Mittuniversitetet
MID SWEDEN UNIVERSITY

Försättsblad Prov Original

Kurskod	Provkod	Tentamensdatum
ET095G	T102	2018-08-29
Kursnamn	Elektroteknik GR (B), Introduktion till programmering av...	
Provnamn	Skriftlig examination	
Ort	Sundsvall	
Termin	H18	
Ämne	Elektroteknik	

8982538474

0027798

Exam in Introduction to Embedded System Programming Part 2 (HT2017)

Course codes: ET032G, ET095G

Information

Date:	2018-08-29
Duration:	3 hours
Examiner:	Sebastian Bader
Phone:	+46 10 1428095 or +46 72 7433680
Maximum achievable points:	50
Minimum required points to pass:	25
Aids (Hjälpmedel):	Calculator

General Instructions

- Begin the answer to every question on a separate sheet of paper
- Do not write in red color
- For calculations, the way towards the result has to be clearly understandable
- Direct citations from lecture slides will not be accepted as an own contribution

---

Questions

Question 1:

Together with two other students you have been assigned an embedded system project. Together you are supposed to develop a simple application that reads data from a sensor, performs some calculations on the data and display the results on a display. You are told to create separate function libraries for the sensor and the display.

(a) In your own words, describe what a function library is. (2)

(b) Why could it be useful for your project to develop the libraries for the sensor and display? (4)

(c) What purpose would the usage of API documentation for your libraries fulfill? (2)

Question 2:

Someone in your dormitory frequently forgets to close the door to the fridge in the shared kitchen, leading to much of your food going to waste. For your summer project you decide to build a temperature monitoring system that could be placed inside the fridge. You have connected a TMP36 analog temperature sensor to the ADC of your development board. The ADC has a resolution of 12 bit and operates with a reference voltage of 3 V.

(a) When you test the system, your development board reports a value of 751. What does that mean for the output voltage of the sensor? (4)

(b) What temperature does this represent, considering the relationship provided in Figure 1? (4)

(c) According to its datasheet, the TMP36 sensor has an accuracy of $\pm 2\,^{\circ}\mathrm{C}$. Is the quantization error due to the ADCs resolution smaller or larger than this? (4)

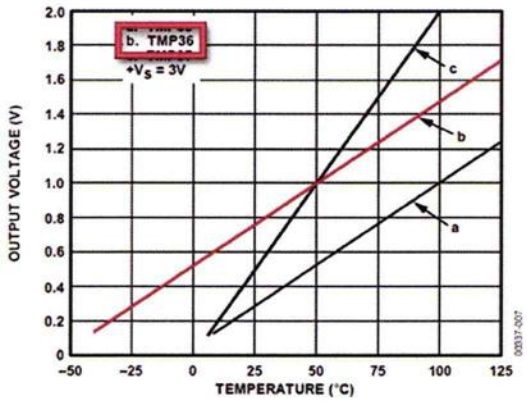

Figure 1: Relationship between output voltage and temperature for the TMP36. Voltage at 0 degrees Celsius is 0.5V.

Question 3:

After having implemented your sensor reading you are planing to add a warning light. You have connected a red LED to one of the MCUs PWM channels. You want the LED to light brighter the warmer it gets in the fridge.

(a) Why should the LED be connected to a PWM channel and not just a common GPIO pin? (4)

(b) Explain how you can utilize the ADCs result as an input to the LED control. (4)

Question 4:

You are considering to add another LED, but there is no additional PWM channel available on the development board you are using.

Explain how you could achieve the same functionality by using the combination of a timer module and a GPIO pin. (4)

Question 5:

The timer you try to use counts by default from zero to its top value (maximum value of the counter register) and then resets back to zero. You observe that the timer periodically resets with $T_{period} = 8.192\,\text{ms}$. You know that the counter register has a bit width of 16 bit.

(a) Calculate the frequency of the clock source connected to the timer. (4)

(b) Explain how T_{period} could be adjusted with the help of a prescaler. (4)

Question 6:

Tinkering with the timer got you interested in configuring a timer without the help of an API. You have been digging in the MCUs documentation and found some description of the RTC timer registers (see Figures 2 and 3).

(a) Provide a line of C code that adjusts the RTC_CTRL register in order to enable the RTC and set COMP0 as the top value of the register. Do not modify other settings of the register. (4)

(b) Provide a line of C code that enables compare match 0 interrupts by altering the RTC_IEN register. Do not modify other settings of the register. (2)

(c) Motivate why using this interrupt is typically an advantage over polling the timer. (4)

19.5.1 RTC_CTRL - Control Register (Async Reg)

For more information about Asynchronous Registers please see Section 5.3 (p. 18) .

Offset	Bit Position																													
0x000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset																														0	0	0
Access																														RW	RW	RW
Name																														COMP0TOP	DEBUGRUN	EN

Bit	Name	Reset	Access	Description
31:3	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
2	COMP0TOP	0	RW	**Compare Channel 0 is Top Value**

When set, the counter is cleared in the clock cycle after a compare match with compare channel 0.

Value	Mode	Description
0	DISABLE	The top value of the RTC is 16777215 (0xFFFFFF)
1	ENABLE	The top value of the RTC is given by COMP0

1	DEBUGRUN	0	RW	**Debug Mode Run Enable**

Set this bit to enable the RTC to keep running in debug.

Value	Description
0	RTC is frozen in debug mode
1	RTC is running in debug mode

0	EN	0	RW	**RTC Enable**

When this bit is set, the RTC is enabled and counts up. When cleared, the counter register CNT is reset.

Figure 2: Description of the RTC_CTRL register

19.5.8 RTC_IEN - Interrupt Enable Register

Offset	Bit Position																													
0x01C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset																														0	0	0
Access																														RW	RW	RW
Name																														COMP1	COMP0	OF

Bit	Name	Reset	Access	Description
31:3	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
2	COMP1	0	RW	**Compare Match 1 Interrupt Enable**
	Enable interrupt on compare match 1.			
1	COMP0	0	RW	**Compare Match 0 Interrupt Enable**
	Enable interrupt on compare match 0.			
0	OF	0	RW	**Overflow Interrupt Enable**

Figure 3: Description of the RTC_IEN register