Text

Description automatically generated

**School of Electrical and Computer Engineering**

**Embedded Systems Design - ECE 5721 - Fall 2022**

**Homework 2**

**Daniel Funke**

**October 18, 2022**

# Chapter 3

1. Consider the scheduling approach described in the section “Creating and Using Tasks.” See Figure 3.4. Assume there is no time taken to switch between tasks, and that the tasks have the following execution times:

|  |  |  |
| --- | --- | --- |
| **Task or Handler** | **Execution time when in flash mode** | **Execution time when in RGB mode** |
| Task\_Read\_Switches | 1 ms | 1 ms |
| Task\_Flash | 100 ms | 1 ms |
| Task\_RGB | 1 ms | 1000 ms |

1. Describe the sequence of events that leads to maximum delay between pressing the switch and seeing the LED flash. Calculate the value of that delay.

The maximum delay would occur when the user presses switch one immediately after Task\_RGB begins execution. This results in Task\_RGB taking the maximum amount of time which is:

1. Describe the sequence of events that leads to maximum delay between releasing the switch and seeing the LED sequence through RGB colors. Calculate the value of that delay.

The maximum delay would occur when the user releases the switch immediately after Task\_Flash begins executing, which would be:

1. (Disregarded per instructions)
2. Would changing the scheduler to call Task\_RGB **before** Task\_Flash change the delays in (a) and (b)? If so, determine the new delays and explain why they changed.

The delays would change in parts (a) and (b) if the order of Task\_RGB and Task\_Flash were revered. The new delays would be:

1. 1101ms
2. Another developer wants to add two more tasks (Task\_X and Task\_Y) to the system of the previous question. Each task can take up to 80 ms to complete.
   1. When will there be maximum delay between pressing the switch and seeing the LED flash? Calculate the value of that delay.

The maximum delay would occur when the user presses the switch immediately after the Task\_RGB begins execution. It would calculate to be:

* 1. When will there be maximum delay between releasing the switch and seeing the LED sequence through RGB colors? Calculate the value of that delay.

The maximum delay will occur when the switch is released immediately following the start of Task\_Flash execution:

* 1. (Disregarded per instructions)

1. Consider the scheduling approach of the section “Interrupts and Event Triggering.” See Figure 3.7. Assume that PORTD\_IRQHandler starts executing as soon as the switch changes from pressed to released or from released to pressed. Also assume there is no time taken to switch between tasks or the handler, and that the tasks and handler have the following execution times:

|  |  |  |
| --- | --- | --- |
| **Task or Handler** | **Execution time when in flash mode** | **Execution time when in RGB mode** |
| PORTD\_IRQ\_Handler | 0.01 ms | 0.01 ms |
| Task\_Flash | 100 ms | 1 ms |
| Task\_RGB | 1 ms | 1000 ms |

1. Describe the sequence of events that leads to maximum delay between pressing the switch and seeing the LED flash. Calculate the value of that delay.

Once again, the maximum delay would occur if the user presses the switch immediately after the Task\_RGB begins execution:

1. Describe the sequence of events that leads to maximum delay between releasing the switch and seeing the LED sequence through RGB colors. Calculate the value that delay.

Once again, the maximum delay would occur when the user presses/releases the switch immediately after execution of Task\_Flash:

1. (Disregarded per instructions)
2. Consider the scheduling approach of the section “Reducing Task Completion Times with Finite State Machines.” See Figure 3.9. Assume there is no time taken to switch between tasks, and that the tasks have the following execution times:

|  |  |  |
| --- | --- | --- |
| **Task or Handler** | **Execution time when in flash mode** | **Execution time when in RGB mode** |
| Task\_Read\_Switches | 1ms | 1ms |
| Task\_Flash | 34 ms | 1 ms |
| Task\_RGB | 1 ms | 334 ms |

1. Describe the sequence of events that leads to maximum delay between pressing the switch and seeing the LED flash. Calculate the value of that delay.

The maximum delay occurs immediately after the start of execution of the Task\_RGB. However, since the task has been broken up into subtasks based on a state variable, the maximum duration of Task\_RGB is 1/3 of previous implementations. Therefore, the longest delay would be:

405ms

1. Describe the sequence of events that leads to maximum delay between releasing the switch and seeing the LED sequence through RGB colors. Calculate the value of that delay.

The maximum delay would occur when the switch is released immediately after the Task\_Flash begins execution:

1. (Disregarded per instructions)

# Chapter 4

11. We wish to enable IRQ13 but disable IRQ24. What values need to be loaded into which register bits, and what is the sequence of CMSIS calls to accomplish the same?

To enable IRQ13, a ‘1’ value needs to be written into the 13th bit of the Interrupt Set Enable Register (ISER) register. To disable IRQ24, a ‘1’ value needs to be written into the 24th bit of the Interrupt Clear Enable Register (ICER register). This can also be achieved by the following sequence of CMSIS calls:

1. // Enable IRQ13 aka “UART\_IRQn”
2. NVIC\_EnableIRQ(UART1\_IRQn);
3. // Disable IRQ24 aka “USB0\_IRQn”
4. NVIC\_DisableIRQ(USB0\_IRQn);

12. We wish to determine if IRQ7 has been requested. Which register and which bit will indicate this? What is the CMSIS call that will reveal the information?

To determine if IRQ7 has been requested, we need to examine the 7th bit of the Interrupt Set Pending Register (ISPR). This can also be achieved by the following CMSIS call:

1. // Determine if interrupt 7 aka “LLWU\_IRQn” has been
2. // requested
3. NVIC\_GetPendingIRQ(LLWU\_IRQn)

13. Which register can an exception handler use to determine if it is servicing exception number 0x21? What value will the register have? What is the CMSIS interface code to read the IPSR?

An exception handler can check the Interrupt Program Status Register (IPSR) to determine which exception number it is servicing. Bits 0 through 8 indicate the exception number, so for exception number 21, the value of the register will be 0x00000015. This value can be read by the following CMSIS call:

1. // Get value of IPSR register
2. uint32\_t exceptionNum = \_\_get\_IPSR();

# Chapter 6

2. Consider an 8-bit ADC with a reference voltage of 2.7V operating in singled-ended mode. What input voltage range will lead to an output code of 0x34?

1. Consider a 10-bit DAC with reference voltage of 2.7V. Given that the input code is 0x104, what is the output voltage?

8. How would you configure the comparator in the KL25Z to trigger whenever the input voltage rises above 2.0V? Assume the reference voltage is 3.3V.

Calculate VOSEL:

1. // Assign clock to comparator peripheral
2. SIM->SCGC4 |= SIM\_SCGC4\_CMP\_MASK;
4. // Enable Comparator
5. CMP0->CR1 = CMP\_CR1\_EN\_MASK;
7. // Select Input Channels (CMP DAC is channel 7)
8. CMP0->MUXCR = CMP\_MUXCR\_PSEL(5) | CMP\_MUXCR\_MSEL(7);
10. // Enable DAC, set output voltage to 3.3V (38)
11. CMP0->DACCR = CMP\_DACCR\_DACEN\_MASK | CMP\_DACCR\_VOSEL(38);
13. // Enable Interrupt for comparator on both edges
14. CMP0->SCR = CMP\_SCR\_IEF\_MASK | CMP\_SCR\_IER\_MASK;

# References

|  |  |
| --- | --- |
| [1] | A. Dean, Embedded Systems Fundamentals with Arm Cortex-M based Microcontrollers, Cambridge: Arm Education Media, 2017. |