Laboratorium z podstaw techniki cyfrowej
Studia inżynierskie stacjonarne, II rok III semestr, 2017/2018

Cz.1.

Przygotowanie – zapoznać się z działaniem podstawowych elementów logicznych (bramek, przerzutników oraz liczników).
Korzystając z zestawów do ćwiczeń z PTC wykonać:

1. zademonstrować działanie dostępnych elementów w zestawie,
2. zaprojektować:
   A. (tablica charakterystyczna, tablica stanów, tablica Karnaugha, tablica wzbudzeń, równanie logiczne przerzutnika, schemat blokowy) i połączyć przerzutnik asynchroniczny RS w oparciu o bramki NAND,
   B. zaprojektować (jw.) i połączyć przy pomocy dostępnych elementów przerzutnik synchroniczny typu T.
   C. (jw.) i połączyć przy pomocy bramek NAND przerzutnik JK.
   D. Zbudować na przerzutnikach D uniwersalny rejestr.

Cz.2.
Korzystając z zestawów płytek stykowych uniwersalnych zaprojektować i połączyć układ pełniący funkcję rejestru równoległego 4 bitowego z wpisem równoległym. Zastosować elementy 7400.

Cz.3. (oprac. dr Rafał Walkowiak)

W ramach laboratorium używamy:
- oprogramowanie: QUARTUS 13.0 sp1 firmy Altera
- sprzęt - karty DE2 firmy Terasic;
- część ćwiczeń
Przygotowanie do zajęć
W ramach przygotowania do zajęć proszę pobrać (dostępny darmowo na cele edukacyjne) i zainstalować
na własnych komputerach w celu przygotowywania projektów laboratoryjnych ze strony:
http://dl.altera.com/13.0sp1/?edition=web pakiety:
- Quartus II Software (includes Nios II EDS)
- Cyclone II, Cyclone III, Cyclone IV device support

Regulamin przedmiotu i laboratorium

1. Wątpliwości dotyczące zawartości regulaminu laboratorium należy zgłaszać prowadzącemu zajęcia – instruktorowi (e-mail lub podczas zajęć).
2. Każdy student otrzymuje identyfikator określający parametry zadania do przygotowania.
3. Instruktor przydziela komputer i płytę DE2 do wykorzystania podczas zajęć, po zakończeniu zajęć proszę oddać płytę DE2 instruktorowi.
4. Laboratorium polega na zaliczeniu przewidzianych regulaminem ćwiczeń (wymienionych poniżej).

Zaliczenie odbywa się na podstawie następujących elementów:
- zaliczeniu testu wejściowego jeśli jest (obowiązkowo),
- zaliczeniu ustnym (prezentacja układu, odpowiedzi na pytania) i uruchomieniu w symulatorze lub/i na płycie FPGA (wg zalecenia) przygotowanego projektu układu,
- sprawozdania z projektami układów – sprawozdanie niezbędne do zaliczenia układu zawiera temat, projekt, spis wykorzystanych elementów i krótki opis działania układu.

5. Układy cyfrowe można wprowadzać do sytemu Quartus poza zajęciami, lecz zaliczenia odbywają się wyłącznie na zajęciach.
6. Zaliczanie układu jest możliwe pod warunkiem zgłoszenia gotowości (poprawnie działający układ) przed upływem czasu zajęć, koniec zajęć powoduje przesunięcie zaliczania układu na początek kolejnych zajęć. Kryterium poprawności działania układu określa opis ćwiczenia.
7. Ocena ostateczna jest zależna od liczby wykonanych ćwiczeń, sposobu i terminu zaliczenia ćwiczeń, odpowiedzi ustnej przy zaliczaniu działających układów, stopnia realizacji projektów oraz złożoności prezentowanych układów.
8. Termin prezentacji wykonanych ćwiczeń mija na ostatnich zajęciach semestru.

Lista ćwiczeń laboratoryjnych
Ćwiczenia laboratoryjne podzielone są na cztery grupy (GR1-GR4):

1. projektowanie układów cyfrowych metoda strukturalną w oparciu o standardowe elementy składowe: bramki, przerzutniki, liczniki; przygotowanie układu polega na **narysowaniu schematu** w oparciu o elementy biblioteczne dostępne w pakiecie Quartus
2. projektowanie układów cyfrowych metoda strukturalną w oparciu o równania logiczne; przygotowanie układu polega na przygotowaniu **kodu w języki VHDL** zawierającego jednostki struktury układu cyfrowego zdefiniowane za pomocą **równań Boolowskich** np. multipleksery, kodery, sumatory; jednostki strukturalne zgodnie z zasadami konkretyzacji są wykorzystywane do zdefiniowania docelowych układów wyższego poziomu.
3. projektowanie układów cyfrowych przy wykorzystaniu dowolnych **konstrukcji języka VHDL** dla specyfikacji układów cyfrowych metodą strukturalną i przy użyciu opisu zachowania układów (**opis behawioralny**).
4. projektowanie układów cyfrowych przy wykorzystaniu kreatora automatów skończonych.

Oznaczanie sygnałów zewnętrznych projektowanych układów

Projektowane układy mają być realizowane w ramach płyt edykacyjnych DE2 i DE2-70. Na płytach znajdują się układy reprogramowalne Cyclone odpowiednio w DE2 - EP2C35F672C6 , w DE2-70 EP2C70F896C6. Zarówno uruchamianie zaprogramowanego w FPGA układu, jak i testowanie pracy układu w symulatorze wymaga określenia w projekcie typu użytego układu scalonego oraz wskazanie wykorzystywanych wyprowadzeń układu, do których będą podłączone sygnały wejściowe i wyjściowe urządzenia. Przyporządkowanie informacji o wyprowadzeniach jest możliwe po podaniu modelu użytego układu programowalnego opcja Assignments/Device. Wyprowadzenia układów FPGA mają złożone oznaczenia i mogą być trudne do zapamiętania. Poszczególne wyprowadzenia są na stałe podłączone do wielu różnorodnych elementów cyfrowych, których wykorzystanie umożliwia testowanie zaprojektowanego układu. W celu uniknięcia potrzeby wpisywania, w każdym projekcie, złożonych oznaczeń wykorzystywanych wyprowadzeń, możliwe jest zastosowanie prostszego rozwiązania polegającego na **oznaczaniu sygnałów zewnętrznych** projektowanych układów typowymi, łatwymi do zapamiętania **symbolami określającymi elementy podłączone do wykorzystywanych wyprowadzeń układu FPGA**. Ostatecznie powiązanie wprowadzonych przez użytkownika oprogramowania standardowych nazw elementów połączonych z wyprowadzeniami układu FPGA z fabrycznymi oznaczeniami wyprowadzeń (niezbędne do określenia miejsca doprowadzenia sygnału na potrzeby symulacji i programowania układu) **odbywa się automatycznie** za pomocą opcji oprogramowania Quartus - Assignments/Import Assignments i wskazania standardowego pliku (odpowiedniego do wykorzystywanej płyty – układu FPGA) zawierającego właściwe powiązania (nazwa przyłączonego elementu – fabryczne oznaczenie wyprowadzenia układu FPGA). Poleca się korzystać z następujących oznaczeń wyprowadzeń sygnałów projektu (aby automatycznie uzyskać powiązanie z symbolem używanego wyprowadzenia).

Wejścia:

- SW[0] do SW[17] przełączniki bistabilne używane do podawania wartości binarnych
- KEY[0] do KEY[3] przyciski monostabilne z filtracją drgań zestyków, używane do taktowania
- CLOCK_50 sygnał zegarowy o częstotliwości 50 MHz

Wyjścia :

- LEDR[0] do LEDR[17] diody elektroluminescencyjne świecące światłem czerwonym
- LEDG[0] do LEDG[8] diody elektroluminescencyjne świecące światłem zielonym
- HEX0[0] do HEX0[6] sygnały sterujące wyświetlaczem siedmiosegmentowym HEX0 (8 wyświetlaczy 7- mio segmentowych posiada płyta HEX0-HEX7).

Standardowe (używane przez producenta) nazwy wyprowadzeń (różne dla płyt DE2 i DE2-70) zawarte są w plikach DE2_pin_assignments.csv, DE2_70_pin_assignments.csv

W celu ujednolicenia nazw wyprowadzeń dla DE2 i DE2-70 przygotowano plik z wyprowadzeniami FPGA z płyty DE2-70 w którym najczęściej używane wyprowadzenia mają takie same nazwy jak używane w płycie DE2 DE2_70_pin_assignments_mod.csv

Ćwiczenie 1 – TRANSKODER (GR1)

Wykonanie transkodera jako układu kombinacyjnego na bramkach NAND. Zastosować dowolną metodę minimalizacji wyrażeń boolowskich do wyznaczenia postaci funkcji wyjściowych transkodera.

Przygotowanie do ćwiczeń polega na:

- zapoznaniu się z metodami minimalizacji wyrażeń boolowskich,
- przypomnieniem praw logiki niezbędnych do realizacji funkcji wyłącznie za pomocą bramek NAND,

- przygotowania sprawozdania zawierającego syntezę funkcji i schemat logiczny układu.

Realizacja ćwiczenia polega na:

- narysowaniu schematu w programie Quartus,
- sprawdzeniu poprawności działania za pomocą symulacji w Quartus i realizacji w układzie DE2 lub DE2-70.
- wyznaczeniu czasów propagacji sygnałów przez układ na skutek podawania na wejścia układu kolejnych wartości w NKB
- prezentacja działającego układu prowadzącemu zajęcia.

Zadania dla poszczególnych studentów:

Opis zadania składa się z dwóch części: specyfikacja kodu wejściowego (IN) i specyfikacja kodu wyjściowego (OUT). Poniżej obok identyfikatora studenta podano kod wejściowy i wyjściowy transkodera:

1. IN k.binarny OUT 0,7,5,4,3,4,1
2. IN k.binarny OUT 0,2,5,2,4,6,1
3. IN k.binarny OUT 0,3,7,6,5,2,1
4. IN k.binarny OUT 0,5,6,2,7,1,3
5. IN k.binarny OUT 0,4,3,2,5,7,6
6. IN k.binarny OUT 0,2,4,5,1,3,7
7. IN k.binarny OUT 0,5,1,4,2,3,6
8. IN k.binarny OUT 0,5,6,7,3,4,1
9. IN k.binarny OUT 0,4,5,3,6,3,7
10. IN k.binarny OUT 0,2,6,2,4,7,3
11. IN k.binarny OUT 0,5,4,2,1,6,7
12. IN k.binarny OUT 0,5,1,2,4,7,1
13. IN k.binarny OUT 0,4,5,2,1,3,7
14. IN k.binarny OUT 0,7,1,2,3,6,4
15. IN k.binarny OUT 0,3,1,2,7,6,4
16. IN k.binarny OUT 0,3,4,7,1,5,6
17. IN k.binarny OUT 0,4,6,2,7,1,5
18. IN k.binarny OUT 0,2,3,5,7,4,6
19. IN k.binarny OUT 0,6,2,7,4,5,2
20. IN k.binarny OUT 0,3,6,2,4,7,2
21. IN k.binarny OUT 0,5,7,2,6,2,3
22. IN k.binarny OUT 0,6,1,3,2,3,4
23. IN k.binarny OUT 0,3,2,2,6,1,4
24. IN k.binarny OUT 0,2,1,5,5,7,4
25. IN k.binarny OUT 0,3,1,2,7,2,5
26. IN k.binarny OUT 0,5,1,2,2,7,3
27. IN k.binarny OUT 0,2,7,2,6,1,3
28. IN k.binarny OUT 0,4,3,2,1,5,2
29. IN k.binarny OUT 0,2,6,5,3,5,7
30. IN k.binarny OUT 0,4,2,2,3,7,1

31 - 60. kod wejściowy jest kodem wyjściowym ćwiczenia o numerze o 30 mniejszym (powtarzające się w kodzie cyfry należy zamienić na inne w kodzie nie występujące), a kod wyjściowy jest kodem 1,2,4,4,5,6,3.

Jeżeli wynik minimalizacji doprowadzi do układu zawierającego mniej niż 4 bramki, to należy zrealizować kolejne zadanie z listy.

Ćwiczenie 2 - Synteza liczników synchronicznych i asynchronicznych (GR1)

Zakres ćwiczenia: Zapoznanie się z metodą syntezy liczników przy wykorzystaniu przerzutników i podstawowych układów bramek. Zaprojektowanie wybranych układów liczników i wprowadzenie do edytora schematów układów. Uruchomienie i analiza w symulatorze częstotliwości granicznej pracy układów.

Przygotowanie do ćwiczeń polega:

1. zapoznanie się z zasadami działania i budowy liczników synchronicznych i asynchronicznych,
2. zaprojektowanie liczników i wprowadzenie do edytora schematów zaprojektowanych układów,
3. ocena przy użyciu symulatora (w trybie rzeczywistym- czasowym) poprawności pracy przygotowanych liczników,

4. w sprawozdaniu, wykonanym przed przystąpieniem do zajęć w pracowni zamieścić projekty liczników (tablice minimalizacji, uzyskane funkcje, schemat).

Praca na zajęciach polega na rozwiązaniu trudności w uruchomieniu projektów i zapewnieniu poprawnej ich pracy; wyznaczenie częstotliwości granicznej pracy układów na drodze symulacji; zaprezentowaniu prowadzącemu zajęcia projektów (sprawozdanie) oraz działających układów (symulacja i rzeczywista w FPGA). Kryterium poprawności pracy licznika jest jego praca z wymaganą kolejnością stanów przy wolnym taktowaniu i odpowiedni podział częstotliwości wejściowej przy szybkim taktowaniu.

Wersje układów do przygotowania:

- licznik synchroniczny liczący w kodzie wykorzystywanym w Ćw.1 (kod niebinarny, uniklane wartości, pozostałe pominąć), przy użyciu przerzutników D,
- licznik synchroniczny liczący w kodzie wykorzystywanym w Ćw.1 (kod niebinarny, uniklane wartości, pozostałe pominąć), przy użyciu przerzutników JK,
- licznik asynchroniczny liczący (przerzutniki D) w kodzie modulo (8+ RESZTA(Identyfikator/8))

Wartościowym uzupełnieniem wykonanego ćwiczenia jest umiejętność prezentacji teoretycznej analiza częstotliwości granicznej pracy liczników; jako częstotliwość graniczną rozumiemy częstotliwość pracy zapewniającą właściwy kod zliczania (dla liczników synchronicznych) oraz poprawne dzielenie częstotliwości wejściowej dla liczników asynchronicznych.

Ćwiczenie 3 - Skracanie liczników (GR1)

Zakres ćwiczenia: Zapoznanie się z działaniem liczników scalonych (układy wg technologi TTL): 7493, 74161(asynchroniczny reset), 74163 (synchroniczny reset), 74193. Zastosowanie liczników scalonych do budowy liczników w kodzie binarnym metodą skracania.

Przygotowanie do ćwiczeń:

1. zapoznanie się z zasadami działania układów - Opis metody skracania liczników
2. zaprojektowanie liczników modulo N (N liczba określona jako parametr inny dla każdego studenta):
   - układ liczący w górę – użycie 7493,
   - układ liczący w górę - użycie 74161,
   - układ liczący w górę - użycie 74163,
   - jeden układ liczący w górę i w dół w zależności od wartości na wejściu wyboru kierunku – użycie 74193;

1. (opcjonalnie) teoretyczna analiza częstotliwości granicznej pracy liczników. Częstotliwość graniczna w przypadku tych liczników jest to maksymalna częstotliwość, przy której licznik dzieli częstotliwość wejściową przez N;
2. sprawozdaniu, wykonanym przed przystąpieniem do realizacji ćwiczenia należy zamieścić opis działania użytych układów rejestrów scalonych, projekty liczników i teoretyczną analizę częstotliwości granicznej pracy liczników;
3. ocena przy użyciu symulatora (w trybie rzeczywistym- czasowym) poprawności pracy przygotowanych liczników.

Praca na zajęciach polega na rozwiązaniu trudności w uruchomieniu projektów i zapewnieniu poprawnej ich pracy; wyznaczenie częstotliwości granicznej pracy układów na drodze symulacji; zaprezentowaniu prowadzącemu zajęcia projektów (sprawozdanie) oraz działających układów (symulacja i rzeczywista w FPGA).

Kryterium poprawności pracy licznika jest podział częstotliwości wejściowej przy szybkim taktowaniu. Wyznaczenie N dla poszczególnych studentów posiadających identyfikator z ćwiczenia 1: N = 15+ Identyfikator*2.

Ćwiczenie 4 – REKLAMA TEKSTOWA (GR2)

Celem ćwiczenia jest wykorzystanie języka VHDL do opisu sposobu wykorzystania prostych elementów wejścia-wyjścia płyty DE2 i podłączenie ich do układu FPGA. Wejściem układu będą przełączniki SW0-SW17 a wyjściem wyświetlacze siedmiosegmentowe HEX0-HEX7. Zastosowanie strukturalnej metody projektowania i **równań Boolowskich** określających sposób przetwarzania sygnałów w jednostkach układu.

Zaprojektować układ pozwalający na wyświetlanie na 7 wyświetlaczach siedmiosegmentowych HEX0 do HEX6 dowolnego słowa ZBUDOWANEGO z podzbioru następujących znakow: A,B,C,E,F,G, H,I,J,L,O,P,S.

Proszę znaki kodować na 3 bitach. Należy wyświetlać litery z powyższej listy, student umożliwia wykorzystanie w tekście spacji (kod 0) i **5 różnych liter** pobieranych kolejno z powyższej listy cyklicznie począwszy od litery numer= 1+ RESZTA(Identyfikator/13). Napis składający się z wybranych liter zakodowanych na wejściach SW0- SW14 powinien przesuwać się cyklicznie między wyświetlaczami siedmiosegmentowymi pod wpływem zmian na wejściach SW15-SW17 określających miejsce wyświetlania informacji. Do wykonania układu należy przygotować następujące elementy składowe (komponenty układu):

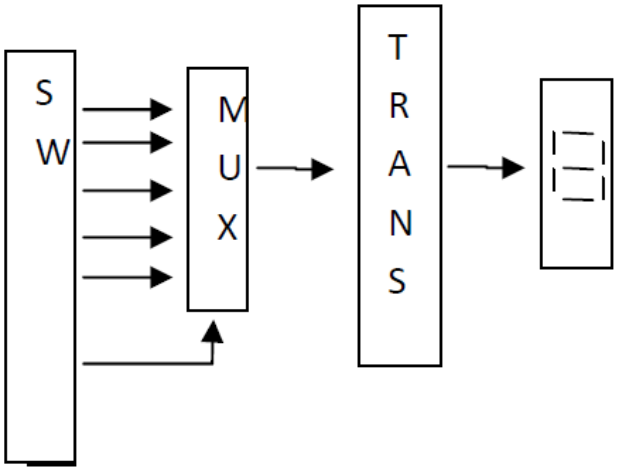
- multiplekser 1 z 8 – wybiera jedno z 8 wejść informacyjnych dostarczające kody znaków (lub kod spacji) do transkodera (bin-7seg)
- transkoder - kod liter na kod wyświetlacza 7 segmentowego

Korzystając z procedury konkretyzacji należy użyć zaprojektowane komponenty wielokrotnie do zbudowania układu pozwalającego na wyświetlenie tekstu 8 znakowego składającego się z (podanych za pomocą kodów na wejściach SW0-SW14) 5 liter uzupełnionych spacjami. Strukturę kodu podano poniżej z komentarzami.

W ramach wykładu (i w wielu innych miejscach) dostępny jest kod wyświetlacza siedmiosegmentowego – proszę zwrócić uwagę na konieczną polaryzację sygnałów sterujących diodami – do zapalenia segmentu niezbędny jest poziom niski sygnału (0 logiczne)

Zaliczenie projektu wymaga prezentacji poprawnie działającego układu w DE2 wyświetlającego zadane litery w zadanej kolejności. Schemat blokowy modułu związanego z jednym wyświetlaczem 7-mio segmentowym zawiera poniższy rysunek, przy deklaracji interfejsu układu proszę zwrócić uwagę na właściwe dla stosowanej karty nazwy wyprowadzeń w plikach przyporządkowań (Assignments)

```
LIBRARY ieee;
USE ieee.std logic 1164.all;
ENTITY CW4 IS
PORT ( SW : IN STD LOGIC VECTOR(17 DOWNTO 0);
HEX0 : OUT STD LOGIC VECTOR(0 TO 6));
END CW4;
ARCHITECTURE strukturalna OF CW4 IS
CONSTANT SPACJA: STD_LOGIC_VECTOR(2 DOWNTO 0):=”000”; -- KOD SPACJI
--DEKLARACJA KOMPONENTÓW
COMPONENT mux3bit 8to1 -- muliptekser
PORT ( S, U0, U1, U2, U3, U4, U5,U6,U7: IN STD LOGIC VECTOR(2 DOWNTO 0);
//WEKTOR STERUJĄCY I 8 wektorów INFORMACYJNYCH
M0 : OUT STD LOGIC VECTOR(2 DOWNTO 0));
END COMPONENT;
COMPONENT char7seg -- transkoder
PORT ( C : IN STD LOGIC VECTOR(2 DOWNTO 0);
Display : OUT STD LOGIC VECTOR(0 TO 6));
END COMPONENT;
```




```
SIGNAL M : STD LOGIC VECTOR(2 DOWNTO 0);
BEGIN
-- KONKRETYZACJA UŻYCIA KOMPONENTÓW
MUX0: mux 3bit 8to1 PORT MAP (SW(17 DOWNTO 15), SW(14 DOWNTO 12), SW(11 DOWNTO 9),
SW(8 DOWNTO 6), SW(5 DOWNTO 3), SW(2 DOWNTO 0),SPACJA,SPACJA,SPACJA, M0);
-- KONKRETYZACJE KOLEJNYCH MULTIPLEKSERÓW UKŁADU
H0: char7seg PORT MAP (M0, HEX0);
-- KONKRETYZACJE KOLEJNYCH TRANSKODERÓW
END strukturalna;
- - implementacja multipleksera 8 do 1 (wektor 3 bitowy)
LIBRARY ieee;
USE ieee.std logic 1164.all;
ENTITY mux3bit_8to1 IS
PORT ( S, U0, U1, U2, U3, U4, U5,U6,U7: IN STD LOGIC VECTOR(2 DOWNTO 0);
M : OUT STD LOGIC VECTOR(2 DOWNTO 0));
END mux 3bit8to1;
ARCHITECTURE strukturalna OF mux3bit_8to1 IS
. . . do uzupełnienia
END strukturalna;
-- IMPLEMENTACJA TRANSKODERA
LIBRARY ieee;
USE ieee.std logic 1164.all;
ENTITY char7seg IS
PORT ( C : IN STD LOGIC VECTOR(2 DOWNTO 0);
Display : OUT STD LOGIC VECTOR(0 TO 6));
END char 7seg;
ARCHITECTURE strukturalna OF char 7seg IS
. . . douzupełnienia
END strukturalna;
```

Ćwiczenie 5 Sumatory i mnożarki (GR2)

Kolejne kroki ćwiczenia - jednostki projektowe kolejnych poziomów (wykonane za pomocą syntezy strukturalnej):

1. **Sumator jednobitowy pełny**, przerzutnik D: test w symulatorze;

2. **Sumator 8 bitowy** (koncepcja struktury wg Rys. 5.1) i rejestr 8 bitowe w oparciu o elementy 1 bitowe: test w symulatorze;
3. **Sumator sekwencyjny**, struktura układu wg Rys. 5.2: test w symulatorze, analiza prędkości działania za pomocą TimeQuest Analyser.
4. **Sumator/Subtrakor sekwencyjny**, struktura układu wg Rys. 5.3: test na płycie DE2, dopasowanie wyprowadzeń dla potrzeb podawania danych wejściowych SW, KEY i prezentacji wartości HEX (dane wejściowe i wynik wyświetlane szesnastkowo), analiza prędkości działania za pomocą TimeQuest Analyser.
5. Implementacja mnożenia liczb binarnych 4 bitowych:

```
          1001
X         1010 9*10=90
-------------------------------------------------
          0000 mnożna X b0 mnożnej
         1001  przesunięta o 1 pozycję mnożna X b1 mnożnej
        0000   przesunięta o 1 pozycję mnożna X b2 mnożnej
       1001    przesunięta o 1 pozycję mnożna X b3 mnożnej
-------------------------------------------------
       1011010 wynik =90 – suma sum częściowych
```

Implementacja dodawania 4 liczb wymaga 3 sumatorów korzystających ze swoich wyników.

Sekwencyjny układ mnożący liczby 4 bitowe o strukturze z Rysunku 5.4 poszerzonej o rejestry argumentów i wyjściowe analogicznie do sumatora z Rysunku 5.2, dopasowanie wyprowadzeń dla potrzeb podawania danych wejściowych SW, KEY i prezentacji wyników LEDG, HEX (dane wejściowe i wynik wyświetlane szesnastkowo), test na płycie DE2, analiza prędkości działania za pomocą TimeQuest Analyser.

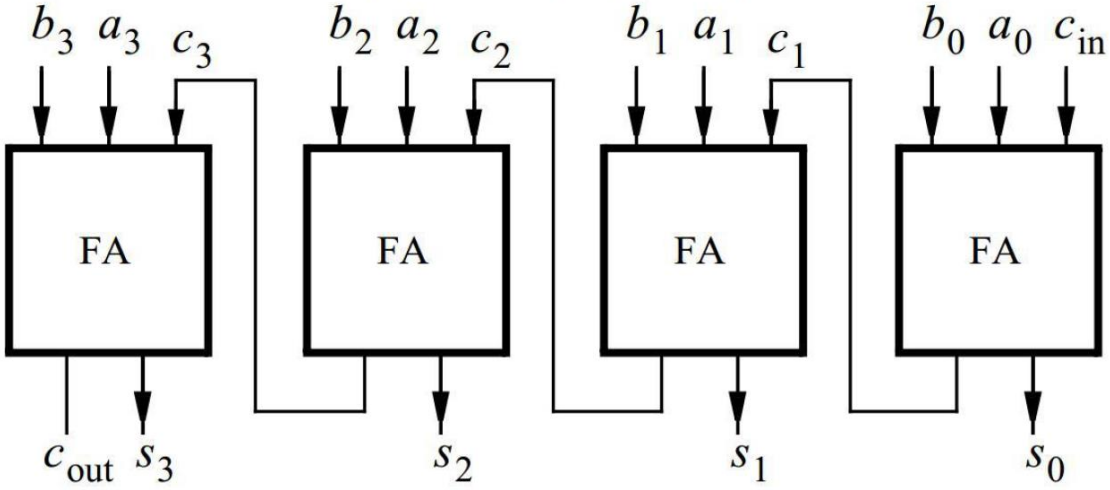
Rysunek 5.1 Sumator 4 bitowy (FA sumator pełny 1 bitowy)

Rysunek 5.2 Sumator sekwencyjny 8 bitowy w oparciu o rejestry 8 bitowe oraz sumator 8 bitowy.

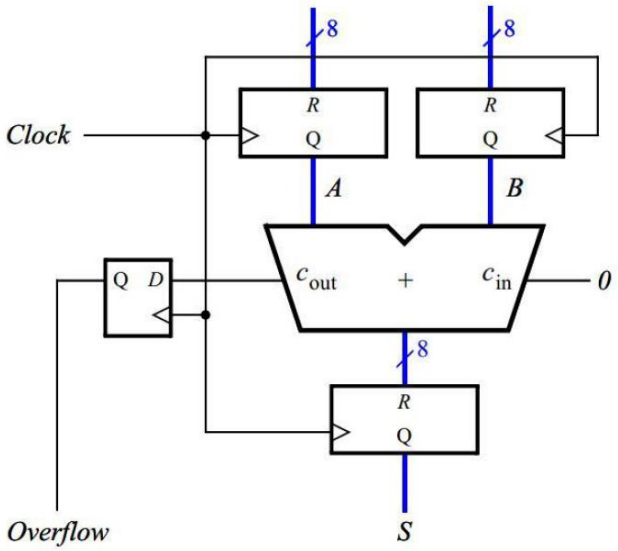
Rysunek 5.3 Sekwencyjny układ dodający lub odejmujący (wielokrotnie uwzględnienie składnika lub odjemnej)

Rysunek 5.4 Struktura 4 bitowego układu mnożącego.

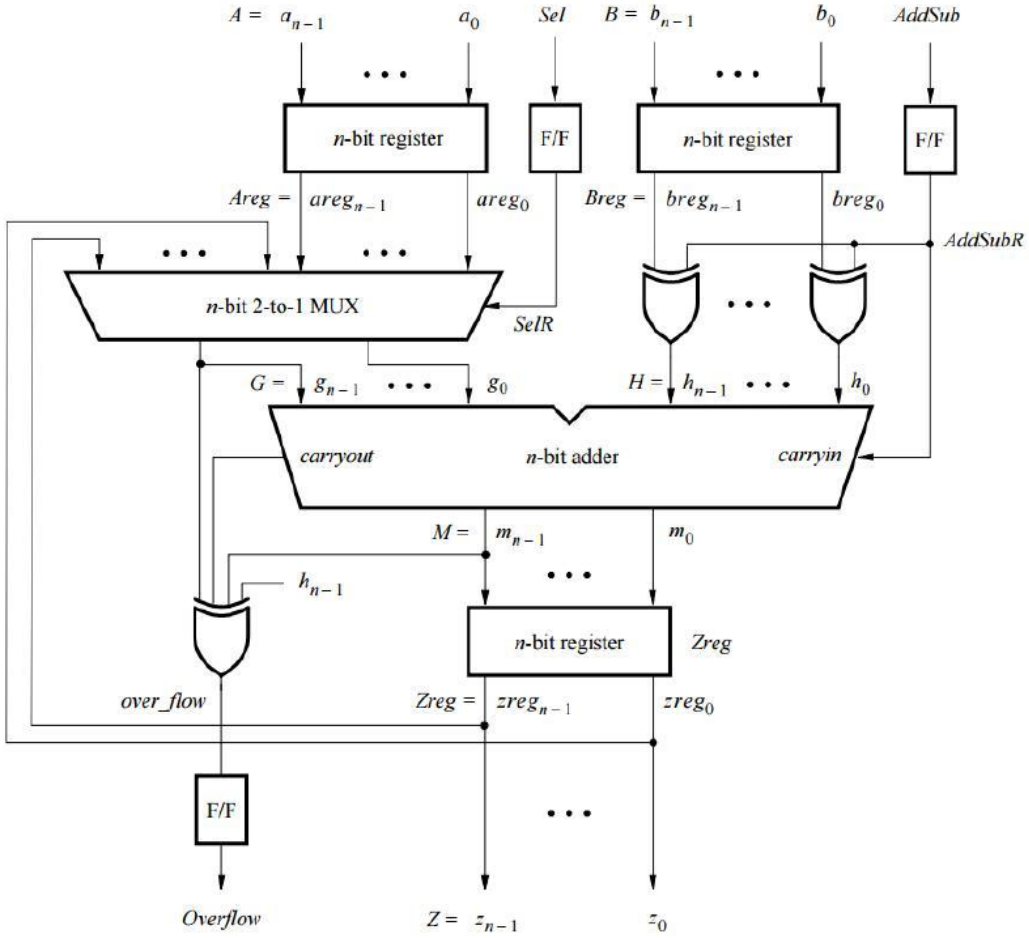
Rysunek 5.1 Sumator 4 bitowy (FA sumator pełny 1 bitowy)



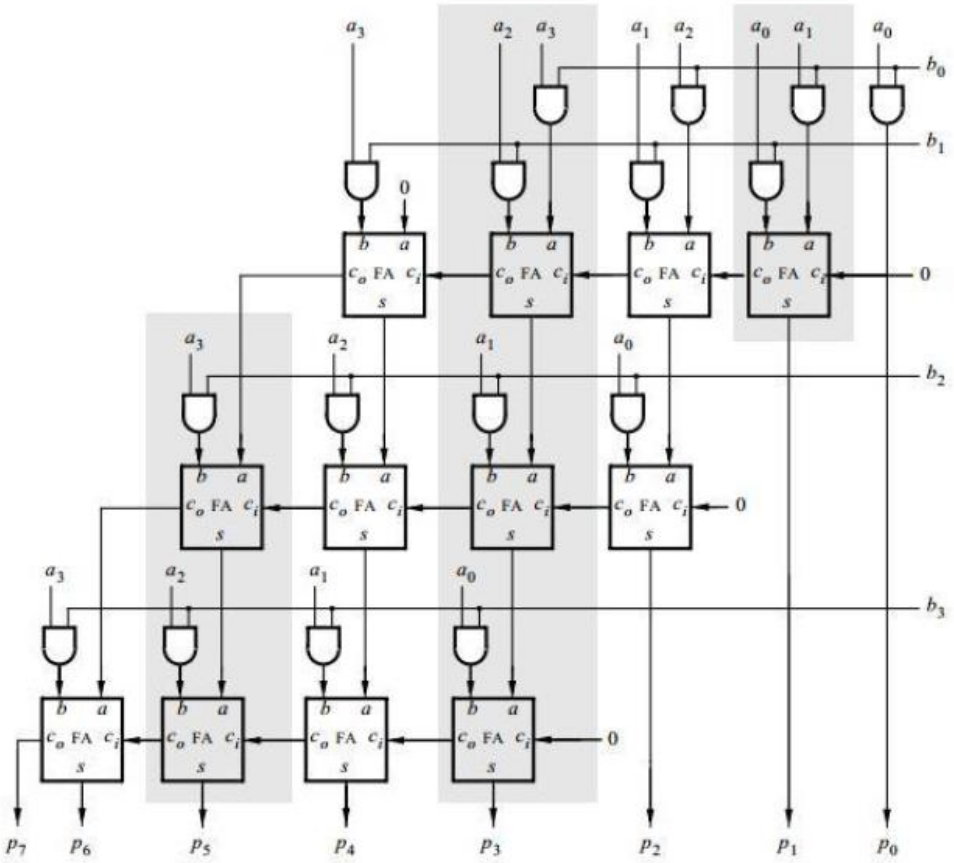
Rysunek 5.2 Sumator sekwencyjny 8 bitowy w oparciu o rejestry 8 bitowe oraz sumator 8 bitowy.



Rysunek 5.3 Sekwencyjny układ dodający lub odejmujący (wielokrotnie uwzględnienie składnika lub odjemnej)



Rysunek 5.4 Struktura 4 bitowego układu mnożącego.



Ćwiczenie 6 Automat (GR4)

Zakres ćwiczenia:

- sposób wykorzystania edytora automatu i kreatora kodu układu sterującego zostanie zaprezentowany na zajęciach poprzedzających wykonanie ćwiczenia;
- projektowanie automatu zgodnie ze specyfikacją i własnymi uszczegółowieniami sposobu działania,
- wprowadzanie do systemu Quartus projektu za pomocą edytora stanów,
- kompilacja i symulacja czasowa układu sterowania,
- uruchomienie automatu za pomocą płyty DE2, wykorzystanie przełączników lub przycisków jako źródeł informacji z czujników i resetu, wyświetlanie za pomocą diod lub wyświetlacza 7segmentowego informacji o stanie automatu oraz wyjściowych sygnałach sterujących,
- proponuje się sterowanie upływem czasu w układzie **w proporcji** do rzeczywistego czasu sterowania np. 1 min czasu rzeczywistego = 10 sek czasu testowanego układu,
- oprócz wejść sygnałów „z czujników" niezbędne będą informacje o upływie czasu pochodzące z liczników sygnalizujących, umożliwiających uruchomienie zliczania, odpowiednie liczniki należy zaprojektować jako oddzielne jednostki projektowe i odpowiednio powiązać (reset, wartość licznika) z układem sterowania.

Każda osoba realizuje układ sterowania jednym urządzeniem. Numer układu cyfrowego zależny jest od identyfikatora: numer = (identyfikator mod 4) + 1.

Wersje projektowanego i uruchamianego układu:

1. **Sygnalizacja uliczna na skrzyżowaniu pełnym** 2 dróg **Opis:** 2 przejścia dla pieszych, przyciski żądania światła dla pieszych, stały cykl zmiany świateł, zapewniony minimalny czas trwania światła zielonego dla pieszych, zapewniony minimalny czas trwania światła zielonego dla samochodów, brak czujników pojazdów.
2. **Sygnalizacja uliczna na skrzyżowaniu pełnym** 2 dróg - ulica główna i boczna

Opis: 2 przejścia dla pieszych: przyciski żądania światła dla pieszych, zmienny cykl zmiany świateł pozwalający na wyjazd z ulicy bocznej na żądanie, zapewniony minimalny czas trwania światła

zielonego dla pieszych, zapewniony minimalny czas trwania światła zielonego dla samochodów, czujniki pojazdów na ulicy bocznej.

3. **Przejazd kolejowy dla linii jednotorowej. Opis:** dwie barierki wjazdowe i 2 barierki zjazdowe, 2 czujniki (masy) po każdej ze stron toru zbliżającego się pociągu (odległość między czujnikami większa od długości pociągu) i czujnik samochodu na przejeździe, sterowanie dla samochodów światłami i barierką, sterowanie światłami STOP dla pociągów, umożliwienie zjazdu samochodu, sygnalizacja robocza stanów: pociąg na przejeździe, samochód na przejeździe, pociągi mogą się zbliżać do przejazdu z 2 stron ale nie równocześnie, pojawienie się pociągów równocześnie z 2 stron powoduje sygnalizację awarii i sygnalizację STOP na sygnalizatorze świetlnym.

4. Pralka automatyczna

Opis:

- układ sterowania umożliwia realizację jednego programu prania,
- sterowanie zaworem pobierania wody, obrotami silnika pralki przy wirowaniu i praniu, grzałką, pompą spustu wody,
- uwzględnienie upływu czasu, czujnika poziomu wody i temperatury,
- proszę uwzględnić typowe etapy prania: pobieranie wody, podgrzewanie wody z praniem, wielokrotne płukanie z pobraniem i spustem wody i wirowanie.

Uwagi do Ćwiczenia 6: poza zajęciami przygotowując projekt warto skorzystać z:
QuartusHelp i informacji z prezentacji:
How to create a State Machine with the Quartus State Machine Wizard

Ćwiczenie 7 Dostęp do pamięci statycznej (do wykonania tylko na płycie laboratoryjnej typ DE2) GR3

Zakres ćwiczenia:

- Przypomnienie zasad działania pamięci statycznej – SRAM parametry czasowe pracy, łączenie układów pamięci.
- Przygotowanie w języku VHDL układu pozwalającego na zapis i odczyt danych do/z pamięci IS61LV25616ALdostępnej w układzie DE2.

Wymagana szczegółowe dla układu, należy:

- zapisać do pamięci 16 liczb (począwszy od adresu równego identyfikatorowi), liczby te mają być pobrane w kolejnych cyklach pracy licznika modulo N liczącego w NKB, gdzie N = Identyfikator_studenta modulo 10)+3
- w wyniku naciśnięciu przycisku KEY[0] następuje zapis danych do pamięci i sygnalizacja zakończenia zapisu zapaleniem diody LEDR[0]
- po naciśnięciu przycisku KEY[1] następuje odczyt z pamięci danych i wyświetlenie za pomocą wyświetlaczy 7-mio segmentowych wartości i adresu – jedna wartość co sekundę i sygnalizacja zakończenia pracy poprzez wygaszenie wyświetlacza i diody LEDR[0], system oczekuje na kolejne rozkazy – zapis lub odczyt.

Nazwy skojarzone z wyprowadzeniami FPGA podłączonymi do SRAM na płycie DE2 (wg pliku DE2_pin_assignments.csv): SRAM_ADDR[0] - SRAM_ADDR[17], SRAM_DQ[0] SRAM_DQ[15],SRAM_WE_N , SRAM_OE_N, SRAM_UB_N, SRAM_LB_N, SRAM_CE_N

Uruchomienie projektu w układzie DE2 powinno być poprzedzone prezentacją prowadzącemu zajęcia kodu układu odpowiadającego za sterowanie pamięcią – w kodzie czytelnie widoczny ma być sposób sterowania pamięcią. Należy zwrócić szczególną uwagę na wykluczający się dostęp do magistrali danych: pamięci i bufora wyjściowego danych. Magistrala danych i adresowa mogą być na stałe podłączone do wyświetlaczy siedmiosegmentowych prezentujących wartości w kodzie szesnastkowym.

Ćwiczenie 8 Prosty procesor DE2 lub DE2-70 GR3-4

Zakres ćwiczenia 7 jest opisany w pliku PTC_cw8.pdf. Pięć wersji procesora jest przydzielanych do studentów przy użyciu identyfikatora studenta i operacji modulo 5, np student o identyfikatorze 30 realizuje wersję piątą projektu, a student o id 1 projekt 1. Możliwe są również inne realizacje zaproponowane przez studentów, a przed przygotowaniem projektu zaakceptowane przez prowadzącego zajęcia. Do zaliczenia ćwiczenia 7 niezbędne jest sprawozdanie (w wersji drukowanej dostępne przy zaliczaniu na zajęciach) zawierające opis koncepcji, realizacji, sposobu uruchomienia i

sposobu działania. Pliki źródłowe projektu i elektroniczną wersję sprawozdania proszę przesłać w terminie do 2 dni po zaliczeniu na adres email prowadzącego zajęcia.

UWAGA : wszystkie projekty powinny być przygotowany samodzielnie (ewentualna pomoc jest dopuszczalna lecz, przed zaliczaniem ćwiczenia należy omówić zakres otrzymanej pomocy).

Zaliczenie projektu wymaga:

- przygotowania projektu,
- rozumienia struktury projektu i umiejętności prezentacji układu i opisu sposobu jego działania,
- umiejętności (zgodnie z instrukcją projektu) prezentacji poprawnie działającego układu na DE2/DE2-70 i/lub umiejętności realizacji symulacji projektu i odpowiedzi na pytania dotyczące koncepcji i zawartości projektu,
- analizy czasowej dla implementacji projektu w FPGA za pomocą TimeQuest Timing Analyser.