RTX přepínací strategie (Scheduling Options)

- **Pre-emptive scheduling**

Každý task má **různou prioritu** a běží až ho přeruší task s vyšší prioritou nebo zablokuje volání funkce OS.

- **Round-Robin scheduling**

Každý task má **stejnou prioritu** a běží fixní periodu nebo ho zablokuje volaní funkce OS.

- **Co-operative multi-tasking**

Každý task má **stejnou prioritu** a Round-Robin je zakázán. Task zablokuje volaní OS nebo funkce **os_tsk_pass()**.

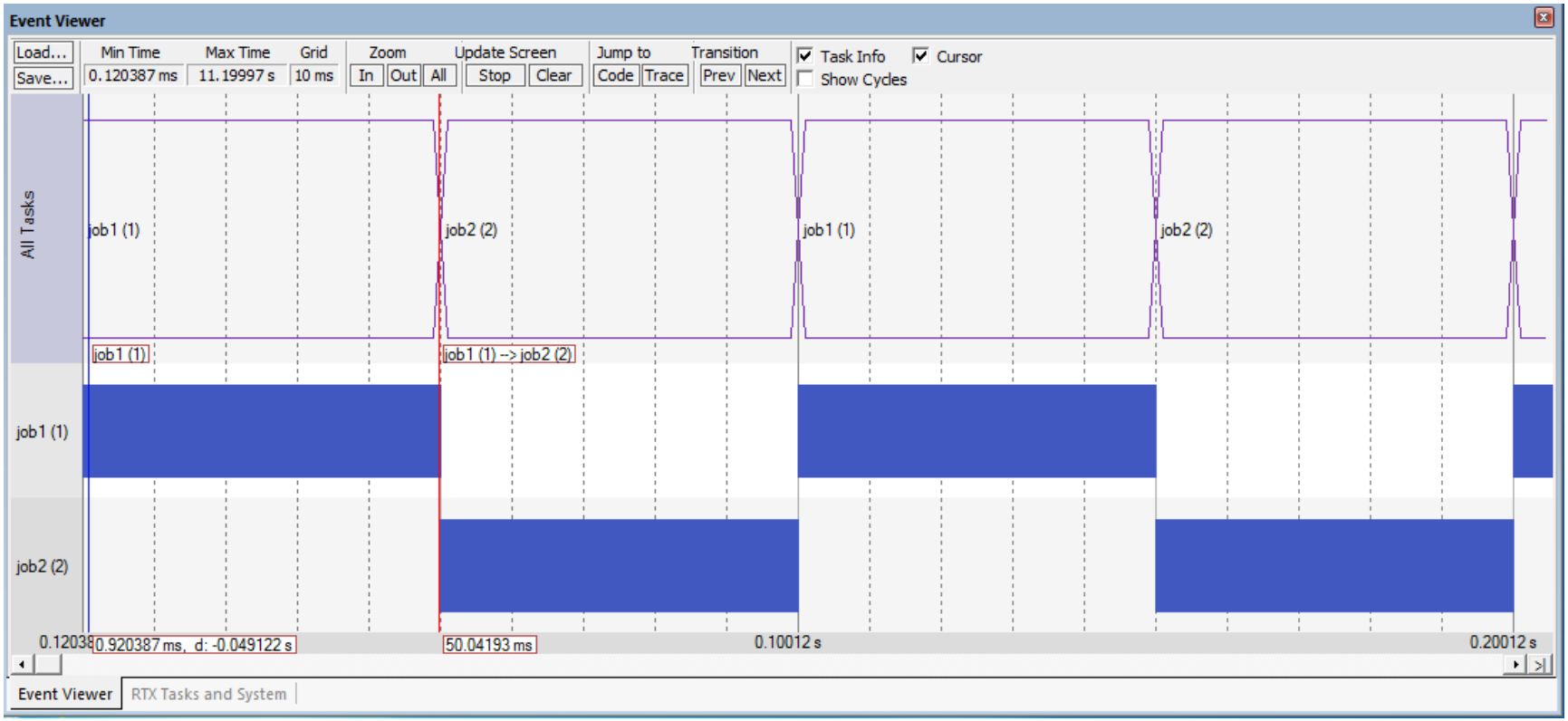
Round-Robin Scheduling

Tasky jsou prováděny po dobu stanoveného časového intervalu a ten se nastavuje. Potom RTX přepne následující task ze stavu **ready** do stavu **RUN** jestli má **stejnou prioritu**.Jestliže žádný jiný task není ve frontě **READY,** aktuální task pokračuje v činnosti. Délka časového intervalu se nastavuje v konfiguračním souboru **RTX_config.c** .

Příklad pro Round-Robin strategií.

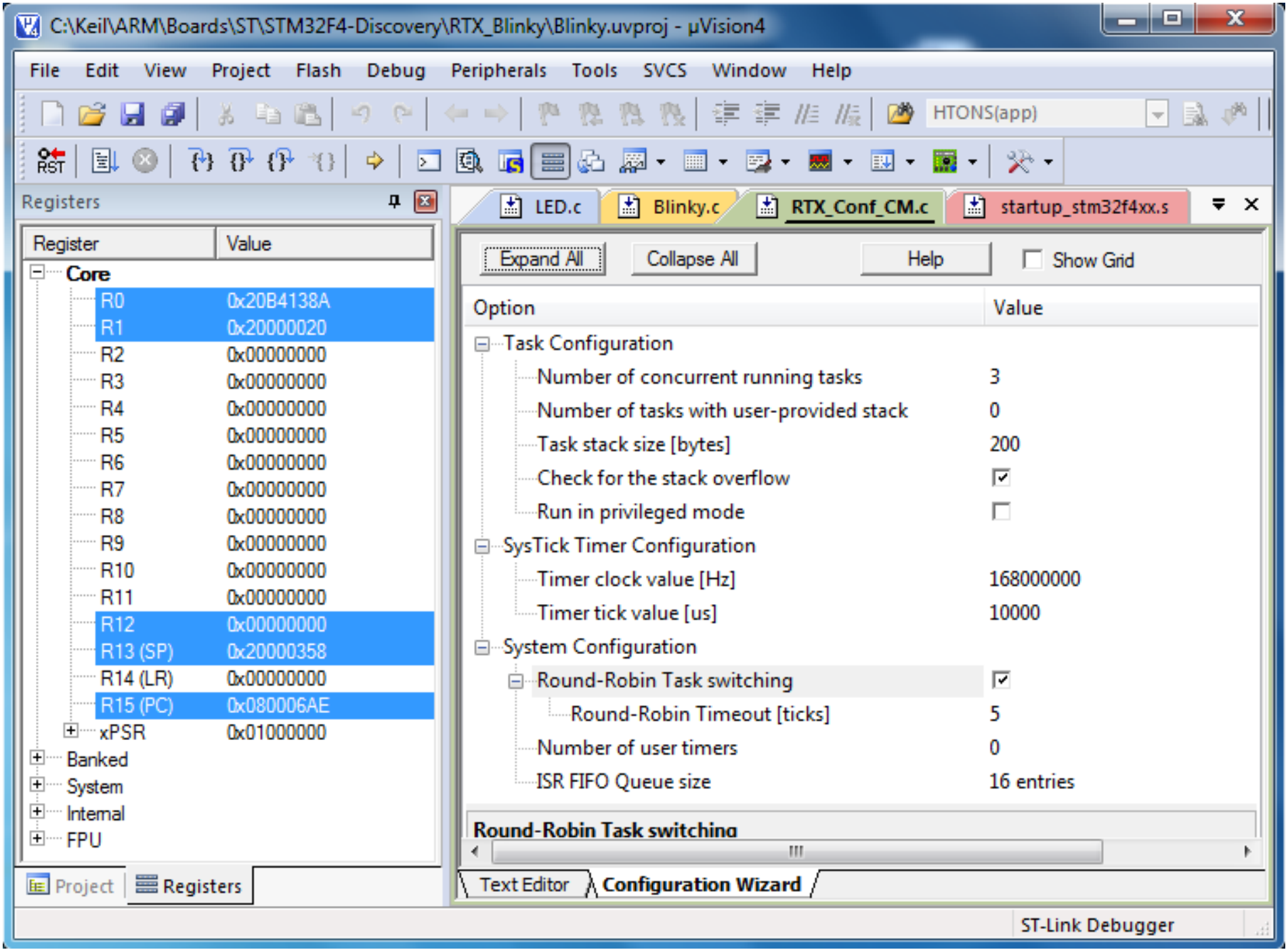
Dva tasky jsou nekonečné smyčky. RTX startuje task 1, jehož funkce se jmenuje **job1**. Tato funkce startuje další task zvaný **job2**.Když **job1** spotřebuje svůj time slice, RTX přepne kontext na **job2**. když **job2** spotřebuje svůj time slice, , RTX přepne kontext na **job1**. Tyto činnosti se opakují neustále.

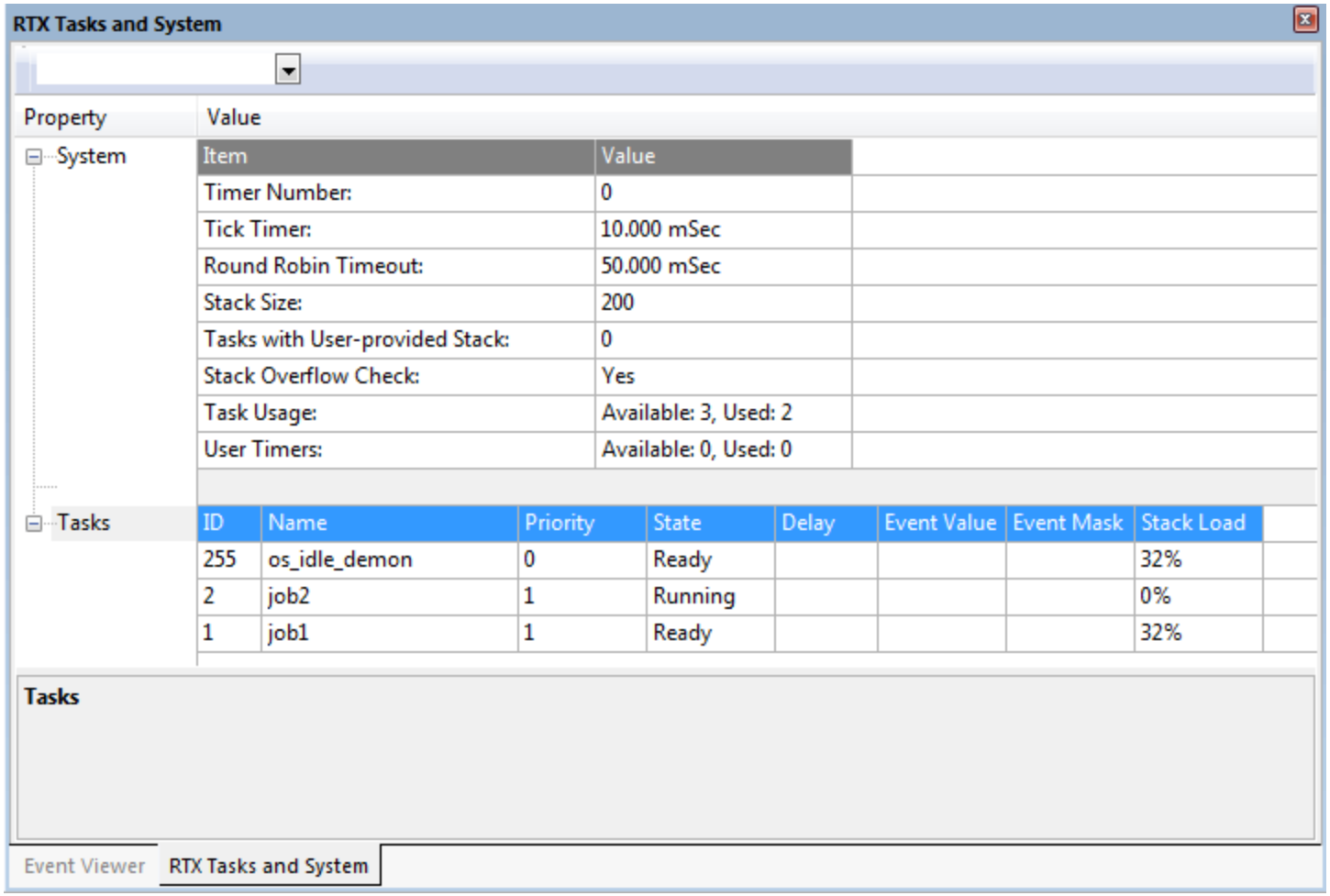
```
#include <rtl.h>

int counter1;
int counter2;
__task void job1 (void);
__task void job2 (void);

__task void job1 (void) {
  os_tsk_create (job2, 0);        /* vytvoří task 2 */
  while (1) {
    counter1++;                   /* update počítadlo 1 */
  }
}
__task void job2 (void) {
  while (1) {
    counter2++;                   /* update počítadlo 2 */
  }
}

void main (void) {
  os_sys_init (job1);         /* Inicializuje RTX Kernel and startuje task 1 */
  for (;;);
}
```

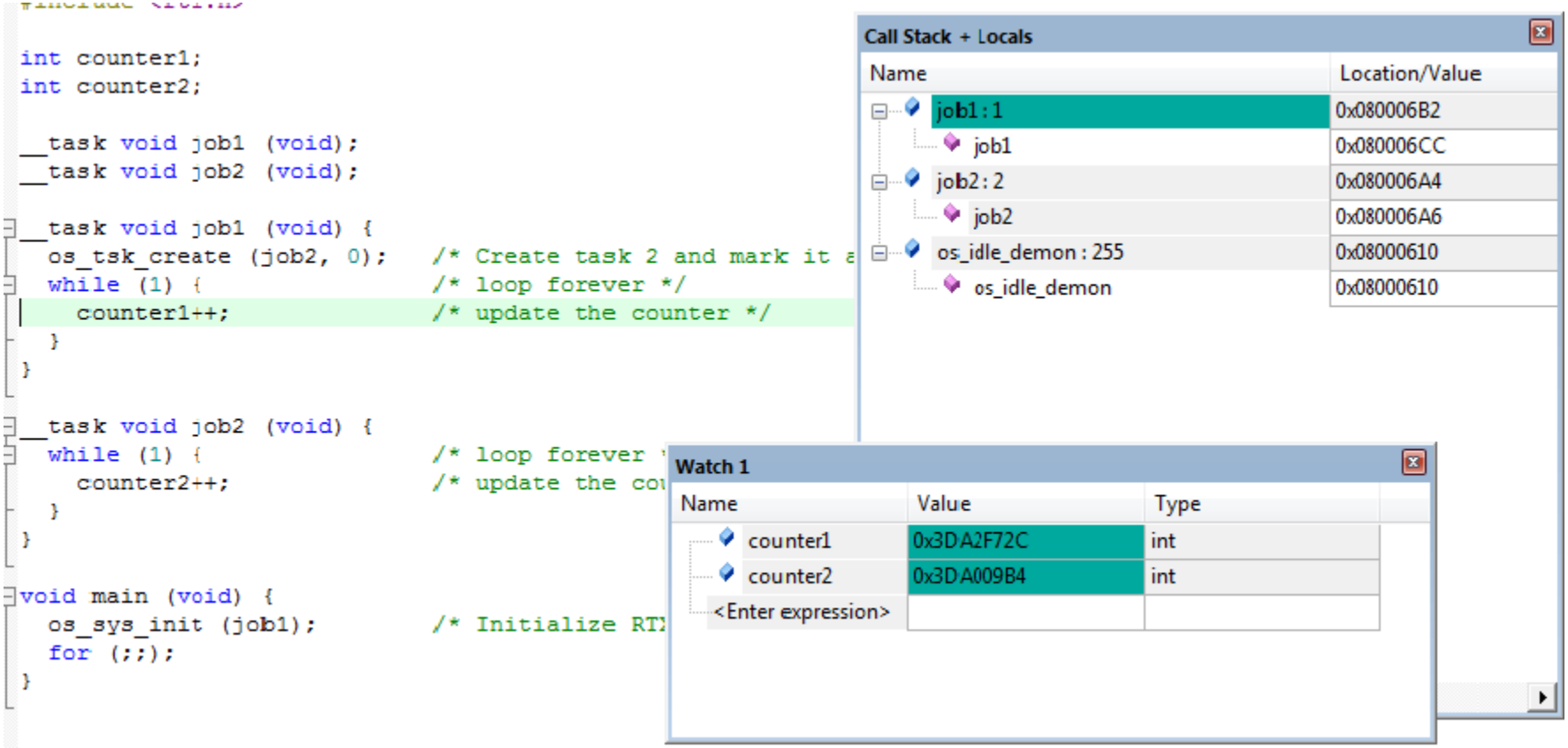


Note

•Rather than wait for a task's time slice to expire, you can use one of the **system wait** functions or the **os_tsk_pass** function to signal to the RTX kernel that it can switch to another task. The system wait function suspends the current task (changes it to the **WAIT_xxx** state) until the specified event occurs. The task is then changed to the **READY** state. During this time, any number of other tasks can run.


C:\Keil\ARM\Boards\ST\STM32F4-Discovery\RTX_Blinky\Blinky.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
HTONS(app)
Registers
Register Value
Core
R0 0x20B4138A
R1 0x20000020
R2 0x00000000
R3 0x00000000
R4 0x00000000
R5 0x00000000
R6 0x00000000
R7 0x00000000
R8 0x00000000
R9 0x00000000
R10 0x00000000
R11 0x00000000
R12 0x00000000
R13 (SP) 0x20000358
R14 (LR) 0x00000000
R15 (PC) 0x080006AE
xPSR 0x01000000
Banked
System
Internal
FPU
Project
Registers
LED.c
Blinky.c
RTX_Conf_CM.c
startup_stm32f4xx.s
Expand All
Collapse All
Help
Show Grid
Option
Value
Task Configuration
Number of concurrent running tasks 3
Number of tasks with user-provided stack 0
Task stack size [bytes] 200
Check for the stack overflow
Run in privileged mode
SysTick Timer Configuration
Timer clock value [Hz] 168000000
Timer tick value [us] 10000
System Configuration
Round-Robin Task switching
Round-Robin Timeout [ticks] 5
Number of user timers 0
ISR FIFO Queue size 16 entries
Round-Robin Task switching
Text Editor
Configuration Wizard
ST-Link Debugger


RTX Tasks and System
Property
Value
System
Item
Value
Timer Number:
0
Tick Timer:
10.000 mSec
Round Robin Timeout:
50.000 mSec
Stack Size:
200
Tasks with User-provided Stack:
0
Stack Overflow Check:
Yes
Task Usage:
Available: 3, Used: 2
User Timers:
Available: 0, Used: 0
Tasks
ID
Name
Priority
State
Delay
Event Value
Event Mask
Stack Load
255
os_idle_demon
0
Ready
32%
2
job2
1
Running
0%
1
job1
1
Ready
32%
Tasks
Event Viewer
RTX Tasks and System


int counter1;
int counter2;
__task void job1 (void);
__task void job2 (void);
__task void job1 (void) {
os_tsk_create (job2, 0); /* Create task 2 and mark it a
while (1) { /* loop forever */
counter1++; /* update the counter */
}
}
__task void job2 (void) {
while (1) { /* loop forever
counter2++; /* update the cou
}
}
void main (void) {
os_sys_init (job1); /* Initialize RT
for (;;);
}
Call Stack + Locals
Name
Location/Value
job1 : 1
0x080006B2
job1
0x080006CC
job2 : 2
0x080006A4
job2
0x080006A6
os_idle_demon : 255
0x08000610
os_idle_demon
0x08000610
Watch 1
Name
Value
Type
counter1
0x3DA2F72C
int
counter2
0x3DA009B4
int
<Enter expression>