FreeRTOS 3/3

Przebieg ćwiczenia

1. Zaimplementuj mechanizm synchronizacji za pomocą semafora binarnego. Po naciśnięciu przycisku, dioda powinna zapalić się na 1s.
   Wskazówki:
   - Użyj przerwania (EXTI) do zwolnienia semafora w reakcji na przycisk
   - Zadanie główne powinno czekać na semafor

   Sprawdź czy semafor można zastąpić mutexem.
2. Zaimplementuj ochronę dostępu do wspólnego zasobu za pomocą mutex. Stwórz dwa zadania:
   - Pierwsze zadanie co 2 sekundy wysyła komunikat: "Task 1: Hello!".
   - Drugie zadanie co 3 sekundy wysyła komunikat: "Task 2: World!".

   Tekst wysyłaj funkcją opracowaną na poprzednich zajęciach. Wysyłane dane obserwuj w terminalu.
3. Różne priorytety a dostęp do zasobów. Zaimplementuj trzy wybrane sygnały kodów błędów BIOS prezentowanych na diodzie LD2. Przykładowe:
   - Sygnał 1: Miganie diody 3 razy (500 ms włączona, 500 ms wyłączona).
   - Sygnał 2: Miganie diody 2 razy (1 sekunda włączona, 1 sekunda wyłączona).
   - Sygnał 3: Miganie diody 2 razy (2 sekundy włączona).

   Stwórz trzy zadania:
   - Zadanie A (wysoki priorytet): Co 3 sekundy próbuje wygenerować sygnał n.1.
   - Zadanie B (średni priorytet): Co 10 sekund próbuje wygenerować sygnał n.2.
   - Zadanie C (niski priorytet): Co 15 sekund próbuje wygenerować sygnał n.3.

   Pomiędzy sekwencjami odczekaj 2 sekundy. Nie doprowadź do zagłodzenia zadań.
4. Producent i konsument z wykorzystaniem semaforów zliczających.
   Producent:
   - Miga diodą LD2 co 2 sekundy.
   - Każde zaświecenie zwiększa wartość semafora zliczającego (np. generuje nowe „zadanie” do wykonania przez konsumenta).

   Konsument:
   - Wyświetla komunikat na porcie szeregowym po naciśnięciu przycisku B1 (synchronizacja semaforem binarnym), jeżeli dostępny jest zasób od producenta.

   Semafor zliczający:
   - Pojemność „kolejki” wynosi 5.

   Obserwuj, jak działa mechanizm ograniczenia dostępu do zasobów.

Funkcje

HAL

- void **HAL_GPIO_WritePin** (GPIO_TypeDef * GPIOx, uint16_t GPIO_Pin, GPIO_PinState PinState)
- GPIO_PinState **HAL_GPIO_ReadPin** (GPIO_TypeDef * GPIOx, uint16_t GPIO_Pin)
- void **HAL_Delay** (uint32_t Delay)
- uint32_t **HAL_GetTick** (void)
- HAL_StatusTypeDef **HAL_ADC_Start** (ADC_HandleTypeDef * hadc)
- HAL_StatusTypeDef **HAL_ADC_PollForConversion** (ADC_HandleTypeDef * hadc, uint32_t Timeout)
- uint32_t **HAL_ADC_GetValue** (ADC_HandleTypeDef * hadc)
- __weak void **HAL_GPIO_EXTI_Callback**(uint16_t GPIO_Pin)

FreeRTOS

- osStatus **osDelay** (uint32_t millisec)
- int32_t **osSemaphoreWait** (osSemaphoreId semaphore_id, uint32_t millisec)
- osStatus **osSemaphoreRelease** (osSemaphoreId semaphore_id)
- osStatus **osMutexWait** (osMutexId mutex_id, uint32_t millisec);
- osStatus **osMutexRelease** (osMutexId mutex_id);

Materiały dodatkowe

- Description of STM32F4 HAL and low-layer drivers - User manual
- FreeRTOS API categories