



Politechnika Wrocławska

Wydział Elektroniki, Fotoniki i Mikrosystemów

Weryfikacja Systemów Cyfrowych

Projekt indywidualny

Automat sprzedający napoje

Marcel Chołodecki, 275818

Grupa USOS nr 4 (Piątek TP 13:15)

3 lutego 2025

Spis treści

1	Wstęp	3
2	Działanie automatu	3
3	Weryfikacja działania automatu	3
3.1	Zapis i odczyt danych z rejestrów	4
3.2	Zamówienie kawy z domyślną mocą napoju	5
3.3	Zamówienie soku z mocą napoju większą od stanu zasobnika	6
3.4	Zamówienie herbaty przy wrzuceniu zbyt małej liczby monet	7
3.5	Reset rejestrów	8
4	Wnioski	9
	Spis rysunków	9

1. Wstęp

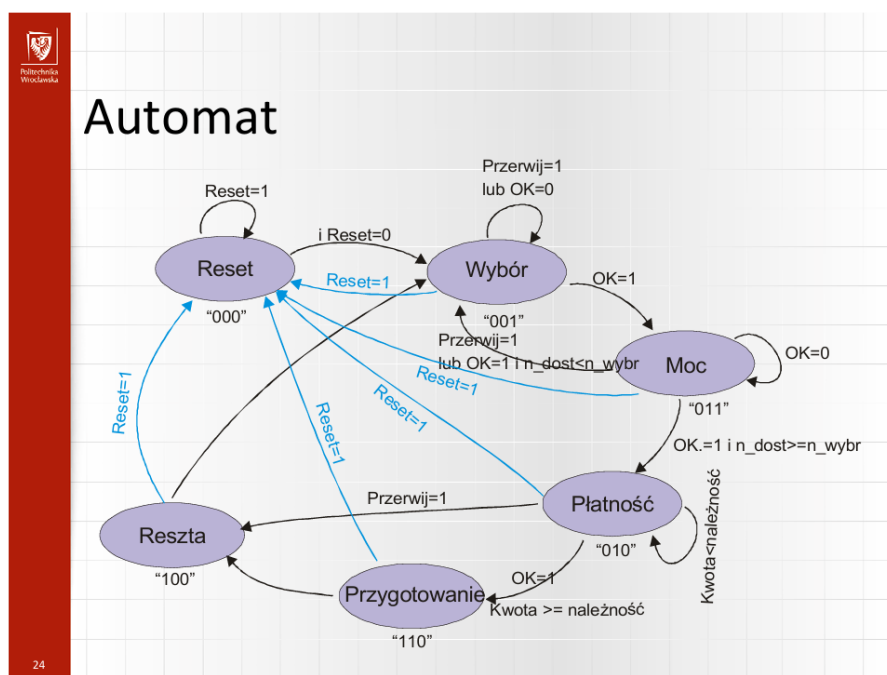
Celem projektu było stworzenie testbenchu, napisanego w języku Verilog, pozwalającego na weryfikację działania automatu sprzedającego napoje. Zbadano przypadek poprawnego zakupu napoju oraz dwa przypadki szczególne, w których, zgodnie ze specyfikacją, automat nie powinien pozwolić na zakup.

2. Działanie automatu

Automat, według specyfikacji, cechuje się sześcioma wewnętrznymi stanami, przedstawionymi na Rys. 1. Pozwala na zakup trzech rodzajów napojów (kawy, herbaty oraz soku) o trzech poziomach intensywności. Zawiera w sobie rejestry, przechowujące ceny i stan zasobnika dla poszczególnych napojów oraz liczbę wrzuconych do automatu monet.

Dostęp do rejestrów realizowany jest poprzez wejścia adres, nr_w oraz data_we. Dane odczytane z rejestrów dostępne są na wyjściu dane_wy.

Maszyna wyposażony jest w 3 przyciski, którymi użytkownik nawigować może interfejs automatu - są to przyciski OK, SELECT oraz BREAK.



Rysunek 1. Graf stanów wewnętrznych badanego automatu

3. Weryfikacja działania automatu

Podczas weryfikacji działania układu korzystano z zadań napisanych w języku Verilog, porównujących uzyskane dane z oczekiwanymi oraz informującymi o wyniku porównania

na konsoli, wykorzystując polecenie `$display`. Przykładowe zadanie, realizujące porównanie wartości odczytanych z rejestru i oczekiwanych, przedstawiono na Rys. 2.

```

605 // Zadanie ostrzeżenia o nieprawidłowym zapisie do rejestru
606 task assertRegisterReadout;
607     input[16*8:1] registerName; // Nazwa rejestru w formacie string, zawarta w wyświetlanej wiadomości
608     input[7:0] obtained; // Odczytana wartość z rejestru (w założeniu reg dane_wy)
609     input[7:0] expected; // Oczekiwana wartość
610
611     begin
612         if (obtained == expected) begin
613             $display("%d ns: Zawartość rejestru [%s] prawidłowa (%d)", $time, registerName, obtained);
614         end else begin
615             $display("%d ns: Nieprawidłowa zawartość rejestru [%s]. Oczekiwano %d, otrzymano %d", $time, registerName, $signed(expected), $signed(obtained));
616         end
617     end
618 endtask

```

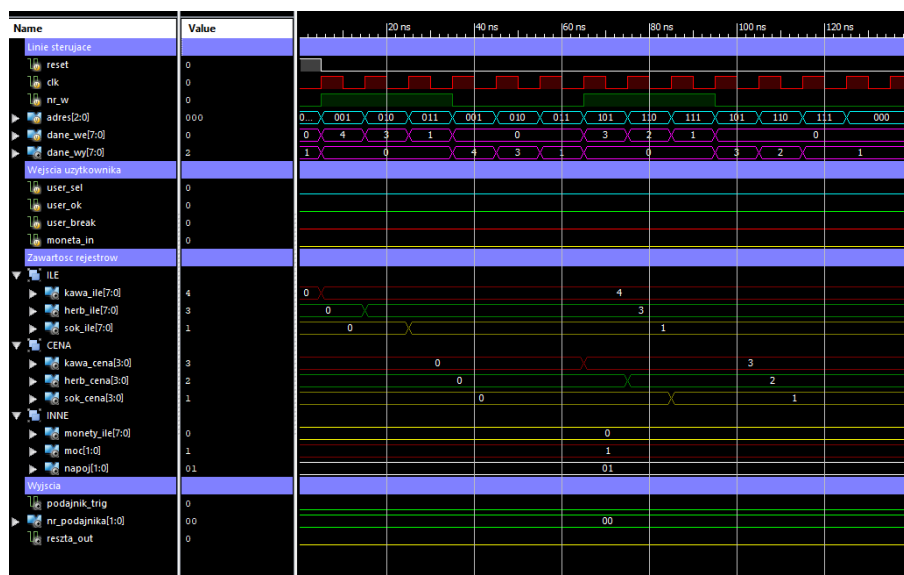
Rysunek 2. Przykładowe zadanie wykorzystane w projekcie

3.1. Zapis i odczyt danych z rejestrów

Sprawdzono poprawność implementacji systemu rejestrów poprzez zapisanie do nich parametrów napojów, wykorzystywanych w dalszych krokach weryfikacji:

- Kawa: ilość 4, cena 3
- Herbata: ilość 3, cena 2
- Sok: ilość 1, cena 1

Następnie odczytano zapisane wartości i porównano je z oczekiwanymi, wykorzystując zadanie `assertRegisterReadout`. Przebiegi czasowe sygnałów przedstawiono na Rys. 3. Wyjście uzyskane na konsoli przedstawiono na Rys. 4. Zachowanie maszyny stanów było zgodne z oczekiwaniami.



Rysunek 3. Przebiegi czasowe dla weryfikacji działania rejestrów układu

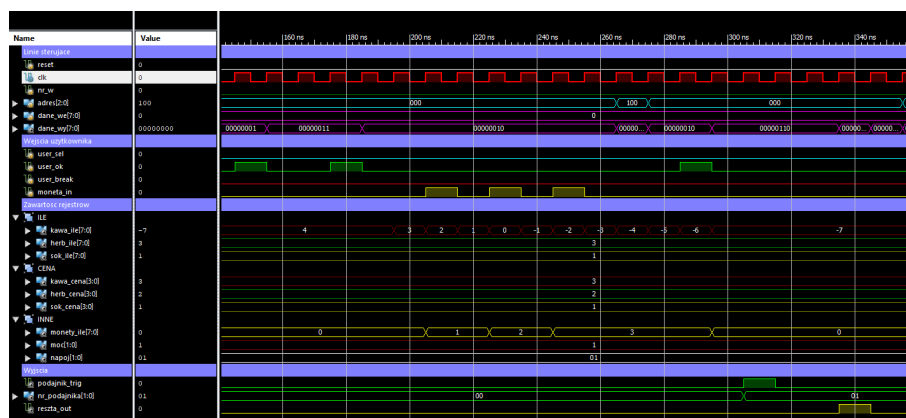
Test 1: Zapis i odczyt wartosci z rejestrow

5 ns: Zmiana wartosci Reset na 0
 45 ns: Zawartosc rejestru [KAWA_ile] prawidłowa (4)
 55 ns: Zawartosc rejestru [HERB_ile] prawidłowa (3)
 65 ns: Zawartosc rejestru [SOK_ile] prawidłowa (1)
 105 ns: Zawartosc rejestru [KAWA_cena] prawidłowa (3)
 115 ns: Zawartosc rejestru [HERB_cena] prawidłowa (2)
 125 ns: Zawartosc rejestru [SOK_cena] prawidłowa (1)

Rysunek 4. Wyjście konsoli dla weryfikacji działania rejestrów układu

3.2. Zamówienie kawy z domyślną mocą napoju

Sprawdzono działanie układu przy zamówieniu domyślnego napoju (kawy) z domyślną intensywnością (1) oraz przy wrzuceniu odpowiedniej ilości monet. Automat pozwolił na zakup napoju, jednak zaobserwowano, że rejestr KAWA_ile stale dekrementuje swoją wartość. Pod koniec scenariusza wartość kawy zdążyła zejść poniżej zera (integer underflow). Oprócz tej zależności nie zaobserwowano odstępstw od specyfikacji.



Rysunek 5. Przebiegi czasowe dla poprawnego zamówienia kawy

```

Test 2: Zakup domyslnego napoju z dokladnie wyliczona liczba monet
135 ns: Stan maszyny prawidlowy: 0b001, [ STATE_Wybor]
145 ns: Wcisniecie przycisku OK
165 ns: Stan maszyny prawidlowy: 0b011, [ STATE_Moc]
175 ns: Wcisniecie przycisku OK
195 ns: Stan maszyny prawidlowy: 0b010, [ STATE_Platnosc]
205 ns: Wrzucono monete nr 1
225 ns: Wrzucono monete nr 2
245 ns: Wrzucono monete nr 3
275 ns: Zawartosc rejestru [ MONETY_ile] prawidlowa ( 3)
285 ns: Wcisniecie przycisku OK
305 ns: Stan maszyny prawidlowy: 0b110, [ STATE_Przygotowanie]
335 ns: Wybrany napoj prawidlowy: 0b001
335 ns: Uzyskano prawidlowa moc napoju: 1
345 ns: Stan maszyny prawidlowy: 0b100, [ STATE_Reszta]
355 ns: Stan maszyny prawidlowy: 0b001, [ STATE_Wybor]
365 ns: Zawartosc rejestru [ MONETY_ile] prawidlowa ( 0)
375 ns: Nieprawidlowa zawartosc rejestru [ KAWA_ile]. Oczekiwano 3, otrzymano -7

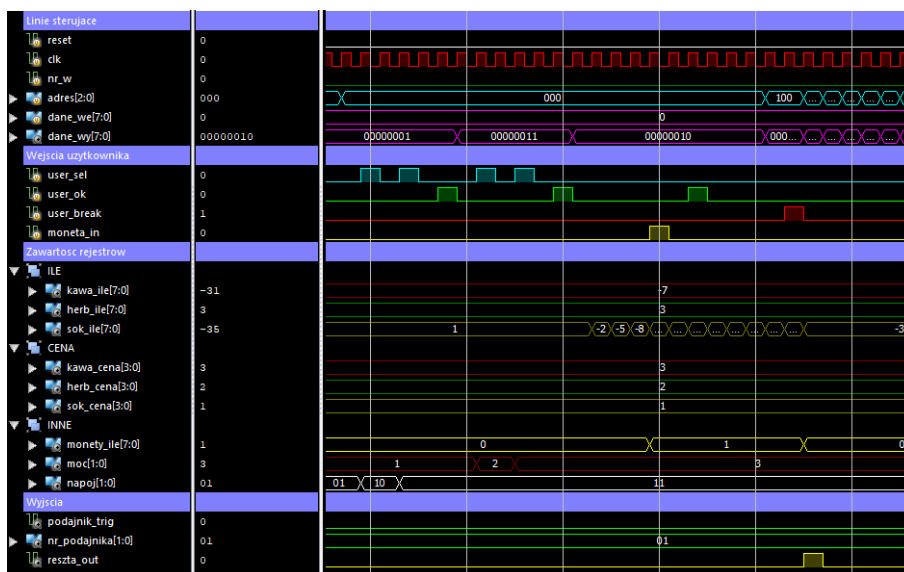
```

Rysunek 6. Wyjście konsoli dla poprawnego zamówienia kawy

3.3. Zamówienie soku z mocą napoju większą od stanu zasobnika

Przy zakupie soku o intensywności napoju większej od stanu zasobnika zaobserwowano, że implementacja automatu jest niezgodna ze specyfikacją - według grafu stanów, wciśnięcie przycisku `user_ok` przy zbyt małym stanie zasobnika powinno spowodować powrót automatu do stanu Wybór. Zauważono jednak, że w tej sytuacji automat przechodzi do stanu Płatność. Wrzucenie monet nie pozwala jednak na wykonanie zakupu - konieczne jest wykorzystanie przycisku `user_break` do przejścia do stanu Reszta, w którym automat zwraca wrzuconą monetę.

Zaobserwowano podobne zachowanie, co w sekcji 3.2 - zawartość rejestru `SOK_ile` stale zmniejsza się o wybraną intensywność napoju.



Rysunek 7. Przebiegi czasowe dla zamówienia soku przy zbyt małym stanie zasobnika

```

Test 3: Zakup soku o intensywnosci napoju wiekszej od stanu zasobnika
385 ns: Zawartosc rejestru [   SOK_ile] prawidłowa ( 1)
395 ns: Stan maszyny prawidłowy: 0b001, [   STATE_Wybor]
395 ns: Wcisniecie przycisku SELECT
415 ns: Wcisniecie przycisku SELECT
435 ns: Wcisniecie przycisku OK
455 ns: Stan maszyny prawidłowy: 0b011, [   STATE_Moc]
455 ns: Wcisniecie przycisku SELECT
475 ns: Wcisniecie przycisku SELECT
495 ns: Wcisniecie przycisku OK
545 ns: Bład maszyny stanow. Oczekiwano 0b001 (   STATE_Wybor), otrzymano 0b010 (   STATE_Platnosc)
545 ns: Wrzucono monete nr      1
565 ns: Wcisniecie przycisku OK
605 ns: Bład maszyny stanow. Oczekiwano 0b110 (STATE_Przygotowanie), otrzymano 0b010 (   STATE_Platnosc)
615 ns: Zawartosc rejestru [  MONETY_ile] prawidłowa ( 1)
615 ns: Wcisniecie przycisku BREAK
635 ns: Stan maszyny prawidłowy: 0b100, [   STATE_Reszta]
645 ns: Zawartosc rejestru [  MONETY_ile] prawidłowa ( 0)
655 ns: Stan maszyny prawidłowy: 0b001, [   STATE_Wybor]
665 ns: Nieprawidłowa zawartosc rejestru [   SOK_ile]. Oczekiwano  1, otrzymano -35

```

Rysunek 8. Wyjście konsoli dla zamówienia soku przy zbyt małym stanie zasobnika

3.4. Zamówienie herbaty przy wrzuceniu zbyt małej liczby monet

Automat, po wrzuceniu zbyt małej liczby monet, zachowywał się zgodnie ze specyfikacją - nie pozwolił na zakup napoju, przeszedł przez wszystkie stany zgodnie z oczekiwaniami. Prawidłowo zwrócił resztę. Zaobserwowano zachowanie tożsame z poprzednimi testami - zawartość rejestru HERBATA_ile stale zmniejsza się.



Rysunek 9. Przebiegi czasowe dla zakupu herbaty przy wrzuceniu zbyt małej ilości monet

Test 4: Zakup herbaty przy wrzuceniu zbyt małej ilości monet

```

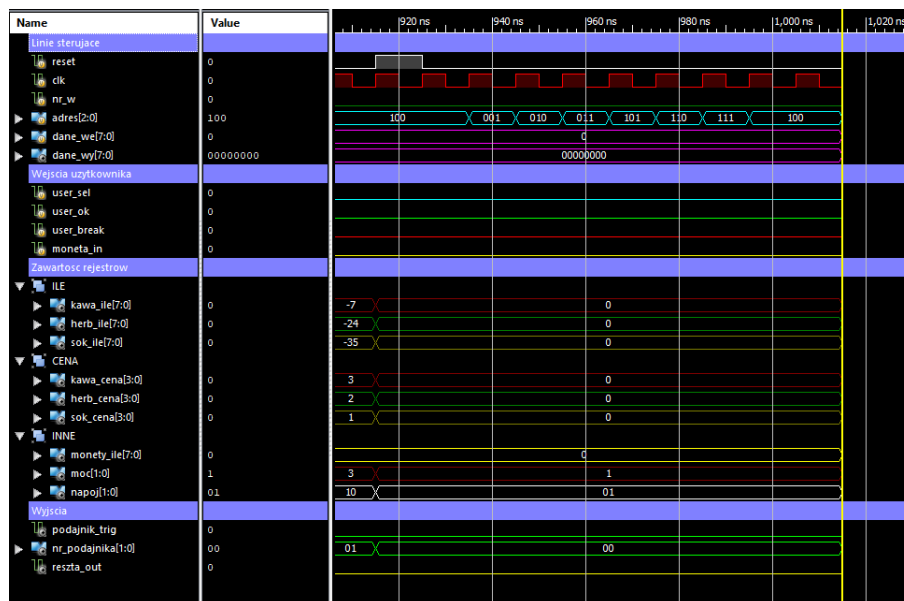
675 ns: Zawartosc rejestru [  HERB_cena] prawidłowa ( 2)
685 ns: Stan maszyny prawidłowy: 0b001, [  STATE_Wybor]
685 ns: Wcisniecie przycisku SELECT
705 ns: Wcisniecie przycisku SELECT
735 ns: Wcisniecie przycisku OK
765 ns: Stan maszyny prawidłowy: 0b011, [  STATE_Moc]
765 ns: Wcisniecie przycisku OK
795 ns: Stan maszyny prawidłowy: 0b010, [  STATE_Platnosc]
795 ns: Wrzucono monetę nr      1
825 ns: Zawartosc rejestru [  MONETY_ile] prawidłowa ( 1)
825 ns: Wcisniecie przycisku OK
855 ns: Stan maszyny prawidłowy: 0b010, [  STATE_Platnosc]
855 ns: Wcisniecie przycisku BREAK
875 ns: Stan maszyny prawidłowy: 0b100, [  STATE_Reszta]
885 ns: Stan maszyny prawidłowy: 0b001, [  STATE_Wybor]
905 ns: Zawartosc rejestru [  MONETY_ile] prawidłowa ( 0)
925 ns: Nieprawidłowa zawartosc rejestru [  HERB_ile]. Oczekiwano  3, otrzymano -24

```

Rysunek 10. Wyjście konsoli dla zakupu herbaty przy wrzuceniu zbyt małej ilości monet

3.5. Reset rejestrów

Reset rejestrów był zgodny ze specyfikacją - automat przeszedł do stanu Reset, wartości rejestrów przyjęły wartość 0.



Rysunek 11. Przebiegi czasowe dla resetu rejestrów


```

Test 5: Weryfikacja poprawności działania stanu Reset
925 ns: Zmiana wartości Reset na 1
935 ns: Stan maszyny prawidłowy: 0b000, [    STATE_Reset]
935 ns: Zmiana wartości Reset na 0
955 ns: Zawartosc rejestru [    KAWA_ile] prawidłowa ( 0)
965 ns: Zawartosc rejestru [    HERB_ile] prawidłowa ( 0)
975 ns: Zawartosc rejestru [    SOK_ile] prawidłowa ( 0)
985 ns: Zawartosc rejestru [    KAWA_cena] prawidłowa ( 0)
995 ns: Zawartosc rejestru [    HERB_cena] prawidłowa ( 0)
1005 ns: Zawartosc rejestru [    SOK_cena] prawidłowa ( 0)
1015 ns: Zawartosc rejestru [    MONETY_ile] prawidłowa ( 0)

```

Rysunek 12. Wyjście konsoli dla resetu rejestrów

4. Wnioski

Występujące rozbieżności implementacji automatu od specyfikacji wskazują, że należałoby przeprowadzić bardziej dogłębną weryfikację. Stale zmniejszające się wartości rejestrów oraz przejścia do nieodpowiednich stanów wewnętrznych sugerują możliwe błędy w kodzie języka opisu sprzętu.

Spis rysunków

1	Graf stanów wewnętrznych badanego automatu	3
2	Przykładowe zadanie wykorzystane w projekcie	4
3	Przebiegi czasowe dla weryfikacji działania rejestrów układu	4
4	Wyjście konsoli dla weryfikacji działania rejestrów układu	5
5	Przebiegi czasowe dla poprawnego zamówienia kawy	5
6	Wyjście konsoli dla poprawnego zamówienia kawy	6
7	Przebiegi czasowe dla zamówienia soku przy zbyt małym stanie zasobnika .	6
8	Wyjście konsoli dla zamówienia soku przy zbyt małym stanie zasobnika . .	7
9	Przebiegi czasowe dla zakupu herbaty przy wrzuceniu zbyt małej ilości monet	7
10	Wyjście konsoli dla zakupu herbaty przy wrzuceniu zbyt małej ilości monet	8
11	Przebiegi czasowe dla resetu rejestrów	8
12	Wyjście konsoli dla resetu rejestrów	9