IN4343 – Lab 2

Casper van Wezel (4209192) & Erwin de Haan (4222814) – 2017-03-11

# Part 1

## Overhead of timer handler

Hypothesis: The time will increase linearly as a function of NUMTASKS, due

1. This data was produced using a script the takes the average for every period of a time of 5 seconds. This is more than five thousand periods.

|  |  |
| --- | --- |
| **NUMTASKS** | **Time (us)** |
| 1 | 41,3 |
| 2 | 120 |
| 3 | 157 |
| 4 | 196 |
| 5 | 232 |
| 6 | 271 |
| 7 | 308 |
| 8 | 345 |
| 9 | 381 |
| 10 | 420 |
| 15 | 607 |
| 20 | 793 |
| 24 | 942 |
| 25 | 980 |

2. The period of the timer is 976,5 us. So 25 tasks is the threshold value (see table above).

3. The compiler will optimize the loop out of the final machine code. This means a lot of code can be dropped for NUMTASKS = 1

4. Make the NUMTASKS equal to the number of registered tasks. In the case of Tst1 3. Change the clock ticks per second to 2 and divide the periods of the tasks by 512.

## Event latency

5.

This data was produced use a script that measured over 11 events the delay between the intr\_num rising edge and the “green” rising edge. The CountDelay was removed and only the Green task was registered.

|  |  |  |
| --- | --- | --- |
| **PRIORITY** | **Time (us)** | **Time (us) Lower** |
| 0 | 261 | 51 |
| 1 | 224 | 51 |
| 2 | 187 | 47 |
| 3 | 149 | 51 |
| 4 | 112 | 47 |

An “empty” interrupt handler takes 250 us, this was measured in the same way as for question one, but one task was registered. The ISR with one Task activation was measured to take 297 or 301 us.

6.