Tenniel Takenaka-Fuller
Github root directory: https://github.com/TennielTakenaka/sturdy-carnival/tree/master/Lab%201

Date Submitted: **12/4/19**

Task 01:

Youtube Link: https://youtu.be/5vN4CUxyeCA

Modified Code:

```
/* TI-RTOS Header files */
#include <xdc/std.h>
#include <ti/sysbios/BIOS.h>
#include <ti/sysbios/knl/Task.h>

#include <ti/drivers/GPIO.h>

/* Example/Board Header files */
#include "Board.h"

void myDelay(int count);

/* Could be anything, like computing primes */
#define FakeBlockingSlowWork()   myDelay(12000000)
#define FakeBlockingFastWork()   myDelay(2000000)

Task_Struct workTask;
/* Make sure we have nice 8-byte alignment on the stack to avoid wasting memory */
#pragma DATA_ALIGN(workTaskStack, 8)
#define STACKSIZE 1024
static uint8_t workTaskStack[STACKSIZE];

void doUrgentWork(void)
{
    GPIO_write(Board_GPIO_LED1, Board_GPIO_LED_OFF);
    FakeBlockingFastWork(); /* Pretend to do something useful but time-consuming */
    GPIO_write(Board_GPIO_LED1, Board_GPIO_LED_ON);
}

void doWork(void)
{
    GPIO_write(Board_GPIO_LED0, Board_GPIO_LED_OFF);
    FakeBlockingSlowWork(); /* Pretend to do something useful but time-consuming */
    GPIO_write(Board_GPIO_LED0, Board_GPIO_LED_ON);
}

Void workTaskFunc(UArg arg0, UArg arg1)
{
    while (1) {

        /* Do work */
        doWork();

        /* Wait a while, because doWork should be a periodic thing, not continuous.*/
```

Grading scheme: 30% Coding, 30% Documentation, 40% Execution/Video.

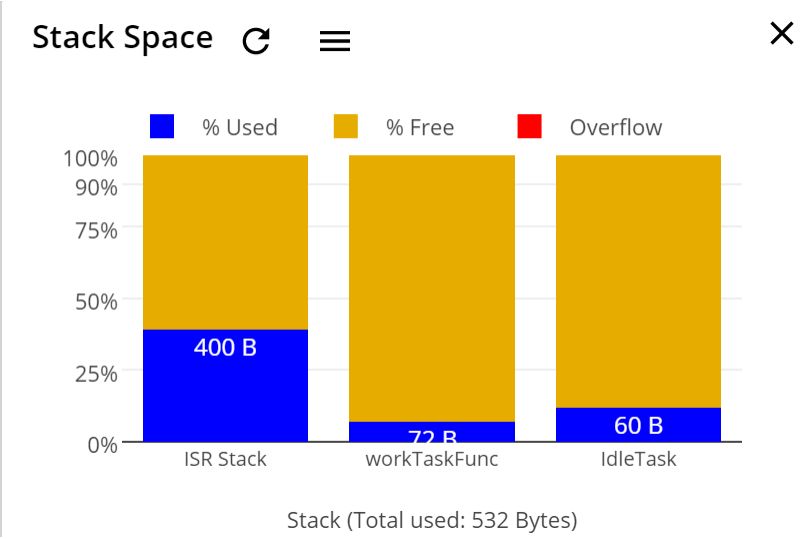
```
        myDelay(24000000);
    }
}

/*
 *  ======== main ========
 *
 */
int main(void)
{
    Board_initGeneral();
    GPIO_init();

    /* Set up the led task */
    Task_Params workTaskParams;
    Task_Params_init(&workTaskParams);
    workTaskParams.stackSize = STACKSIZE;
    workTaskParams.priority = 2;
    workTaskParams.stack = &workTaskStack;

    Task_construct(&workTask, workTaskFunc, &workTaskParams, NULL);

    /* Start kernel. */
    BIOS_start();

    return (0);
}

/*
 *  ======== myDelay ========
 *  Assembly function to delay. Decrements the count until it is zero
 *  The exact duration depends on the processor speed.
 */
__asm("    .sect \".text:myDelay\"\n"
      "    .clink\n"
      "    .thumbfunc myDelay\n"
      "    .thumb\n"
      "    .global myDelay\n"
      "myDelay:\n"
      "    subs r0, #1\n"
      "    bne.n myDelay\n"
      "    bx lr\n");
```

Task 02:

Youtube link: https://youtu.be/5vN4CUxyeCA

Grading scheme: 30% Coding, 30% Documentation, 40% Execution/Video.

Tenniel Takenaka-Fuller
Github root directory: https://github.com/TennielTakenaka/sturdy-carnival/tree/master/Lab%201



Enable logs:

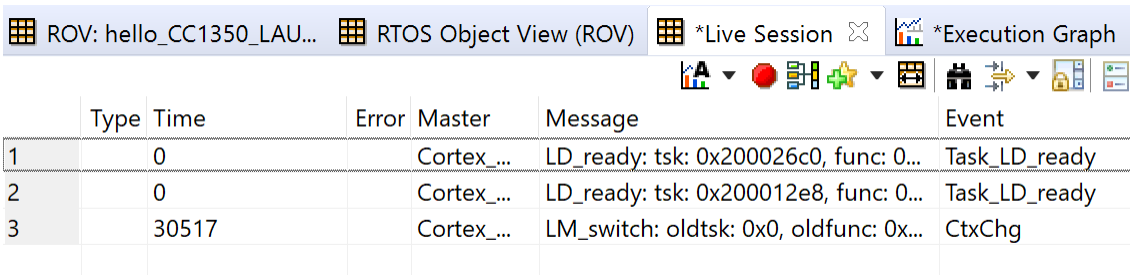
```
*/
BIOS.logsEnabled = true;
//BIOS.logsEnabled = false;
```

Enable store to flash:

```
383 else if (Program.cpu.deviceName.match(/CC26/)) {
384     ROM.romName = ROM.CC2650;
385 }
386 /*else if (Program.cpu.deviceName.match(/CC13/)) {
387     ROM.romName = ROM.CC1350;
388 }
389 */
```

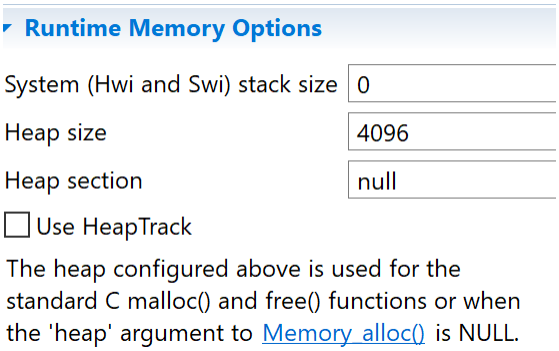
Added at bottom of config file:

```
609 var LoggingSetup = xdc.useModule('ti.uia.sysbios.LoggingSetup');
610 LoggingSetup.sysbiosLoggerSize = 1024;
611 LoggingSetup.loadLogging = false;
```



	Type	Time	Error	Master	Message	Event
1		0		Cortex_...	LD_ready: tsk: 0x200026c0, func: 0...	Task_LD_ready
2		0		Cortex_...	LD_ready: tsk: 0x200012e8, func: 0...	Task_LD_ready
3		30517		Cortex_...	LM_switch: oldtsk: 0x0, oldfunc: 0x...	CtxChg

Grading scheme: 30% Coding, 30% Documentation, 40% Execution/Video.



Task 03:

Youtube Link: https://youtu.be/lv6XkVPCMO4

Modified Code:

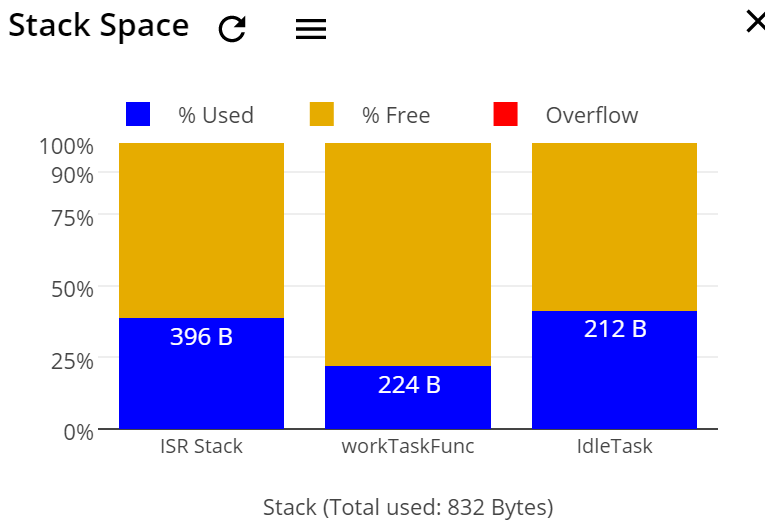
```
/* TI-RTOS Header files */
#include <xdc/std.h>
#include <ti/sysbios/BIOS.h>
#include <ti/sysbios/knl/Task.h>

#include <ti/drivers/GPIO.h>
#include <ti/sysbios/knl/Clock.h>
/* Example/Board Header files */
#include "Board.h"

void myDelay(int count);

/* Could be anything, like computing primes */
#define FakeBlockingSlowWork()   myDelay(12000000)
#define FakeBlockingFastWork()   myDelay(2000000)

Task_Struct workTask;
/* Make sure we have nice 8-byte alignment on the stack to avoid wasting memory */
#pragma DATA_ALIGN(workTaskStack, 8)
#define STACKSIZE 1024
static uint8_t workTaskStack[STACKSIZE];

void doUrgentWork(void)
{
    GPIO_write(Board_GPIO_LED1, Board_GPIO_LED_OFF);
    FakeBlockingFastWork(); /* Pretend to do something useful but time-consuming */
    GPIO_write(Board_GPIO_LED1, Board_GPIO_LED_ON);
}

void doWork(void)
{
    GPIO_write(Board_GPIO_LED0, Board_GPIO_LED_OFF);
    FakeBlockingSlowWork(); /* Pretend to do something useful but time-consuming */
```

Grading scheme: 30% Coding, 30% Documentation, 40% Execution/Video.

```
    GPIO_write(Board_GPIO_LED0, Board_GPIO_LED_ON);
}

Void workTaskFunc(UArg arg0, UArg arg1)
{
    while (1) {

        /* Do work */
        doWork();

        /* Wait a while, because doWork should be a periodic thing, not continuous.*/
        //myDelay(24000000);
        Task_sleep(500 * (1000 / Clock_tickPeriod));
    }
}

/*
 *  ======== main ========
 *
 */
int main(void)
{
    Board_initGeneral();
    GPIO_init();

    /* Set up the led task */
    Task_Params workTaskParams;
    Task_Params_init(&workTaskParams);
    workTaskParams.stackSize = STACKSIZE;
    workTaskParams.priority = 2;
    workTaskParams.stack = &workTaskStack;

    Task_construct(&workTask, workTaskFunc, &workTaskParams, NULL);

    /* Start kernel. */
    BIOS_start();

    return (0);
}

/*
 *  ======== myDelay ========
 *  Assembly function to delay. Decrements the count until it is zero
 *  The exact duration depends on the processor speed.
 */
__asm("    .sect \".text:myDelay\"\n"
      "    .clink\n"
      "    .thumbfunc myDelay\n"
      "    .thumb\n"
      "    .global myDelay\n"
      "myDelay:\n"
      "    subs r0, #1\n"
      "    bne.n myDelay\n"
      "    bx lr\n");
```

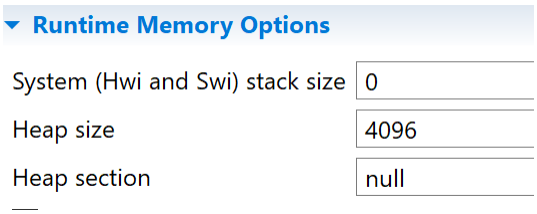
Grading scheme: 30% Coding, 30% Documentation, 40% Execution/Video.

Tenniel Takenaka-Fuller
Github root directory: https://github.com/TennielTakenaka/sturdy-carnival/tree/master/Lab%201



ROV: hello_CC1350_LAUNCHXL_tirtos_ccs.out - Cortex_M3_0 | RTOS Object View (ROV) | *Live Session | *Execution Graph

Type	Time	Error	Master	Message	Event	EventClass	Data1	Data2
	71730499267		Cortex_...	LM_switch: oldtsk: 0x200026c0, old...	CtxChg	TSK	wor...	
	72724884033		Cortex_...	LD_block: tsk: 0x200012e8, func: 0...	Task_LD_block	Unknown	wor...	
	72724914550		Cortex_...	LM_sleep: tsk: 0x200012e8, func: 0...	Task_LM_sleep	Unknown	wor...	
	72724914550		Cortex_...	LM_switch: oldtsk: 0x200012e8, old...	CtxChg	TSK	ti_sy...	
	73225006103		Cortex_...	LD_ready: tsk: 0x200012e8, func: 0...	Task_LD_ready	Unknown	wor...	
	73225006103		Cortex_...	LM_switch: oldtsk: 0x200026c0, old...	CtxChg	TSK	wor...	
	74219390869		Cortex_...	LD_block: tsk: 0x200012e8, func: 0...	Task_LD_block	Unknown	wor...	



Task 04:

Youtube Link: https://youtu.be/60jNbsg7I1A

Modified Code:

```
   /* TI-RTOS Header files */
  #include <xdc/std.h>
  #include <ti/sysbios/BIOS.h>
  #include <ti/sysbios/knl/Task.h>
  #include <ti/sysbios/knl/Clock.h>
```

Grading scheme: 30% Coding, 30% Documentation, 40% Execution/Video.

```
#include <ti/drivers/GPIO.h>

/* Example/Board Header files */
#include "Board.h"

void myDelay(int count);

/* Could be anything, like computing primes */
#define FakeBlockingSlowWork()   myDelay(12000000)
#define FakeBlockingFastWork()   myDelay(2000000)

Task_Struct workTask;
Task_Struct urgentWorkTask;
/* Make sure we have nice 8-byte alignment on the stack to avoid wasting memory */
#pragma DATA_ALIGN(workTaskStack, 8)
#define STACKSIZE 1024
static uint8_t workTaskStack[STACKSIZE];
static uint8_t urgentWorkTaskStack[STACKSIZE];

void doUrgentWork(void)
{
    GPIO_write(Board_GPIO_LED1, Board_GPIO_LED_OFF);
    FakeBlockingFastWork(); /* Pretend to do something useful but time-consuming */
    GPIO_write(Board_GPIO_LED1, Board_GPIO_LED_ON);
}

void doWork(void)
{
    GPIO_write(Board_GPIO_LED0, Board_GPIO_LED_OFF);
    FakeBlockingSlowWork(); /* Pretend to do something useful but time-consuming */
    GPIO_write(Board_GPIO_LED0, Board_GPIO_LED_ON);
}

Void workTaskFunc(UArg arg0, UArg arg1)
{
    while (1) {

        /* Do work */
        doWork();

        /* Wait a while, because doWork should be a periodic thing, not
continuous.*/
        //myDelay(24000000);
        Task_sleep(500 * (1000 / Clock_tickPeriod));
    }
}

Void urgentWorkTaskFunc(UArg arg0, UArg arg1)
{
    while (1) {

        /* Do work */
        doUrgentWork();
```

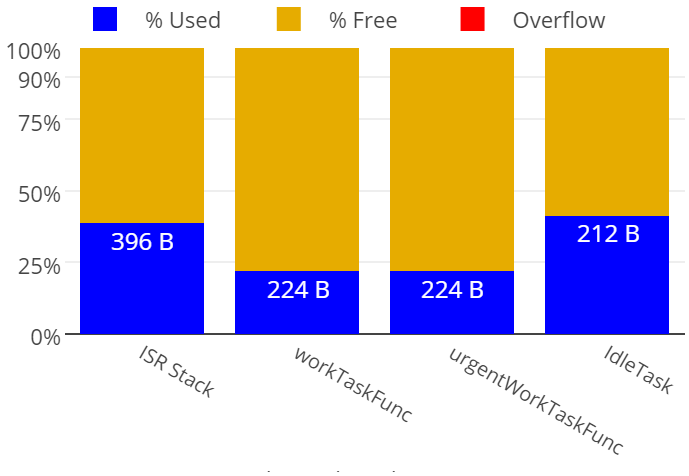
Grading scheme: 30% Coding, 30% Documentation, 40% Execution/Video.

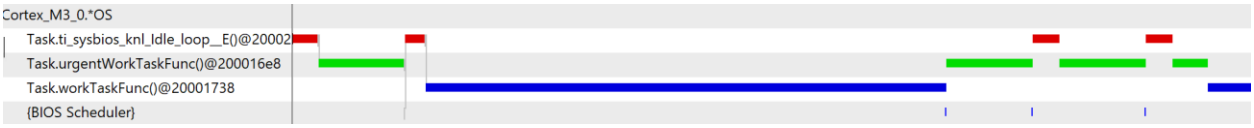
```
            /* Wait a while, because doWork should be a periodic thing, not
continuous.*/
            //myDelay(24000000);
            Task_sleep(50 * (1000 / Clock_tickPeriod));
        }
    }

    /*
     *  ======== main ========
     *
     */
    int main(void)
    {
        Board_initGeneral();
        GPIO_init();

        /* Set up the led task */
        Task_Params workTaskParams;
        Task_Params_init(&workTaskParams);
        workTaskParams.stackSize = STACKSIZE;
        workTaskParams.priority = 2;
        workTaskParams.stack = &workTaskStack;

        Task_construct(&workTask, workTaskFunc, &workTaskParams, NULL);

        workTaskParams.priority = 1;
        workTaskParams.stack = &urgentWorkTaskStack;

        Task_construct(&urgentWorkTask, urgentWorkTaskFunc, &workTaskParams, NULL);

        /* Start kernel. */
        BIOS_start();

        return (0);
    }

    /*
     *  ======== myDelay ========
     *  Assembly function to delay. Decrements the count until it is zero
     *  The exact duration depends on the processor speed.
     */
    __asm("    .sect \".text:myDelay\"\n"
          "    .clink\n"
          "    .thumbfunc myDelay\n"
          "    .thumb\n"
          "    .global myDelay\n"
          "myDelay:\n"
          "    subs r0, #1\n"
          "    bne.n myDelay\n"
          "    bx lr\n");
```

Grading scheme: 30% Coding, 30% Documentation, 40% Execution/Video.

Tenniel Takenaka-Fuller
Github root directory: https://github.com/TennielTakenaka/sturdy-carnival/tree/master/Lab%201

Stack Space





	Type	Time	Error	Master	Message	Event	EventClass	Data1	Data2	SeqNo	Logger	Module
12		117620880126		Cortex_...	LD_block: tsk: 0x20001738, func: 0...	Task_LD_block	Unknown	wor...		1278	SYSBI...	ti.sysb...
13		117620910644		Cortex_...	LM_sleep: tsk: 0x20001738, func: 0...	Task_LM_sleep	Unknown	wor...		1279	SYSBI...	ti.sysb...
14		117620910644		Cortex_...	LM_switch: oldtsk: 0x20001738, old...	CtxChg	TSK	urg...		1280	SYSBI...	_ti.uia....
15		117786773681		Cortex_...	LD_block: tsk: 0x200016e8, func: 0...	Task_LD_block	Unknown	urg...		1281	SYSBI...	ti.sysb...
16		117786804199		Cortex_...	LM_sleep: tsk: 0x200016e8, func: 0...	Task_LM_sleep	Unknown	urg...		1282	SYSBI...	ti.sysb...
17		117786804199		Cortex_...	LM_switch: oldtsk: 0x200016e8, old...	CtxChg	TSK	ti_sy...		1283	SYSBI...	_ti.uia....
18		117836914062		Cortex_...	LD_ready: tsk: 0x200016e8, func: 0...	Task_LD_ready	Unknown	urg...		1284	SYSBI...	ti.sysb...

BIOS - All Top-level BIOS Options

Welcome | System Overview | Runtime | Error Handling | Device Support | Advanced

Basic | xdc.runtime

Name	Value	Summary
▾ Memory		
instanceHeap	null	module's instance heap
instanceSection	null	memory section for module's instances
memoryPolicy	DELETE_POLICY	module's memory policy
namedModule	false	true => module's name is on target
namedInstance	false	true => instances have string names
▾ Diagnostics		
logger	null	module's logger
diags_ASSERT	ALWAYS_ON	module's Diags assert mode
diags_ENTRY	ALWAYS_OFF	module's function entry Diags mode
diags_EXIT	ALWAYS_OFF	module's function exit Diags mode
diags_INTERNAL	ALWAYS_OFF	module's internal assert mode

Grading scheme: 30% Coding, 30% Documentation, 40% Execution/Video.

Tenniel Takenaka-Fuller
Github root directory: https://github.com/TennielTakenaka/sturdy-carnival/tree/master/Lab%201

Task 05: CHANGING PRIORITY [FROM TASK 4]

Youtube Link: https://youtu.be/pLeWJ3xMKJ8

Modified Code:

```
   /* TI-RTOS Header files */
  #include <xdc/std.h>
  #include <ti/sysbios/BIOS.h>
  #include <ti/sysbios/knl/Task.h>
  #include <ti/sysbios/knl/Clock.h>

  #include <ti/drivers/GPIO.h>

  /* Example/Board Header files */
  #include "Board.h"

  void myDelay(int count);

  /* Could be anything, like computing primes */
  #define FakeBlockingSlowWork()   myDelay(12000000)
  #define FakeBlockingFastWork()   myDelay(2000000)

  Task_Struct workTask;
  Task_Struct urgentWorkTask;
  /* Make sure we have nice 8-byte alignment on the stack to avoid wasting memory */
  #pragma DATA_ALIGN(workTaskStack, 8)
  #define STACKSIZE 1024
  static uint8_t workTaskStack[STACKSIZE];
  static uint8_t urgentWorkTaskStack[STACKSIZE];

  void doUrgentWork(void)
  {
      GPIO_write(Board_GPIO_LED1, Board_GPIO_LED_OFF);
      FakeBlockingFastWork(); /* Pretend to do something useful but time-consuming */
      GPIO_write(Board_GPIO_LED1, Board_GPIO_LED_ON);
  }

  void doWork(void)
  {
      GPIO_write(Board_GPIO_LED0, Board_GPIO_LED_OFF);
      FakeBlockingSlowWork(); /* Pretend to do something useful but time-consuming */
      GPIO_write(Board_GPIO_LED0, Board_GPIO_LED_ON);
  }

  Void workTaskFunc(UArg arg0, UArg arg1)
  {
      while (1) {

          /* Do work */
          doWork();

          /* Wait a while, because doWork should be a periodic thing, not
continuous.*/
```

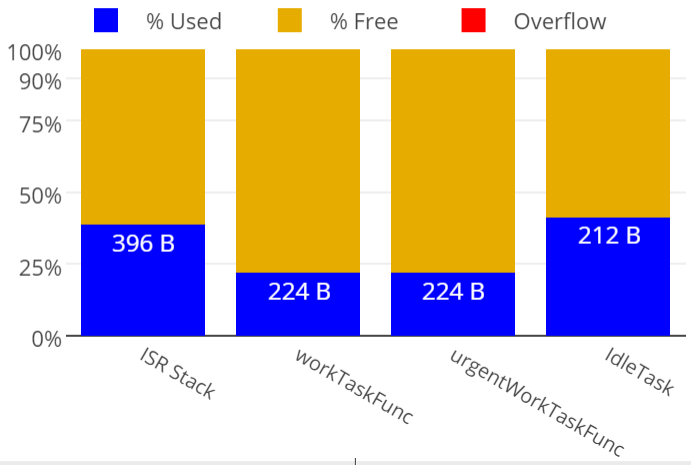
Grading scheme: 30% Coding, 30% Documentation, 40% Execution/Video.

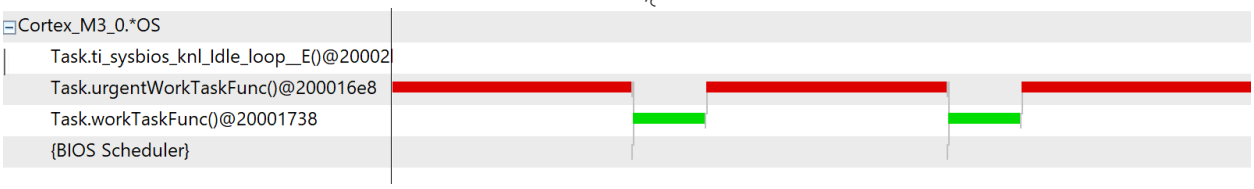
```
            //myDelay(24000000);
            Task_sleep(500 * (1000 / Clock_tickPeriod));
        }
    }

    Void urgentWorkTaskFunc(UArg arg0, UArg arg1)
    {
        while (1) {

            /* Do work */
            doUrgentWork();

            /* Wait a while, because doWork should be a periodic thing, not
continuous.*/
            //myDelay(24000000);
            Task_sleep(50 * (1000 / Clock_tickPeriod));
        }
    }

    /*
     *  ======== main ========
     *
     */
    int main(void)
    {
        Board_initGeneral();
        GPIO_init();

        /* Set up the led task */
        Task_Params workTaskParams;
        Task_Params_init(&workTaskParams);
        workTaskParams.stackSize = STACKSIZE;
        workTaskParams.priority = 2;
        workTaskParams.stack = &workTaskStack;

        Task_construct(&workTask, workTaskFunc, &workTaskParams, NULL);

        workTaskParams.priority = 3;
        workTaskParams.stack = &urgentWorkTaskStack;

        Task_construct(&urgentWorkTask, urgentWorkTaskFunc, &workTaskParams, NULL);

        /* Start kernel. */
        BIOS_start();

        return (0);
    }

    /*
     *  ======== myDelay ========
     *  Assembly function to delay. Decrements the count until it is zero
     *  The exact duration depends on the processor speed.
     */
    __asm("     .sect \".text:myDelay\"\n"
          "     .clink\n"
```

Grading scheme: 30% Coding, 30% Documentation, 40% Execution/Video.

```
"    .thumbfunc myDelay\n"
"    .thumb\n"
"    .global myDelay\n"
"myDelay:\n"
"    subs r0, #1\n"
"    bne.n myDelay\n"
"    bx lr\n");
```

Stack Space





Type	Time	Error	Master	Message	Event	EventClass	Data1	Data2
	102998962402		Cortex_...	LM_switch: oldtsk: 0x20001738, old...	CtxChg	TSK	urg...	
	103164825439		Cortex_...	LD_block: tsk: 0x200016e8, func: 0...	Task_LD_block	Unknown	urg...	
	103164855957		Cortex_...	LM_sleep: tsk: 0x200016e8, func: 0...	Task_LM_sleep	Unknown	urg...	
	103164855957		Cortex_...	LM_switch: oldtsk: 0x200016e8, old...	CtxChg	TSK	wor...	
	103214874267		Cortex_...	LD_ready: tsk: 0x200016e8, func: 0...	Task_LD_ready	Unknown	urg...	
	103214874267		Cortex_...	LM_switch: oldtsk: 0x20001738, old...	CtxChg	TSK	urg...	
	103380737304		Cortex_...	LD_block: tsk: 0x200016e8, func: 0...	Task_LD_block	Unknown	urg...	
	103380767822		Cortex_...	LM_sleep: tsk: 0x200016e8, func: 0...	Task_LM_sleep	Unknown	urg...	

Grading scheme: 30% Coding, 30% Documentation, 40% Execution/Video.

Tenniel Takenaka-Fuller
Github root directory: https://github.com/TennielTakenaka/sturdy-carnival/tree/master/Lab%201

BIOS - All Top-level BIOS Options

Welcome | System Overview | Runtime | Error Handling | Device Support | Advanced

Basic | xdc.runtime

Name	Value	Summary
▾ Memory		
instanceHeap	null	module's instance heap
instanceSection	null	memory section for module's instances
memoryPolicy	DELETE_POLICY	module's memory policy
namedModule	false	true => module's name is on target
namedInstance	false	true => instances have string names
▾ Diagnostics		
logger	null	module's logger
diags_ASSERT	ALWAYS_ON	module's Diags assert mode
diags_ENTRY	ALWAYS_OFF	module's function entry Diags mode
diags_EXIT	ALWAYS_OFF	module's function exit Diags mode

Grading scheme: 30% Coding, 30% Documentation, 40% Execution/Video.