Annals of DAAAM for 2010 & Proceedings of the 21st International DAAAM Symposium, Volume 21, No. 1, ISSN 1726-9679
ISBN 978-3-901509-73-5, Editor B. Katalinic, Published by DAAAM International, Vienna, Austria, EU, 2010
Make Harmony Between Technology and Nature, and Your Mind will Fly Free as a Bird
DAAAM Symposium

EXPERIMENTAL PERFORMANCE MEASURING ON REAL-TIME OPERATING SYSTEMS FOR EMBEDDED DEVICES

PAVLAS, L[ibor]

Abstract: *The goal of this paper is to show the results of measuerement of performance on real time operating systems and to evaluate real time operating systems for microcontrollers by speed of interacting to various input signals, with respect to different load during various situations. The measurement method is based on measuring time difference between input rising edge and output rising edge. This paper also evaluates operating systems by their properties.*
Key words: *RTOS, microcontrollers, operating systems, performance*

1. INTRODUCTION

There are a couple of real-time operating systems for microcontrollers and there is a question, which is better for what kind of problem (Ganssle & Barr, 2009).

In my work I focused on measuring delays in real-time operating systems in predefined situations. In this paper I will describe, how the measure was made and what was measured.

2. REAL TIME OPERATING SYSTEMS

There are a couple of real-time operating systems for microcontrollers, one of the most known operating system is uC/OS-II from STMicroelectronics (Labrosse, 1999). This operating system is free for non-commercial use (Labrosse, 2002). The next well known real-time operating system is uC/OS-III also from STMicroelectronics, also free for non-commercial use (Labrosse, 2009).

Next operating system, which I tested, is FreeRTOS from Real Time Engineers Ltd. It is open source project and if you do not need support, it is for free.

FreeRTOS can be downloaded from internet for free, you can develop both non-commercial and commercial products, but without any warranty and support. If you want any of this, you have to buy commercial product OpenRTOS and for critical use SafeRTOS.

There are other RTOS (Real-time operating systems) for microcontrollers, but for our research I used these three, because of availability of all ports, necessary for compiling to machine code for our destination hardware.

3. USED EQUIPMENT

For my research I used Micrium STM32CMICOS-EVAL target board with ARM microcontroller STM32F107, oscilloscope Tektronix TDS 220. For program development, uploading and debugging was used IAR Embedded Workbench Evaluation 5.41.

4. EXPERIMENTAL MEASUREMENT

Measuring was done by two channel oscilloscope Tektronix TDS 220. One channel was for measuring time, when the input subject occurred and second channel measured delay, when microprocessor answered correctly to input subject.

There were five types of measurement on each RTOS:

- Delay in answer to change of input pin, programmed as a one task made as an endless loop.
- The same problem as above, but RTOS has one more running task.
- The same problem as first, but there is one more task and IRQ handler interacting on the same input pin as an endless loop.
- The same problem as a first, but there is an IRQ handler interacting on the same input pin as an endless loop.
- Delay in Answer to an IRQ.

Every experiment was measured for couple of minutes with input signal period 10 [Hz], every response was stored in oscilloscope and we read minimum and maximum and calculated average value. The results of our experiment are in table 1, 2, 3.

Measured task	Running tasks			Rising edge			Falling edge		
	Endless loop	Another task	IRQ handling	min	average	max	min	average	max
Endless loop	Y	Y	Y	13,8	14,1	14,3	13,8	14,1	14,3
Endless loop	Y	N	Y	13,8	14,1	14,5	13,8	14,15	14,5
Endless loop	Y	Y	N	1,24	1,48	1,73	1,25	1,495	1,74
Endless loop	Y	N	N	1,23	1,23	1,23	1,19	1,44	1,7
IRQ	Y	Y	Y	5,76	5,82	5,88	5,8	5,86	5,92

Tab. 1. Measured delays in uC/OS-III (measured in [ms])

Measured task	Running tasks			Rising edge			Falling edge		
	Endless loop	Another task	IRQ handling	min	average	max	min	average	max
Endless loop	Y	Y	Y	2,6	2,75	2,9	2,6	2,73	2,86
Endless loop	Y	N	Y	2,52	3,15	3,78	2,52	3,27	4,02
Endless loop	Y	Y	N	0,53	0,63	0,73	0,56	0,6	0,76
Endless loop	Y	N	N	0,55	0,66	0,77	0,55	0,66	0,768
IRQ	Y	Y	Y	1,13	1,16	1,2	1,15	1,17	1,2

Tab. 2. Measured delays in uC/OS-II (measured in [ms])

Measured task	Running tasks			Rising edge			Falling edge		
	Endless loop	Another task	IRQ handling	min	average	max	min	average	max
Endless loop	Y	Y	Y	5,24	7,28	9,32	5,2	7,22	9,24
Endless loop	Y	N	Y	5,2	6,7	8,2	5,3	6,75	8,2
Endless loop	Y	Y	N	1,08	1,96	2,84	1,06	1,77	2,48
Endless loop	Y	N	N	1,06	2,24	3,42	1,08	1,75	2,42
IRQ	Y	Y	Y	2,02	2,24	2,46	2,06	2,185	2,31

Tab. 3. Measured delays in freeRTOS (measured in [ms])

From tables you can see that the slowest answer is in the case, when you have to react to input signal using endless loop and there is also running IRQ handler, handling the same signal. This is because, when the change of this signal is done, first microcontroller has to run IRQ handler subroutine, store and restore its registers to stack and after this can be executed the instructions in endless loop. On the other hand, when there is only endless loop running, without any other tasks and IRQ handler, it is much faster, then IRQ handler.
Reason, why endless loop is faster in response, is because when you call interrupt, you have to store registers and program counter (return address) to stack and change program counter. But on the other hand, interrupt handler has small differences in delay, because when the interrupt occurs, microcontroller has to do always almost the same operations, does not matter what kind of operation was done before. There are only few things, what can change the delay of interrupt answer, those are another interrupts, occurring before, or during our interrupt and DMA transfers.

5. MEASURED GRAPHS

The results of the measurement can be seen at graphs 1, 2 and 3. The darkest part of bar graph is the minimum time, needed to respond, the lighter parts of bar graph are average and maximal value of time, needed to respond to input signal.

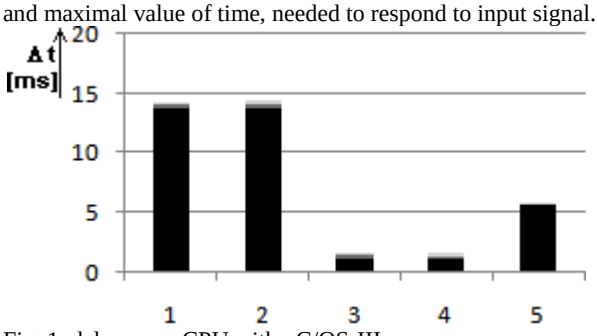

Fig. 1. delays on µCPU with uC/OS-III

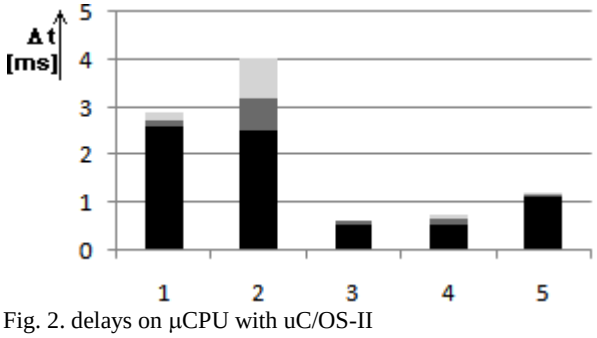

Fig. 2. delays on µCPU with uC/OS-II

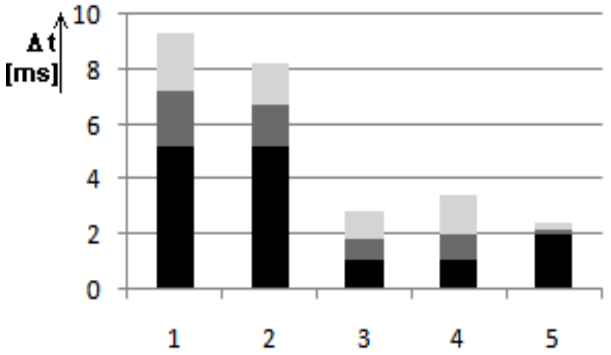

Fig. 3. delays on µCPU with FreeRTOS

The first bar is measurement with endless loop with IRQ and another task, second is endless loop with IRQ and without second task. Third is loop with only second task and fourth bar is only the loop running without anything else. The last bar is reply on IRQ, when other two tasks are running.

6. CONCLUSION

During our experiment, we compared three real time operating systems and we can see those things:

- RTOS uC/OS-III has the smallest ratio between minimal and maximal time needed to make respond.
- uC/OS-II has the fastest respond to input signal.
- uC/OS-III is unfortunately the slowest operating system, with this target board.
- When we used FreeRTOS, we get minimal difference, between respond, when we used endless loop and IRQ.

If there is a decision, which operating system we should use, we have to ask, how we will use it. If we need software for free, with middle-fast time response, we should use FreeRTOS. If speed is our primary requirement, we should choose uC/OS-II and if we need also support in future, there is a uC/OS-III, which is new generation of uC/OS-II, so we can assume that there will be great support from Micrium.

7. ACKNOWLEDGEMENTS

Many thanks to my colleagues at University of West Bohemia: Petr Kratochvil, Stepan Jenicek and Richard Lipka. Many thanks go also to my advisor Doc. Ing. Vlastimil Vavricka, CSc., for his continuous support in the Ph.D. program.

8. REFERENCES

Ganssle, J. & Barr, M. (2009). *Embedded Systems Dictionary*, CMP Books, 1578201209, Gilroy CA, USA

Labrosse, J.J. (1999). Embedded Systems Building Blocks, Second Edition: Complete and Ready-to-Use Modules in C, CMP Books Distribution Center, 978-0879306045, Gilroy CA, USA

Labrosse, J.J. (2002). *uC/OS-II The Real-Time Kernel, second edition*, CMP Books Distribution Center, 1-57820-103-9, Gilroy CA, USA

Labrosse, J.J. (2009). *uC/OS-III The Real-Time Kernel*, Micrium press, 978-0-9823375-3-0, Weston FL, USA

http://www.freertos.org/ - The FreeRTOS Project, *Accessed on: 2010-05-20*

http://micrium.com/page/home - Micrium RTOS and Tools, *Accessed on: 2010-04-15*