

Projektowanie systemów cyfrowych

Układ wykonawczy (operacyjny)

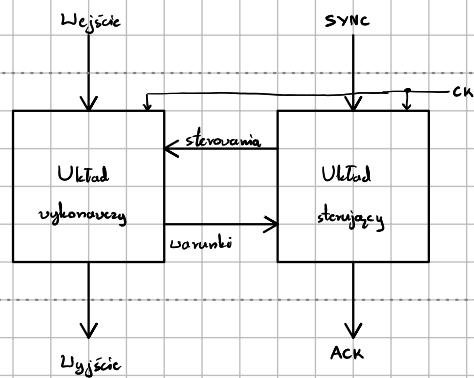
- wykonuje określone operacje
- obsługuje wejścia/wyjścia

Układ sterujący

- kontroluje kolejność wykonywania operacji
- warunki jako wejście
- sterowania jako wyjście

Mожет быть реализовано как

- zwykły automat } nie jest wygodne
- rozdzielacz sterujący }
- układ mikroprogramowany w pamięci ROM



Układy komunikują się sygnałami sterowania i warunków

Układy są taktowane wspólnym zegarem

Mogą (albo muszą) zastosować zegar dławiaczy

- jeden z układów otrzymuje zamieniony sygnał zegarowy
- układ sterowania elektrycznie otrzymuje sygnał opóźniony o poł taktu
- wymusza kolejność między układami

Jest wymagany w automatach Moore'a, gdzie w komórce warunkowej sprawduje się warunek ustalony w ostatniej komórce operacyjnej

Siec działania algorytmu

Schemat blokowy, składa się z klatek operacyjnych i warunkowych, kroków Start i Stop.

Klatka warunkowa

- warunkowe rozgałęzienie

Klatka operacyjna

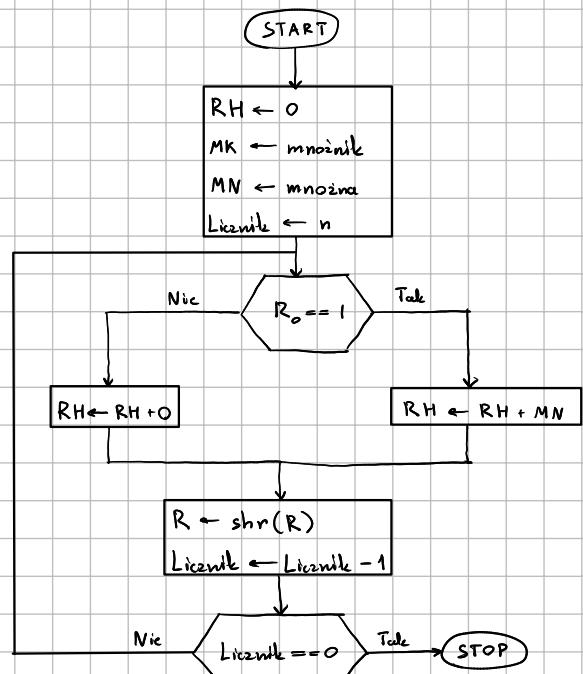
- synchroniczne operacje
- wszystkie operacje wykonują się w 1 taktie zegara

Siec nie zawiera operacji kombinacyjnych, tylko synchronicznych
(np. nie dodawanie ale zapisanie wyniku dodawania do rejestru)

Zamiast sprawdzania kilku warunków na raz, można ustawić

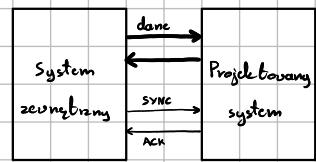
pomiędzy sprawdzaniem klatek operacyjnych, które nie wykonują żadnej operacji (NOP - no operation)

Przykładowa sieć działań dla algorytmu mnożenia



Przeplot (handshaking)

Prosty protokół komunikacyjny, umożliwiający komunikację systemu z otoczeniem zapewniający poprawność odczytywanych i zapisywanych danych



SYNC - synchronize - jest gotowe nowe wejście

ACK - acknowledge - jest gotowe wyjście

Mожет być zrealizowany synchronicznie lub asynchronicznie

Koniec przy każdej próbie odczytu wejścia

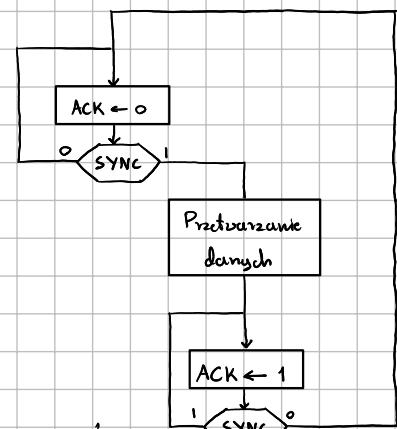
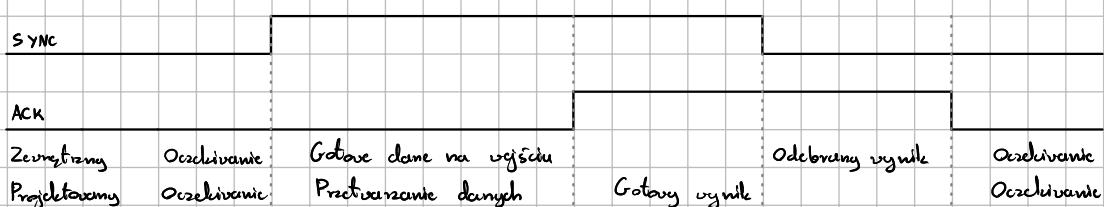
Odczyt szeregowy - pełna przeplotu przy odczycie każdej danej

Produkowanie sygnału ACK

np. przesuwnik asynchroniczny RS

na podstawie sygnałów sterowania

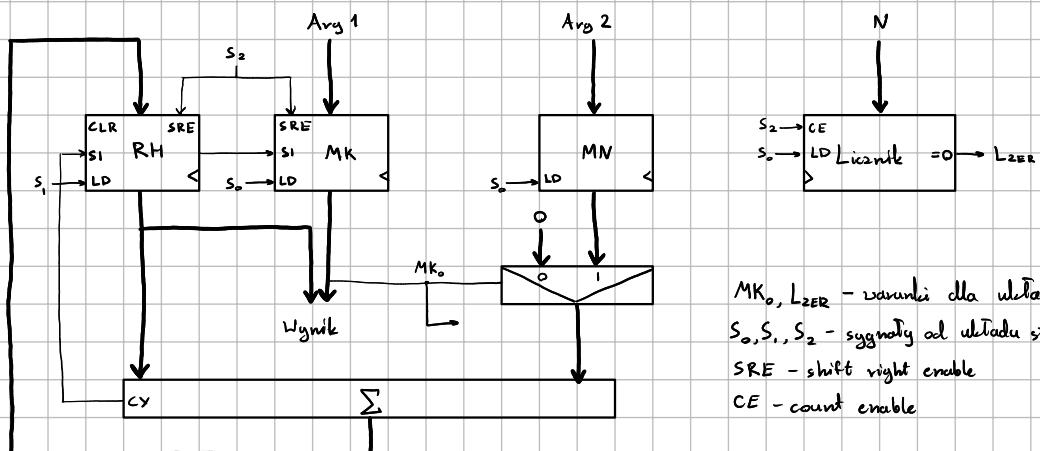
synchroniczne lub asynchroniczne



Projekt układu wykonawczego dla algorytmu mnożenia

- Rejestr na wynik i mnożnik R
- Rejestr na mnożnica
- Licznik getli
- Sumator - kombinacyjny
- Muliplikser - rozgałęzienia warunkowe

Trzeba ustalić szerokość szyn danych



MK₀, L_{2ER} - warunki dla układu sterowania

S₀, S₁, S₂ - sygnały do układu sterowania

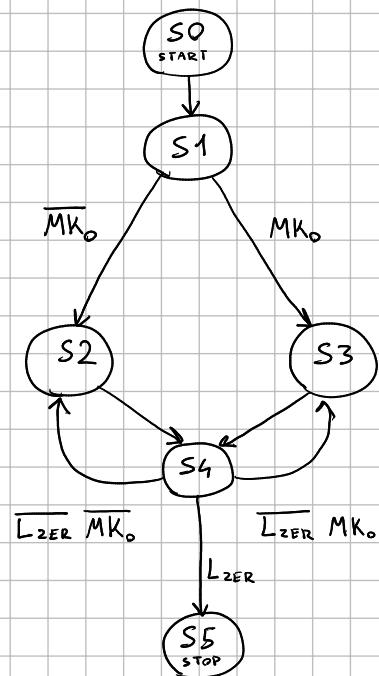
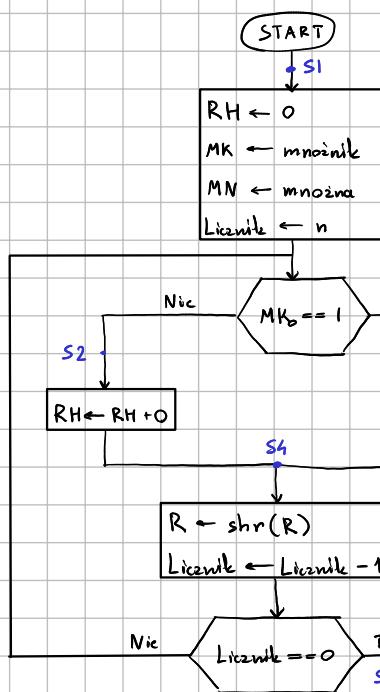
SRE - shift right enable

CE - count enable

Realizacja układu sterowania

Sequencja działań → graf automatu → tabela automatu

Stworzenie grafu automatu na podstawie sekwencji działań
ustawiaj stan przed każdą kroką operacyjną



Stworzenie tabeli na podstawie grafu

S \ S _i	00	01	11	10	S ₂	S ₃	S ₄	Liggicie
0	1	1	1	1	0	0	0	
1	2	3	3	2	0	0	1	Ładowanie MK, MN i Liczniłka, zerowanie RH
2	4	4	4	4	0	0	0	Ładowanie RH + 0 do RH
3	4	4	4	4	0	1	0	Ładowanie RH + MN do RH
4	2	3	5	5	1	0	0	Przesunięcie R w prawo, dekrementacja Liczniłka
5	5	5	5	5	0	0	0	

Realizacja jako związkły automat

Każda zmiana w projekcie wymaga zaprojektowania automatu od nowa

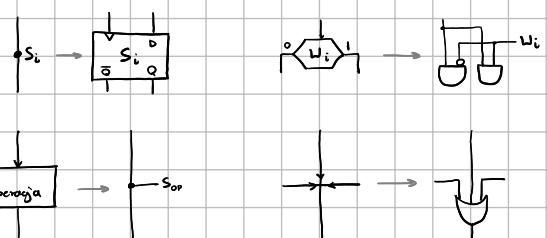
Niepraktyczne, niewygodne rozwiązywanie

Realizacja jako rozdzielacz sterujący
Zakodowane stanów w kodzie 1:n

Stworzenie schematu na podstawie sekwencji działań według schematu

Tylko 1 z przewodników D zaznacza wartość 1 w każdym momencie

Musi zostać zainicjowany w odpowiednim stanie po przywołaniu zasilania

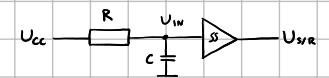


Układ iniekcji z bramką Schmitta

Zapewnia, że układ przejmuje określony stan przy ustawieniu zasilania

Układ RC zapewnia stopniowe narastanie napięcia.

Użytkowne opóźnienie zostało czas na ustalenie przesztuknika w pożądany stan

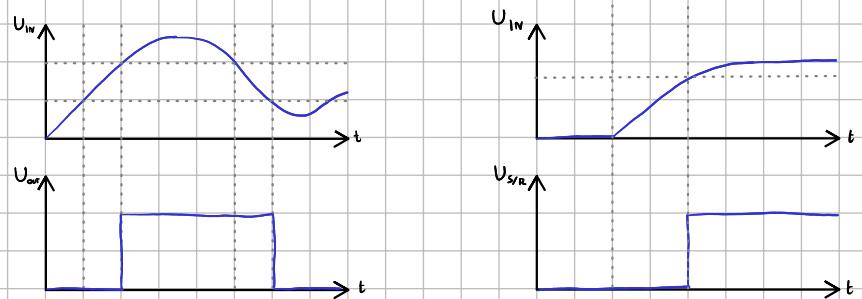


Najpierw wszystkie przesztukiują się, zatrzymują.

Po czacie, do pierwszego przesztuknika zatrudnia się 1.



Bramka Schmitta - histeresa



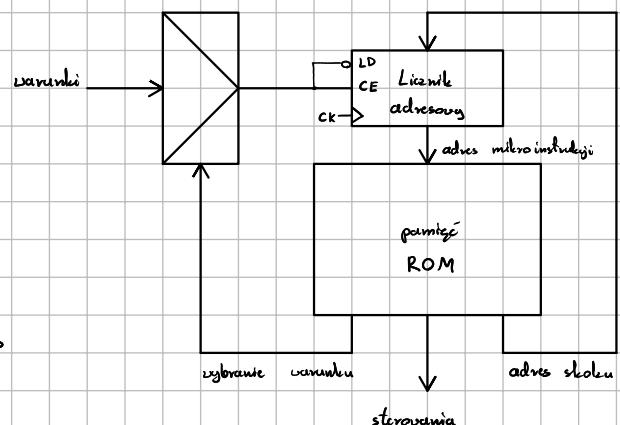
Mikroprogramowany układ sterujący

realizuje automat synchroniczny

Pamięć ROM zawiera mikroinstrukcje mikroprogramu.

Mikroinstrukcja składa się z

- kodu warunku
- wyjścia - sygnału sterowania dla układu wykonawczego
- adresu skoku warunkowego



Multipleser

- wybiera za pomocą kodu sygnał warunku z układu wykonawczego
- wyjście steruje licznikiem

Licznik adresowy

- przechowuje adres wykonywanej mikroinstrukcji
- warunkowo albo się inkrementuje - następna instrukcja albo uzytuje adres - skok warunkowy

Zapisanie mikroprogramu na podstawie tabeli automatu

SYN, L ₂ , MK ₀	000 001 010 011 100 101 110 111	S ₃ S ₂ S ₁ S ₀
1	1 1 1 1 2 2 2 2	0 0 0 1
2	4 3 4 3 4 3 4 3	0 0 0 0
3	4 4 4 4 4 4 4 4	0 0 1 0
4	2 2 5 5 2 2 5 5	0 1 0 0
5	1 1 1 1 5 5 5 5	1 0 0 0

1. 0001; if SYNC then goto 2 else goto 1
2. 0000; if MK₀ then goto 3 else goto 4
3. 0010; if TRUE then goto 4 else goto -
4. 0100; if L₂ then goto 5 else goto 2
5. 1000; if SYNC then goto 5 else goto 1

Budowa układu mikroprogramowanego wymaga instrukcji typu

X. wyjście; if warunek then następna instrukcja else goto adres skoku

Trzeba zmodyfikować instrukcje

5. 1000; if SYNC then goto 5 else goto 1

↓

5. 1000; if SYNC then goto 6 else goto 5

6. 0000; if FALSE then goto - else goto 1

w klasycznej wersji
może być tylko 1 skok

Mikroprogram w pamięci ROM

Warunkom i adresom trzeba przypisać kody
odróżnione dla warunków X i not X

1. 0001; if SYNC then goto 2 else goto 1
2. 0000; if MK₀ then goto 3 else goto 4
3. 0010; if TRUE then goto 4 else goto -
4. 0100; if L₂ then goto 5 else goto 2
5. 1000; if ~~SYNC~~ then goto 6 else goto 5
6. 0000; if FALSE then goto - else goto 1

Warunek	Kod	Adres	Kod
SYNC	000	1	000
not SYNC	001	2	001
MK ₀	010	3	010
L ₂	011	4	011
TRUE	100	5	100
FALSE	101	6	101

Zaszentosć ROM

adres instrukcji	warunki			operacje			adres skoku			
	u ₂	u ₁	u ₀	s ₃	s ₂	s ₁	s ₀	a ₂	a ₁	a ₀
0 0 0	0	0	0	0	0	0	1	0	0	0
0 0 1	0	1	0	0	0	0	0	0	1	1
0 1 0	1	0	0	0	0	0	1	0	-	-
0 1 1	0	1	1	0	1	0	0	0	0	1
1 0 0	0	0	1	1	0	0	0	1	0	0
1 0 1	1	0	1	0	0	0	0	0	0	0

Modyfikacje klasycznego układu

Dla specyficznych potrzeb projektu można wprowadzić modyfikacje do klasycznego układu

Mozna np. wprowadzić kompresję mikroinstrukcji w celu zaoszczędzenia pamięci

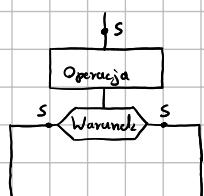
Przykłady

- multiplexer adresowy - więcej możliwości warunkowych skoków
- układ modyfikujący adresu } dekodery do skompresowanych
- układ modyfikujący sterowania } adresów sterowań

Realizacja automatau Moore'a i Mealy'ego

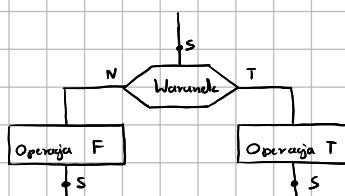
Automat Moore'a

- stan przed klatką operacyjną
- pełna musi zatrzymać stan



Automat Mealy'ego

- stan przed klatką warunkową
- klatka operacyjne muszą być rozdzielone stanem



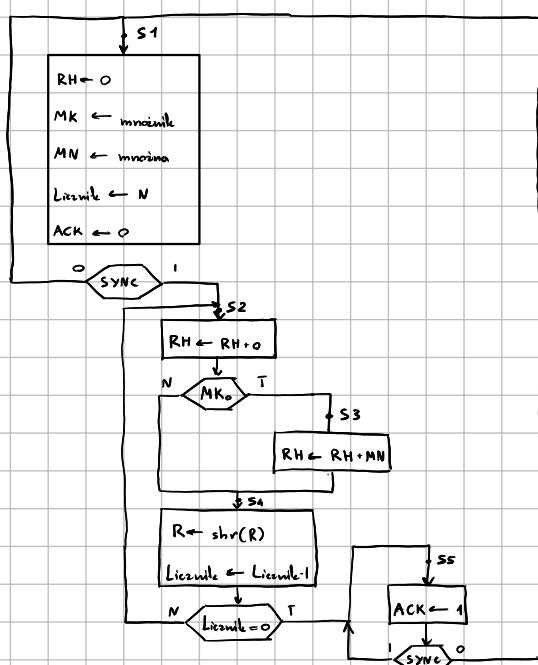
Adres: operacja; if warunek

then goto adres +1
else goto adres skoku

Adres: if warunek

then operacja T; goto adres +1
else operacja F; goto adres skoku

Układ mnożacy w NKB, z przepłotem realizacji

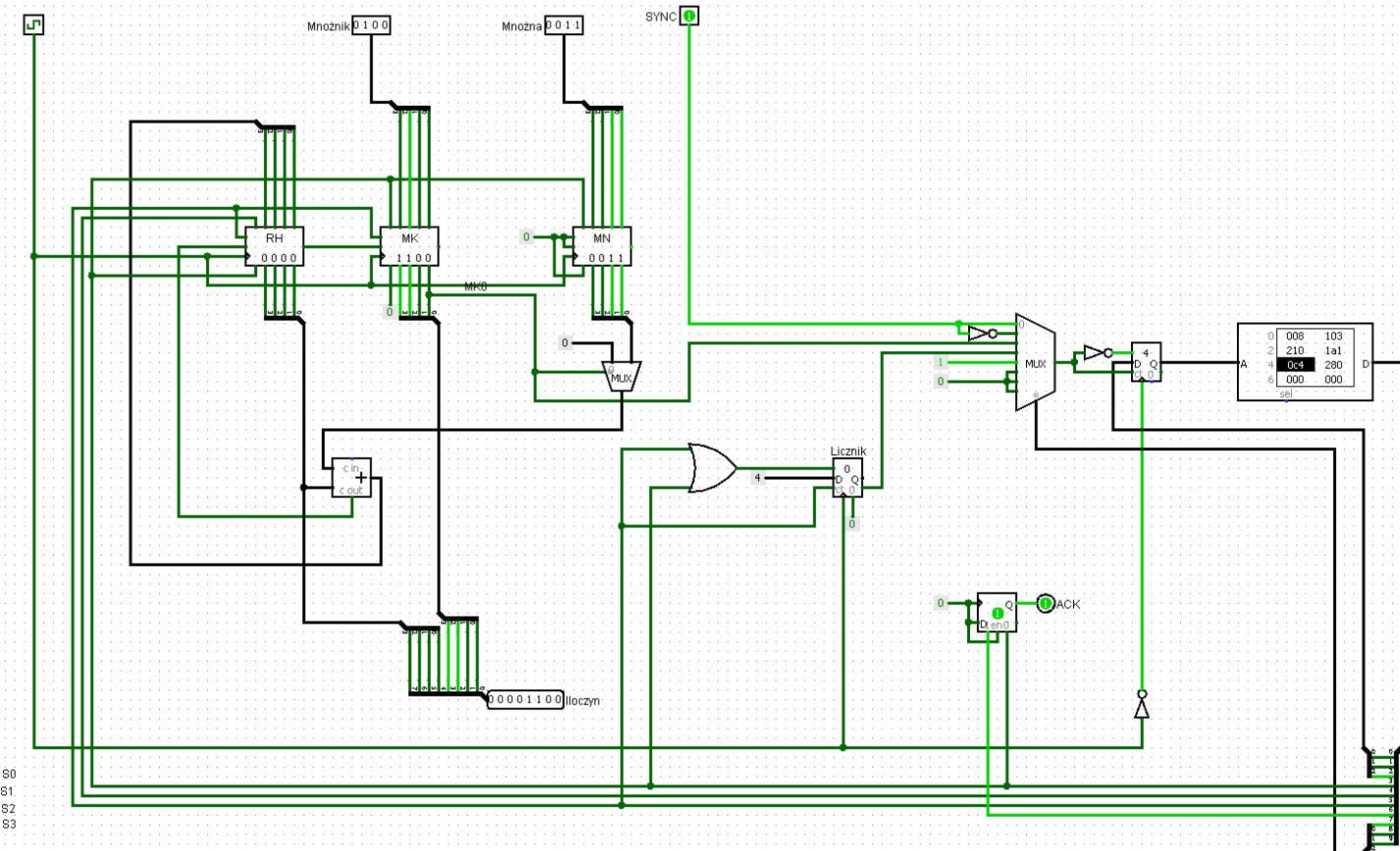


STAN	000	001	010	011	100	101	110	111	S ₂	S ₂	S ₁	S ₀
1	1	1	1	1	2	2	2	2	0	0	0	1
2	4	3	4	3	4	3	4	3	0	0	0	0
3	4	4	4	4	4	4	4	4	0	0	1	0
4	2	2	5	5	2	2	5	5	0	1	0	0
5	1	1	1	1	5	5	5	5	1	0	0	0

1. 0001; if SYNC then goto 2 else goto 1
2. 0000; if MK₀ then goto 3 else goto 4
3. 0010; if TRUE then goto 4 else goto -
4. 0100; if L₂ then goto 5 else goto 2
5. 1000; if SYNC then goto 6 else goto 5
6. 0000; if FALSE then goto - else goto 1

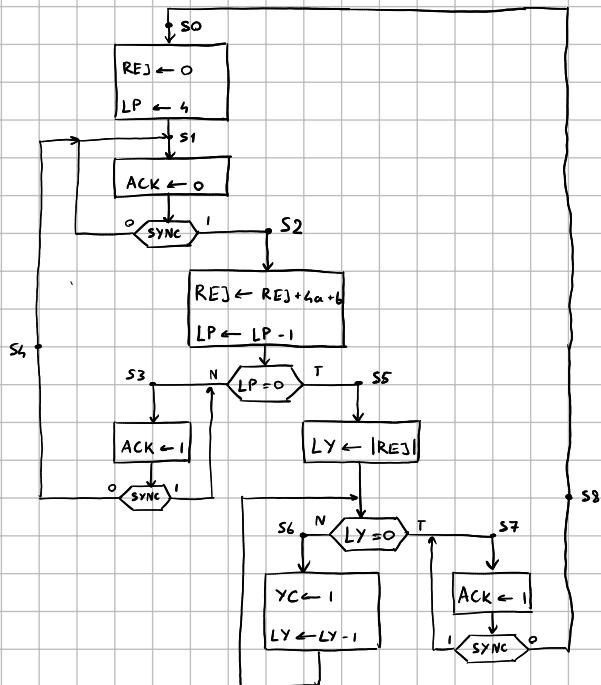
adres instrukcji	warunki	operacje	adres stroku					HEX		
			u ₂	u ₁	u ₀	s ₃	s ₂	s ₁	s ₀	
0 0 0	0 0 0	0 0 0 1	0	0	0	0	0	0	0	0x008
0 0 1	0 1 0	0 0 0 0	0	1	1	0	0	0	0	0x103
0 1 0	1 0 0	0 0 1 0	-	-	-	0	0	1	0	0x210
0 1 1	0 1 1	0 1 0 0	0	0	1	0	0	1	0	0xA1
1 0 0	0 0 1	1 0 0 0	1	0	0	0	0	0	0	0x0C4
1 0 1	1 0 1	0 0 0 0	0	0	0	0	0	0	0	0x280

warunek	kad	adres	kad
SYNC	000	1	000
not SYNC	001	2	001
MK ₀	010	3	010
L ₂	011	4	011
TRUE	100	5	100
FALSE	101	6	101



Projekt učebadu

- dostaje na wejściu szeregowo 4 pary liczb w U2 (a_i, b_i)
- oblicza wartość $Y = \sum_{i=1}^4 (4a_i + b_i)$
- przepuszcza na wyjście $|Y|$ impulsów zegara
- wykorzystuje preplot

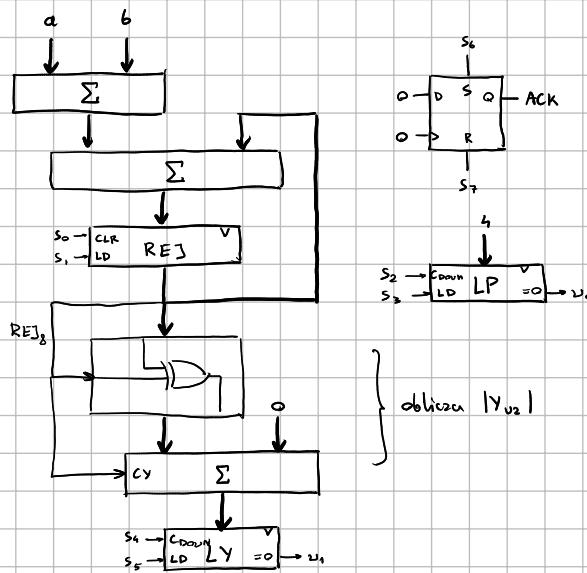


```

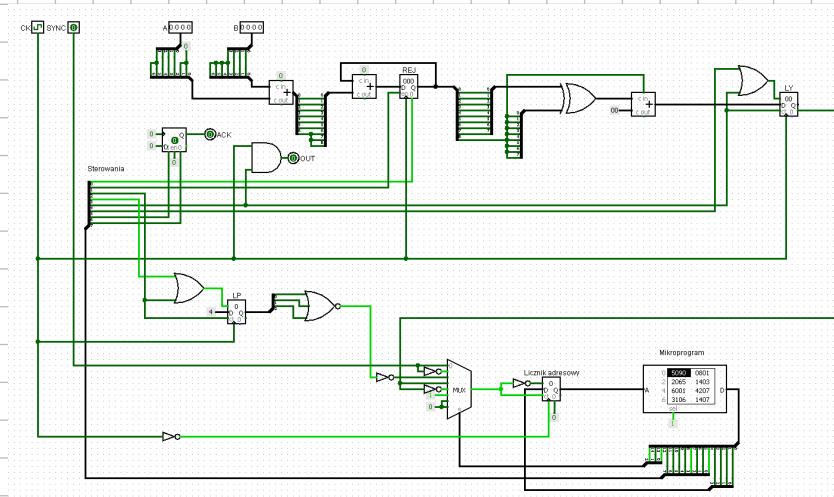
0. REJ=0, LP=4;
1. ACK=0;
2. REJ=REJ+4a+b, LP=LP-1;
3. ACK=1;
4. NOP;
5. LY=|REJ|;
6. YC=1; LY=LY-1;
7. ACK=1;
8. NOP;
  
```

if TRUE goto 1 else -
 if SYNC goto 2 else goto 1
 if not LP=0 goto 3 else goto 5
 if not SYNC goto 4 else goto 3
 if FALSE - else goto 1
 if not LY=0 goto 6 else goto 7
 if LY=0 goto 7 else goto 6
 if not SYNC goto 8 else goto 7
 if False - else goto 0

Warunek	Kod	Adres	Kod
SYNC	000	0	0000
not SYNC	001	1	0001
not W ₀	010	2	0010
W ₁	011	3	0011
not W ₁	100	4	0100
TRUE	101	5	0101
FALSE	110	6	0110
		7	0111
		8	1000



numer	u ₂	u ₁	u ₀	s ₇	s ₆	s ₅	s ₄	s ₃	s ₂	s ₁	s ₀	a ₂	a ₁	a ₀	HEX	
0000	1	0	1	0	0	0	0	1	0	0	1	0	0	0	0	0x5090
0001	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0x0801
0010	0	1	0	0	0	0	0	0	1	1	0	0	1	0	1	0x2065
0011	0	0	1	0	1	0	0	0	0	0	0	0	0	1	1	0x1403
0100	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0x6001
0101	1	0	0	0	0	1	0	0	0	0	0	0	1	1	1	0x4207
0110	0	1	1	0	0	0	1	0	0	0	0	0	1	1	0	0x3106
0111	0	0	1	0	1	0	0	0	0	0	0	0	1	1	1	0x1407
1000	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0x6000



bity liczb U2 są rozszerzone
mnożenie przez przesunięte bitowe

Mikrogram

Liczmi adresowy

MR