Lab. 2. Priorytety przerwań.

Poprawnie oprogramowany (*lab2.c*) mikrokontroler wykonuje trzy główne zadania:

- W pętli głównej (*main*): inkrementowana jest zmienna *main_var*. Wartości *main_var* oraz *int_var* po konwersji na bitmapy są przesyłane interfejsem *SPI* do sterownika wyświetlacza *LCD*. Opóźnienie wykonywania pętli (0.5 s) odmierzane jest licznikiem *TIM6*.
  Opcjonalnie: wymuszenie stanu niskiego linii wejściowej nr 2 portu G wstrzymuje wykonywanie pętli głównej aż do powrotu stanu wysokiego (jest to symulacja „zawieszenia się" programu głównego).

- Ze stałą częstotliwością, tj. w chwili przepełnienia licznika *TIM7*, wywoływana jest *ISR* zmieniająca stan linii 13 portu G na przeciwny (sygnalizator: zielona *LED*).
  Okresowo wywoływana *ISR* może np. sterować liniami multipleksowanego wyświetlacza, skanować rzędy klawiatury, służyć do odmierzania czasu (inkrementacja zmiennej *int_var*), itp.

- Reaguje na pojawienie się stanu wysokiego na linii 0 portu A zgłoszeniem przerwania zewnętrznego *EXTI0* (reakcja na zbocze rosnące: 0->1). Wywołana *ISR* na czas trwania stanu wysokiego na linii A0 wymusza stan wysoki linii 14 portu G (sygnalizator: czerwona *LED*).
  Wyjście z *ISR* zostaje przyblokowane aż do powrotu linii A0 do stanu niskiego. Przed powrotem z *ISR* linii G14 przywrócony zostaje stan niski. Symuluje to wydłużenie czasu wykonywania *ISR EXTI0*, lub wręcz „zawieszenie się" tej procedury.

Zadania:

1. Uzupełnić, skompilować i uruchomić program (punkty 1. i 2.) z parametrami podanymi przez prowadzącego, następnie zbadać zachowanie układu.

> Sprawdzić: czy zablokowanie *ISR EXTI0* wpływa na wykonanie pętli głównej oraz *ISR TIM7*?

Opcjonalnie: czy zablokowanie pętli głównej (dopóki linia G2 znajduje się w stanie niskim) wpływa na wykonywanie procedur obsługi przerwań?

2. Ustawić priorytety przerwań (p. 3) tak, aby w żadnym przypadku *ISR TIM7* nie została zablokowana – np. „zawieszeniem się" *ISR EXTI0* (co może w realnym świecie skutkować np. uszkodzeniem wyświetlacza na skutek przepływu przez diody *DEL* prądu o zbyt dużym natężeniu, przez zbyt długi czas).

Priorytety nadaje się w zakresie od 0 do 15 ustawiając starsze 4 bity (bajtu) w odpowiednim rejestrze *IP[n]* kontrolera *NVIC* (opis w *Programming Manual*, rozdział 4.3.7). Np. `NVIC->IP[20] = 0x90;` oznacza nadanie priorytetu 9 przerwaniu o numerze 20 (CAN1_RX0). **Uwaga: im mniejsza wpisana do rejestru liczba, tym wyższy priorytet.**

> Sprawdzić zachowanie programu po zmianie priorytetów przerwań. Jak teraz przebiega wykonanie pętli głównej oraz *ISR TIM7* podczas „blokady" *ISR EXTI0*?

Opcjonalnie: sprawdzić blokowanie wykonywania pętli głównej dopóki linia G2 znajduje się w stanie niskim. Jak wpływa zablokowanie pętli głównej na wykonywanie obu *ISR*?

3. Zwiększyć częstotliwość przepełnień TIM7 (>= 4 Hz) i obserwować zmiany wartości *main_var* i *int_var*.

4. Zadanie dodatkowe. Jeżeli program będzie działał zgodnie z ww. założeniami, należy uruchomić zabezpieczenie – układ czuwający *Independent Watchdog* wpisem: `IWDG->KR = 0xCCCC;`. Następnie odblokować możliwość konfiguracji (`IWDG->KR = 0x5555;`), ustawić preskaler (rozdział 21.3.3 i tablica 108 w *Reference Manual*) i zbadać zachowanie urządzenia.

> Wybrać (**jedno!**) odpowiednie miejsce przeładowania *watchdoga* (wpisem: `IWDG->KR = 0xAAAA;`); są trzy możliwości: w *main* – w pętli głównej, w *ISR TIM7*, w *ISR EXTI0*.

Miejsce przeładowania należy wybrać tak, żeby **zablokowanie jakiejkolwiek z powyższych funkcji skutkowało zadziałaniem zabezpieczenia,** natomiast w trakcie poprawnej pracy *reset* nie występował.

Wyjaśnienia skrótów:

EXTIn – External Interrupt – zewnętrzny sygnał zgłoszenia przerwania podawany na *n*-tą linię wejściową portu GPIO mikrokontrolera.

GPIO – General Purpose Input/Output – uniwersalny port wejścia/wyjścia. W przypadku STM32F4xx porty GPIO mają po 16 niezależnych linii.

IP[n] – n-th Interrupt Priority – tutaj: rejestr przechowujący priorytet *n*-tego przerwania.

ISR – Interrupt Service Routine, IRQ Handler, Interrupt Request Handler – procedura obsługi przerwania.

LED – Light Emitting Diode – dioda elektroluminescencyjna (pot. świecąca) przyrząd półprzewodnikowy, który spolaryzowany w kierunku przewodzenia emituje energię świetlną (niekoniecznie w zakresie widzialnym).

LCD – Liquid Crystal Display – wyświetlacz ciekłokrystaliczny.

NVIC – Nested Vectored Interrupt Controller – kontroler przerwań (zintegorwany w strukturze STM32F4xx).

SPI – Serial Peripheral Interface – interfejs szeregowy, często stosowany do komunikacji między blokami systemu mikroprocesorowego (najczęściej wewnątrz urządzenia).

Watchdog – układ nadzorujący prawidłowe wykonywanie programu (działanie urządzenia). Tutaj: licznik zliczający od zadanej wartości w dół (do zera, z ustaloną częstotliwością). Wyzerowanie licznika wywołuje reset procesora. W poprawnie działającym urządzeniu licznik musi być (w określonym miejscu programu) cyklicznie przeładowywany wartością początkową w odstępach czasu krótszych niż czas potrzebny do jego wyzerowania. „Zawieszenie się” programu (w dowolnej jego części) bądź uszkodzenie sprzętowe powinno skutkować nieprzeładowaniem licznika na czas i wymuszeniem resetu systemu. Podczas resetu wszystkie podzespoły wykonawcze powinny pozostawać w stanie bezpiecznym (np. silnik i grzałka – wyłączone). Generalnie raz uruchomionego watchdoga nie można programowo zatrzymać.