System Verification

Tasks

- Note: all tasks execution time is calculated from the actual implemented tasks using GPIOs and the logic analyzer.

Name	Periodicity (ms)	deadline (ms)	excution Time (ms)
1. Button_1_Monitor	50	50	0.008
1. Button_2_Monitor	50	50	0.008
3. Periodic Transmitter	100	100	0.0096
4. Uart Receiver	20	20	0.017
5. Load 1 Simulation	10	10	5
8. Load 2 Simulation	100	100	12

1. Using Analytical Method

a. System Hyperperiod

- It's the Least Common Multiple of all tasks periods
- H = LCM(50, 50, 100, 20, 10, 100) = 100

b. CPU Load

- U = (E1 + E2 + E3 + E4 + E5 + E6) / H
  where E is the Execution time and H is the Hyperperiod.
- U = (0.008*2 + 0.008*2 + 0.0096 + 0.017*5 + 5*10 + 12) / 100 = 0.621 (62.1 %)

c. System schedulability check using URM and Time Demand Analysis Techniques

1. Rate-Monotonic utilization bound

$$\sum_{i=1}^{n}\frac{C_i}{P_i} <= n(2^{\frac{1}{n}}-1)$$

$$L.H.S = \sum_{i=1}^{n}\frac{C_i}{P_i} = \frac{0.008}{50} + \frac{0.008}{50} + \frac{0.0096}{100} + \frac{0.017}{20} + \frac{5}{10} + \frac{12}{100} = 0.621$$

$$R.H.S = n(2^{\frac{1}{n}}-1) = 6(2^{\frac{1}{6}}-1) = 0.735$$

Since L.H.S <= R.H.S then the system is scheduable.

2. Time Demand Analysis

- First We Sort the tasks making the highest priority at the first. And since we are using Rate-Monotonic Scheduler the smaller the periodicity the higher the priority.

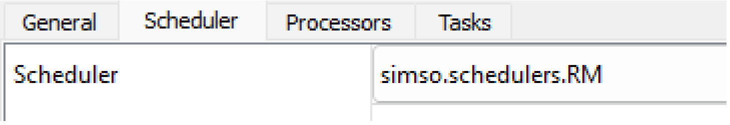
Name	Periodicity (ms)	deadline (ms)	excution Time (ms)
5. Load 1 Simulation	10	10	5
4. Uart Receiver	20	20	0.017
1. Button_1_Monitor	50	50	0.008
2. Button_2_Monitor	50	50	0.008
3. Periodic Transmitter	100	100	0.0096
8. Load 2 Simulation	100	100	12

- Choose the critical instant 0 then:
  $W_1(10) = 5 + 0 = 5 < deadline$
  $W_4(20) = 0.017 + \frac{20}{10} * 5 = 10.017 < deadline$
  $W_1(50) = 0.008 + \frac{50}{10} * 5 + \frac{50}{20} * 0.017 = 25.059 < deadline$
  $W_2(50) = 0.008 + \frac{50}{10} * 5 + \frac{50}{20} * 0.017 + \frac{50}{50} * 0.008 = 25.067 < deadline$
  $W_3(100) = 0.0096 + \frac{100}{10} * 5 + \frac{100}{20} * 0.017 + \frac{100}{50} * 0.008 + \frac{100}{50} * 0.008 = 50.1266 < deadline$
  $W_4(100) = 12 + \frac{100}{10} * 5 + \frac{100}{20} * 0.017 + \frac{100}{50} * 0.008 + \frac{100}{50} * 0.008 + \frac{100}{100} * 0.0096 = 62.1266 < deadli$
- Since all tasks are less than the deadline. The system is scheduable.

---

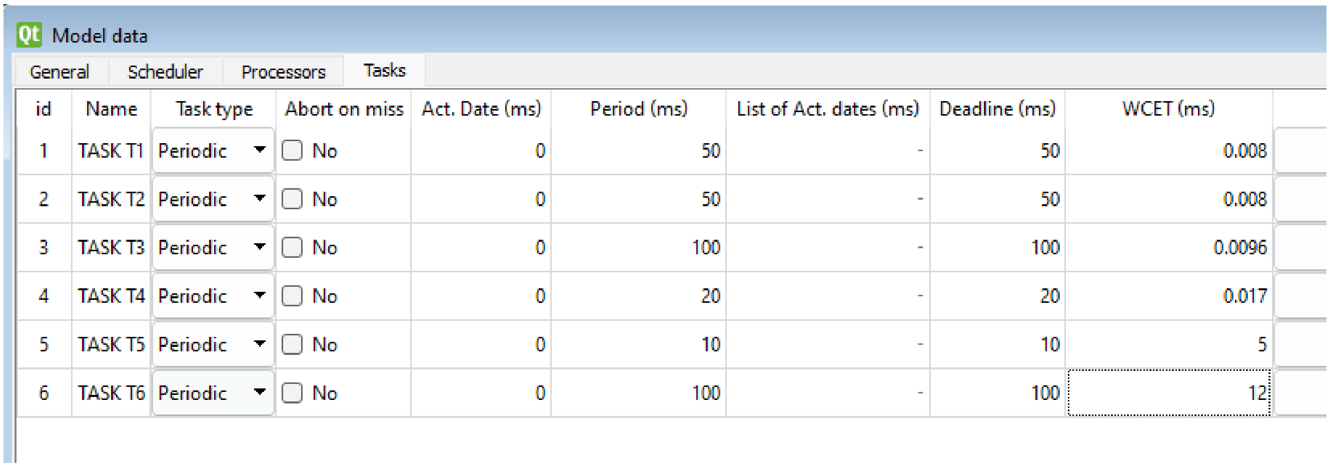
2. Using SIMSO offline simulator:

- Scheduler used: Fixed priority rate monotonic as required.



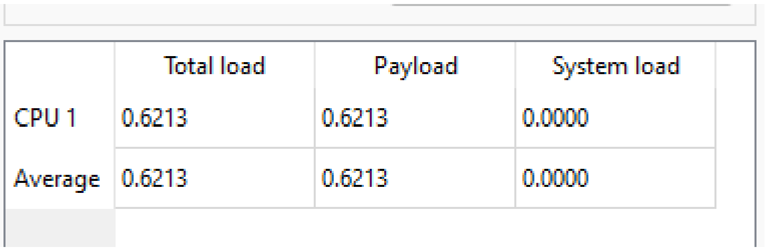
Scheduler simso.schedulers.RM

- Tasks



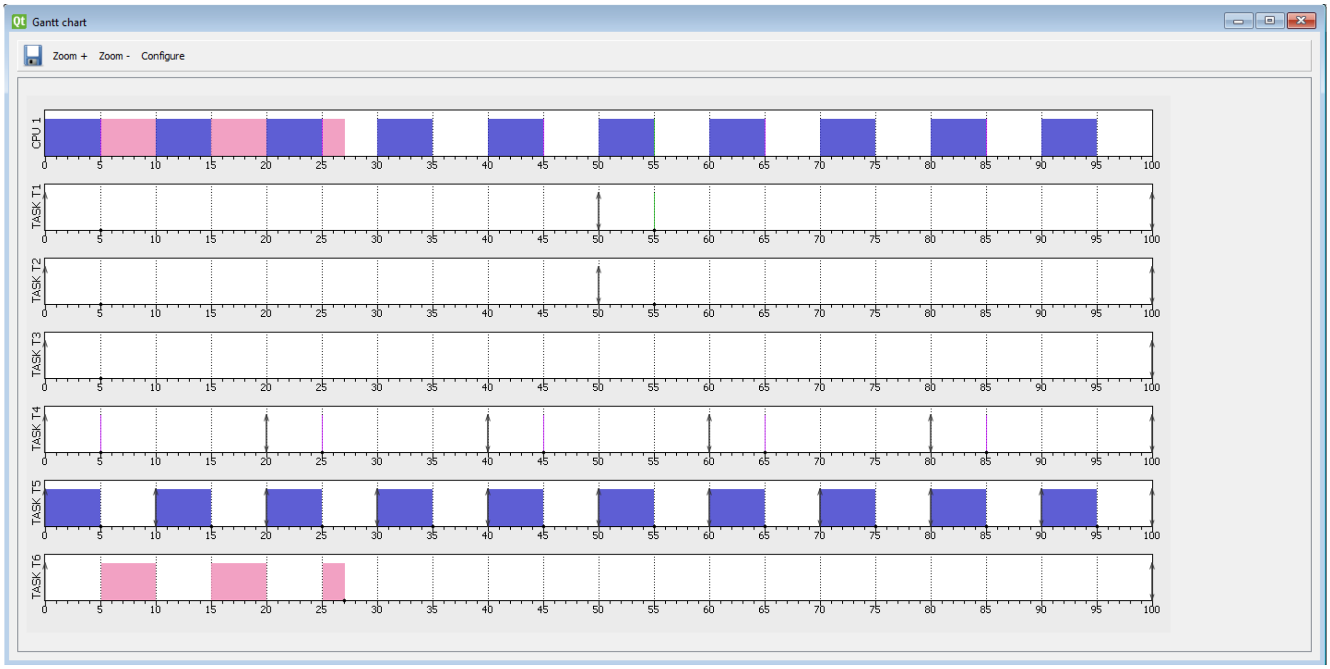
id	Name	Task type	Abort on miss	Act. Date (ms)	Period (ms)	List of Act. dates (ms)	Deadline (ms)	WCET (ms)
1	TASK T1	Periodic	☐ No	0	50	-	50	0.008
2	TASK T2	Periodic	☐ No	0	50	-	50	0.008
3	TASK T3	Periodic	☐ No	0	100	-	100	0.0096
4	TASK T4	Periodic	☐ No	0	20	-	20	0.017
5	TASK T5	Periodic	☐ No	0	10	-	10	5
6	TASK T6	Periodic	☐ No	0	100	-	100	12

- The CPU load is the same as calculated in the analytical method.

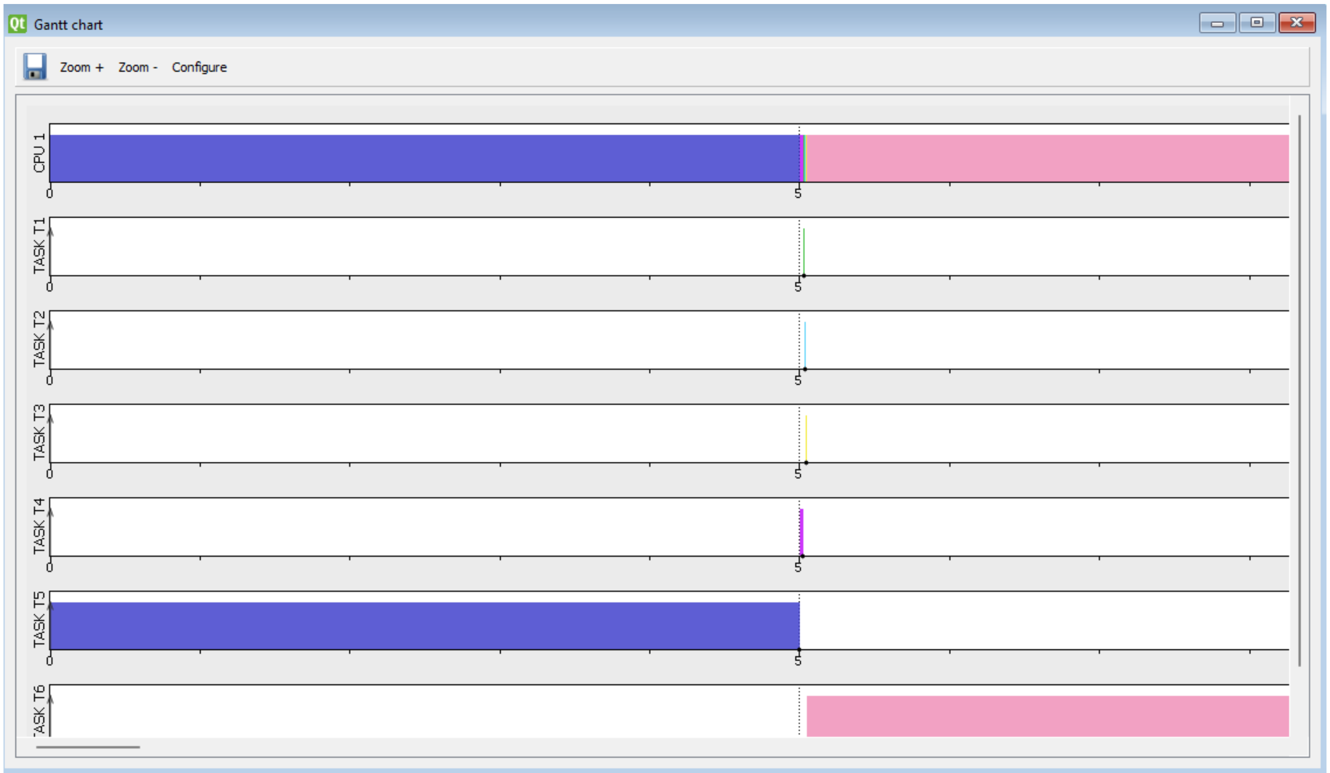


	Total load	Payload	System load
CPU 1	0.6213	0.6213	0.0000
Average	0.6213	0.6213	0.0000

- Gnatt chart over the Hyperperiod

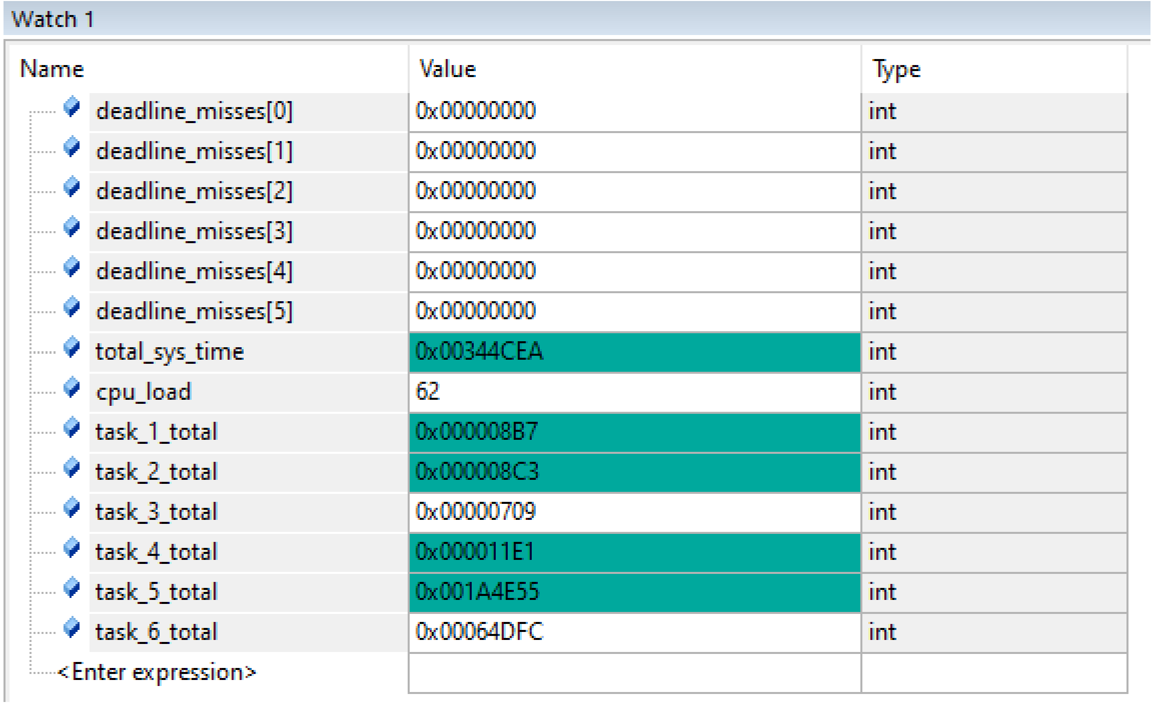


- zoomed in



3. Using Keil Simulator At Runtime

1. Calculate the CPU usage time using timer 1 and trace macros

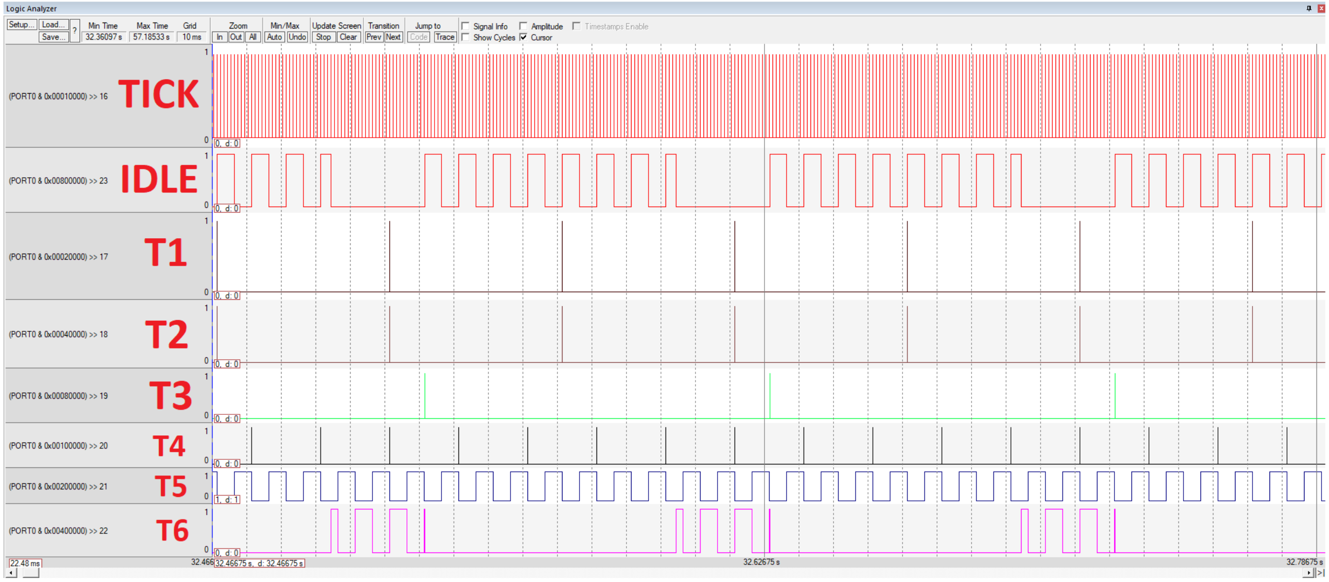

Watch 1

Name	Value	Type
deadline_misses[0]	0x00000000	int
deadline_misses[1]	0x00000000	int
deadline_misses[2]	0x00000000	int
deadline_misses[3]	0x00000000	int
deadline_misses[4]	0x00000000	int
deadline_misses[5]	0x00000000	int
total_sys_time	0x00344CEA	int
cpu_load	62	int
task_1_total	0x000008B7	int
task_2_total	0x000008C3	int
task_3_total	0x00000709	int
task_4_total	0x000011E1	int
task_5_total	0x001A4E55	int
task_6_total	0x00064DFC	int
<Enter expression>		

- Note: The `cpu_load` is the same as the on obtained using the analytical method and using the SIMSO offline simulator

- Note: None of the tasks miss the deadline!

2. Using trace macros and GPIOs, plot the execution of all tasks, tick, and the idle task on the logic analyzer


Logic Analyzer

Min Time 32.36097 s
Max Time 57.18533 s
Grid 10 ms

- Note: The above chart was using the implemented EDF scheduler. As you can see tasks with closer deadline preempts other tasks.
- Note: The IDLE task never interfered with my other main tasks so my modification in the IDLE task function to make sure it never preempts my main tasks is successful. As I made sure It always offsets the maximum main tasks deadline by a user defined amount So It never executes unless there are no more tasks in the ready queue.