Batch: A2 Roll No.: 16010322014
Experiment / assignment / tutorial No. 04
Grade: AA / AB / BB / BC / CC / CD /DD
Signature of the Staff In-charge with date

TITLE: To perform resource synchronization using semaphores
AIM: Write a FreeRTOS based program to receive message using semaphores & UART0 of LPC 2148

1. **OUTCOME:** Implement Open Source RTOS for resource sharing using inter task communication

Components Required: -

□ **Hardware:**

- LPC2148 Microcontroller Board
- UART0 interface for serial communication
- Serial communication cable (e.g., USB to UART converter)
- PC or terminal software (e.g., PuTTY, Tera Term) for UART communication

□ **Software:**

- Keil µVision IDE ARM Compiler
- FreeRTOS kernel source files

Procedure :-

Step 1: Setup the Project

- Create a new project for LPC2148 in your IDE.
- Add startup files and system initialization code.
- Add FreeRTOS source files to the project.
- Configure clock and initialize UART0 peripheral.

Step 2: Initialize UART0

- Configure UART0 for desired baud rate, data bits, parity, stop bits.
- Enable UART0 receive interrupt if needed.

Step 3: Create Semaphore

- Define and create a binary semaphore using xSemaphoreCreateBinary().
- The semaphore will be used to signal that UART data is available.

Step 4: Create Tasks

- **UART Receiver Task:** Waits on the semaphore to receive notification that UART data is ready, then reads data from UART.
- **Other Task (optional):** Could signal the semaphore or process received data.

Step 5: UART Interrupt Handler

- In the UART0 ISR, when data is received, give (release) the semaphore using xSemaphoreGiveFromISR() to unblock the UART Receiver Task.

Step 6: Task Function Logic

- UART Receiver Task blocks on the semaphore using xSemaphoreTake().
- When semaphore is taken (UART data ready), read the UART data and process it.

Step 7: Start Scheduler

- Call vTaskStartScheduler() to start FreeRTOS.

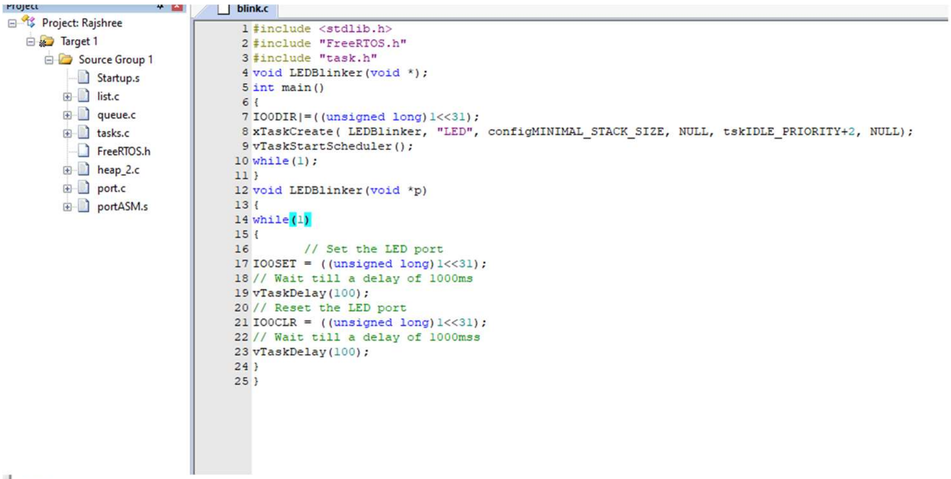
Step 8: Debug and Test

- Connect UART0 to PC terminal software.
- Build and flash the program.
- Send data from terminal, observe task receiving data upon semaphore signaling.

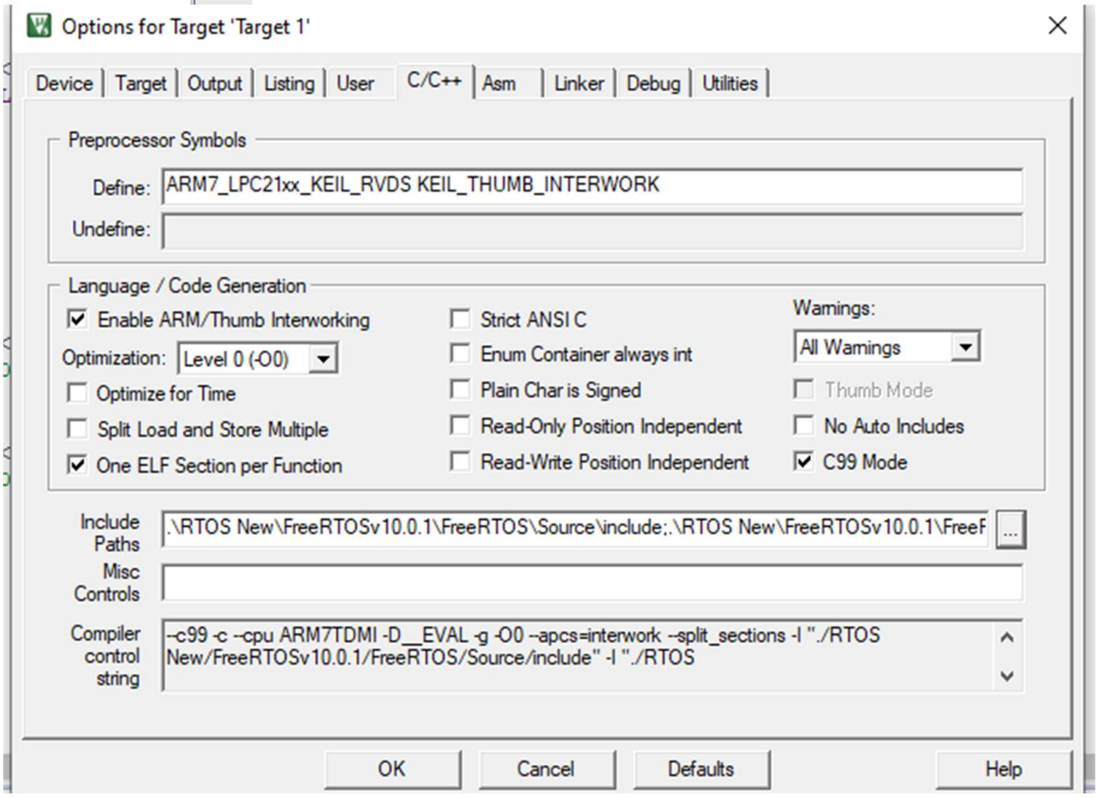
Mention and Describe the FreeRTOS APIs related to Semaphore

xSemaphoreCreateBinary()	Creates a binary semaphore (initially empty).
xSemaphoreGive()	Releases (gives) a semaphore from a task context, unblocking any task waiting on it.
xSemaphoreGiveFromISR()	Releases a semaphore from an ISR context (interrupt service routine).
xSemaphoreTake()	Attempts to take (acquire) a semaphore, optionally blocking if not available.
vSemaphoreDelete()	Deletes a semaphore and frees resources.
xSemaphoreCreateMutex()	Creates a mutex semaphore for mutual exclusion (not strictly binary).

Observations: -



```c
#include <stdlib.h>
#include "FreeRTOS.h"
#include "task.h"
void LEDBlinker(void *);
int main()
{
IO0DIR|=((unsigned long)1<<31);
xTaskCreate( LEDBlinker, "LED", configMINIMAL_STACK_SIZE, NULL, tskIDLE_PRIORITY+2, NULL);
vTaskStartScheduler();
while(1);
}
void LEDBlinker(void *p)
{
while(1)
{
        // Set the LED port
IO0SET = ((unsigned long)1<<31);
// Wait till a delay of 1000ms
vTaskDelay(100);
// Reset the LED port
IO0CLR = ((unsigned long)1<<31);
// Wait till a delay of 1000mss
vTaskDelay(100);
}
}
```



Preprocessor Symbols

Define: ARM7_LPC21xx_KEIL_RVDS KEIL_THUMB_INTERWORK

Undefine:

Language / Code Generation

☑ Enable ARM/Thumb Interworking

Optimization: Level 0 (-O0)

□ Optimize for Time

□ Split Load and Store Multiple

☑ One ELF Section per Function

□ Strict ANSI C

□ Enum Container always int

□ Plain Char is Signed

□ Read-Only Position Independent

□ Read-Write Position Independent

Warnings: All Warnings

□ Thumb Mode

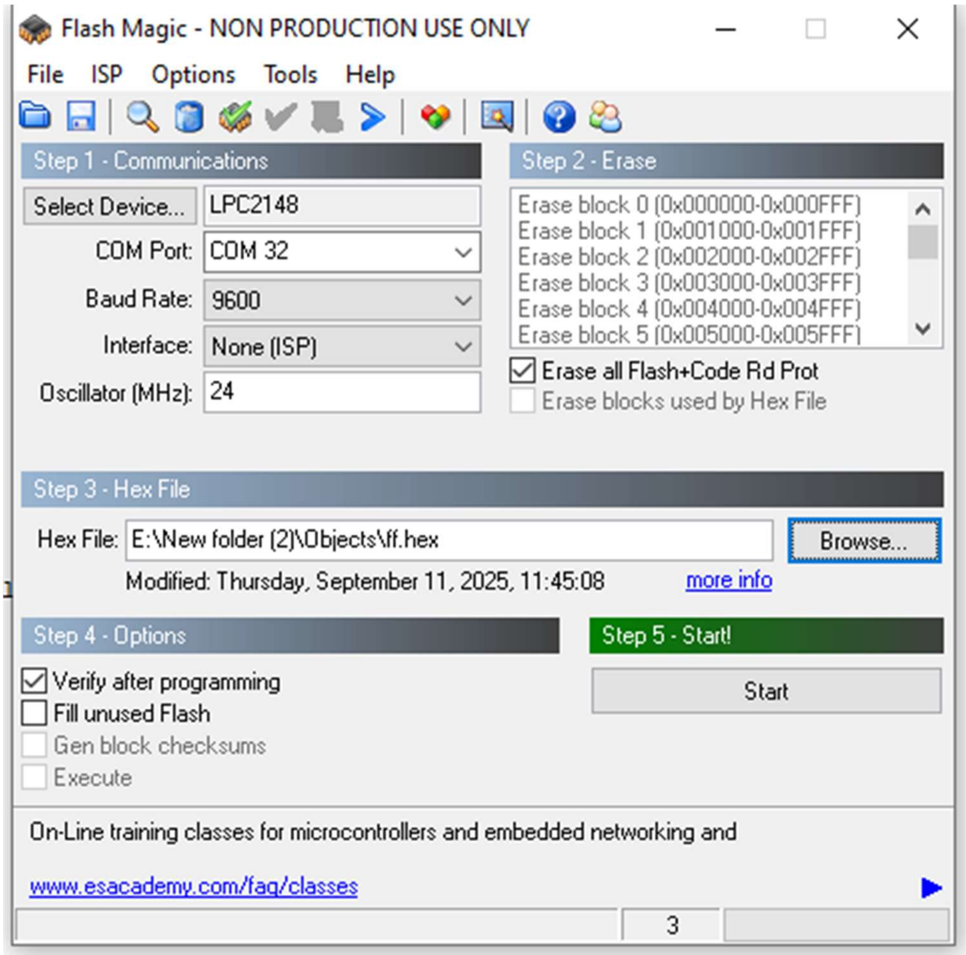
□ No Auto Includes

☑ C99 Mode

Include Paths: .\RTOS New\FreeRTOSv10.0.1\FreeRTOS\Source\include;.\RTOS New\FreeRTOSv10.0.1\FreeF

Misc Controls:

Compiler control string: --c99 -c --cpu ARM7TDMI -D__EVAL -g -O0 --apcs=interwork --split_sections -I "./RTOS New/FreeRTOSv10.0.1/FreeRTOS/Source/include" -I "./RTOS



Select Device... LPC2148

COM Port: COM 32

Baud Rate: 9600

Interface: None (ISP)

Oscillator (MHz): 24

Erase block 0 (0x000000-0x000FFF)
Erase block 1 (0x001000-0x001FFF)
Erase block 2 (0x002000-0x002FFF)
Erase block 3 (0x003000-0x003FFF)
Erase block 4 (0x004000-0x004FFF)
Erase block 5 (0x005000-0x005FFF)

☑ Erase all Flash+Code Rd Prot
□ Erase blocks used by Hex File

Hex File: E:\New folder (2)\Objects\ff.hex

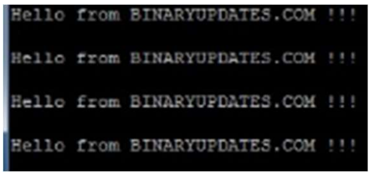
Modified: Thursday, September 11, 2025, 11:45:08

☑ Verify after programming
□ Fill unused Flash
□ Gen block checksums
□ Execute

On-Line training classes for microcontrollers and embedded networking and

www.esacademy.com/faq/classes

3



```
Hello from BINARYUPDATES.COM !!!

Hello from BINARYUPDATES.COM !!!

Hello from BINARYUPDATES.COM !!!

Hello from BINARYUPDATES.COM !!!
```

Conclusion:
Using FreeRTOS semaphores with UART0 on the LPC2148 enables effective resource synchronization and safe message handling. This demonstrates practical inter-task communication for efficient resource sharing in embedded systems.

Signature of faculty in-charge with date