

Exercise 13)

$$f = \frac{1}{T}$$

$$\underline{10 \text{ Hz}}$$

$$10 \text{ Hz} = \frac{1}{T}$$

$$\text{Cycles} = f \cdot \text{Duration}$$

$$100 = 10 \text{ Hz} \cdot 10$$

$$T = \frac{1}{10 \text{ Hz}} = 0.1$$

$$\frac{0.1}{2} = 0.05 \text{ sec} = 50 \text{ ms}$$

```
12 void flashLed(uint16_t GPIO_Pin, int delay, int cycles)
13 {
14     for(int i = 0; i < cycles; i++)
15     {
16         HAL_GPIO_WritePin(GPIOD, GPIO_Pin, GPIO_PIN_SET);
17         osDelay(delay);
18         HAL_GPIO_WritePin(GPIOD, GPIO_Pin, GPIO_PIN_RESET);
19         osDelay(delay);
20     }
21 }
```

```
484 void StartGreenTask(void const * argument)
485 {
486     /* USER CODE BEGIN StartGreenTask */
487     /* Infinite loop */
488     for(;;)
489     {
490
491         flashLed(LD4_Pin, 50, 100); //10Hz, 10 cycles
492         set_flag(1);
493         flashLed(LD4_Pin, 500, 10); //1Hz 10 cycles
494         reset_flag(1);
495         flashLed(LD4_Pin, 50, 100); //10Hz, 10 cycles
496
497         vTaskSuspend(NULL);
498
499     }
500     /* USER CODE END StartGreenTask */
501 }

510 void StartBlueTask(void const * argument)
511 {
512     /* USER CODE BEGIN StartBlueTask */
513     /* Infinite loop */
514     for(;;)
515     {
516
517         HAL_GPIO_WritePin(GPIOD, LD6_Pin, GPIO_PIN_SET);
518         osDelay(50);
519         HAL_GPIO_WritePin(GPIOD, LD6_Pin, GPIO_PIN_RESET);
520         osDelay(50);
521
522         if((check_flag(1)==Set) && (check_flag(2)==Set))
523         {
524
525             HAL_GPIO_WritePin(GPIOD, LD6_Pin, GPIO_PIN_SET);
526             osDelay(500);
527             HAL_GPIO_WritePin(GPIOD, LD6_Pin, GPIO_PIN_RESET);
528             osDelay(500);
529         }
530
531         else if((check_flag(1)==Reset) && (check_flag(2)==Reset))
532         {
533
534
535             HAL_GPIO_WritePin(GPIOD, LD6_Pin, GPIO_PIN_SET);
536             osDelay(50);
537             HAL_GPIO_WritePin(GPIOD, LD6_Pin, GPIO_PIN_RESET);
538             osDelay(50);
539         }
540
541     }
542 }
543 /* USER CODE END StartBlueTask */
```

```

553 void StartRedTask(void const * argument)
554 {
555     /* USER CODE BEGIN StartRedTask */
556     /* Infinite loop */
557     for(;;)
558     {
559         flashLed(LD5_Pin, 50, 150); //10Hz, 15 cycles
560         set_flag(2);
561         flashLed(LD5_Pin, 500, 10); //1Hz 10 cycles
562         reset_flag(2);
563         flashLed(LD5_Pin, 50, 50); //10Hz, 5 cycles
564
565         vTaskSuspend(NULL);
566
567     }
568     /* USER CODE END StartRedTask */
569 }
570

```

Exercise 14)

```

32 const uint32_t GreenSignal = 0x05;
33 const uint32_t RedSignal = 0x04;

505 void StartGreenTask(void const * argument)
506 {
507     /* USER CODE BEGIN StartGreenTask */
508     /* Infinite loop */
509     for(;;)
510     {
511         osSignalWait(GreenSignal, osWaitForever); //wait for signal
512
513         flashLed(LD4_Pin, 50, 50); //10Hz, 5 cycles
514     }
515     /* USER CODE END StartGreenTask */
516 }

525 void StartBlueTask(void const * argument)
526 {
527     /* USER CODE BEGIN StartBlueTask */
528     /* Infinite loop */
529     for(;;)
530     {
531         flashLed(LD6_Pin, 500, 10); //1Hz, 10 cycles
532
533         osSignalSet(GreenTaskHandle, GreenSignal); //sends signal to GreenTask
534
535         osDelay(6000); //delay 6 seconds
536
537
538         osSignalSet(RedTaskHandle, RedSignal); //sends signal to redTask
539
540     }
541     /* USER CODE END StartBlueTask */
542 }

```

```

551 void StartRedTask(void const * argument)
552 {
553     /* USER CODE BEGIN StartRedTask */
554     /* Infinite loop */
555     for(;;)
556     {
557         osSignalWait(RedSignal, osWaitForever); //wait for signal
558
559         flashLed(LD5_Pin, 50, 50); //10Hz, 5 cycles
560     }
561     /* USER CODE END StartRedTask */
562 }
563

```

Exercise 15)

```

141 /* USER CODE BEGIN RTOS_SEMAPHORES */
142 /* add semaphores, ... */
143 osSemaphoreWait(RedEFHandle, 1); //takes semaphore to make task wait
144 osSemaphoreWait(GreenEFHandle, 1); //takes semaphore to make task wait
145 /* USER CODE END RTOS_SEMAPHORES */

519 void StartGreenTask(void const * argument)
520 {
521     /* USER CODE BEGIN StartGreenTask */
522     /* Infinite loop */
523     for(;;)
524     {
525
526         osSemaphoreWait(GreenEFHandle, portMAX_DELAY); //wait for EventFlag
527
528         flashLed(LD4_Pin, 50, 50); //10Hz, 5 cycles
529     }
530     /* USER CODE END StartGreenTask */
531 }

540 void StartBlueTask(void const * argument)
541 {
542     /* USER CODE BEGIN StartBlueTask */
543     /* Infinite loop */
544     for(;;)
545     {
546         flashLed(LD6_Pin, 500, 10); //1Hz, 10 cycles
547
548         osSemaphoreRelease(GreenEFHandle); //Send Event flag to Green
549
550         osDelay(6000); //delay 6 seconds
551
552
553         osSemaphoreRelease(RedEFHandle); //Send Event flag to Red
554     }
555     /* USER CODE END StartBlueTask */
556 }

```

```

565 void StartRedTask(void const * argument)
566 {
567     /* USER CODE BEGIN StartRedTask */
568     /* Infinite loop */
569     for(;;)
570     {
571         osSemaphoreWait(RedEFHandle, portMAX_DELAY);
572
573         flashLed(LD5_Pin, 50, 50); //10Hz, 5 cycles
574
575     }
576     /* USER CODE END StartRedTask */
577 }
---
```

Exercise 16)

```

136 /* USER CODE BEGIN RTOS_SEMAPHORES */
137 /* add semaphores, ... */
138 osSemaphoreWait(SemaSync1Handle, 1); //takes semaphore to make task wait
139 osSemaphoreWait(SemaSync2Handle, 1); //takes semaphore to make task wait
140 /* USER CODE END RTOS_SEMAPHORES */

510 void StartGreenTask(void const * argument)
511 {
512     /* USER CODE BEGIN StartGreenTask */
513     /* Infinite loop */
514     for(;;)
515     {
516         flashLed(LD4_Pin, 50, 50); //10Hz, 5 cycles
517
518         osSemaphoreWait(SemaSync2Handle, portMAX_DELAY);
519         flashLed(LD4_Pin, 500, 10); //1Hz, 10 cycles
520         osSemaphoreRelease(SemaSync1Handle);
521
522     }
523     /* USER CODE END StartGreenTask */
524 }

534 void StartRedTask(void const * argument)
535 {
536     /* USER CODE BEGIN StartRedTask */
537     /* Infinite loop */
538     for(;;)
539     {
540         flashLed(LD5_Pin, 500, 10); //1Hz, 10 cycles
541
542         osSemaphoreRelease(SemaSync2Handle);
543         flashLed(LD5_Pin, 50, 50); //10Hz, 5 cycles
544         osSemaphoreWait(SemaSync1Handle, portMAX_DELAY);
545
546     }
547     /* USER CODE END StartRedTask */
548 }
---
```

