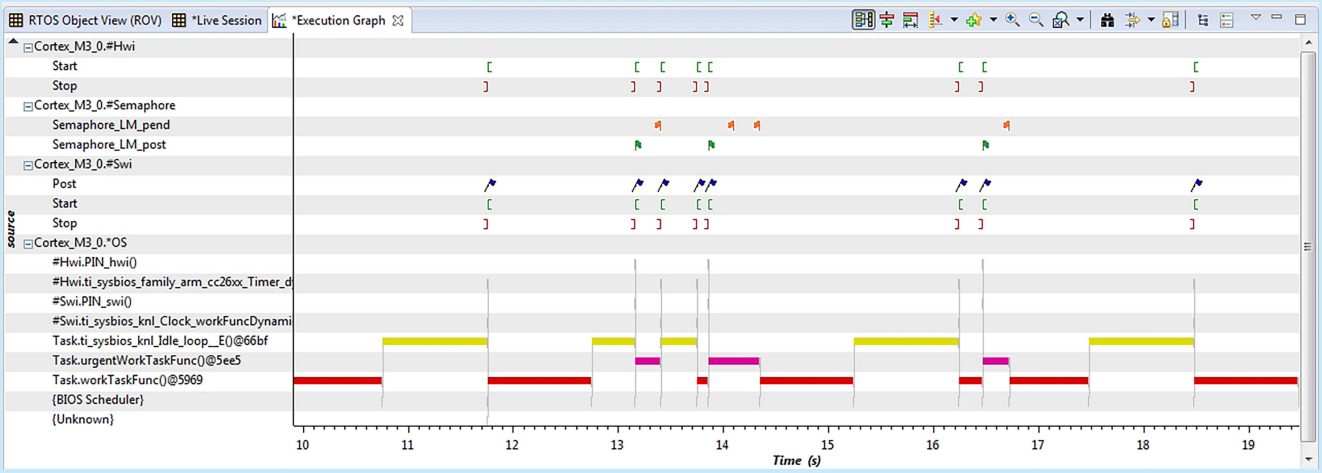


Systemy dla Internetu Rzeczy (6)

System operacyjny czasu rzeczywistego TI-RTOS – zadania i przerwania

Podstawą efektywnego działania systemów wbudowanych jest praca w czasie rzeczywistym. Istotnym elementem procesu tworzenia i uruchamiania oprogramowania dla tych systemów jest możliwość podglądania działania wątków. Z taką sytuacją mamy do czynienia w wypadku układu CC2650 SensorTag firmy Texas Instruments. Dlatego producent przygotował wersję systemu operacyjnego czasu rzeczywistego TI-RTOS zawierającego rozszerzenia debugowe oraz ściśle powiązanego ze stosem komunikacyjnym.

W artykułach z cyklu „Systemy dla Internetu Rzeczy" omówiono zestaw CC2650 SensorTag [1], jego użytkowanie [2] oraz moduły rozszerzeń Debug DevPack, Display DevPack (LCD screen) i LED Audio DevPack [3]. Moduł CC1310 LaunchPad został omówiony w kolejnym odcinku [4].

W poprzednim odcinku kursu przedstawiono system operacyjny czasu rzeczywistego TI-RTOS dla serii układów scalonych CC13xx/CC26xx [5]. Praktyczne ćwiczenie z prostym programem dotyczyło zestawu CC2650 SensorTag.

Dokumentacja

Linki do aktualnych wersji podstawowej dokumentacji można znaleźć na stronie TI-RTOS [6] oraz na stronach TI Wiki [13, 14]. Opis rdzenia systemu jest opisany w dokumencie *SYS/BIOS (TI-RTOS Kernel) User's Guide* [9]. Opis ogólny systemu jest zamieszczony w dokumencie *TI-RTOS 2.20 User's Guide* [11]. Opis wersji dla procesorów serii CC26xx jest zamieszczony w *TI-RTOS 2.20 for CC13xx/CC26xx SimpleLink Getting Started Guide* [8] oraz w *CC2640/CC2650 Bluetooth low energy Software Developer's Guide* [10].

Podstawowym źródłem dokumentacji zainstalowanej lokalnie w komputerze PC w wersji systemu TI-RTOS jest plik *Documentation_Overview_cc13xx_cc26xx.html* zamieszczony w ścieżce *C:\ti\tirtos_cc13xx_cc26xx_2_20_01_08\docs*. Pliki opisu są w formacie pdf i htm. Jest również dokumentacja drajwerów peryferyjnych w formacie Doxygen. W pliku są też odnośniki do stron internetowych. Na portalu społecznościowym *TI E2E Community* znajduje się bardzo przydatna strona *CC2640/CC2650 Getting Started and FAQ* [12]. Jest ona często aktualizowana i zawiera odpowiedzi na najczęściej zadawane pytania.

Warsztaty *SimpleLink Academy*

Bardzo ciekawą pomocą dla każdego, kto zaczyna pracować z procesorami rodziny CC13xx/CC26xx, są ćwiczenia warsztatowe *SimpleLink Academy* [16]. Dostępnych jest wiele ćwiczeń z dokładnym opisem oraz kodem źródłowym. Dla wielu ćwiczeń jest udostępniony zapis wideo.

Ćwiczenie *TI-RTOS Basic – Lab 1*

To ćwiczenie jest dalszym ciągiem ćwiczenia przedstawionego w poprzednim odcinku kursu [5]. Postępowanie jest opisane w dwóch wariantach:

- Kontynuacja poprzedniego ćwiczenia z wykorzystaniem poprzednio utworzonego folderu roboczego.
- Utworzenie nowego folderu roboczego i rozpoczęcie pracy z systemem TI-RTOS od obecnego materiału.

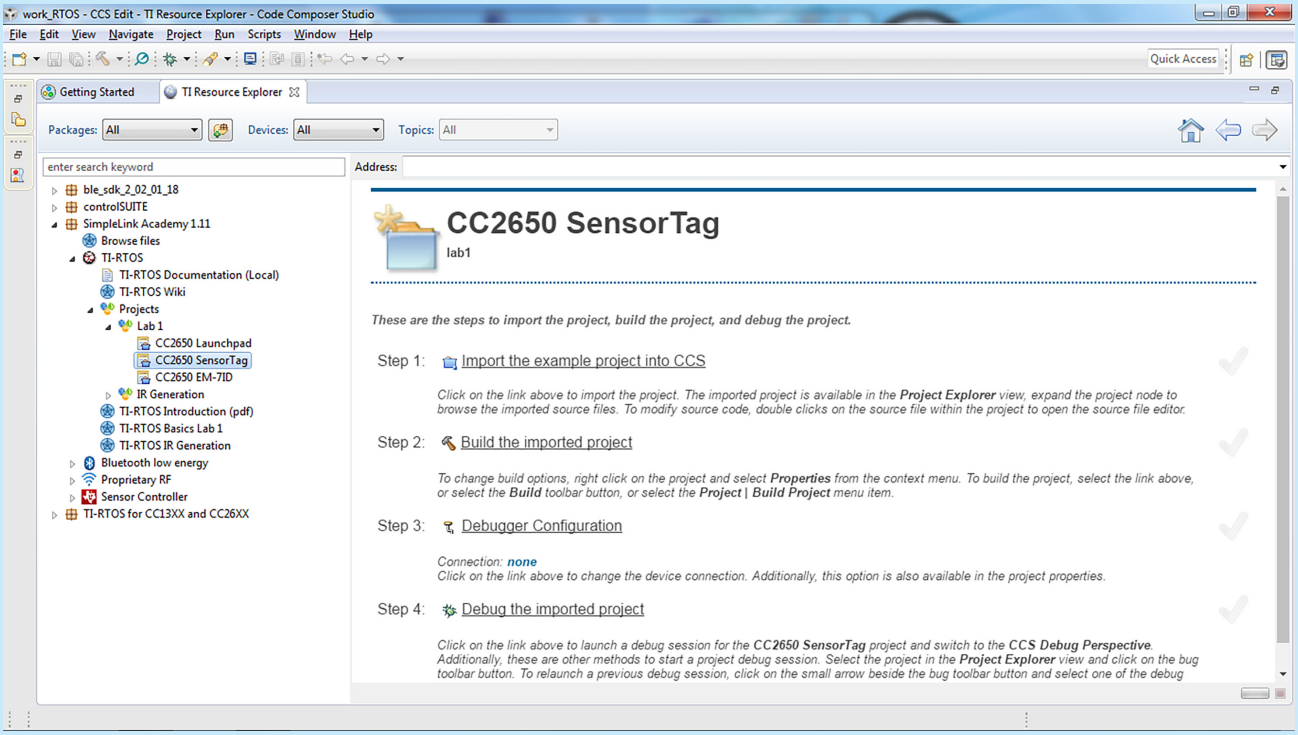
Ćwiczenie pokazuje (krok po kroku) praktyczne używanie podstawowych elementów systemu operacyjnego czasu rzeczywistego TI-RTOS. Jest ono wzorowane na ćwiczeniu *TI-RTOS Basic* w portalu *SimpleLink Academy* [16]. Zostały jednak wprowadzone liczne zmiany i rozszerzenia.

Wymagania sprzętowe

- Do pracy potrzebny jest zestaw CC2650 SensorTag z dołączonym modułem rozszerzeń *Debug DevPack*, połączony z komputerem kablem USB-A USB-Micro.
- Komputer PC z systemem operacyjnym Windows 7 (lub nowszym).

Ćwiczenie można też wykonywać z użyciem modułu uruchomieniowego CC2650 LanuchPad lub modułu CC2650EM Evaluation Module z płytą SmartRF06 Evaluation Board.

Poprzednie części kursu i dodatkowe materiały dostępne są na FTP: ftp://ep.com.pl, user: 44747, pass: 3qwdwa8u



Rysunek 1. Ładowanie projektu *TI-RTOS Basic - Lab 1*

Wymagania programowe

- Zainstalowany program CCS v7 (lub nowszy)
- Zainstalowany program BLE-STACK V2.2.1 (Support for CC2640/CC2650) [7].

Instalowany jest:

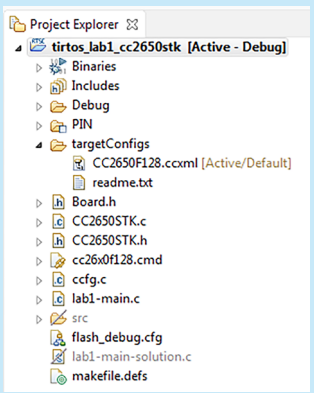
- stos BLE w ścieżce C:\ti\simplelink\ble_sdk_2_02_01_18
- biblioteki systemu operacyjnego TI-RTOS w ścieżce C:\ti\tirtos_cc13xx_cc26xx_2_20_01_08
- biblioteki zewnętrzne tego systemu w ścieżce C:\ti\xdctools_3_32_00_06_core.

- Zainstalowane źródła pakietu *SimpleLink Academyv1.11* w ścieżce C:\ti\simplelink_academy_01_11_00_0000

Istotna jest powyższa kolejność instalowania oprogramowania. Po zainstalowaniu każdego pakietu należy uruchomić CCSv7. Pozwala to na zbudowanie przez CCSv7 bazy projektów przykładowych dostarczanych przez pakiet.

Uruchamianie CCSv7

1. Do zestawu CC2650 SensorTag dołącz moduł rozszerzeń Debug DevPack (dokładniejszy opis w [2]). Połącz go z komputerem kablem USB-A USB-Micro. Na płytce modułu Debug DevPack zaczyna świecić dioda LED, co sygnalizuje, że moduł jest gotowy do pracy.
2. Otwórz Menadżer urządzeń i czekaj, aż zostaną zainstalowane wszystkie drajwery sprzętowe.
3. Uruchom program CCSv7. Kliknij dwukrotnie na jego ikonę.
4. W oknie *Workspace Launcher* pozostaw (lub wpisz) ścieżkę do folderu roboczego, np. <C:\home_dir**work_ART5**>. Kliknij *OK*.
5. Obserwuj informacje na pasku w prawym dolnym rogu. Dotyczą one ładowania modułów środowiska Eclips oraz sprawdzania dostępności aktualizacji. Najlepiej zaczekać na zakończenie tych prac.
6. Zamknij okno *Updates Avilable* (jeśli się pojawi) lub wykonaj aktualizację.



Rysunek 2. Drzewo projektu *tirtos_lab1_cc2650stk (s1)*

Wykorzystanie projektu z poprzedniego odcinka kursu

W poprzednim odcinku kursu został utworzony folder roboczy *C:\home_dir\work_ART5\tirtos_lab1_cc2650stk* z projektem ***tirtos_lab1_cc2650stk***. Teraz można z niego ponownie skorzystać. Środowisko CCSv7 jest już gotowe do pracy. Jeśli masz otworzone środowisko CCSv7 po zakończeniu poprzedniego ćwiczenia, to przejdź od razu do punktu 12.

Ładowanie nowego projektu

Jeśli zaczynamy dopiero pracować z projektem *tirtos_lab1_cc2650stk*, można go teraz załadować w sposób podobny jak w poprzednim odcinku kursu.

7. Otwórz okno *Resource Explorer* z menu *View* → *Resource Explorer Classic*.
8. W zakładce *TI Resource Explorer* rozwiń listę SimpleLink Academy 1.11 → TI-RTOS → Projects → Lab 1.
9. Kliknij na linię ***CC2650 SensorTag***.

Po prawej stronie zostanie wyświetlona instrukcja, jak postępować w czterech krokach (**rysunek 1**).

10. Kliknij na odnośnik kroku 1. Po załadowaniu projektu zostanie pokazana w oknie *Project Explorer* linia projektu ***tirtos_lab1_cc2650stk*** *[Active – Debug]*. Otwierane są także w oknach edycji pliki *lab1-main.c* oraz *lab1-main-solution.c*.
11. Kliknij na zakładkę *TI Resource Explorer*. Zielony znaczek √ pojawi się na prawo od odnośnika kroku 1.

Jeśli znaczek się nie pojawi, należy kliknąć w zakładce *TI Resource Explorer* na linię *CC2650 SensorTag*.

Zadania systemu TI-RTOS

Nowa wersja projektu ma monitorować stan przycisku Board_BUTTON0 co 10 ms i po wykryciu jego przyciśnięcia wykonywać funkcję *doUrgentWork*. Reprezentuje ona watek realizujący najważniejszą

Poprzednie części kursu i dodatkowe materiały dostępne są na FTP: **ftp://ep.com.pl**, user: **44747**, pass: **3qwdwa8u**

część działania. Następna wersja projektu wykorzystuje przerwanie sprzętowe do obsługiwania reakcji na przyciśnięcie przycisku. Drajwer wyprowadzenia (PIN) sam realizuje powiązanie funkcji *callback* z tym zdarzeniem.

Tworzenie nowego zadania

12. W oknie *„Project Explorer”* perspektywy *CCS Edit* rozwiń drzewo projektu *tirtos_lab1_cc2650stk* (**rysunek 2**).
13. W oknie *Project Explorer* kliknij prawym przyciskiem myszki na plik *lab1-main.c* i z podręcznego menu wybierz *Exclude from Building*.
14. W oknie *Project Explorer* kliknij prawym przyciskiem myszki na plik *lab1-main-solution.c* i z podręcznego menu wybierz *Copy*.
15. Kolejny raz kliknij prawym przyciskiem myszki na plik *lab1-main-solution.c* i z podręcznego menu wybierz *Paste*.
16. W oknie *Name Conflict* wpisz nową nazwę pliku ***lab1-main-Task2.c***. Kliknij *OK*.
17. Otwórz plik ***lab1-main-Task2.c***. Kliknij dwukrotnie na linię z jego nazwą.
18. Opatrz komentarzem linie kodu 145-146 (wpisz // na początku każdej linii).
19. Opatrz komentarzem linię 118.
20. Wstaw kopie linii 110 po linii 118 jako nowa linia 119.
21. Skróć czas oczekiwania dwukrotnie.
22. Opatrz komentarzem linię 128.

Kod powinien wyglądać jak na **rysunku 3**.

23. Otwórz plik ***flash_debug.cfg***. Dwukrotnie kliknij na linię z jego nazwą.
24. Na dole okna *flash_debug.cfg* zobacz, czy otworzona jest zakładka *TI-RTOS* (a nie *cfg Script*) (**rysunek 4**).
25. W panelu *Outline* po prawej stronie zaznacz ***LoggingSetup*** (kliknij na linię).
26. W polu *RTOS Execution Analysis* sprawdź, czy są zaznaczone pola: SWI, HWI and Semaphores.

Teraz podczas kolejnego budowania projektu obsługa monitowania tych elementów systemu TI-RTOS będzie dołączona do projektu.

Zbuduj projekt *tirtos_lab1_cc2650stk*

27. W celu zbudowania projektu wybierz z menu *Project* → ***Clean***. Nie używaj przycisku *Build* lub przycisku *Debug*.
28. W oknie *Clean* kliknij na *OK*. Zielony znaczek √ pojawi się na prawo od odnośnika kroku.
29. Jeśli znaczek się nie pojawił, to trzeba w zakładce *TI Resource Explorer* kliknąć na linię projektu *CC2650 SensorTag*.

W projekcie już został wcześniej zdefiniowany typ emulatora *„Texas Instruments XDS110 USB Debug Probe”*. Dlatego zielony znaczek przy odnośniku kroku 3 jest ustawiony.

Debaguj projekt *tirtos_lab1_cc2650stk*

30. Kliknij na przycisk *Debug*.
31. Czekaj, aż kursor w oknie edycji pliku ***lab1-main-Task2.c*** zostanie umieszczony w pierwszej linii funkcji *main()*.

```
109         /* Sleep */
110         Task_sleep(1000 * (1000 / Clock_tickPeriod));
111     }
112 }
113
114 Void urgentWorkTaskFunc(UArg arg0, UArg arg1)
115 {
116     while (1) {
117
118 //          Semaphore_pend(urgentWorkSem, BIOS_WAIT_FOREVER);
119         Task_sleep(1000/2 * (1000 / Clock_tickPeriod));
120
121         /* Do work */
122         doUrgentWork();
123     }
124 }
125
126 void pinInterruptHandler(PIN_Handle handle, PIN_Id pinId)
127 {
128 //    Semaphore_post(urgentWorkSem);
129 }
```

Rysunek 3. Kod po modyfikacjach

Uruchom program tirtos_lab1_cc2650stk

32. Na pasku narzędzi perspektywy *CCS Debug* kliknij na przycisk *Resume*.

Program zaczyna pracować. Dioda LED1 (czerwona) zestawu CC2650STK SensorTag zapala się na 1 sekundę z jednosekundową przerwą.

33. Po pięciokrotnym zaświeceniu diody LED kliknij na przycisk *Suspend* (Halt/Pause). Spowoduje to zatrzymanie działania programu.

Okno RTOS Object View

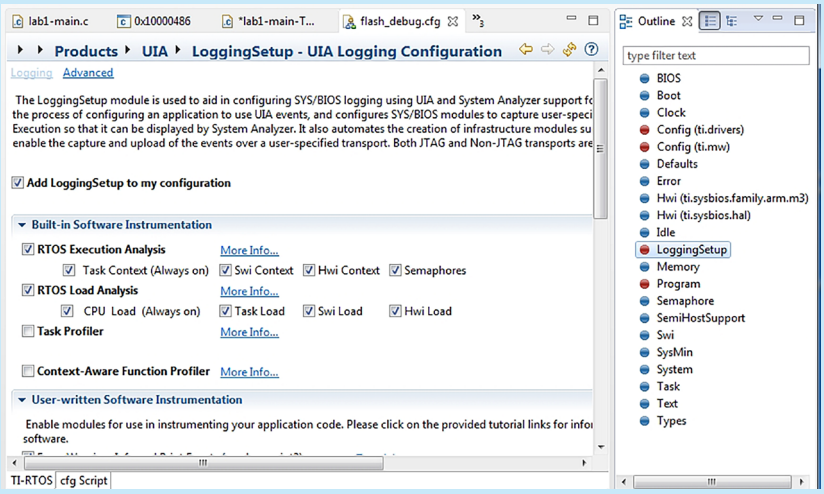
Okno RTOS Object View (lub w skrócie ROV) pozwala na wgląd w stan obiektów systemu TI-RTOS [17]. Informacja jest aktualizowana przez łącze JTAG po zatrzymaniu pracy procesora.

34. Otwórz ROV z menu *Tools* → *RTOS Object View (ROV)*.

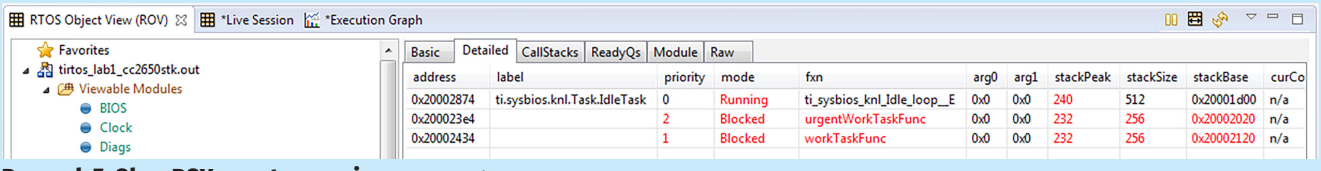
Otwierana jest tabelka z informacją o obiektach systemu TI-RTOS.

35. Zamknij okno *Console*. W zakładce *RTOS Object View (ROV)* rozwiń drzewo *tirtos_lab1_cc2650stk.out*. Wybierz pozycję *Task* i otwórz zakładkę *Detailed* (**rysunek 5**).

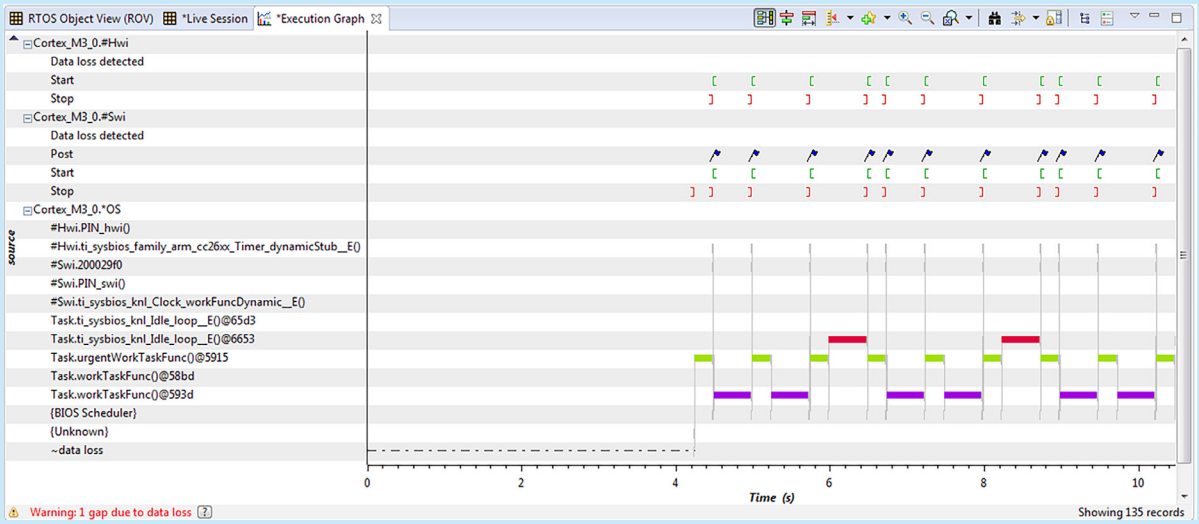
W oknie można zobaczyć, które zadanie jest aktualnie wykonywane (Running), które jest zablokowane (Blocked), a które jest gotowe (Ready). Obecnie jest widoczne zadanie tła – IdleTask (uruchomione) i możemy sprawdzić wielkość jego stosu (512) oraz jego



Rysunek 4. Okno edycji pliku *flash_debug.cfg*



Rysunek 5. Okno ROV po zatrzymaniu pracy programu



Rysunek 6. Okno *Execution Graph* po zatrzymaniu pracy programu

maksymalny rozmiar (240). Są jeszcze dwa zadania, które są zablokowane.

Okno Execution graph

Okno *Execution Graph* pokazuje informacje o stanie systemu TI-RTOS podczas pracy w czasie [15]. Lista wątków jest pokazana w lewej kolumnie. Uwidoczniona jest aktywność wątków zdefiniowanych przez użytkownika (HWI, SWI, Semaphore), jego zadań (Task) oraz wątków systemu operacyjnego. Kolorowa linia pokazuje, kiedy wątek jest aktywny. Dokładniejszy opis jest zamieszczony w dokumencie [15].

36. Z menu wybierz *Tools* → *RTOS Analyzer* → *Execution Analysis*.
37. W oknie *Analysis Configuration* zaznacz tylko *Execution Graph*, pozostaw resztę ustawień bez zmian i naciśnij przycisk *Start*.
38. W nowym oknie zakładki *Execution Graph* rozwiń widoczność wątków *Cortex_M3_0.*OS* (**rysunek 6**).

W oknie zakładki *Execution Graph* kliknij na linię czasu (Time (s)), przytrzymaj i kółkiem myszy zmień skalę wykresu tak, aby zobaczyć obraz podobny do pokazanego na rysunku 6. Teraz są aktywne naprzemiennie dwa zadania – *WorkTaskFunc* oraz *urgentWorkTaskFunc*. Pomiędzy nimi jest aktywowane zadanie tła *knl_Idle_loop*.

Symbole stosowane na wykresie okna *Execution Graph*

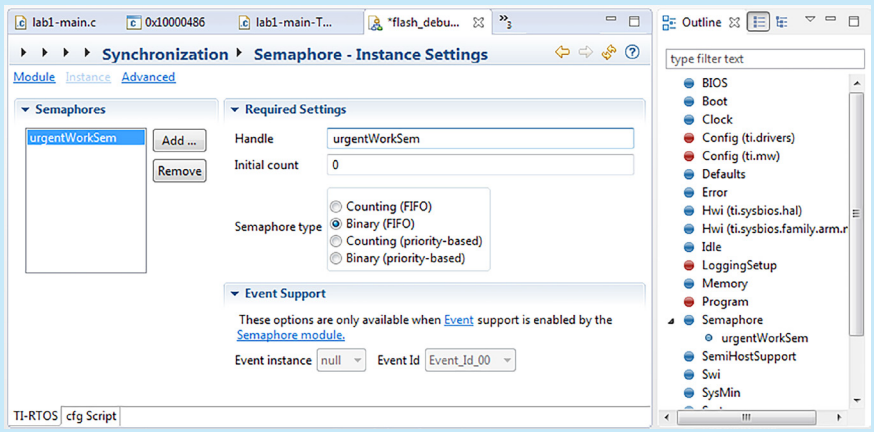
Przerwania sprzętowe. Nawiasy „[” oraz „]” oznaczają początek i koniec pracy wątku. Jeśli wystąpi utrata danych, wtedy pokazywana jest ikonka .

Przerwania programowe. Ikonka informuje, że wątek został aktywowany (post). Nawiasy „[” oraz „]” oznaczają początek i koniec pracy wątku.

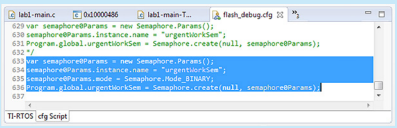
Tworzenie semafora

Teraz zostanie utworzony semafor w sposób statyczny przy zastosowaniu pliku konfiguracyjnego *flash_debug.cfg*.

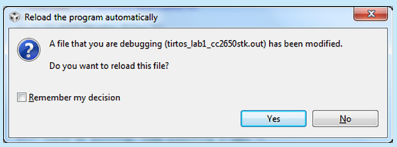
39. Przejdź do perspektywy *CCS Edit*.
40. Otwórz okno edycji pliku ***flash_debug.cfg***.
41. W panelu *Outline* po prawej stronie zaznacz ***Semaphore*** (kliknij na linię).
42. W menu okna wybierz zakładkę *Instance*.
43. Kliknij na przycisk *Add*.
44. W linii *Handle* wpisz *urgentWorkSem*.



Rysunek 7. Okno edycji pliku *flash_debug.cfg* podczas tworzenia semafora



Rysunek 8. Dodane linie tworzenia semafora w pliku *flash_debug.cfg*



Rysunek 9. Okno ładowania automatycznego

45. W polu *Semaphore type* wybierz typ *Binary (FIFO)* (**rysunek 7**).
46. Zapisz plik *flash_debug.cfg* z menu File → Save lub klinij na ikonkę *Save*.

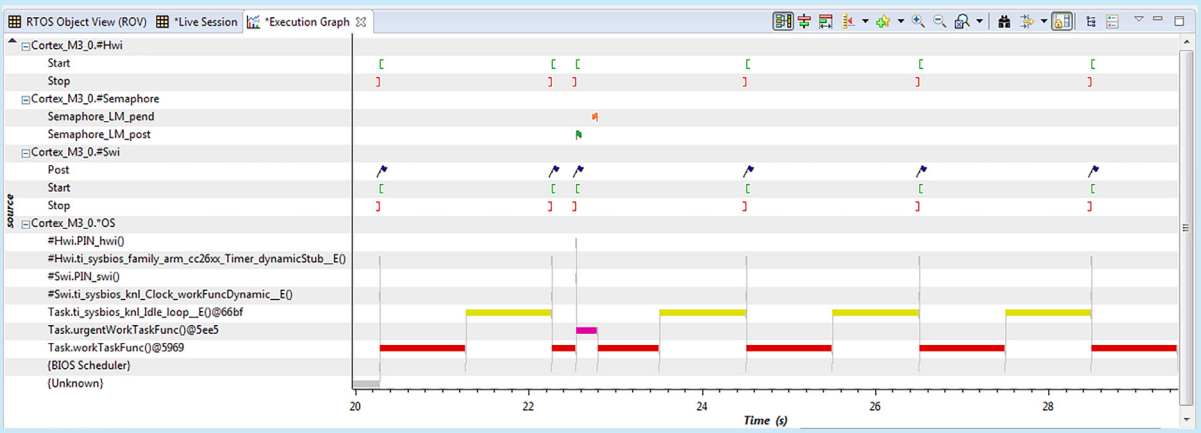
Nowo utworzony semafor *urgentWorkSem* jest dostępny w globalnej przestrzeni nazw (symboli adresowych) projektu. Zostanie on automatycznie użyty podczas budowania projektu.

47. W oknie edycji pliku *flash_debug.cfg* kliknij na zakładkę *cfg Script*).
48. Przejdź na koniec pliku.

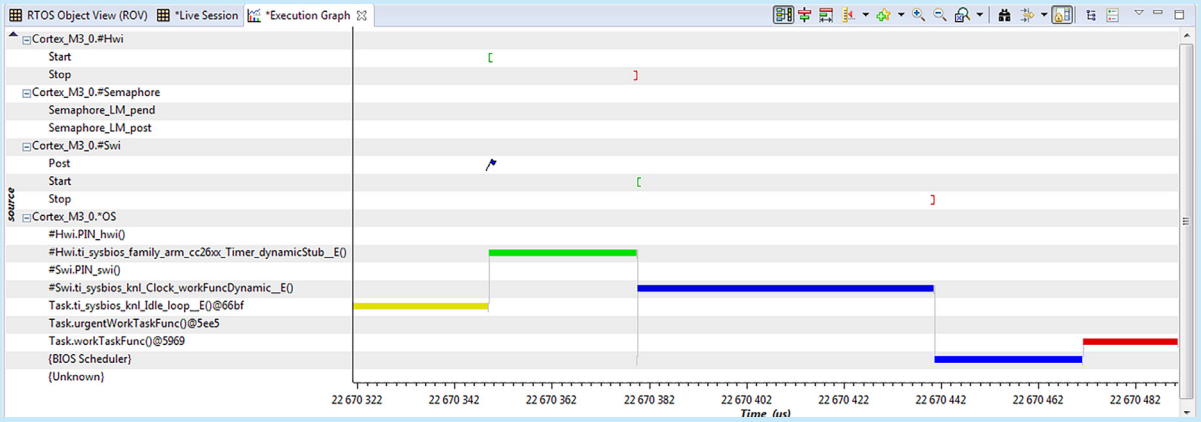
Właśnie zostały dodane nowe cztery linie do pliku *flash_debug.cfg* (**rysunek 8**).

Używanie semafora

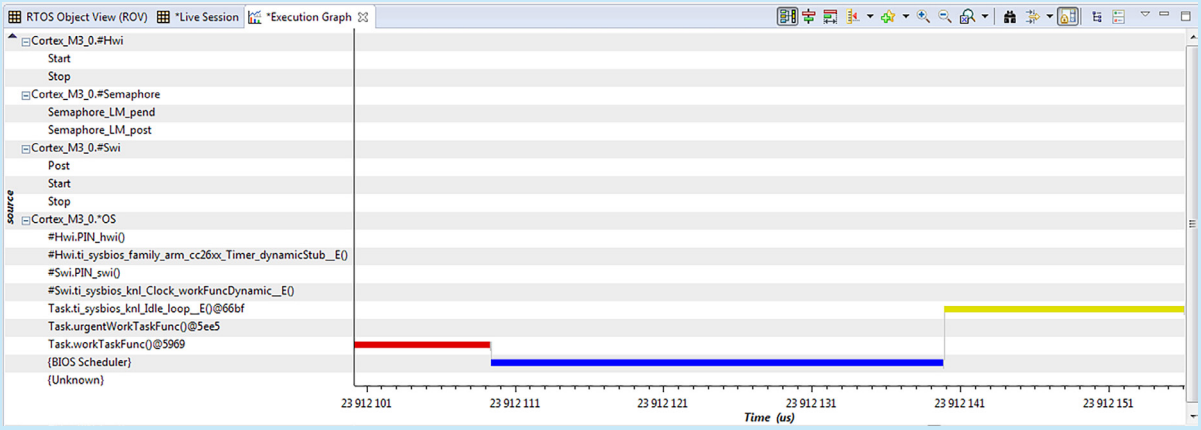
49. W oknie *Project Explorer* kliknij prawym klawiszem myszki na plik *lab1-main-Task2.c* i z podręcznego menu wybierz *Exclude from Building*.



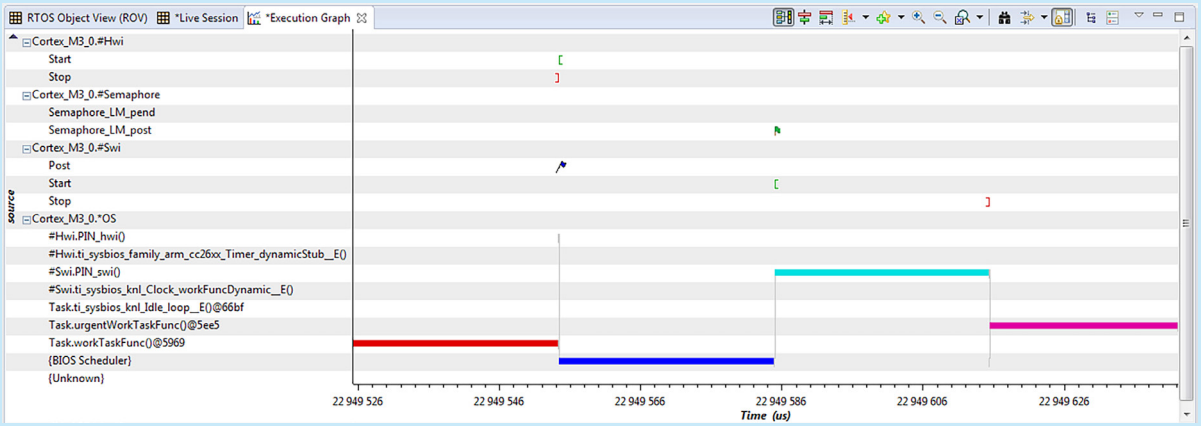
Rysunek 10. Okno *Execution Graph* po zatrzymaniu pracy programu *lab1-main-solution.c*



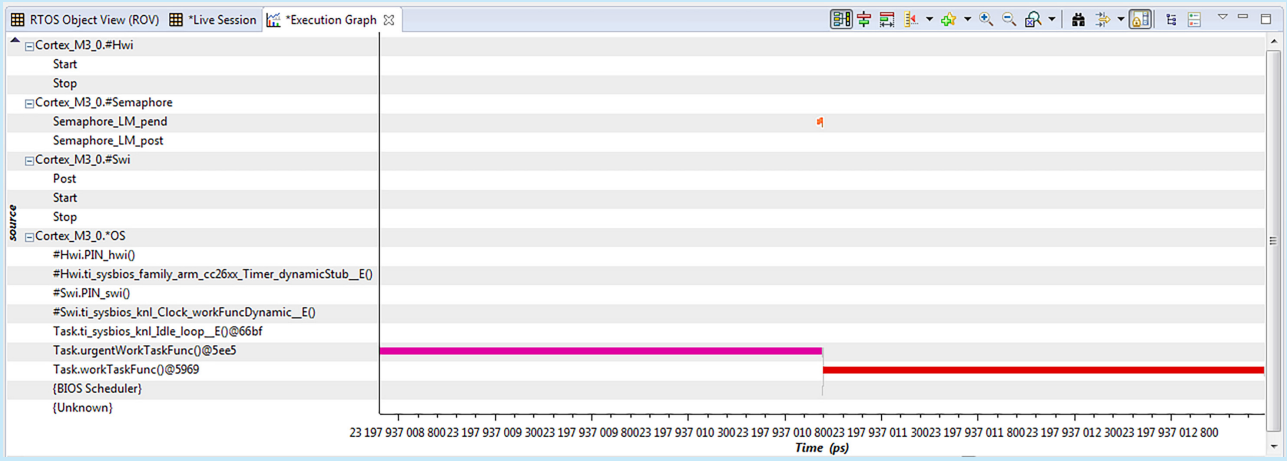
Rysunek 11. Początek pracy zadania *workTaskFunc*



Rysunek 12. Koniec pracy zadania *workTaskFunc*



Rysunek 13. Przyciśnięcie przycisku



Rysunek 14. Zwolnienie przycisku

50. W oknie *Project Explorer* kliknij prawym klawiszem myszki na plik *lab1-main-solution.c* i z podręcznego menu wybierz *Exclude from Building*. Oznacza to włączenie tego pliku do budowania.

Teraz przerwanie sprzętowe HWI tylko ustawia semafor, co powoduje odblokowanie oczekującego zadania *urgentWorkTaskFunc*.

51. Zbuduj projekt. Wybierz z menu *Project* → ***Clean***. Nie używaj przycisku *Build* lub przycisku *Debug*.
52. W oknie *Clean* kliknij na *OK*.
53. Po wyświetleniu okna *Reload the program automatically* kliknij przycisk *Yes* (**rysunek 9**).
54. Czekaj, aż kursor w oknie edycji pliku ***lab1-main-solution.c*** zostanie umieszczony w pierwszej linii funkcji *main()*.
55. Przejdź do perspektywy *CCS Debug*.

Uruchom ponownie program tirtos_lab1_cc2650stk

56. Na pasku narzędzi perspektywy *CCS Debug* kliknij na przycisk *Resume*.
57. Gdy dioda LED zostanie zaświecona, drugi raz przyciśnij i zwolnij prawy przycisk modułu CC2650 SensorTag.

Literatura:

1. Systemy dla Internetu Rzeczy (1). Zestaw CC2650 SensorTag, „Elektronika Praktyczna”, 12/2016
2. Systemy dla Internetu Rzeczy (2). Użytkowanie zestawu CC2650 SensorTag, „Elektronika Praktyczna”, 1/2017
3. Systemy dla Internetu Rzeczy (3). Moduły rozszerzeń DevPack dla zestawu SensorTag, „Elektronika Praktyczna”, 2/2017
4. Systemy dla Internetu Rzeczy (4). Zestaw CC1310 LaunchPad, „Elektronika Praktyczna”, 3/2017
5. Systemy dla Internetu Rzeczy (5). System operacyjny czasu rzeczywistego TI-RTOS – pierwszy program, „Elektronika Praktyczna”, 4/2017
6. TI-RTOS: Real-Time Operating System (RTOS) for Microcontrollers (MCU), ***https://goo.gl/34YH5f***
7. BLE-STACK V2.2.1 (Support for CC2640/CC2650) v2.2.1, 28-OCT-2016, ***https://goo.gl/00dsyZ***
8. TI-RTOS 2.20 for CC13xx/CC26xx SimpleLink Getting Started Guide (SPRUHU7D.pdf), 17 Jun 2016, ***https://goo.gl/f199w8***
9. SYS/BIOS (TI-RTOS Kernel) v6.46 User's Guide (SPRUEX3Q.pdf), 16 Jun 2016, ***https://goo.gl/v6wXTd***
10. CC2640/CC2650 Bluetooth low energy Software Developer's Guide (SWRU393D.pdf) 15 October 2016, ***https://goo.gl/6mka3t***
11. TI-RTOS 2.20 User's Guide (SPRUHD4M.pdf), 17 Jun 2016, ***https://goo.gl/NW3s85***
12. CC2640/CC2650 Getting Started and FAQ, 2016 Oct 31, ***https://goo.gl/UIP40E***
13. TI-RTOS (TI WIKI), ***https://goo.gl/K1FTyv***
14. Category:SYSBIOS (TI WIKI), ***https://goo.gl/jlDFap***
15. System Analyzer User's Guide (SPRUH43.pdf) (ver. „F” March 2014), ***https://goo.gl/Trk7dp***
16. SimpleLink Academy (v1.11 – November 4th 2016), ***https://goo.gl/vBP9Nq***
17. Runtime Object Viewer, Using ROV for Eclipse-Based Debugging, ***https://goo.gl/rgY25o***

58. Po pięciokrotnym zaświeceniu się diody LED kliknij na przycisk *Suspend* (Halt/Pause).
59. Zobacz wykres w oknie zakładki *Execution Graph* (**rysunek 10**).
60. W oknie zakładki *Execution Graph* rozciągnij wykres w okolicach przełączania z systemowego zadania tła do zadania *workTaskFunc* (**rysunek 11**).

Zakończenie zliczania czasu uśpienia powoduje zgłoszenie przerwania sprzętowego. Wątek przerwania sprzętowego aktywuje funkcję zegara. Po jego zakończeniu startuje praca wątku przerwania programowego (zegarowego) i następnie sterowanie jest przekazywane do programu szeregującego zadania (scheduler). Zadanie *workTaskFunc* jest gotowe i zostaje uruchomione – zaczyna świecić dioda LED.

61. W oknie zakładki *Execution Graph* rozciągnij wykres w okolicach przełączania z systemowego zadania *workTaskFunc* do zadania tła (**rysunek 12**).

Zadanie *workTaskFunc* kończy pracę – dioda LED gaśnie. Zadanie zostaje zablokowane i program szeregujący zadania uruchamia zadanie tła.

62. W oknie zakładki *Execution Graph* rozciągnij wykres w okolicach przyciśnięcia przycisku (**rysunek 13**).

Widoczna jest cała sekwencja działań. Zadanie *workTaskFunc* pracuje normalnie. Pojawia się przerwanie sprzętowe HWI, które powołuje przerwanie programowe SWI. Po zakończeniu pracy HWI startuje program szeregujący, który uruchamia oczekujące SWI. SWI ustawia semafor i kończy pracę. Powoduje to zmianę stanu zadania *urgentWorkTaskFunc* z zablokowanego do gotowego. Uruchamiane jest zadanie oczekujące o najwyższym priorytecie, czyli zadanie *urgentWorkTaskFunc*.

63. W oknie zakładki *Execution Graph* rozciągnij wykres w okolicach zwolnienia przycisku (**rysunek 14**).

Po zwolnieniu przycisku stan semafora zostaje zmieniony i zadanie *urgentWorkTaskFunc* blokuje się i kończy pracę. Uruchamiane jest zadanie oczekujące o najwyższym priorytecie, czyli zadanie *workTaskFunc*.

Podsumowanie

Ćwiczenia 5 i 6 kursu „Systemy dla Internetu Rzeczy” pozwalają na praktyczne zapoznanie się z aplikacją pracującą z wykorzystaniem systemu operacyjnego czasu rzeczywistego. Kolejne modyfikacje projektu pozwalały na poznawanie zastosowania zadań, przerwań i semaforów. Cenną zaletą systemu TI-RTOS jest możliwość graficznego wglądu w pracę wątków z bezinwazyjnym podglądem zależności czasowych ich pracy.

Henryk A. Kowalski
kowalski@ii.pw.edu.pl

Poprzednie części kursu i dodatkowe materiały dostępne są na FTP: **ftp://ep.com.pl**, user: **44747**, pass: **3qwdwa8u**