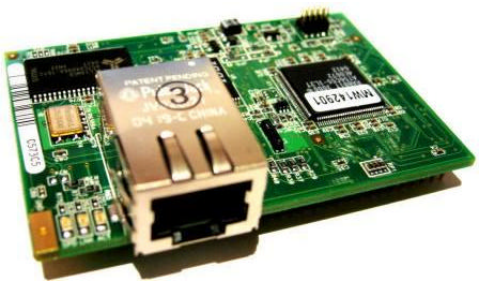
TN001

RABBIT micro controller

Extension of system timer clock to ms resolution
Technical note



Authors: Gerhard Burger
Version: 1.0
Last update: 07.11.2005
File:
Attachments: no attachments
Current versions available on: www.burger-web.com

Table of versions

Version	Date	Remarks
1.0	6.1.2005	First version
		Please check internet for updates www.burger-web.com

1 Overview

This document outlines how to modify the ZWorld library vdriver.lib to implement a system timer clock with ms-resolution. The description starts with a short problem description, shows an easy example and outlines the necessary modification step by step.

2 Introduction

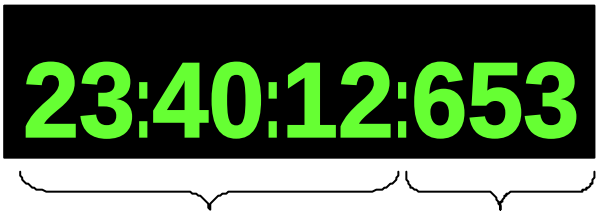
Some times it can be useful in a project to have an exact clock e.g. for capturing data with a time stamp. Having the Rabbit Micro Controller, a RTC with a resolution of 1 second is provided. Beyond that some timer variables as ms_timer, s_timer etc. are available with the Z-World libraries but what happens if an exact time stamp is needed. **The modification can be done in around 30 Minutes.**

2.1.1 The goal

The goal of this patch is to extend the system timer by a variable **MS_TIMER_CLOCK** for getting a clock with a resolution of 1ms. The MS_TIMER_CLOCK variable runs from 0 to 999 and is synchronized with the **SEC_TIMER** variable. The variable MS_TIMER_CLOCK has the type **unsigned int**.

2.1.2 Example

The example shows how to read the clock system clock variable having a time stamp with ms-resolution. The picture below shows the extension of the clock by the new timer value MS_TIMER_CLOCK.



Calculated by RTC or by SEC_TIMER **MS_TIMER_CLOCK**

The value can easily be read by the same way as the SEC_TIMER value. MS_TIMER_CLOCK is synchronized wit the SEC_TIMER value. A small example that shows the current SEC_TIMER value and MS_TIMER_CLOCK value is shown below.

Example program how to use the MS_TIMER_CLOCK variable

```
// Example program of modified vdriver.lib
// this program shows how to use the modified library vdriver.lib with the MS_TIMER_CLOCK value
// Author: Gerhard Burger
// Date: 6. November 2005
// Feedback: please over: www.burger-web.com
// Updates: www.burger-web.com
//
// program description
// first step: SEC_TIMRER value is inited by the Real time clock value (it is assumed that
// the RTC is inited with the right time value
// second step: program prints out the current second value (holding the time and date in
// a resolution of 1 second and print out the milliseconds
void main(void)
{
      SEC_TIMER = read_rtc();         // initialize SEC_TIMER to new RTC
      printf("System time is %d seconds and %d2 milliseconds \r\n",SEC_TIMER,MS_TIMER_CLOCK);
}
```

2.1.3 The Modification

Having a system timer clock with ms resolution the library vdriver.lib has to be modified. Changes have to be made on 5 positions in the vdriver.lib and around 30 minutes are needed.
All positions in the file are indicated with the label “@Modification, so that you can find them very easy.
Attached a small description of the five Modification steps

2.2 Modification step 1

Define an external shared unsigned int variable MS_TIMER_CLOCK. This variable can be accessed by the application.

Modification 1

```
extern char __TICK_COUNTER;
extern char __MS_COUNTER;
extern int __SEC_COUNTER;
extern unsigned int __MCOS_COUNTER;
extern unsigned int __MCOS_CNT_VAL;
extern shared unsigned long MS_TIMER;
extern shared unsigned long TICK_TIMER;
extern shared unsigned long SEC_TIMER;

// @MODIFICATION------------------------------  22. November 2004 ------------------------------additional code for 1000 Hz counter
// Extension to the system clock to MS-TIMERS
// Modification 1 to 5
extern shared unsigned int MS_TIMER_CLOCK;

extern char  bios_intnesting;
extern char  bios_swpend;

/*** EndHeader */

char  bios_intnesting;
char  bios_swpend;
unsigned int __MCOS_COUNTER;
char  __TICK_COUNTER;
char __MS_COUNTER;
int __SEC_COUNTER;
shared unsigned long SEC_TIMER;
shared unsigned long MS_TIMER;
shared unsigned long TICK_TIMER;
unsigned int __MCOS_CNT_VAL;


// @MODIFICATION---------------------------------  22. November 2004 ----------------------------------------------
additional code for 1000 Hz counter
// Extension to system Clock to MS-TIMERS
// Modification 2 of  5
shared unsigned int MS_TIMER_CLOCK;

char pisr_inuse;

#if __SEPARATE_INST_DATA__
// This overwrites the periodic interrupt relay
#rcodorg periodic_intvec apply
#asm
    jp periodic_isr
#endasm
#rcodorg rootcode resume
#endif

#asm xmem nodebug
;
```

Figure 1: Modification step 1 (indicated in blue color)

2.3 Modification step 2

Define a shared unsigned int variable MS_TIMER_CLOCK. This variable holds the milliseconds for the system clock.

Modification 1

```
extern char __TICK_COUNTER;
extern char __MS_COUNTER;
extern int __SEC_COUNTER;
extern unsigned int __MCOS_COUNTER;
extern unsigned int __MCOS_CNT_VAL;
extern shared unsigned long MS_TIMER;
extern shared unsigned long TICK_TIMER;
extern shared unsigned long SEC_TIMER;

// @MODIFICATION---------- 22. November 2004 ----------additional code for 1000 Hz counter
// Extension to the system clock to MS-TIMERS
// Modification 1 to 5
extern shared unsigned int MS_TIMER_CLOCK;

extern char  bios_intnesting;
extern char  bios_swpend;

/*** EndHeader */

char  bios_intnesting;
char  bios_swpend;
unsigned int __MCOS_COUNTER;
char  __TICK_COUNTER;
char __MS_COUNTER;
int __SEC_COUNTER;
shared unsigned long SEC_TIMER;
shared unsigned long MS_TIMER;
shared unsigned long TICK_TIMER;
unsigned int __MCOS_CNT_VAL;


// @MODIFICATION---------- 22. November 2004 ----------additional code for 1000 Hz counter
// Extension to system Clock to MS-TIMERS
// Modification 2 of 5
shared unsigned int MS_TIMER_CLOCK;

char pisr_inuse;

#if __SEPARATE_INST_DATA__
// This overwrites the periodic interrupt relay
#rcodorg periodic_intvec apply
#asm
     jp periodic_isr
#endasm
#rcodorg rootcode resume
#endif

#asm xmem nodebug
;
```

Figure 2: Modification step 2

2.4 Modification step 3

Jump to calculation of MS_TIMER_CLOCK

Modification 3

```
  inc   (hl)              ; 8
  jr    nz,.DoneTICK      ; 5
  inc   hl                ; 2   this runs every 16384 s.
  inc   (hl)              ; 8
.DoneTICK:

  ld    a,(__MS_COUNTER)  ; 9   this runs every 1/1024th s.
  add   a,250             ; 4
  ld    (__MS_COUNTER),a  ; 10

// @MODIFICATION---------- 22. November 2004 ----------additional code for 1000 Hz counter
;// Extension to system Clock to MS-TIMERS
; Modification 3 of 5

; früher  jr    nc, .Done_MS      ; 5   skip 6 times per 256 entries because  ;      (250/256)*1024 = 1000
  jr    nc,  .Done_MS_TIMER_CLOCK   ;  5  skip  6  times  per  256  entries  because  ;
(250/256)*1024 = 1000

  ld    hl,  MS_TIMER     ; 6   this runs 1000 time per s.
  inc   (hl)              ; 8
  jr    nz,.Done_MS       ; 5
  inc   hl                ; 2   this runs ~ every 32/125 s.
  inc   (hl)              ; 8
  jr    nz,.Done_MS       ; 5
  inc   hl                ; 2   this runs ~ every 8192/125 s.
  inc   (hl)              ; 8
  jr    nz,.Done_MS       ; 5
  inc   hl                ; 2   this runs ~ every 2097152/125 s.
  inc   (hl)              ; 8
.Done_MS:


;@MODIFICATION----------  22. November 2004 ----------additional code for 1000 Hz counter
;Extension to system Clock to MS-TIMERS
; Modification 4 of 5
;
  ld    hl,  MS_TIMER_CLOCK  ; 6   this runs 1000 time per s. // load address of MS_Timer_CLOCK
  inc   (hl)              ; 8                    // icnrement ms
  jr    nz,.Done_MS_TIMER_CLOCK   ; 5  Check overflow
  inc   hl                ; 2   this runs ~ every 32/125 s.  if overflow
  inc   (hl)              ; 8
.Done_MS_TIMER_CLOCK:
;----------additional  code  for  1000  Hz
counter

  ld    hl,TICK_TIMER+1   ; 6
  ld    h,(hl)            ; 5
```

Figure 3: Modification step 3

2.5 Modification step 4

Increment the MS_TIMER_CLOCK variable and set to zero on overflow

Modification 4

```
  inc   (hl)              ; 8
  jr    nz,.DoneTICK      ; 5
  inc   hl                ; 2   this runs every 16384 s.
  inc   (hl)              ; 8
.DoneTICK:

  ld    a,(__MS_COUNTER)  ; 9   this runs every 1/1024th s.
  add   a,250             ; 4
  ld    (__MS_COUNTER),a  ; 10

// @MODIFICATION---------- 22. November 2004 ----------additional code for 1000 Hz counter
;// Extension to system Clock to MS-TIMERS
; Modification 3 of 5

; früher  jr    nc, .Done_MS      ; 5   skip 6 times per 256 entries because ;      (250/256)*1024 = 1000
  jr    nc,  .Done_MS_TIMER_CLOCK   ;  5  skip  6  times  per  256  entries  because  ;
(250/256)*1024 = 1000

  ld    hl,  MS_TIMER     ; 6   this runs 1000 time per s.
  inc   (hl)              ; 8
  jr    nz,.Done_MS       ; 5
  inc   hl                ; 2   this runs ~ every 32/125 s.
  inc   (hl)              ; 8
  jr    nz,.Done_MS       ; 5
  inc   hl                ; 2   this runs ~ every 8192/125 s.
  inc   (hl)              ; 8
  jr    nz,.Done_MS       ; 5
  inc   hl                ; 2   this runs ~ every 2097152/125 s.
  inc   (hl)              ; 8
.Done_MS:


;@MODIFICATION----------  22. November 2004 ----------additional code for 1000 Hz counter
;Extension to system Clock to MS-TIMERS
; Modification 4 of 5
;
  ld    hl,  MS_TIMER_CLOCK  ; 6   this runs 1000 time per s. // load address of MS_Timer_CLOCK
  inc   (hl)              ; 8                    // icnrement ms
  jr    nz,.Done_MS_TIMER_CLOCK   ; 5  Check overflow
  inc   hl                ; 2   this runs ~ every 32/125 s.  if overflow
  inc   (hl)              ; 8
.Done_MS_TIMER_CLOCK:
;-------------------------------------------------------------------------------------------additional  code  for  1000  Hz
counter

  ld    hl,TICK_TIMER+1   ; 6
  ld    h,(hl)            ; 5
```

Figure 4: Modification step 4

2.6 Modification step 5

Set variable MS_TIMER_CLOCK to 0 if 999ms have been reached.

Modification 5

```
  jr    z,.DoneTimers       ; 5

  ld    hl,__SEC_COUNTER    ; 6
  xor   a,(hl)              ; 6
  ld    (hl),a              ; 6

;@MODIFICATION----------  22. November 2004 ----------additional code for 1000 Hz counter
; Extension to system Clock to MS-TIMERS
; Modification 5 to 5
;
   xor   a                  ; clear a
   ld    (MS_TIMER_CLOCK),a
   ld    (MS_TIMER_CLOCK+1),a
;-------------------------------------------------------------------------------additional code for 1000 Hz counter

   ld    hl,SEC_TIMER        ; 6
   inc   (hl)                ; 8   this runs ~ every 1 s.
   jr    nz,.DoneTimers      ; 5
   inc   hl                  ; 2
   inc   (hl)                ; 8   this runs ~ every 256 s.
   jr    nz,.DoneTimers      ; 5
```

Figure 5: Modification step 5

3 Compatibility and side effects

I tested the modification on the libraries of the compiler versions:

- Dynamic C 8.01
- Dynamic C8.61 and
- Dynamic C9.02

On all of them it works very fine, also in more complex projects. I have not detected any side effects.

4 Download

The modified library vdriver.lib of version 8.61 as described above can be downloaded on:
http://www.burger-web.de/www.burger-web.com/downlods

5 Installation

1. Move to directory of Dynamic C compiler /Lib
2. Create a copy of vdriver.lib e.g. vdriver.lib_original
3. Either do modification of vdriver lib as described or use the modified vdriver.lib that can be downloaded on www.burger-web.com
4. run the sample program for test

Disclaimer

5.1 Limited Warranty and Disclaimer of Warranty

THIS SOFTWARE AND ACCOMPANYING WRITTEN MATERIALS (INCLUDING INSTRUCTIONS FOR USE) ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND. FURTHER, the author DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS REGARDING THE USE, OR THE RESULTS OF USE, OF THE SOFTWARE OR WRITTEN MATERIALS IN TERMS OF CORRECTNESS, ACCURACY, RELIABILITY, CURRENTNESS, OR OTHERWISE. THE ENTIRE RISK AS TO THE RESULTS AND PERFORMANCE OF THE SOFTWARE IS ASSUMED BY YOU. IF THE SOFTWARE OR WRITTEN MATERIALS ARE DEFECTIVE YOU, AND NOT the author OR ITS DEALERS, DISTRIBUTORS, AGENTS, OR EMPLOYEES, ASSUME THE ENTIRE COST OF ALL NECESSARY SERVICING, REPAIR, OR CORRECTION.

THE ABOVE IS THE ONLY WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, THAT IS MADE BY the author, ON THIS PRODUCT. NO ORAL OR WRITTEN INFORMATION OR ADVICE GIVEN BY the author, ITS DEALERS, DISTRIBUTORS, AGENTS OR EMPLOYEES SHALL CREATE A WARRANTY OR IN ANY WAY INCREASE THE SCOPE OF THIS WARRANTY AND YOU MAY NOT RELY ON ANY SUCH INFORMATION OR ADVICE.

NEITHER the author NOR ANYONE ELSE WHO HAS BEEN INVOLVED IN THE CREATION, PRODUCTION OR DELIVERY OF THIS PRODUCT SHALL BE LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL OR INCIDENTAL DAMAGES (INCLUDING DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, AND THE LIKE) ARISING OUT OF THE USE OR INABILITY TO USE SUCH PRODUCT EVEN IF the author HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

5.2 ACKNOWLEDGMENT

BY USING THIS PRODUCT YOU ACKNOWLEDGE THAT YOU HAVE READ THIS LIMITED WARRANTY, UNDERSTAND IT, AND AGREE TO BE BOUND BY ITS' TERMS AND CONDITIONS. YOU ALSO AGREE THAT THE LIMITED WARRANTY IS THE COMPLETE AND EXCLUSIVE STATEMENT OF AGREEMENT BETWEEN THE PARTIES AND SUPERSEDE ALL PROPOSALS OR PRIOR AGREEMENTS, ORAL OR WRITTEN, AND ANY OTHER COMMUNICATIONS BETWEEN THE PARTIES RELATING TO THE SUBJECT MATTER OF THE LIMITED WARRANTY.