Hvilket element i struct file indeholder denne information?

```
ssize_t my_read(struct file *filep, char __user *ubuf,
size_t count, loff_t *f_pos)
```

- ○ f_owner

- ✔ (●) f_inode
- ( ) f_path
- ( ) f_pos

Spørgsmål 9 5 / 5 point

Hvilken makro gør det ud for følgende kode:

```
static struct device_attribute dev_attr_adxl345_data_rate = {
  .attr = {
    .name = "adxl345_data_rate",
    .mode = 0666},
  .show  = adxl345_data_rate_show,
  .store = adxl345_data_rate_store,
};
```

- ( ) `DEVICE_BOOL_ATTR(adxl345_data_rate);`
- ✔ (●) `DEVICE_ATTR_RW(adxl345_data_rate);`
- ( ) `DEVICE_ATTR_RO(adxl345_data_rate);`
- ( ) `DEVICE_ATTR_WO(adxl345_data_rate);`

Spørgsmål 10 5 / 5 point

I hvilken folder finder jeg attributter tilhørende devicet "mydevice21"?

- ( ) /dev/mydevice21
- ( ) /proc/devices
- ✔ (●) /sys/class/myclass/mydevice21
- ( ) /sys/firmware/devicetree/base/soc/mydevice21

Spørgsmål 11 5 / 5 point

Øvelse 8 omhandlede bl.a. brugen af timers. I den forbindelse kom vi i emnet ind på at man kan oprette en global instans, eller oprette en timer som en del af den struct som beskriver vores gpio device:

```
/* One timer defined globally */

struct timer_list my_timer;

/* Timer defined as part of device struct */

struct gpio_dev {
 int gpio;
 int delay;
 struct timer_list my_timer;
};
struct gpio_dev my_gpio_devs[12];
```

Hvad kunne man i øvelsen opnå ved at indsætte my_timer i struct'en?

- ( ) At LED'er brugte samme timer og derved blinkede med samme hastighed
- ( ) At man kunne bruge makroer når attributterne skulle oprettes
- ( ) At LED'er blinkede mere præcist, da vi opnåede en højere præcision på timeren
- ✔ (●) At LED'er brugte individuelle timers og derved kunne blinke med individuel hastighed

Spørgsmål 12 5 / 5 point

Med hvilken Linux kommando kan man se indholdet af kernel loggen?

- ( ) ps -A
- ✔ (●) dmesg
- ( ) ls -l

- ○ top

Spørgsmål 13 5 / 5 point

I øvelse 7 omkring SPI drivers arbejdede I blandt andet med hot-plugging. Hvilken funktion i driveren kaldes, når Linux opdager at der findes et device som kan anvende denne driver?

- ✓◉ probe()
- ○ read()
- ○ init()
- ○ open()

Spørgsmål 14 5 / 5 point

Hvilket device tree udsnit konfigurer SPI interfacet korrekt ifht. det viste udsnit fra et datablad? Bemærk at det er angivet hvordan at SDI (MOSI) skal være gyldig omkring clock rising edge.

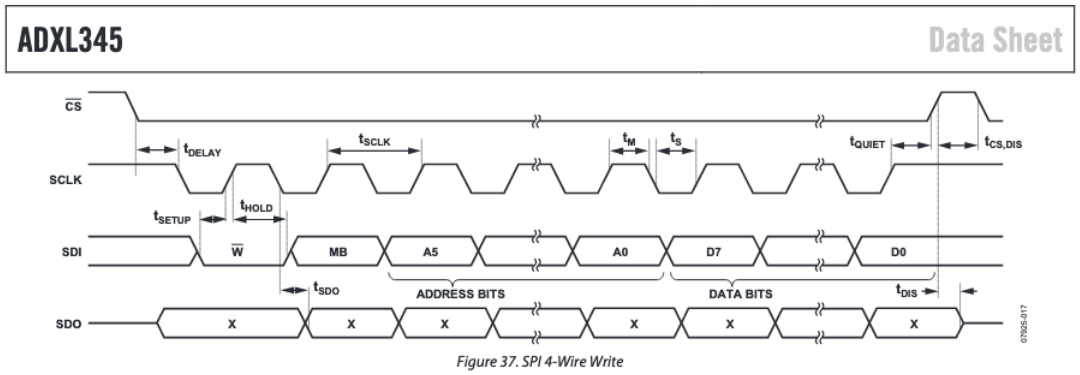


Figure 37. SPI 4-Wire Write

t_{SETUP}	5		ns	SDI valid before SCLK rising edge
t_{HOLD}	5		ns	SDI valid after SCLK rising edge
t_{SDO}		40	ns	SCLK falling edge to SDO/SDIO output transition

- ○
```
target = <&spi0>;
__overlay__ {
  adxl345: adxl345@0 {
    compatible = "ad,adxl345";
    reg = <0>;
    /* spi-cpha; */
    /* spi-cpol; */
  };
};
```
- ○
```
target = <&spi0>;
__overlay__ {
  adxl345: adxl345@0 {
    compatible = "ad,adxl345";
    reg = <0>;
    spi-cpha;
    /* spi-cpol; */
  };
};
```
- ○
```
target = <&spi0>;
__overlay__ {
  adxl345: adxl345@0 {
    compatible = "ad,adxl345";
    reg = <0>;
    /* spi-cpha; */
    spi-cpol;
  };
};
```
- ✓◉
```
target = <&spi0>;
__overlay__ {
  adxl345: adxl345@0 {
```

```
        compatible = "ad,adxl345";
        reg = <0>;
        spi-cpha
        spi-cpol
      };
    };
```

Spørgsmål 15 5 / 5 point

Match driver type og device

✔ __4__ A/D converter	**1**. Platform driver
✔ __1__ Switch	**2**. Block driver
✔ __2__ USB drive	**3**. Network interface
✔ __3__ Ethernet device	**4**. SPI driver

Spørgsmål 16 5 / 5 point

I forbindelse med interrupts, hvad bør/må udføres i top-half?

- ✔ (●) Ting som er tidskritiske
- ( ) Ting som allokerer memory
- ( ) Ting som kræver meget processering
- ( ) Ting som bruger floating-point operationer

Udført

E2022 - Resultater

Forsøg 3 ud af Ubegrænset

Skriftlig 10. jun. 2025 22:22 - 11. jun. 2025 04:42

Your quiz has been submitted successfully.

Forsøgsscore **70 / 100 - 70 %**
Samlet karakter (Højeste forsøg) **70 / 100 - 70 %**

Spørgsmål 1 (Obligatorisk) 5 / 5 point

Hvilken bash kommando kan man benytte for at se hvilken folder man står i?

- less
- ✓ pwd
- mv
- dmesg

Spørgsmål 2 (Obligatorisk) 5 / 5 point

Med hvilken bash kommando kan man krypteret kopiere en lokal fil til en folder på en remote host? F.eks. fra en lokal Ubuntu til en remote Raspberry Pi

- ssh root@10.9.8.2
- scp root@10.9.8.2:file_to_copy .
- ✓ scp file_to_copy root@10.9.8.2:
- cp /media/raspberrypi/file_to_copy /home/stud/

Spørgsmål 3 (Obligatorisk) 5 / 5 point

Hvilken device tree fragment kode initialiserer interfacet korrekt til den viste chip, når den sidder på SPI bus nummer 0 og bruger slave select 0 (Slave select er benævnt CSN i figuren)?

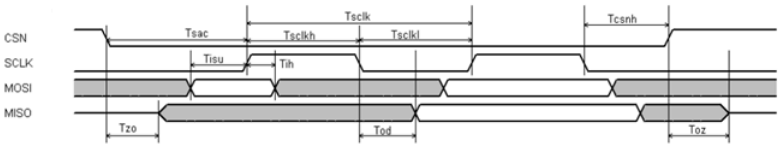


Figure 6-1 SPI Interface Timing

Table 6-7 SPI Interface Timing

Parameter	Description	VCC(I/O)=3.3V		Units
		Min	**Max**	
Tsclk	SPI clock period	33	-	ns
Tsclkl	SPI clock low duration	13	-	ns
Tsclkh	SPI clock high duration	13	-	ns
Tsac	SPI access time	16	-	ns
Tisu	Input Setup	11	-	ns
Tih	Input Hold	3	-	ns
Tzo	Output enable delay	0	16	ns
Toz	Output disable delay	0	16	ns
Tod	Output data delay	0	12	ns
Tcsnh	CSN hold time	0	-	ns

○
```
fragment@0 {
    target = <&spi0>;
    __overlay__ {
        myspi@1 {
            compatible = "myspi";
            reg = <1>;
            /* spi-cpha; */
            spi-cpol;
        };
    };
};
```

○
```
fragment@1 {
    target = <&spi0>;
    __overlay__ {
        myspi@0 {
            compatible = "myspi";
            reg = <0>;
            spi-cpha;
            /* spi-cpol; */
        };
    };
};
```

✓●
```
fragment@2 {
    target = <&spi0>;
    __overlay__ {
        myspi@0 {
            compatible = "myspi";
            reg = <0>;
            /* spi-cpha; */
            /* spi-cpol; */
        };
    };
};
```

○
```
fragment@3 {
    target = <&spi0>;
    __overlay__ {
        myspi@1 {
            compatible = "myspi";
            reg = <1>;
            /* spi-cpha; */
            /* spi-cpol; */
        };
    };
};
```

Spørgsmål 4 5 / 5 point

I øvelsen omkring memory konfigurerede vi GPIO19 således at den udsendte et PWM signal med en given duty-cycle. Hvilken anden GPIO på BCM2835 kan også benyttes til PWM?

○ 17

✓● 18

○ 20

○ 21

Spørgsmål 5 (Obligatorisk) 5 / 5 point

I hvilken folder oprettes attribute filer når en driver med device attributes loades?

- ○ /dev/
- ○ /boot/overlays/
- ✓◉ /sys/class/myclass/mydevice1/
- ○ /proc/devices/

Spørgsmål 6 (Obligatorisk) 5 / 5 point

I hvilken folder skal jeg lægge en device tree blob overlay fil (.dtbo) for at den kan indlæses automatisk under boot?

- ○ /lib/modules
- ○ /sys/firmware/devicetree/
- ✓◉ /boot/overlays/
- ○ /dev

Spørgsmål 7 (Obligatorisk) 5 / 5 point

Angiv hvilke gpio funktioner som skal kaldes i hvilke device driver funktioner

✓ __4__ my_drv_remove()	**1**. gpio_request()
✓ __1__ my_drv_probe()	**2**. gpio_set_value()
✓ __3__ my_drv_read()	**3**. gpio_get_value()
✓ __2__ my_drv_write()	**4**. gpio_free()

Spørgsmål 8 (Obligatorisk) 5 / 5 point

I en drivers interrupt service rutine skal vi vente 6 us. Men hvilken funktion kan dette implementeres?

- ✓◉ udelay()
- ○ gettimeofday()
- ○ msleep()
- ○ usleep()

Spørgsmål 9 (Obligatorisk) 5 / 5 point

Hvad er den primære forskel på top- og bottom half "interrupt" handlers?

- ○ At bottom-half kan afbryde top-half interrupts
- ○ Den ene kører i kernel space, mens den anden kører i user space
- ✓◉ At top-half kører i interrupt kontekst, mens at bottom-half kører i process kontekst, dvs. afviklingen kan her afbrydes af Linux
- ○ At der kun kan være én interrupt kilde som starter et top-half interrupt, mens at flere kilder kan starte et bottom-half interrupt

Spørgsmål 10 (Obligatorisk) 5 / 5 point

På et Linux system, i hvilken fil kan jeg se hvilke interrupts som er i brug?

- ✓◉ /proc/interrupts
- ○ /sys/class/interrupts
- ○ /dev/
- ○ /sys/firmware/devicetree/interrupts

Spørgsmål 11 (Obligatorisk) 5 / 5 point

En platform driver med gpios skal implementeres. Angiv den korrekte rækkefølge af funktionerne i probe():

- ✓ __3__ gpio_direction_input()
- ✓ __4__ device_create()
- ✓ __2__ gpio_request()
- ✓ __1__ of_gpio_count()

Spørgsmål 12 (Obligatorisk) 5 / 5 point

I hvilken Linux source tree subfolder finder man headerfilerne som man benytter i sine drivers, f.eks. spi.h? (source tree: ~/sources/rpi-5.4.83/)

- ./sys/firmware
- ✓ ./include
- ./arch/arm/boot/overlays
- ./drivers/headers

Spørgsmål 13 (Obligatorisk) 5 / 5 point

Hvad betyder det når man i et Device Tree skriver <&spi0> som i f.eks:

target = <&spi0>;

- Man laver en henvisning til et element i driveren, f.eks. en spi device pointer
- ✓ Man laver en henvisning til en label et andet sted i Device Tree
- Man laver en henvisning til et element i /boot/config.txt
- Man overfører værdien af spi0 *by reference*

Spørgsmål 14 (Obligatorisk) 5 / 5 point

Hvad hedder det sted hvor processorens memory management unit gemmer informationer om rettigheder til et hukommelsesområder?

- Cache
- ✓ Page Table
- Memory Page
- Virtual Memory

Spørgsmål 15 (Obligatorisk) 0 / 5 point

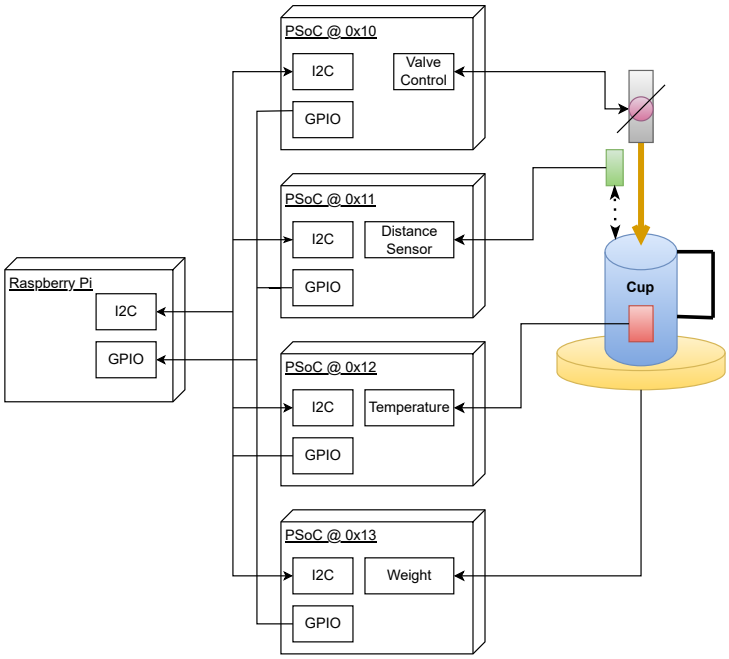
Hvilket kodeeksempel initialiserer attribute groups korrekt for følgende attribute? *static ssize_t lcd_status_store(struct device *dev, struct device_attribute *attr, static char *buf)*

- ✗
```
DEVICE_ATTR_RW(lcd_status);
static struct attribute *lcd_attrs[] = {
  &dev_attr_lcd_status.attr, NULL, }
ATTRIBUTE_GROUPS(lcd);
```
-
```
DEVICE_ATTR_RO(lcd_status);
static struct attribute *lcd_attrs[] = {
  &dev_attr_lcd_status.attr, NULL, }
ATTRIBUTE_GROUPS(lcd);
```
- ➡
```
DEVICE_ATTR_WO(lcd_status);
static struct attribute *lcd_attrs[] = {
  &dev_attr_lcd_status.attr, NULL, }
ATTRIBUTE_GROUPS(led);
```
-
```
DEVICE_ATTR_RW(led_status);
static struct attribute *led_attrs[] = {
  &dev_attr_lcd_status.attr, NULL, }
ATTRIBUTE_GROUPS(led);
```

Spørgsmål 16 0 / 25 point

Bemærk at dette spørgsmål består af 3 underspørgsmål.

Til et 3. semester projekt ønskes udviklet en automatisk sodavands dispenser. Systemet består af en Raspberry Pi, via hvilken man kan bestille sin drik, og fire PSoCs som hver især er tilsluttet en sensor eller aktuator.



De fire PSoCs har hver sin I2C adresse og er forbundet via en fælles I2C bus til Raspberry Pi (RPI), som er I2C Master og den eneste som kan tage initiativ til at kommunikere. Hver PSoC har desuden en GPIO udgang som er forbundet til en fælles GPIO indgang på RPI. Disse forbindelser benyttes til et shared interrupt, som den enkelte PSoC kan aktivere, såfremt den har brug for at kommunikere, f.eks. ved en alarm. Konkret sætter en PSoC sit GPIO output lavt når den vil kommunikere og holder det lavt indtil at RPI har læst dens data via I2C.

Spørgsmål 1) I en driver som skal køre på RPI, hvordan vil du konfigurere interruptet? Argumentér hvorfor.

I kodeudsnittet herunder er vist en device datastruktur til de fire PSoC devices, som er initialiseret med I2C addresserne. Desuden er init funktionen vist, hvor *request_irq* skal kaldes for at initialisere interrupt(s).

Spørgsmål 2) Implementer koden i *my_init()* som initialiserer interrupt(s).

Når man læser data fra en PSoC over I2C, returnerer den en byte med følgende indhold:

IRQ	D6	D5	D4	D3	D2	D1	D0
7	6	5	4	3	2	1	0

D6-D0: 7-bit data (temperatur, vægt etc)

IRQ: 1-bit hvorvidt PSoC skal serviceres (1=IRQ Request, 0=None)

I2C data kan læses med hjælpefunktionen *i2c_read()* vist i kodeudsnittet. Vi antager i denne opgave at i2c kommunikationen er ikke-blokerende og derfor godt må foregå i interrupt service rutinen (ISR).

Spørgsmål 3) Implementer *my_isr()* som læser data fra PSoC(s), gemmer det i device's datastruktur, såfremt den er årsagen til interruptet og returnér i henhold hertil.

C

```c
/* I2C read helper function, */
/* assume implemented elsewhere */
/* Returns negative value on error */
int i2c_read(int i2c_addr, u8 *data);

struct psocdev {
  int i2c_addr;
  u8 data;
};
struct psocdev psocdevs[4] = {{0x10,0}, {0x11,0}, {0x12,0}, {0x13,0}};

static my_init(void) {
  const int gpio = 23;
  [...]

  /* Question 2: Write the code to */
  /* initialize interrupt(s) */
  request_irq(...)

  [...]
}
```

```
22
23 irqreturn_t my_isr(int irq_no, void *dev_id) {
24   struct psocdev *my_psoc = (struct psocdev *)dev_id;
25
26   /* Question 3: Write code to implement ISR */
27   /* - Read data */
28   /* - Store in device's struct */
29   /* - Return as appropriate */
30
31 }
```

Dette spørgsmål er ikke blevet bedømt.

Det rigtige svar vises ikke for spørgsmål, der kræver skriftlige svar.

Udført

F2023-PREEXAM - Resultater

Forsøg 3 ud af Ubegrænset

Skriftlig 10. jun. 2025 20:30 - 10. jun. 2025 20:32

Your quiz has been submitted successfully.

Forsøgsscore **70 / 100 - 70 %**
Samlet karakter (Sidste forsøg) **70 / 100 - 70 %**

Spørgsmål 1

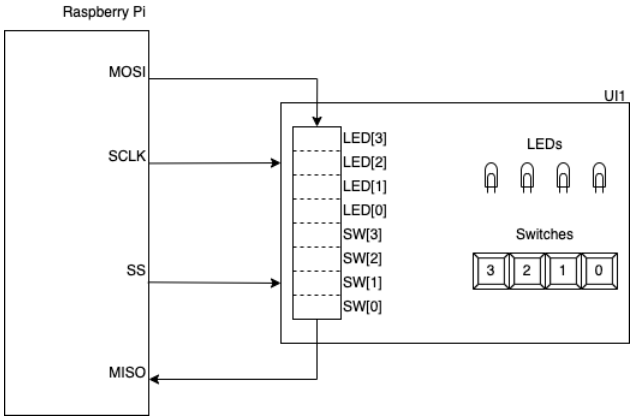
I forbindelse med interrupts, hvad må udføres i top-half? (Vælg 2)

- [x] Opgaver som er tidskritiske
- [ ] Opgaver som kræver meget processering
- [ ] Opgaver som bruger floating-point operationer
- [x] Opgaver som ikke kan blokere
- [ ] Opgaver som afventer input fra andre enheder

Spørgsmål 2

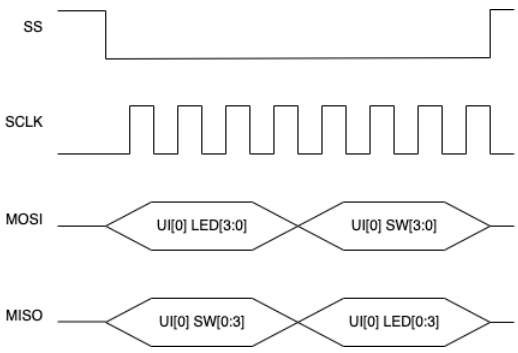
Bemærk at dette spørgsmål indeholder 3 underopgaver.

I forbindelse med et semesterprojekt er udviklet et modulært brugerinterface (UI). Et UI modul indeholder 4 LEDs og 4 switches. Figuren herunder illustrerer et enkelt UI modul tilsluttet en Raspberry Pi (RPI).



Til kommunikation anvendes SPI. Hvert modul indeholder et skifte register som indeholder den aktuelle tilstand af de fire switches og hvilken tilstand som de fire LED'er skal have.

Når RPI, via MOSI, sender 8-bit til modul UI1 sendes den eksisterende tilstand af UI1's 8-bits tilbage til RPI. Dvs. for at udveksle data med et modul skal man sende og modtage 8-bit som vist i figuren herunder.



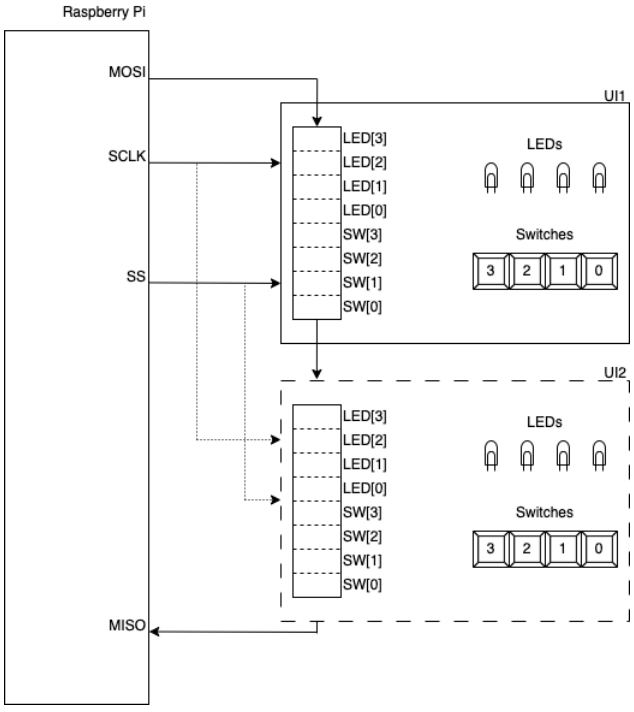
Tilstanden af de fire leds og switches på et modul repræsenteres ved binære værdier i uidev struct'en vist i kodeudsnittet herunder. Er uidevs[0].led = b1111 = 0xF = 15d skal alle fire LED'er på modul (indeks 0) være tændt. Tilsvarende afspejler uidevs[0].sw tilstanden af de 4 switches på et modul.

Spørgsmål 1 (46%): Implementér den nødvendige kode i funktionen *ui_spi_exchange*, som gør det muligt at skrive til og læse fra et UI modul. Når funktionen er udført skal LED'erne på UI modulerne være sat som beskrevet i uidevs[0].led, ligesom at uidevs[0].switch skal være opdateret med tilstanden af switchene på modulerne. Det antages at der er et modul og kun et SPI interface.

Spørgsmål 2 (34%): Implementer driverens read funktion*, ui_drv_read,* som giver brugeren en streng med tilstanden af switchene udtrykt som en hex-værdi. Med et UI modul er der et hex ciffer, dvs: "0xF".

```c
/* UI Module device struct */
/* assume initialized in probe */
struct uidev {
  uint8_t led;
  uint8_t sw;
};

struct uidev uidevs[255];
uint8_t uidevs_len = 1; /* Assume one module */
/* Assume only one SPI device and */
/* it is initialized in probe */
struct spidev spi;

/* Helper function to exchange data over SPI */
void ui_spi_exchange(spidev *spi, struct uidev **uidevs, int uidevs_len){

  /* Question 1: Write the code to */
  /* - Build transfers to exchange data with UI */
  /* - Exchange message */
  /* - Update data */
}

/* Character Driver Read File Operations Method */
ssize_t ui_drv_read(struct file *filep, char __user *ubuf,
                    size_t count, loff_t *f_pos)
{
  /* Question 2: Write the code to */
  /* exchange data over SPI */
  /* Send string of SW values formatted as hex to user */
  /* Return appropriately */
}
```

Hardwaren tillader at man kan kæde flere moduler sammen, dette kræver blot at man udveksler flere bits i sin SPI data overførsel. Dette er vist i følgende figur.



Spørgsmål 3 (20%): Hvordan kan man i device tree angive antallet af moduler og hvordan kan dette udnyttes i driverens probe funktion til at sætte værdien af *uidevs_len?* (Angiv svar som tekst og/eller pseudokode)

Det rigtige svar vises ikke for spørgsmål, der kræver skriftlige svar.

Spørgsmål 3

Hvilket kodeeksempel initialiserer attribute groups korrekt for følgende attributes?

```c
static ssize_t ui_status_store(struct device *dev, struct device_attribute *attr, static char *buf)

static ssize_t ui_status_show(struct device *dev, struct device_attribute *attr, char *buf)
```

✓ ◉
```c
DEVICE_ATTR_RW(ui_status);
static struct attribute *ui_attrs[] = {
  &dev_attr_ui_status.attr, NULL, }
ATTRIBUTE_GROUPS(ui);
```

○
```c
DEVICE_ATTR_RW(ui_status);
static struct attribute *ui_attrs[] = {
  &dev_attr_ui_status_show.attr, &dev_attr_ui_status_store.attr, NULL, }
ATTRIBUTE_GROUPS(ui);
```

○
```c
DEVICE_ATTR_WO(ui_status);
static struct attribute *ui_attrs[] = {
  &dev_attr_ui_status_show.attr, &dev_attr_ui_status_store.attr, NULL, }
ATTRIBUTE_GROUPS(ui_status);
```

○
```c
DEVICE_ATTR_RW(ui_status);
static struct attribute *led_attrs[] = {
  &dev_attr_lcd_status.attr, NULL, }
ATTRIBUTE_GROUPS(ui_status);
```

Spørgsmål 4

En platform driver opretter devices i probe() med *device_create()*. Når man fjerner den kørende driver med rmmod, hvad er rækkefølgen af funktionskald i driveren som gør at de oprettede devices nedlægges? Bemærk! Visse af funktionerne kan kalde andre af de viste. Den ønskede rækkefølge er fra kaldsøjeblikket.

✓ __3__ my_driver_remove()
✓ __2__ platform_driver_unregister()
✓ __4__ device_destroy()
✓ __1__ my_driver_exit()

Spørgsmål 5

Hvad kalder man den enhed som indeholder de senest benyttede Page Tables?

✓ ◉ TLB
○ Heap
○ Slab Allocator
○ VMA

Spørgsmål 6

Med hvilket program kan man se information om memory og CPU forbrug samt sætte prioritering af programmer?

○ ls
○ pwd
✓ ◉ top
○ mem

Spørgsmål 7

En driver anvender en timerfunktion, som skal toggle en gpio port 50 gange i sekundet. Hvilket af følgende kodeudsnit implementerer dette korrekt?

○
```c
static void some_timer_callback(struct timer_list *t){
  mod_timer(&my_timer, jiffies + secs_to_jiffies(0.02));
  gpio_set_value(19, gpio_get_value(19) ^ 1);
}
```

✓ ◉
```c
static void some_timer_callback(struct timer_list *t){
  unsigned long its_time = jiffies + msecs_to_jiffies(20);
  mod_timer(&my_timer, its_time);
  gpio_set_value(19, gpio_get_value(19) ^ 1);
}
```

○
```c
static void some_timer_callback(struct timer_list *t){
  udelay(20000);
  mod_timer(&my_timer, jiffies + secs_to_jiffies(0.02));
  gpio_set_value(19, gpio_get_value(19) ^ 1);
}
```

○
```c
static void some_timer_callback(struct timer_list *t){
  gpio_set_value(19, gpio_get_value(19) ^ 1);
  mdelay(20);
  some_timer_callback();
}
```

Spørgsmål 8

Følgende c-applikation afvikles og anvender tilhørende driver:

```c
// User space program
int main(int narg, char *argp[]) {
  int fp = open("/dev/gpio19", O_RDWR);
  char buf[] = {"sluk"};
  int ret = write(fp, buf, strlen(buf)+1);
  printf("Return value: %d\n", ret);
  return 0;
}
```

```c
// Snippet from kernel driver fops write
ssize_t cdrv_write(struct file *filep, const char __user *ubuf,
                           size_t count, loff_t *f_pos)
{
  char kbuf[6];
  int len;
  ... // Handle length..
  int ret = copy_from_user(kbuf, ubuf, len);
  ... // Use data from user for something..
  return len;
}
```

Hvilken output giver applikationen?

○ "Return value: 3"

○ "Return value: 4"

✓ ◉ "Return value: 5"

○ "Return value: 6"

Spørgsmål 9

I hvilke situationer vil man vælge at bruge udelay frem for usleep? (Vælg 2)

✓ ☐ Når der er behov for at frigive CPU ressourcer i systemet

✓ ☑ Når præcision er vigtigt

✓ ☐ Når præcision ikke er afgørende

- [x] Når forsinkelsen er større end 5 ms
- [x] Når man ikke må blokere, f.eks. i et interrupt

Spørgsmål 10

Hvor kan du finde information omkring hvordan man anvender i2c i Linux device drivers?

- [x] <source_tree>/Documentation/i2c/writing-clients.rst
- [ ] <source_tree>/Documentation/i2c/i2cbus-protocol.rst
- [ ] <source_tree>/Documentation/i2c/i2c-howto.rst
- [ ] <source_tree>/Documentation/i2c/i2c-drivers.rst

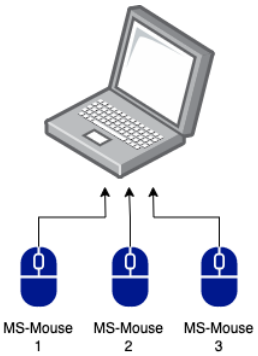
Spørgsmål 11

Hvad er den korrekte fejlhåndtering når man anvender copy_from_user()?

- [ ]
```c
int a = copy_from_user(c,d,e);
if(a>0)
  goto err;
```
- [ ]
```c
int a = copy_from_user(c,d,e);
if(a<0)
  goto err;
```
- [x]
```c
int a = copy_from_user(c,d,e);
if(a!=0)
  goto err;
```
- [ ]
```c
int a = copy_from_user(c,d,e);
if(a==0)
  goto err;
```

Spørgsmål 12

Givet følgende system:

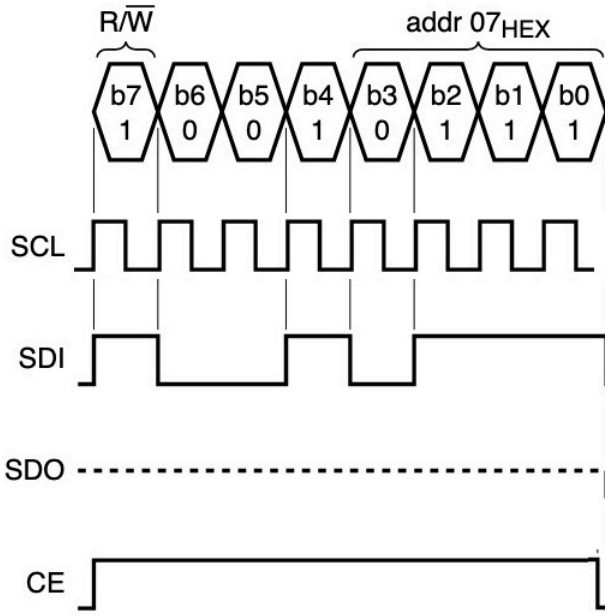


Hvor mange major og minor numre skal MS-Mouse driveren bruge?

- [ ] 3 major og 3 minor numre
- [ ] 3 major og 1 minor numre
- [x] 1 major og 3 minor numre
- [ ] 1 major og 1 minor numre

Spørgsmål 13

Hvordan skal CPHA og CPOL konfigureres for viste SPI Device (CE = Slave Select, SCL = Serial Clock, SDI = MOSI, SDO = MISO):



- ○ CPOL = 0
  CPHA = 0
- ○ CPOL = 1
  CPHA = 0
- ✓◉ CPOL = 0
  CPHA = 1
- ○ CPOL = 1
  CPHA = 1

Spørgsmål 14

Hvilket device tree fragment kode initialiserer interfacet korrekt til et spi device som benytter slave select 1 på spi bus nummer 2?

- ○
```
fragment@0 {
    target = <&spi2>;
    __overlay__ {
        myspi@1 {
            compatible = "myspi";
            reg = <0>;
            /* spi-cpha; */
            /* spi-cpol; */
        };
    };
};
```
- ○
```
fragment@1 {
    target = <&spi1>;
    __overlay__ {
        myspi@1 {
            compatible = "myspi";
            reg = <2>;
            spi-cpha;
            /* spi-cpha; */
            /* spi-cpol; */
        };
    };
};
```

○
```
fragment@2 {
    target = <&spi0>;
    __overlay__ {
        myspi@0 {
            compatible = "myspi";
            reg = <2>;
            /* spi-cpha; */
            /* spi-cpol; */
        };
    };
};
```

✓◉
```
fragment@3 {
    target = <&spi2>;
    __overlay__ {
        myspi@1 {
            compatible = "myspi";
            reg = <1>;
            /* spi-cpha; */
            /* spi-cpol; */
        };
    };
};
```

Spørgsmål 15

Hvilken fil skal jeg modificere, hvis jeg gerne vil have at min Raspberry Pi automatisk indlæser et device tree overlay?

- ○ /etc/modules.load/my_overlay.conf
- ○ /etc/modprobe.d/my_overlay.conf
- ✓◉ /boot/config.txt
- ○ /sys/firmware/devicetree

Udført