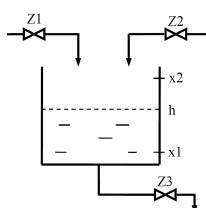


Materiały do przedmiotu "Systemy wbudowane"

Opracował: Zbigniew Świder

Układy sekwencyjno-czasowe, cz. II

4. Zbiorniki z uzależnieniem czasowym (c.d)

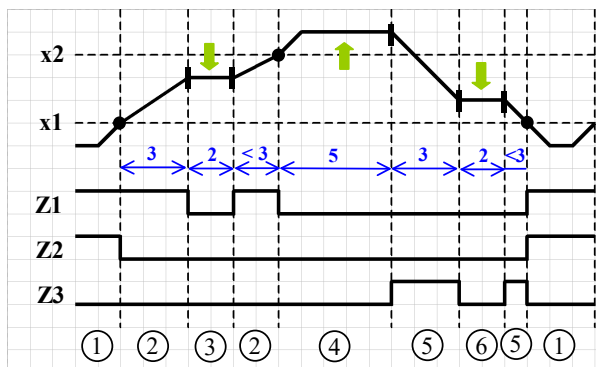


Zadanie 4. Zaprojektuj układ sterowania według algorytmu:

- $h < x1$ - nalewamy z obu Z1+Z2 do x1
- $h \geq x1$ - dalej nalewamy tylko z Z1 do x2 w cyklu 3+2 s
- $h \geq x2$ - zamykamy, czekamy 5 s (pauza)
- wylewamy z Z3 do x1 w cyklu 3+2s.

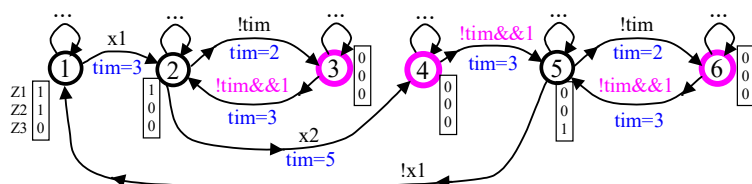
Rozwiązanie.

Warunek "nalewamy tylko z Z1 do x2 w cyklu 3+2 s" oznacza, że nalewamy z Z1 przez 3 s, następnie przerwa 2 s, potem znowu z Z1 przez 3 s, przerwa 2 s - i tak dalej aż do momentu, gdy poziom przekroczy x2. Wtedy przerywamy nalewanie i przechodzimy do następnego etapu. Zwróćmy uwagę, że w trakcie tej 2 sekundowej pauzy poziom wody się nie zmienia (wszystkie zawory są zamknięte).



Również w czasie 5 s oczekiwania (już po napełnieniu zbiornika) poziom wody także się nie zmienia (pozioma linia na przebiegach), podobnie podczas 2 s pauzy przy wylewaniu.

Graf i oznaczenie stanów podano poniżej. Zwróćmy uwagę na warunki przejścia ze stanów opisanych jako "dokładnie", jeśli z tego stanu wychodzi **tylko jedna strzałka** (bo wszystkie zawory Z1, Z2, Z3 są zamknięte, a więc poziom wody nie może się zmieniać).



- 1 - nalewanie z obu (Z1 + Z2)
- 2 - nalewanie z Z1 maksymalnie 3 s
- 3 - pauza w nalewaniu dokładnie 2 s
- 4 - oczekiwanie (pauza) dokładnie 5 s
- 5 - wylewanie z Z3 maksymalnie 3 s
- 6 - pauza w wylewaniu dokładnie 2 s

Jeśli więc, ze stanu opisanego jako "dokładnie" wychodzi tylko jedna strzałka, to warunkiem przejścia do następnego będzie $!tim \& \& 1$ (iloczyn !tim ze stałą 1). W tym przypadku, warunkiem przejścia musi być iloczyn !tim ze stałą - nie wolno pisać iloczynu ze zmienną (to byłoby błędem).

Program w języku C:

```
char x1, x2, Z1, Z2, Z3, stan=1, tim; // definicje, cykl 0.1 s
// -----
x1=aK1; x2=aK2; // wczytaj wartości wejść
switch(stan) {
    case 1: Z1=1; Z2=1; Z3=0;
            if (x1) { tim=30; stan=2; } // timer: 3.0/0.1=30
            break;

    case 2: Z1=1; Z2=0; Z3=0;
            if (!tim) { tim=20; stan=3; } // skończył się czas -> 3
            else // albo
            if (x2) { tim=50; stan=4; } // napełniono zbiornik -> 4
            break;

    case 3: Z1=0; Z2=0; Z3=0;
            if (!tim&&1) { tim=30; stan=2; } // koniec paury -> 2
            break;

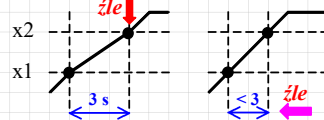
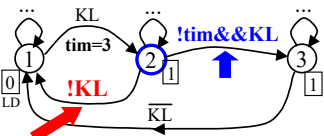
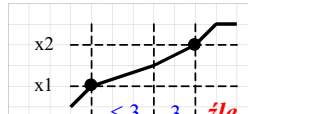
    case 4: Z1=0; Z2=0; Z3=0;
            if (!tim&&1) { tim=30; stan=5; } // oczekiwanie dokładnie 5 s
            break;

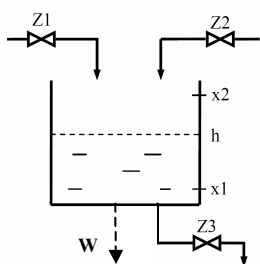
    case 5: Z1=0; Z2=0; Z3=1;
            if (!tim) { tim=20; stan=6; } // skończył się czas -> 6
            else // albo
            if (!x1) stan=1; // opróżniono zbiornik -> 1
            break;

    case 6: Z1=0; Z2=0; Z3=0;
            if (!tim&&1) { tim=30; stan=5; } // koniec paury -> 5
            break;
} // ---- koniec "switch"

if (tim) --tim; // dekrementuj timer co cykl
L1=Z1; L2=Z2; L3=Z3; // zaświeć diody
```

Najczęściej popełniane błędy to:

<p>Zbiornik nie może się nalać <u>dokładnie</u> w 3 sekundy !!!</p> <p>Ostatni cykl nalewania (czy też wylewania) trwa zawsze mniej niż zadany czas (warunkiem jest np. "< 3" - natomiast nie może być równy 3). Również zbiornik nie może się nalać (czy wylać) w jednym cyklu - musi co najmniej raz przełączyć sterowanie (wyjścia).</p>	 <p><i>złe - zbiornik nie może się nalać/wylać w jednym cyklu (np. nalewanie w cyklu 3+3)</i></p>
<p>Jeszcze raz przypomnijmy, że napisanie na jednej strzałce iloczynu (&&) z timerem (np. !tim&&KL), a na drugiej strzałce warunku bez iloczynu z timerem (np. samo !KL) jest ewidentnym błędem w rozwiązaniu, więc albo na obu strzałkach jest iloczyn z timerem ("<u>dokładnie</u>"), albo nie ma go na żadnej ("<u>maksymalnie</u>").</p>	
<p>Nie można na przebiegach napisać warunku "<= 3" - musi być albo mniejsze, albo równe (w zależności jak jest narysowane i przez ile kratek) - nie można pisać "<i>mniejsze lub równe</i>". Podobnie, napisanie 3 zamiast < 3 (gdy etap jest narysowany przez 2 kratki) jest błędem.</p>	



Zadanie 5. Zaprojektuj układ sterowania według algorytmu:

$h < x1$ - nalewamy z obu Z1+Z2 do x1

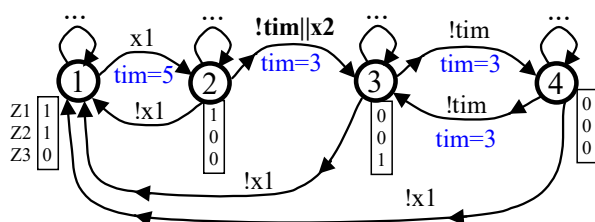
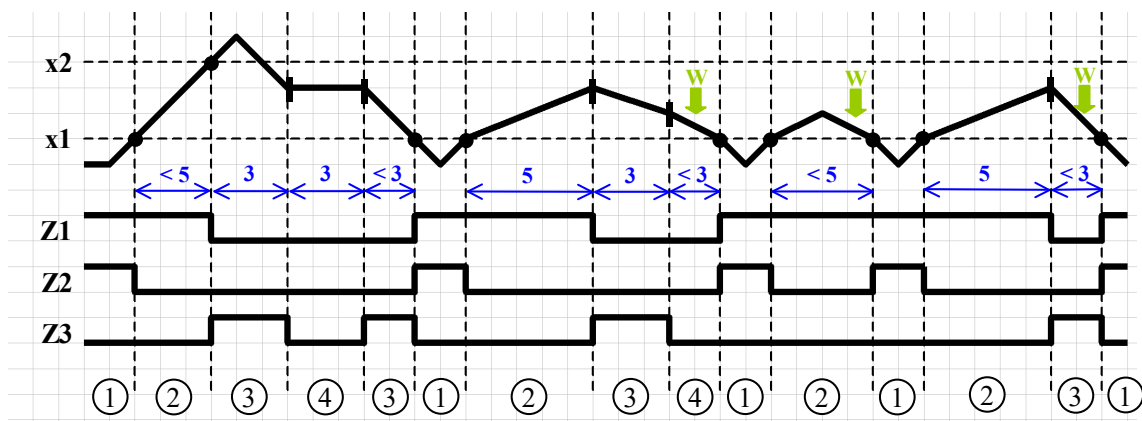
$h \geq x1$ - dalej nalewamy tylko z Z1 do x2, ale nie dłużej niż 5s

- zamykamy Z1, wylewamy z Z3 do x1 w cyklu 3+3s.

- dodatkowo, w każdej chwili może być ręcznie otwarty zawór W (zakłócenie - woda może się szybciej wylewać niż nalewać)

Rozwiązanie.

Przebiegi czasowe (z podziałem na stany) przedstawiono poniżej.



- 1 - nalewanie z obu (Z1 + Z2)
- 2 - nalewanie z Z1 maksymalnie 5 s
- 3 - wylewanie maksymalnie 3 s
- 4 - pauza maksymalnie 3 s

```

char x1, x2, Z1, Z2, Z3, stan=1, tim; // definicje, cykl 0.1 s
// -----
x1=aK1; x2=aK2; // wczytaj wartości wejść
switch(stan) {
  case 1: Z1=1; Z2=1; Z3=0;
    if (x1) { tim=50; stan=2; }
    break;

  case 2: Z1=1; Z2=0; Z3=0;
    if (!tim|x2) { tim=20; stan=3; } else if (!x1) stan=1;
    break;

  case 3: Z1=0; Z2=0; Z3=1;
    if (!tim) { tim=30; stan=4; } else if (!x1) stan=1;
    break;

  case 4: Z1=0; Z2=0; Z3=0;
    if (!tim) { tim=30; stan=3; } else if (!x1) stan=1;
    break;
}

if (tim) --tim; // dekrementuj timer co cykl
L1=Z1; L2=Z2; L3=Z3; // zaświeć diody
  
```