University of South-Eastern Norway

Faculty of Technology, Natural Sciences and Maritime Sciences
Campus Porsgrunn

Industrial IT and Automation (IIA)

Course IIA2017
Industrial Information Technology

Multitasking and real-time assignment (GuiSw)
Version 1.0

Nils-Olav Skeie

March 22, 2021

Multitasking and real-time assignment tasks descriptions

1.1 Introduction

This assignment contains a combination of theory, practical exercises using both a .NET application and developing your own .NET application, and time requirement calculation regarding time requirements in a multitasking system. The assignment is using a downloadable application 1) giving a set of theory exercises, 2) running a set of threads (tasks) and 3) giving a set of specifications for different real-time operating systems (RTOS). Use the knowledge from the multitasking and real-time system section in this course to evaluate and solve these assignment tasks. The goals of this assignment is to use the theory to understand 1) the running times of tasks in an application when sharing one or several resources, 2) how to develop a simple multitasking application sharing one or several resources, and 3) how to estimate any time requirements for a process.

The results of the assignment must be documented in a technical report, delivered as a PDF file in *the Learning Management System (LMS),* within the deadline (due date), and must be approved to pass this course. At least 60% of the tasks must be solved to get the assignment approved. Practice in writing a technical report with the introduction, theory, results and conclusion parts for the assignment with references to the different exercises. Remember to deliver within the due time. If an extension is needed, ask BEFORE the due time. This assignment is a soft real-time task!

The usage of the .NET application is described in section A. See Figure A.1 for Scheduler Setup tab page of the application. The version history of this document:

Version	**Description**	
0.1	First version, include console application and RTOS specification page.	NOS−08
0.2	Extend document and application with GUI, runtime window and real-time evaluation.	NOS−10
0.3	Extend document and application configuration and analysis part.	NOS−12
0.4	Extend document and application with theory section.	NOS−14
0.5	Extend document with more analysis, pseudo coding and development of C# application.	NOS−17
0.6	Minor adjustment for some of the task descriptions.	NOS−17
0.7	Simplifying the task sections.	NOS−18
0.8	Add more hints for the evaluation of the real-time system.	NOS−19
0.9	Minor adjustments and updates for the assignment report.	NOS−20
1.0	Minor adjustments and updates for the assignment report.	NOS−21

1.2 Report Introduction (1%)

Use the template available in the LMS. You can use Latex and the book template with the first page most similar to the existing template. **Make a screen dump of the Scheduler Setup tab page for the introduction section of your report**. This screen dump is needed to be able to evaluate your answers. The screen dump must be made of the main screen, after the setup section is performed. Use figure text, references to the figures in your text, and add references to support your statements. A technical report must contain several references, at least this task description and may be the course lecture document (Skeie n.d.).

An introduction informs the reader what the report is about and sets the project in a wider context. The introduction should also provide the background information that the reader needs to understand the topic in the report. The introduction should 1) short introduce the context of the topic, 2) explain the problem and/or motivation for the topic, 3) aims and purpose of the project, and 4) briefly outlines the report structure (not necessary in this report).

1.3 Theory (20%)

Use the Theory Exercise tab for the theory part. **Start this section with a screen dump of the application with the Theory Exercise tab activated.** Please use your own words in answering, a good training for the real-life as well as the final test. When copy any information, use references.

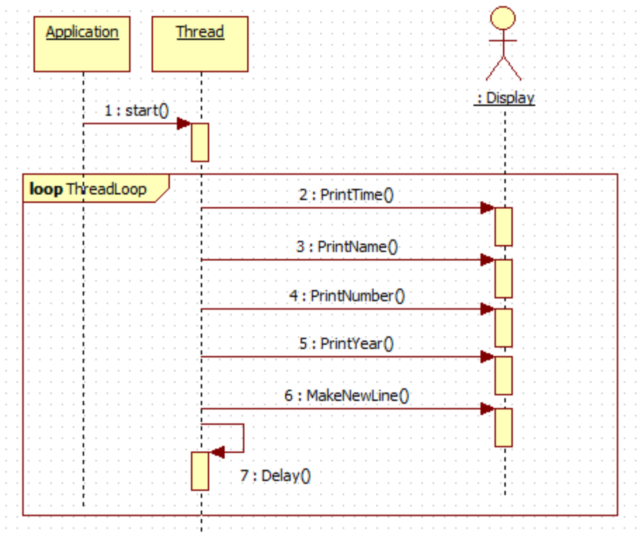


Figure 1.1: A System Sequence Diagram (SSD) for the application.

These exercises are according to the “Theory Exercise Tab” in the application. Limit your answers to maximum six sentences for each exercise. The tasks should be:

- make an image of the application with the "Theory Exercise" tab activated.
- make your own answers for section #1, #2, #3, and #4. Some of these questions may be the same.
- This section MUST contain references to support your statements.

1.4 Evaluation of a multitasking system (33%)

Use the Running Tasks tab page for these exercises. Base your evaluation(s) and assumption(s) that the application should be used in a multitasking system with real-time requirements. The application is running in the Windows environment and using the scheduler in your Windows system. Each real-time thread (task) performs a fixed number of loops and in each loop the thread (task) will print some information starting with $[Tx]$ where x is the thread (task) ID. The goal in this part is to evaluate the real-time performance of the system. The real-time performance for a system, or a task, is always about the total time, the maximum time, for execution of the system or the task, and this time depends on the type and number of resources used by the system or the task. Document the following tasks:

- Analyze the system based on the multitasking and real-time requirements, with focus on this application only. A System Sequence Diagram (SSD)[1] (Fowler & Scott 1997) of the application is shown in Figure 1.1. The real-time requirements will be any factor that may affect the running time of the tasks (threads). The real-time requirements involve tasks running in parallel, any common resources used by the application that will affect the timing and the time used by the application. **Hint:** the application consists of several tasks sharing one or several resources. Evaluate these common resources disregarding cpu and computer memory. Is the running time affected by sharing common resources? How to share such common resources in a multitasking system?
- Select the running code 0 in the Scheduler Setup tab page and switch to the Running Tasks tab page. Use the Run button to start the scheduler, and the tasks. Use the *Clear* button if the *Run* button is not activated. All tasks should stop after approximately 5 to 50 seconds. Include two screen dumps of the output from your application in the report, one screen dump for the start of the threads and a screen dump of the end. Include a discussion of the number of loops, the delays in the loop and running time for each thread.

[1]SSD is a type of UML (Unified Modeling Language) sequence diagram used in the analysis steps of a system (system, hardware and/or software).

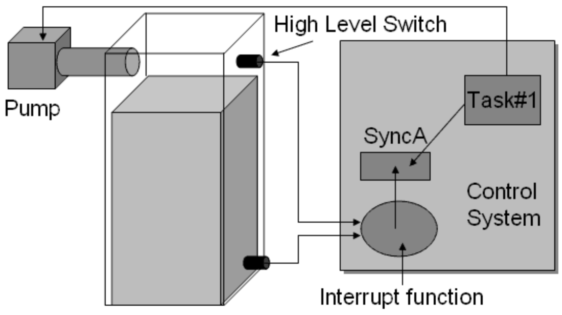


Figure 1.2: A sketch of the control system for the liquid vessel with the pump, two switches and the control system.

- Select the running code 1 in the Scheduler Setup tab page and switch to the Running Tasks tab page. Use the Run button to start the scheduler, and the tasks. Use the *Clear* button if the *Run* button is not activated. All tasks should stop after approximately 5 to 50 seconds. Include two screen dumps of the output from your application in the report, one screen dump for the start of the threads and a screen dump of the end. Include a discussion of the number of loops, the delay in the loops and running time for each thread.
- Select the running code 2 in the Scheduler Setup tab page and switch to the Running Tasks tab page. Use the Run button to start the scheduler, and the tasks. Use the *Clear* button if the *Run* button is not activated. All threads should stop (hang) after a couple of seconds. Use the Process Status application of your operating system to show the status of the application running when in running code 2. Include the columns for CPU time and threads for in the Process Status application. The Process Status application will show more threads then your threads as Windows will add extra threads to handle a console applications like inputs from all the threads. **Hint:** Run the Process Status application several times to check the usage of cpu time. Discuss the results for your application, including a screen dump from the Process Status application. Why is the threads not finishing?
- Make a short conclusion of the results of this system based on 1) the analysis done in the first exercise in this evaluation part, 2) the running time for each threads in code 0 and code 1 and 3) why the threads in code 2 are not finishing. **Hint:** Focus on multitasking and real-time requirements.

1.5 Development of a multitasking system (25%)

Develop your own multitasking application based on your knowledge from the previous parts. Base your application on the source code in the Real-Time "Programming" section in the compendium and use your analysis from the previous part to extend the application. Include any semaphores (or mutex) for a secure usage of any common resources, based on the analysis. A semaphore is defined in C# by making a "Semaphore" variable, and use the methods WaitOne() and Release() to access the semaphore services. Include your name, student number and the semester year in the display information. **Hint:** Sleep(0) statements can be used to test the sharing of common resources. An advice will be to use a console application as using several tasks to write into the same text window in a GUI application will complicate the coding (must include an invoke() statement). Document the following tasks:

- Use pseudo coding describing the functions of the tasks in your system.
- Develop a multitasking application in C# according to the "Scheduler Setup" tab.

1.6 Time requirements for a real-time system (20%)

You are going to select a real-time operating system for your control system running on an one core cpu system. The control system is shown in Figure 1.2.

The system consists of a vessel with liquid and a pump for filling the vessel. The control system has no information about the input or output flow of the vessel. The vessel should **never** be overfilled nor empty, giving that a hard real-time system should be used. Two switches are used, one high level switch and one low level switch. The switches are connected to an external interrupt signal with the highest interrupt priority level in your system, except the timer controlling the scheduler. The system needs to perform maximum 20 instructions in the interrupt function. A synchronization mechanism is used to inform the control task, task#1, to control the pump. The task #1 needs to perform between 150 and 200 instructions before the state of the pump can be changed. The state change delay of the pump is maximum 200 ms. Task #1 will be the only task at the highest priority running level. The pump must be turned off within 250 ms after an active signal from any of the level switches. You can choose between 4 different real-time system with the specification as listed in the *RTOS Specification* tab page in the application. The system will have several other tasks (task#2 - task#N) running on lower priority levels. The tasks should be:

- **Include a screen dump of the *RTOS Specification* tab page in your report.**
- Which operating system (OS) can/will you choose for your control system? **Discuss your answer including a timing diagram of the system!**

1.7 Report conclusion (1%)

Make a conclusion where you summarize the main points or goals of the tasks with focus on your most important findings and/or results. Do not add any new information in the conclusion. Do not focus on the details, focus more on the overview "picture" and examine the greater significance of what you have done. Leave your readers with something to think about.

Appendix A

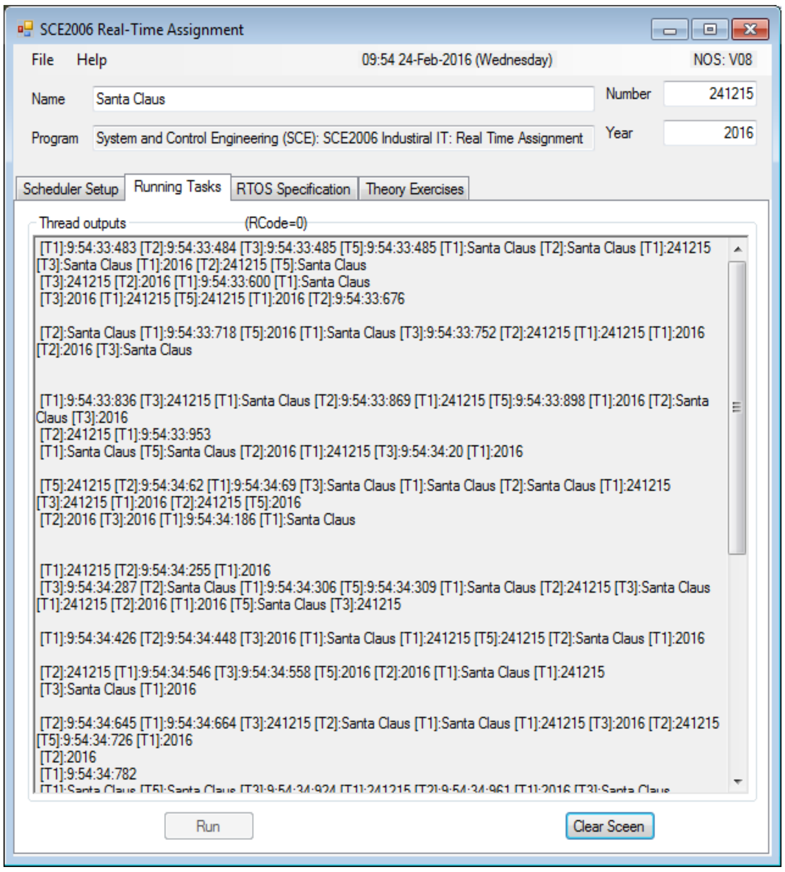
Usage of the application

Download and start the application, fill in your name, student number and semester year (four numbers), and press the “Get System Information” button (startup section). The application contains a configuration part, a simple scheduler, and real-time operating system specifications. The configuration part is using your name, your student number and semester year for loading a set of data for the scheduler and the real-time operating system specifications. A screen dump of the main page, after pressing the “Get System Information” should be included in the introduction of your report.

Use the tab page selection to select the different part of the application. These tabs contains among others information about:

- application setup and configuration,
- showing information from a set of running tasks,
- specification of the different RTOS that can be used in a control system,
- theory section with a set of questions to be answered/explained, with references to support your statements.

Figure A.1 shows the *Running Tasks* tab page running the tasks specified in the *Scheduler Setup* tab page.



09:54 24-Feb-2016 (Wednesday)

Name: Santa Claus

Number: 241215

Program: System and Control Engineering (SCE): SCE2006 Industiral IT: Real Time Assignment

Year: 2016

Thread outputs (RCode=0)

```
 [T1]:9:54:33:483 [T2]:9:54:33:484 [T3]:9:54:33:485 [T5]:9:54:33:485 [T1]:Santa Claus [T2]:Santa Claus [T1]:241215
[T3]:Santa Claus [T1]:2016 [T2]:241215 [T5]:Santa Claus
 [T3]:241215 [T2]:2016 [T1]:9:54:33:600 [T1]:Santa Claus
 [T3]:2016 [T1]:241215 [T5]:241215 [T1]:2016 [T2]:9:54:33:676

 [T2]:Santa Claus [T1]:9:54:33:718 [T5]:2016 [T1]:Santa Claus [T3]:9:54:33:752 [T2]:241215 [T1]:241215 [T1]:2016
[T2]:2016 [T3]:Santa Claus


 [T1]:9:54:33:836 [T3]:241215 [T1]:Santa Claus [T2]:9:54:33:869 [T1]:241215 [T5]:9:54:33:898 [T1]:2016 [T2]:Santa
Claus [T3]:2016
 [T2]:241215 [T1]:9:54:33:953
 [T1]:Santa Claus [T5]:Santa Claus [T2]:2016 [T1]:241215 [T3]:9:54:34:20 [T1]:2016

 [T5]:241215 [T2]:9:54:34:62 [T1]:9:54:34:69 [T3]:Santa Claus [T1]:Santa Claus [T2]:Santa Claus [T1]:241215
[T3]:241215 [T1]:2016 [T2]:241215 [T5]:2016
 [T2]:2016 [T3]:2016 [T1]:9:54:34:186 [T1]:Santa Claus


 [T1]:241215 [T2]:9:54:34:255 [T1]:2016
 [T3]:9:54:34:287 [T2]:Santa Claus [T1]:9:54:34:306 [T5]:9:54:34:309 [T1]:Santa Claus [T2]:241215 [T3]:Santa Claus
[T1]:241215 [T2]:2016 [T1]:2016 [T5]:Santa Claus [T3]:241215

 [T1]:9:54:34:426 [T2]:9:54:34:448 [T3]:2016 [T1]:Santa Claus [T1]:241215 [T5]:241215 [T2]:Santa Claus [T1]:2016

 [T2]:241215 [T1]:9:54:34:546 [T3]:9:54:34:558 [T5]:2016 [T2]:2016 [T1]:Santa Claus [T1]:241215
 [T3]:Santa Claus [T1]:2016

 [T2]:9:54:34:645 [T1]:9:54:34:664 [T3]:241215 [T2]:Santa Claus [T1]:Santa Claus [T1]:241215 [T3]:2016 [T2]:241215
[T5]:9:54:34:726 [T1]:2016
 [T2]:2016
 [T1]:9:54:34:782
 [T1]:Santa Claus [T5]:Santa Claus [T3]:9:54:34:924 [T1]:241215 [T2]:9:54:34:961 [T1]:2016 [T3]:Santa Claus
```

Figure A.1: The *Running Tasks* tab page showing the running of the tasks specified in the *Scheduler Setup* tab page.

Bibliography

Fowler, M. & Scott, K. (1997), *UML Distilled: Applying the Standard Object Modeling Language*, Addison-Wesley, USA.

Skeie, N.-O. (n.d.), 'IIA2017: Industrial information technology', Lecture notes for the master course IIA2017 at the University College of Southeast Norwat (USN) (and SCE2006 at the Telemark University College (TUC)). A master course for data communications, field buses, OPC, process databases,embedded systems, mechatronics, and real-time systems for industrial systems.