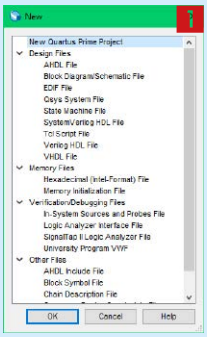
Pierwsze kroki z FPGA (6)

MAXimator jak Arduino – implementacja 32-bitowego mikroprocesora: FPGA dla programistów

Układy FPGA – jak się przekonają Czytelnicy tego artykułu – są w stanie doskonale zastąpić mikrokontrolery i mikroprocesory, nie tracąc przy tym swojej głównej zalety – możliwości elastycznego skonfigurowania sprzętu w sposób wymagany przez użytkownika. Dzięki przedstawionemu projektowi możesz – będąc programistą – całkiem sprawnie wykorzystać znaczną część możliwości i – przy tym – niewielką część zasobów logicznych FPGA MAX10.

Bezpłatna wersja pakietu Quartus Prime to bardzo zaawansowane środowisko dedykowane do obsługi układów programowalnych firmy Altera, przy jego użyciu można implementować projekty cyfrowe w układach PLD (FPGA i CPLD) na różne sposoby. Na przykład, można je budować z bramek logicznych lub bardziej rozbudowanych elementów bibliotecznych, można tworzyć opisy implementowanego sprzętu za pomocą języków opisu sprzętu (VHDL, Verilog lub AHDL), wiele projektów można „wyklikać" za pomocą wbudowanego w oprogramowanie narzędzia Qsys.

W artykule przedstawimy krok po kroku, jak skonfigurować bezpłatny IP core procesora NIOS II i jak go zaimplementować w zestawie MAXimator, a następnie – jak uruchomić prosty program napisany w C. Zaznaczam, że nie trzeba znać VHDL-a ani Verilog-a aby skorzystać z układów, wystarczy znać język programowania C – raj dla programistów!



Tworzenie nowego projektu

Ten etap realizacji projektu przedstawialiśmy już w cyklu „Szkoła MAXimatora", opiszemy go ponownie

Poprzednie części kursu i dodatkowe materiały dostępne są na FTP: ftp://ep.com.pl, user: 66532, pass: 8nnjjeaa

z myślą o Czytelnikach, którzy właśnie teraz zamierzają dołączyć do grona fanów FPGA.

Po uruchomieniu programu Quartus z menu wybieramy *File* → *New...* 1

Następnie wybieramy *New Quartus Prime Project* i naciskamy OK. Pojawia się okienko *New Project Wizard*, w którym klikamy *Next*, wybieramy katalog, w którym będzie projekt, wpisujemy nazwę projektu i ponownie klikamy *Next*. 2

W okienku *Project Type* zostawiamy domyślnie wybraną opcję *Empty project* i klikamy Next, na następnym oknie – *Add Files* – także *Next*. W okienku *Family, Device & Board Settings* wybieramy *Family – MAX 10* i na liście *Available devices* wybieramy układ 10M08DAF256C8GES, zatwierdzając wybór poprzez kliknięcie *Finish*.

Konfigurowanie NIOS-a

Zaczynamy od uruchomienia wbudowanego w pakiet Quartus Prime narzędzia Qsys, co odbywa się poprzez wybranie z menu *Tools* → *Qsys*. W okienku *IP Catalog* wpisujemy NIOS, na samym dole listy wybieramy *Nios II Processor* i klikamy *+Add...* 3

Pojawi się okienko konfiguracji procesora, w którym wybieramy Nios II/e (jedna z trzech konfiguracji tego procesora, najbardziej dopasowana do zasobów MAXi-matora) i klikamy *Finish*. 4

W ten sam sposób dodajemy pamięć RAM i wpisujemy w pole *Total memory size* wartość 32768, co określa pojemność pamięci programu. 5 6

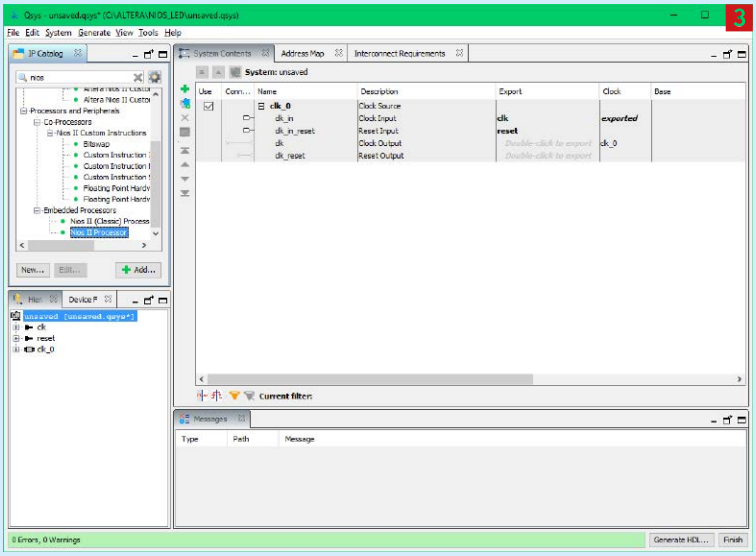
Następnie dodajemy i konfigurujemy timer. 7

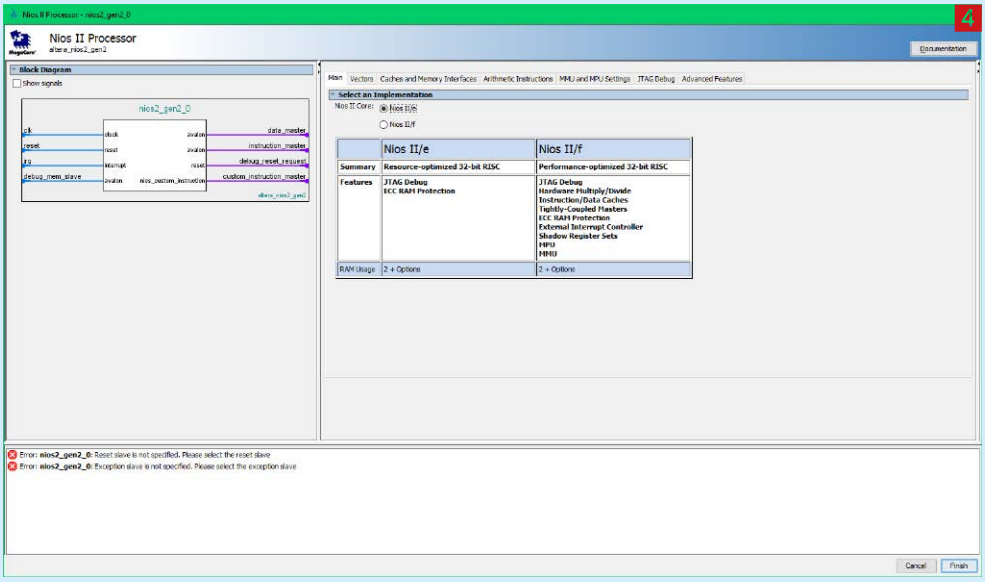
Na potrzeby przykładowego programu w pole *Period* wpisujemy wartość 5000000 (5 milionów) i zmieniamy wartość parametru *Units* na *clocks*. 8

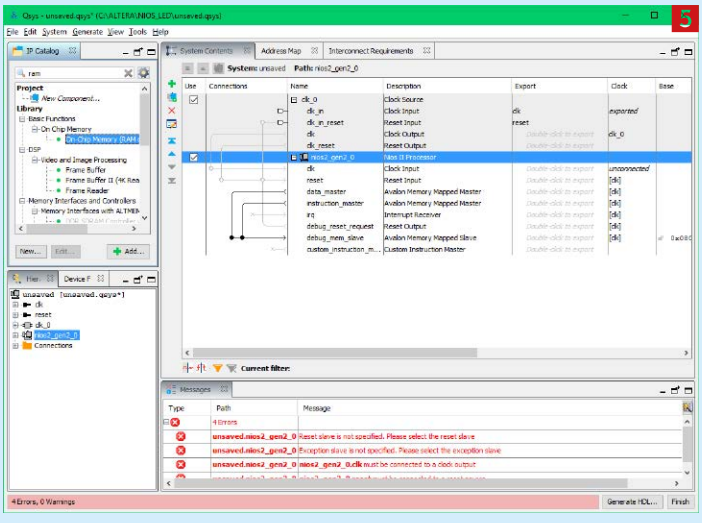
Do rdzenia dodajemy także wyjście PIO – jak na rysunku poniżej. 9

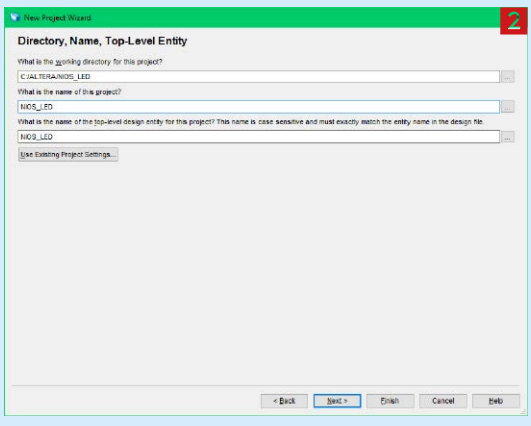
Konfigurujemy je jako wyjście 4 bitowe. 10

W kolejnym kroku musimy połączyć ze sobą wszystkie dotychczas „wyklikane" elementy, które łącznie stworzą nam pożądaną konfigurację mikroprocesorową. Należy więc:


3


4


5



2

- z *nios2_gen2_0 data_master* podpiąć linie do wszystkich s1,
- z *nios2_gen2_0 instruction_master* podpiąć linie do s1 przy *onchip_memory*,
- z *clk_0* łączymy wszystkie *clk* oraz z *clk* w innych modułach, a także *clk_reset* z *reset* w innych modułach,
- w *nios2_gen2_0* podpinamy wszystkie „resety" do *debug_reset_request*,
- z *nios2_gen2_0* podłączamy IRQ z timera,
- przy *pio_0 przy externam_connection* klikamy dwukrotnie szary napis *Double-click to export* i wpisujemy *led*,

z menu wybieramy *System* → *Assign Base Addresses* i klikamy prawym klawiszem myszy na *nios2_gen2_0* wybierając *Edit*, następnie w zakładce *Vectors* zmieniamy *Reset vector memory* oraz *Exception vector memory* na *onchip_memory2_0.s1*. 11

Po wykonaniu tych czynności zapisujemy projekt jako *NIOS_LED*, kompletny projekt powinien wyglądać tak: 12

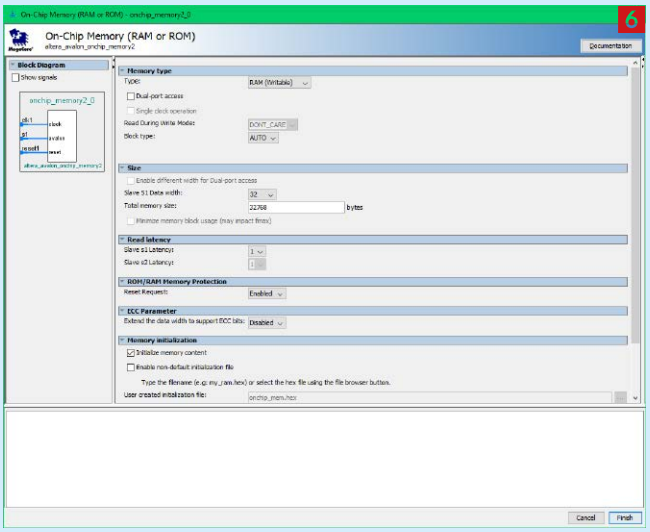
Po kliknięciu *Finish* pakiet Quartus wygeneruje opis sprzętu (skonfigurowanego i wyposażonego w wybrane peryferia rdzenia NIOS), który trzeba zaimportować w głównym projekcie. W tym celu z menu głównego wybieramy *Project* → *Add/Remove Files in Project...* Następnie, przy *File name* klikamy *[...]* i wybieramy: *NIOS_LED/synthesis/NIOS_LED.qip*, po czym klikamy *Add*, *Apply* oraz OK.

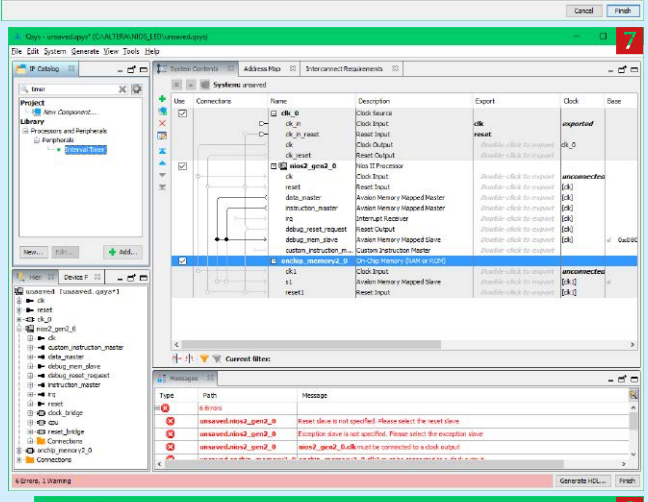
Następnie możemy skompilować projekt, ale – niestety – brakuje jeszcze programu na procesora NIOS oraz nie są podpięte do wyjść diody LED, w związku z czym pojawią się komunikaty błędów 13

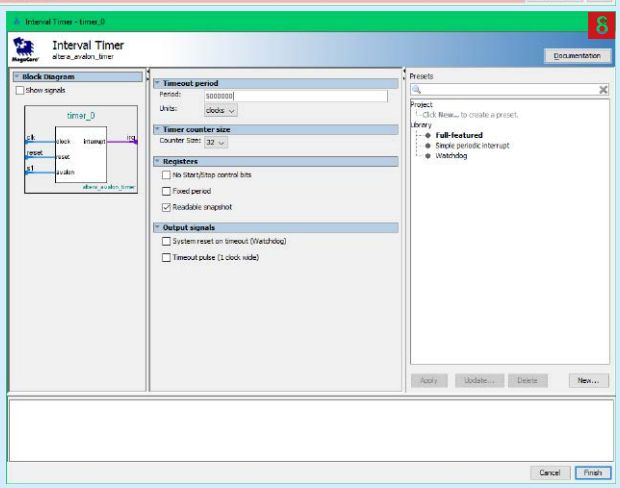
Konfigurowanie wyprowadzeń

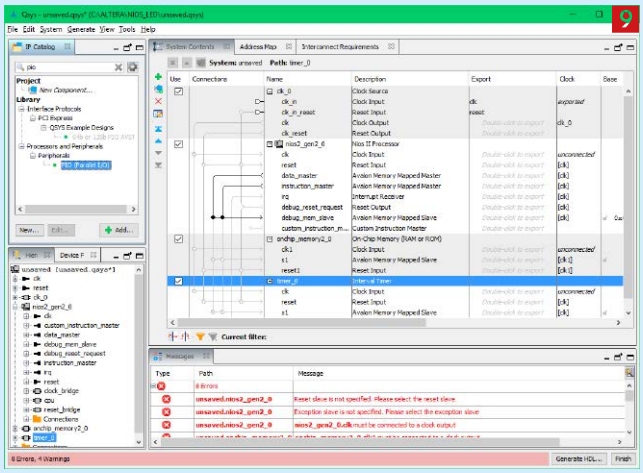
Żeby przypisać sygnały do fizycznych wyprowadzeń uruchamiamy *Pin Planner* (wybierając z menu *Assigments* → *Pin Planner*), następnie podłączamy pokładowe LED-y oraz zegar (sygnał taktujący), przeciągając z dolnej tabelki (*Node Name*) wybrany sygnał na odpowiedni pin układu:

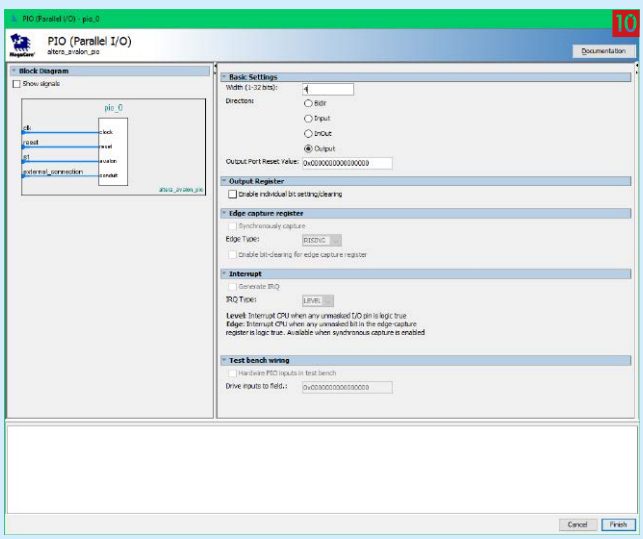
```
clk_clk PIN_L3
led_export[3] PIN_R16
led_export[2] PIN_P16
```





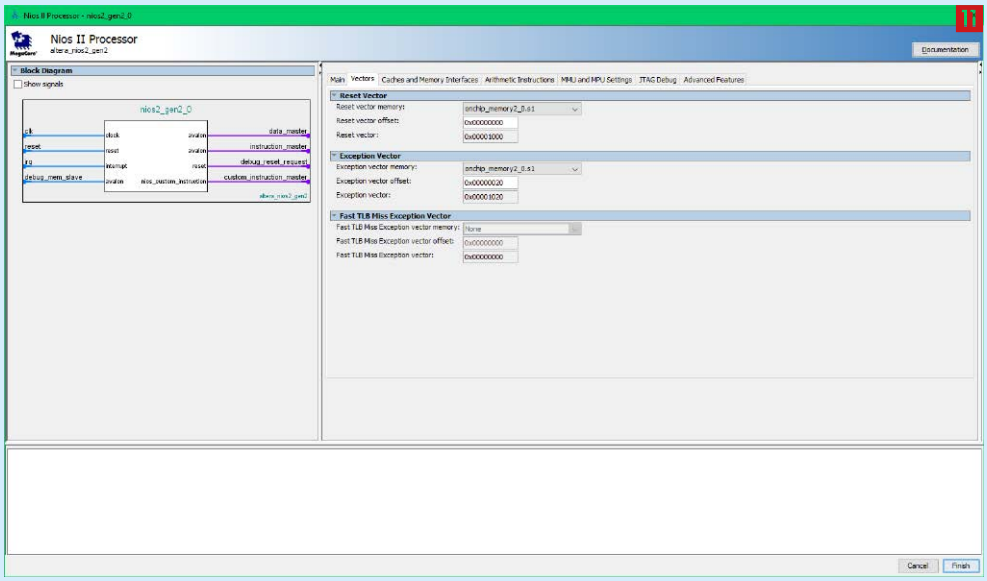






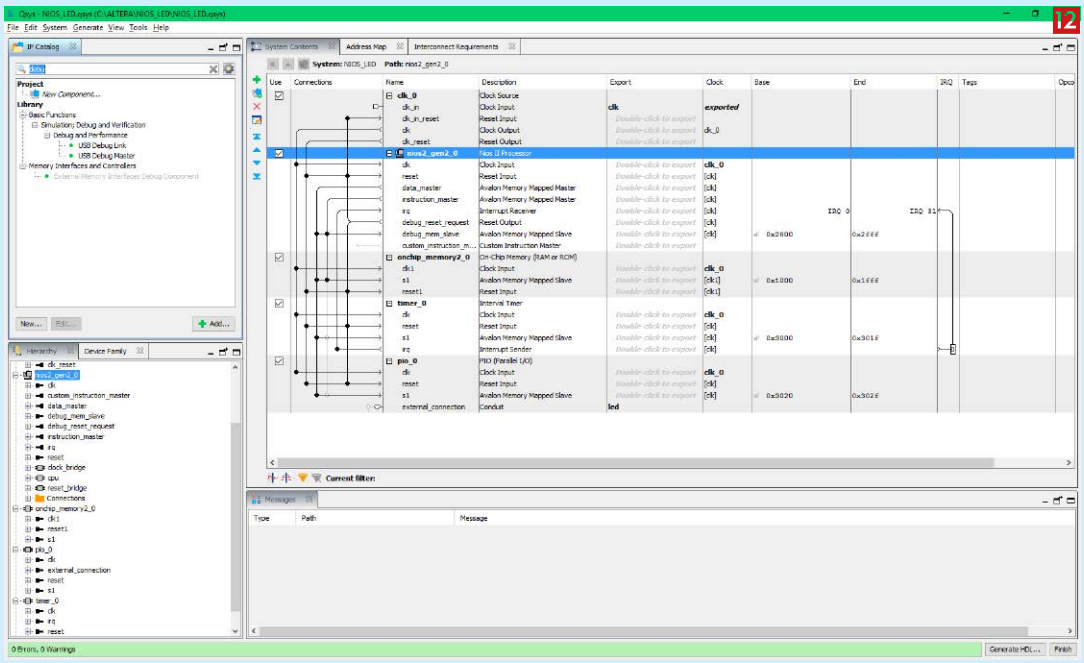
```
led_export[1]
PIN_N16
led_export[0]
PIN_M16
```

Po wykonaniu tych czynności nie ma konieczności zapisywania wprowadzonych modyfikacji, wystarczy zamknąć narzędzie *Pin Planner*. 14

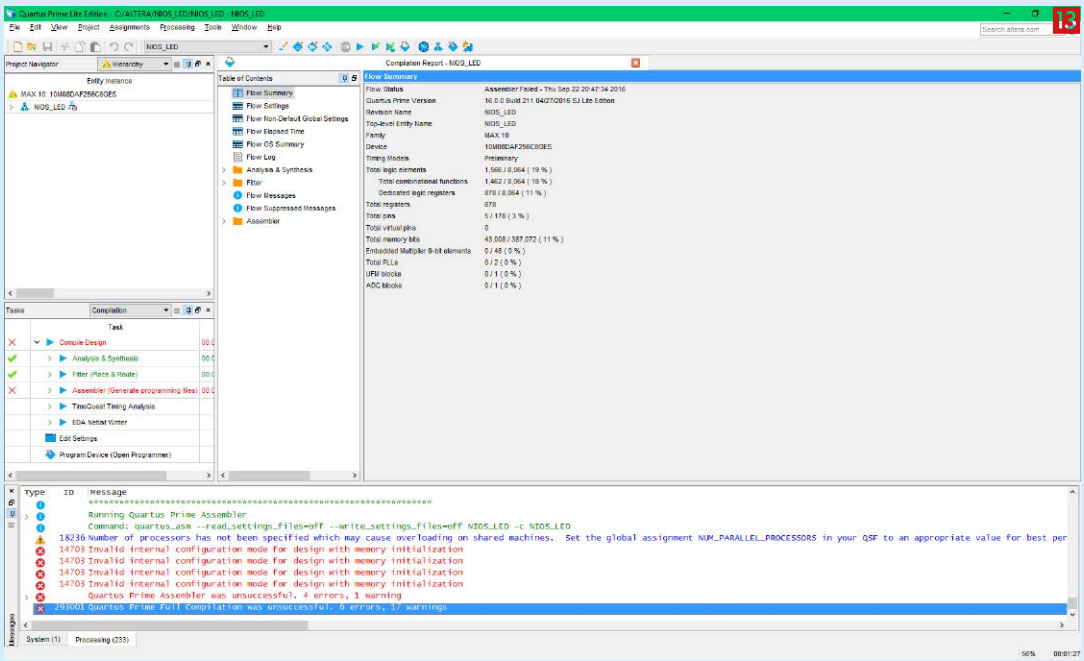

11

Tworzenie programu dla procesora NIOS

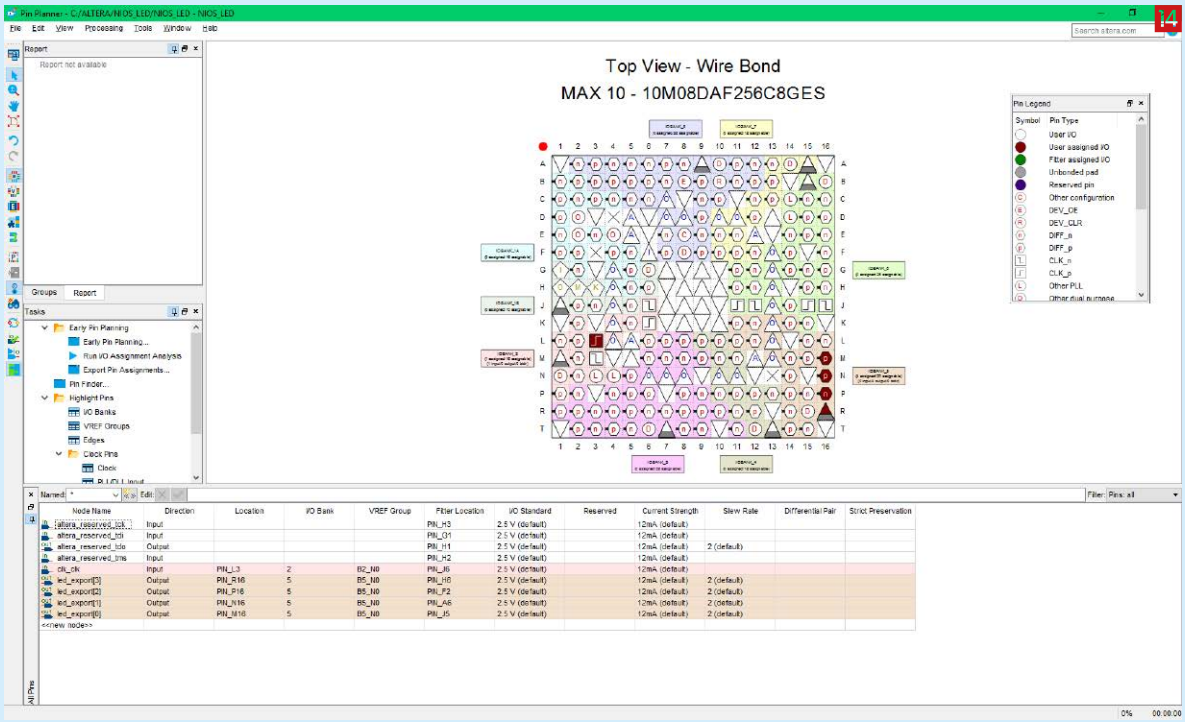
Przygotowany w ten sposób mikroprocesor – jak każdy inny – wymaga napisania programu. Pakiet Quartus Prime wyposażono w komplet narzędzi do realizacji projektów sprzętowo-programowych. Żeby zacząć tworzyć program dla procesora z menu pakietu wybieamy *Tools* → *Nios II Software Build Tools for Eclipse*. W ten sposób uruchomi się IDE Eclipse, które jest domyślnym środowiskiem dla programistów oiszących aplikacje dla NIOS-a.


12

Z menu Eclipse wybieramy *File* → *New* → *Nios II Application and BSP from Template*. W polu *SOPC Information File name* wybieramy plik *NIOS_LED.sopcinfo*, po „przemieleniu" tego pliku, w *Project name* wpisujemy *NIOS_LED* i wybieramy w *Templates „Hello World Small"*, następnie klikając *Finish*. 15


13

W wyniku tych zabiegów utworzą się dwa projekty:



NIOS_LED jest właściwym projektem, w którym piszemy kod, a NIOS_LED_bsp jest projektem powiązanym z NIOS_LED i służy do generowania bibliotek i makr dla danej konfiguracji „wyklikanej” wcześniej w Qsys-ie.

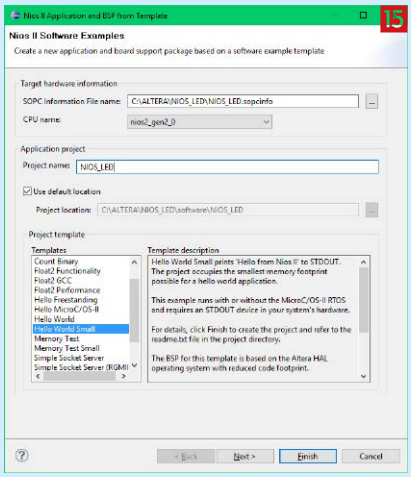
Edytujemy plik *hello_world_small.c* i wpisujemy następujący kod:

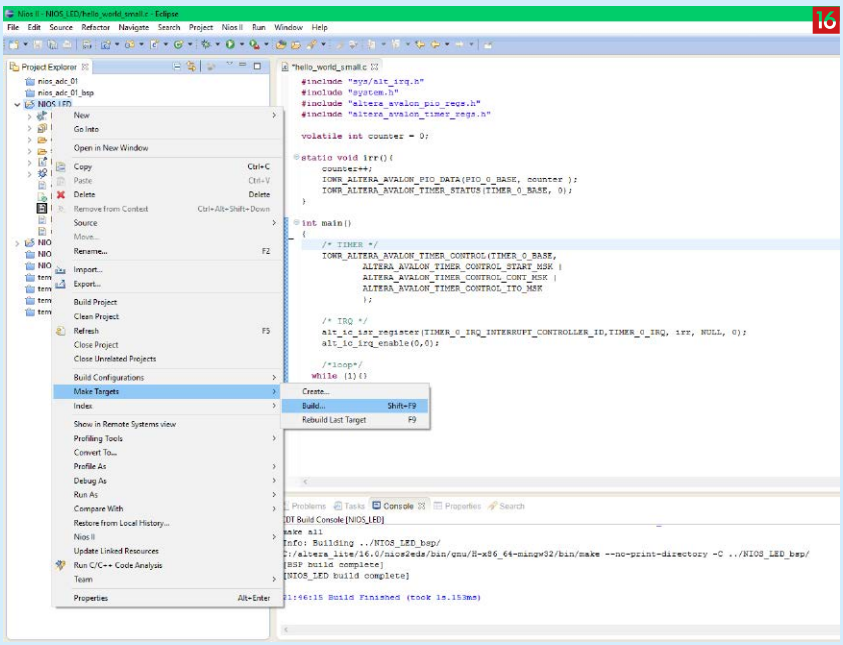
```
#include „sys/alt_irq.h”
#include „system.h”
#include „altera_avalon_pio_regs.h”
#include „altera_avalon_timer_regs.h”

volatile int counter = 0;

static void irr(){
    counter++;
    IOWR_ALTERA_AVALON_PIO_DATA(PIO_0_BASE, counter );
    IOWR_ALTERA_AVALON_TIMER_STATUS(TIMER_0_BASE, 0);
}

int main()
{
    /* TIMER */
    IOWR_ALTERA_AVALON_TIMER_CONTROL(TIMER_0_BASE,
        ALTERA_AVALON_TIMER_CONTROL_START_MSK |
        ALTERA_AVALON_TIMER_CONTROL_CONT_MSK |
```

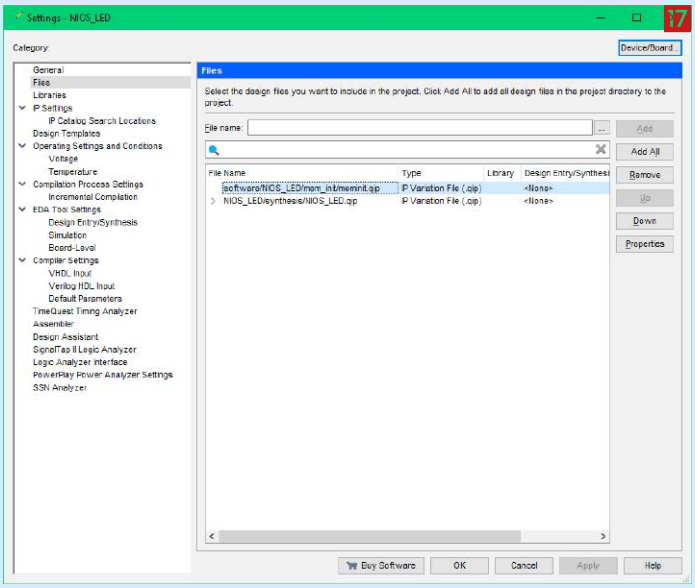




```
        ALTERA_AVALON_TIMER_CONTROL_ITO_
MSK
    );
    /* IRQ */
    alt_ic_isr_register(TIMER_0_IRQ_
INTERRUPT_CONTROLLER_ID,TIMER_0_IRQ, irr,
NULL, 0);
    alt_ic_irq_enable(0,0);
    /*loop*/
    while (1){}
    return 0;
}
```

Po wpisaniu programu kompilujemy go, wybierając z menu *Project* → *Build All*. Klikamy prawym myszy na projekcie *NIOS_LED* i wybieramy *Make Targets* → *Build...* 16

Teraz wybieramy *mem_init_generate* i klikamy *Build*. Po tej operacji wracamy do programu Quartus Prime.



Generowanie wsadu dla procesora

W Quartusie Prime trzeba dodać projekt wygenerowany w Eclipse, w tym celu z menu wybieramy *Project* → *Add/ Remove Files in Project...* Przy *File name* klikamy *[...]* i wybieramy plik: *software/NIOS_LED/mem_init/meminit.qip*, klikając następnie kolejno *Add*, *Apply* i *OK*. 17

Trzeba jeszcze zmienić domyślną konfigurację pamięci Flash układu FPGA. W tym celu w Quartusie wybieramy opcję *Assigments* → *Device...* W wyświetlonym oknie klikamy przycisk *Device and Pin Options...*, później w menu *Category* wybieramy *Configuration* i zmieniamy *Configuration Mode* na *Single Uncompressed Image with Memory Initialization (256Kbits UFM)*. Po tych zmianach klikamy OK, co umożliwia skompilowanie całego projektu wraz z programem dla NIOS. 18

Programowanie FPGA

Po opisanych dotychczas zbiegach zostało nam zaprogramować lub skonfigurować FPGA w zestawie MAXimator do wykonania zadania, co wymaga wykonania następujących, dość oczywistych kroków:

1. Dołączamy płytkę zestawu MAXimator do zasilania.
2. Dołączamy do złącza JTAG programator wchodzący w skład zestawu.
3. Z menu pakietu Quartus wybieramy *Tools* → *Programmer*, następnie w okienku *Programmer* naciskamy *Add File...* i wchodzimy do katalogu *output_files*, w którym wybieramy plik *NIOS_LED.sof* (konfigurujący FPGA poprzez pamięć RAM, zapis konfiguracji nie jest trwały). Alternatywnie, jeżeli chcemy trwale zapisać konfigurację FPGA, należy wybrać plik *NIOS_LED.sof*, który zostanie zapisany w pamięci konfiguracji Flash układu FPGA.
4. Po naciśnięciu przycisku *Start* układ FPGA MAX10 zostanie skonfigurowany, czego efekt będzie widać na pokładowych LED zestawu MAXimator.

Co właściwie zrobiliśmy?

Zrobiliśmy układ mikroprocesorowy z pamięcią RAM, ze sprzętowym timerem, podłączonym do 4 bitowego wyjścia. Timer wywołuje przerwanie co pół sekundy i jest ono obsługiwane przez funkcję *irr()*. Jeśli chodzi o samo programowanie, należy zwrócić uwagę na to, że nie operujemy bezpośrednio na adresach urządzeń a na zdefiniowanych nazwach, jest to bardzo ważne ponieważ przy modyfikacji projektu w Qsys-ie adresy bazowe dla poszczególnych komponentów mogą się zmienić. Wszystkie najważniejsze definicje wygenerowane automatycznie przez Qsys na podstawie zadanych w nim wartości parametrów, znajdują się w pliku *system.h* w projekcie *NIOS_LED.bsp*, warto również zapoznać się z plikami znajdującymi się w katalogu *drivers/inc* w projekcie *NIOS_LED.bsp*. W tych plikach znajdują się makra ułatwiające pracę z komponentami wygenerowanymi w Qsys-ie, przy okazji widać jakie są możliwości sterowania komponentami.

Mam nadzieję, że zachęciłem programistów do sięgnięcia po układy FPGA, w razie problemów, pytań lub propozycji tematów związanych z w/w układami zapraszam na forum ***http://microgeek.eu***.

MARIUSZ KSIĘŻAK
WWW.MICROGEEK.EU