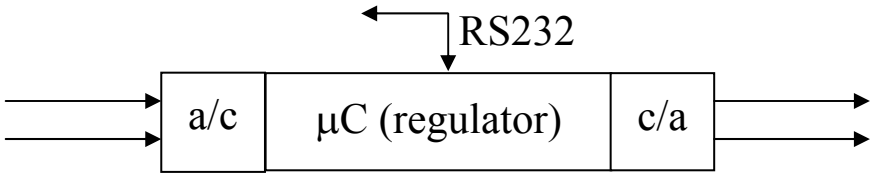
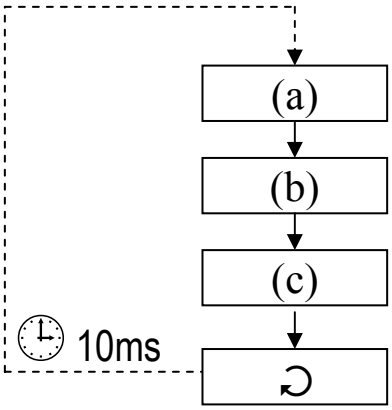
Organizacja oprogramowania

1. Cykliczny program sekwencyjny



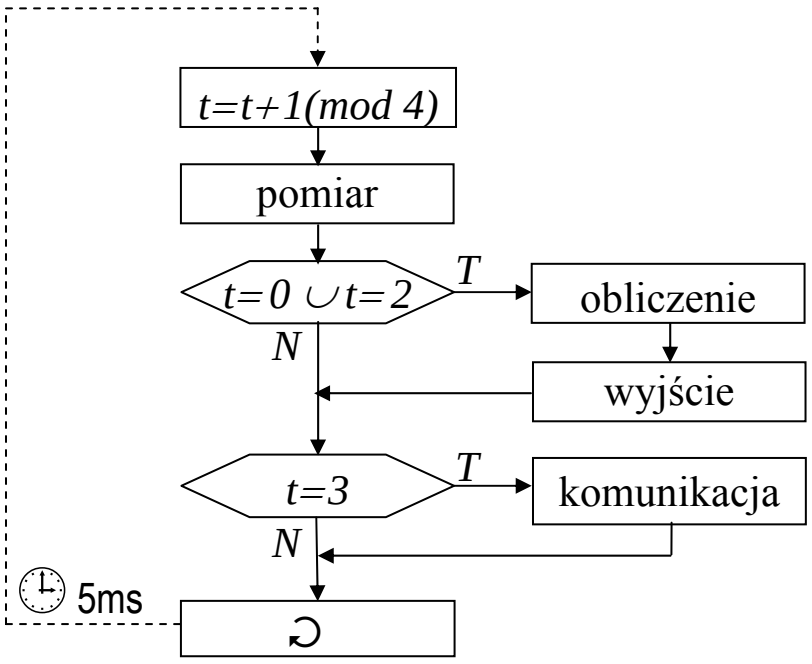
Zadania	czas wykonania	cykl
(a) pomiar	1 ms	≤ 10 ms
(b) obliczenie	2 ms	≤ 10 ms
(c) wyjście	1 ms	≤ 10 ms
	4 ms	



Program sekwencyjny – ręczne rozplanowanie zadań

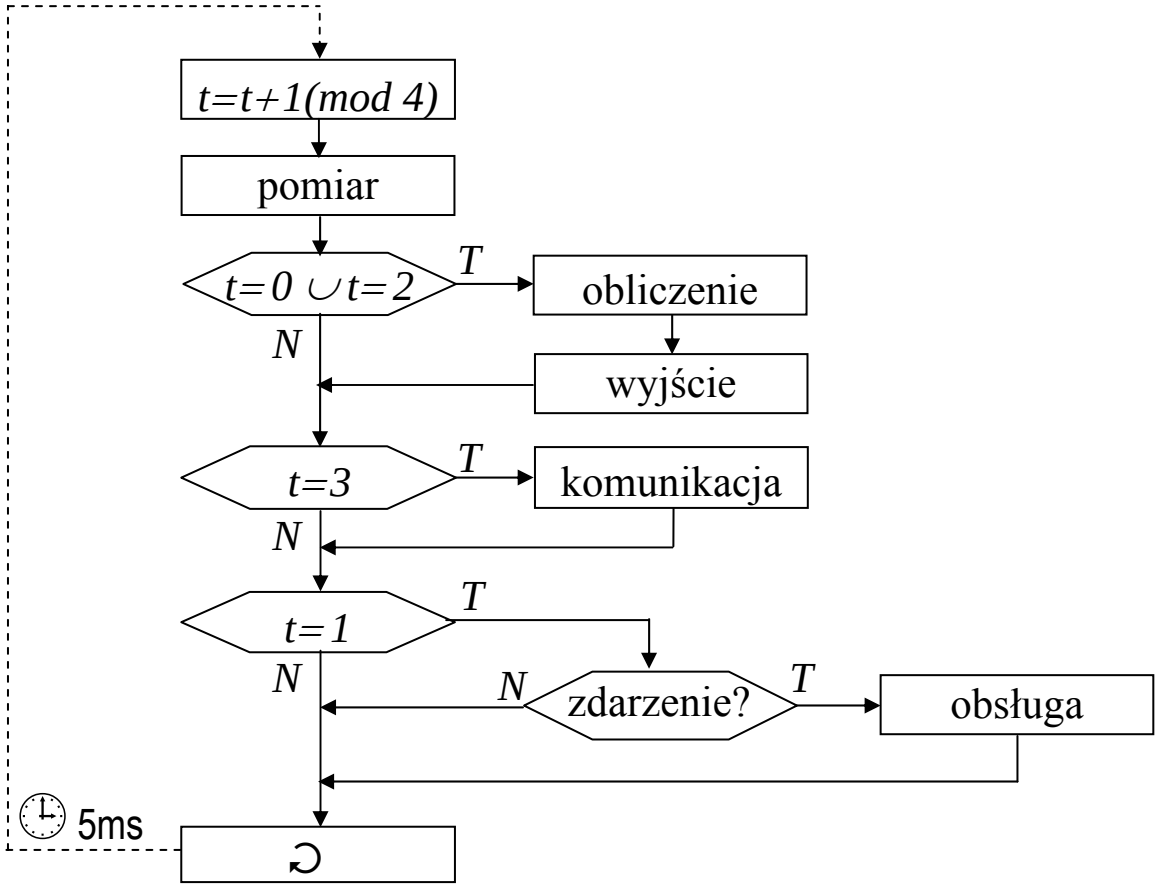
Zadania:	czas wyk.	cykl	0	1	2	3	
(a) pomiar	1 ms	≤ 5 ms	×	×	×	×	20%
(b) obliczenie	2 ms	≤ 10 ms	×		×		20%
(c) wyjście	1 ms	≤ 10 ms	×		×		10%
(d) komunikacja	4 ms	≤ 20 ms				×	20%
	8 ms		4	1	4	4	70%

pre-run scheduling

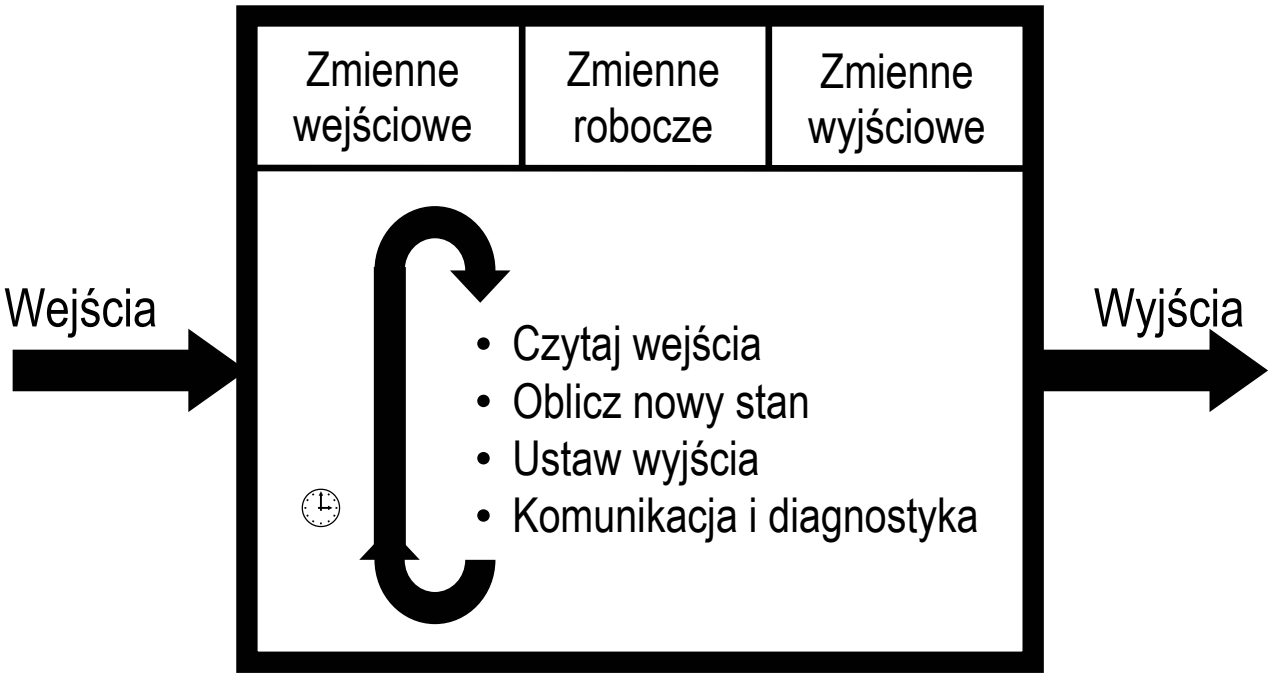


Program sekwencyjny – z uwzględnieniem zdarzeń

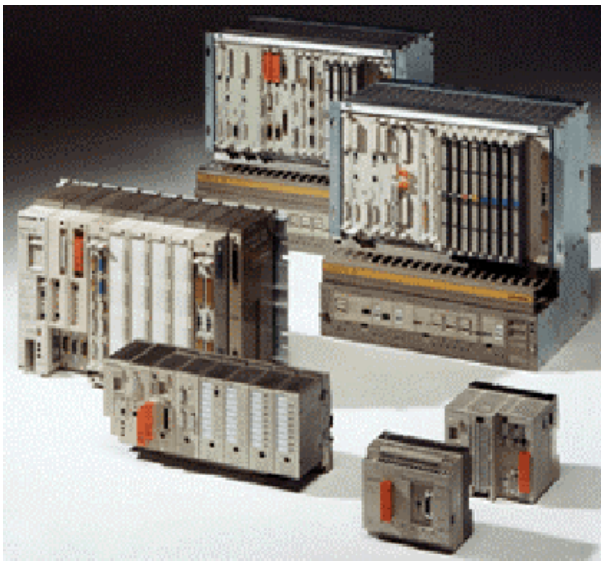
Zadania:	czas wyk.	cykl	0	1	2	3	
(a) pomiar	1 ms	≤ 5 ms	×	×	×	×	20%
(b) obliczenie	2 ms	≤ 10 ms	×		×		20%
(c) wyjście	1 ms	≤ 10 ms	×		×		10%
(d) komunikacja	4 ms	≤ 20 ms				×	20%
(e) przekroczenie	3 ms	≤ 20 ms		×			15%
	11 ms		4	4	4	5	85%



Przykład: sterownik PLC



Siemens, Simatic S5



Alerton, VLC-853

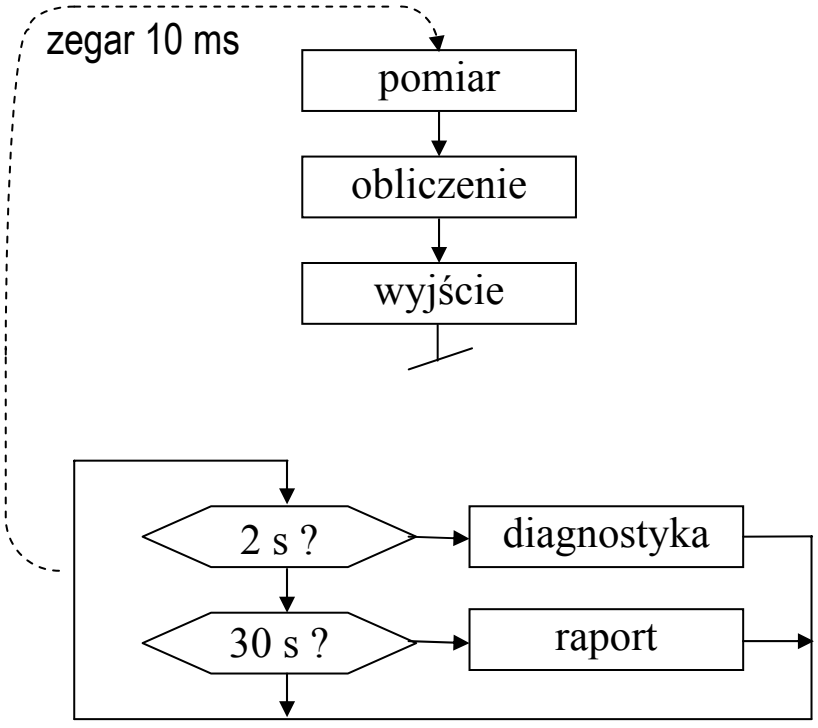
2. System dwuplanowy (*Foreground/Background*)

Zadania:	czas wyk.	cykl	
(a) pomiar	1 ms	≤ 10 ms	10%
(b) obliczenie	2 ms	≤ 10 ms	20%
(c) wyjście	1 ms	≤ 10 ms	10%
(d) diagnostyka	100 ms	≤ 2 s	5%
(d) raport	500 ms	≤ 30 s	1,5%
			46,5%

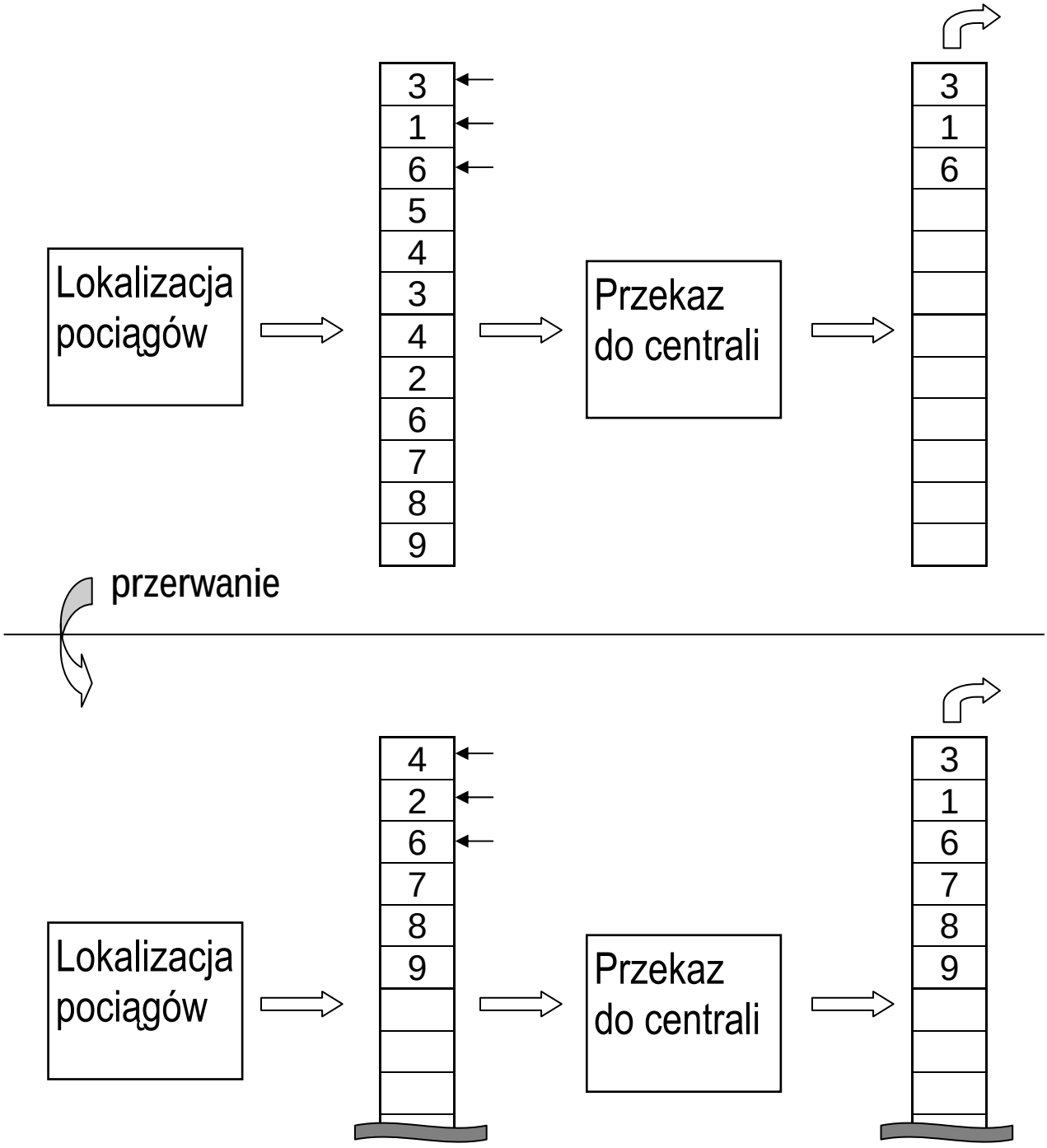
- Zadania planu pierwszego
- Zadania tła

Foreground

Background

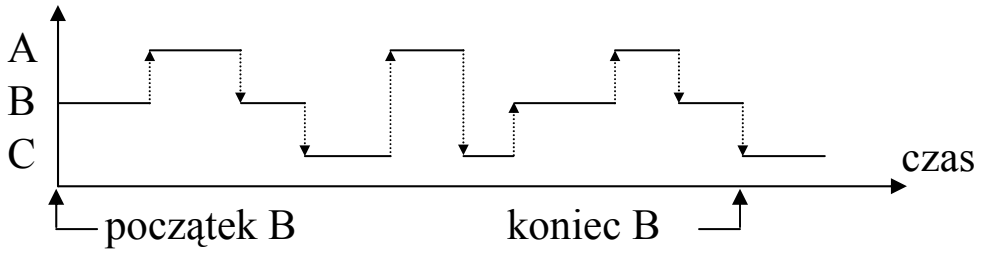


Przykład: komunikacja zadań



3. System wielozadaniowy (*Multitasking*)

- Wiele zadań o różnych charakterystykach czasowych:
  - sterowanie
  - komunikacja w sieci
  - archiwizacja i dokumentacja
  - GUI
- Zadania nie zawsze gotowe



- Złożona synchronizacja i komunikacja zadań

Szeregowanie zadań

Sformułowanie problemu

- Dany zbiór zadań do wykonania
- Dla każdego zadania z_i określone:
  - czas wykonania t_i
  - okres cyklu c_i
- Znaleźć właściwą kolejność wykonania zadań

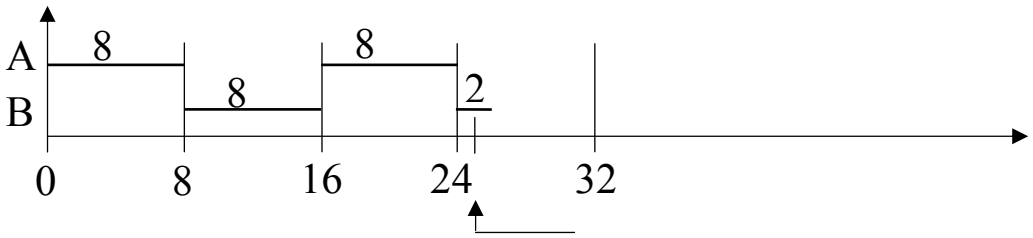
Algorytmy szeregowania

- RMS (*Rate Monotonic Scheduling*)
- EDF (*Earliest Deadline First*)

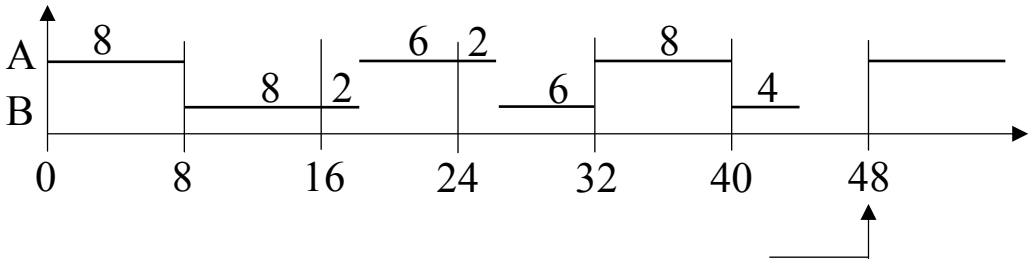
Przykłady

Zadanie	Długość	Cykl	Obciążenie
A	8	16	$8/16 = 50\%$
B	10	25	$10/25 = 40\%$
			$= 90\%$

- ***Rate monotonic scheduling (RMS)***



- ***Earliest deadline first (EDF)***



Szeregowalność

- **RMS – Twierdzenie** (*Liu, Layland, 1973*)

Jeżeli: $\sum_{i=1}^{n} \frac{t_i}{c_i} \leq n\,(2^{\frac{1}{n}} - 1)$

oraz szeregowanie jest zgodne z pilnością (*RMS*)

to wszystkie zadania zostaną wykonane w terminie

- **EDF – Twierdzenie** (*Spuri, Butazzo, Sensini, 1995*)

Jeżeli: $\sum_{i=1}^{n} \frac{t_i}{c_i} \leq 1$

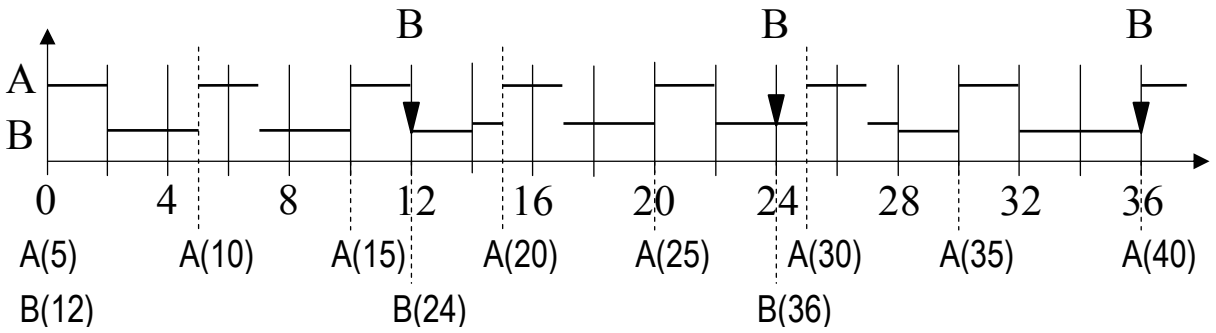
oraz szeregowanie jest zgodne z terminami (*EDF*)

to wszystkie zadania zostaną wykonane w terminie

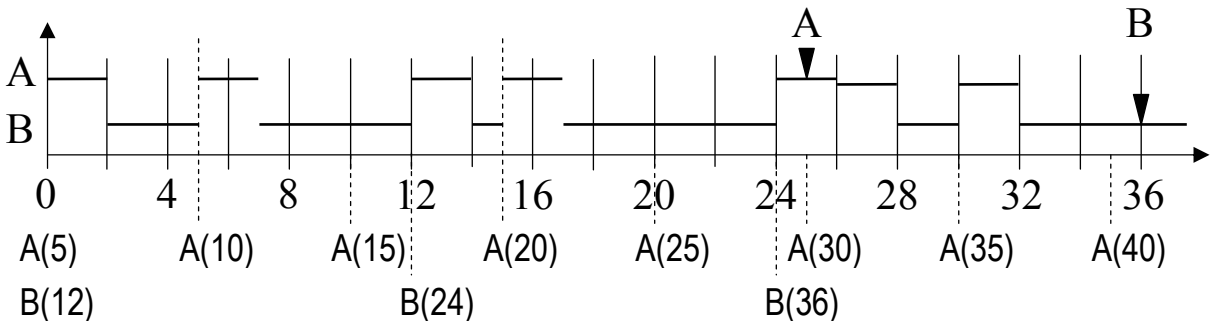
Przewidywalność

Zadanie	Długość	Cykl	Obciążenie
A	2	5	2/5 = 40%
B	8	12	8/12 = 66%
			= 106%

- ***Rate monotonic scheduling (RMS)***



- ***Earliest deadline first (EDF)*** — zmienne priorytety

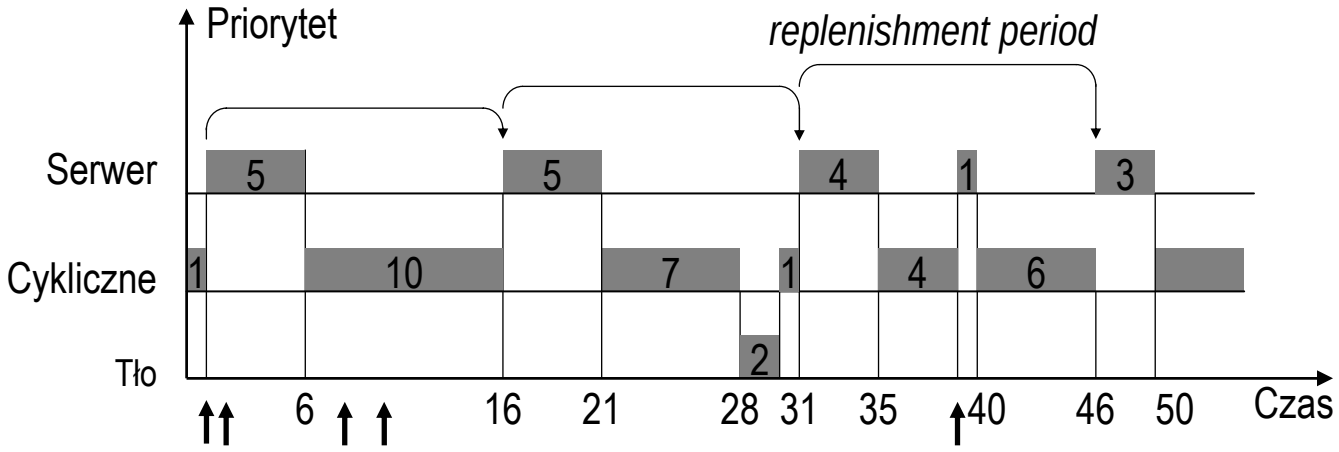


Zadania sporadyczne

- Zadanie obsługujące
- Algorytm *Sporadic Server* (*Sprunt*, *Sha*, *Lehoczky*, *1989*)

Przykład

- zadania cykliczne: czas wykonania 18, okres 30
- serwer sporadyczny: zapas czasu 5, okres odnawiania 15
- zadania sporadyczne: czas obsługi 4, zgłoszenia 1, 2, 8, 10, 39



Synchronizacja zadań

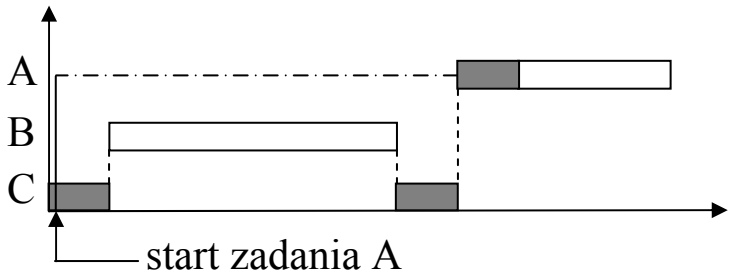
- Dla każdego zadania z_i określone dodatkowo:
  - maksymalny czas zablokowania przez zadania o niższym priorytecie b_i

Twierdzenie (*Sha, Rajkunur, Lehoczky, 1990*)

Jeżeli: $\sum_{i=1}^{n} \frac{t_i}{c_i} + \max_{i=1..n} \left\lceil \frac{b_i}{c_i} \right\rceil \leq n\,(2^{\frac{1}{n}} - 1)$

oraz szeregowanie jest zgodne z pilnością (*RMS*)

to wszystkie zadania zostaną wykonane w terminie



Podejście praktyczne

1. Zadania o ostrych ograniczeniach czasowych (krytyczne)
2. Zadania o łagodnych ograniczeniach czasowych
3. Zadania sporadyczne
4. Szeregowalność zadań krytycznych
5. Szeregowalność wszystkich zadań

- Chwilowe przeciążenia systemu (*transient overload*)