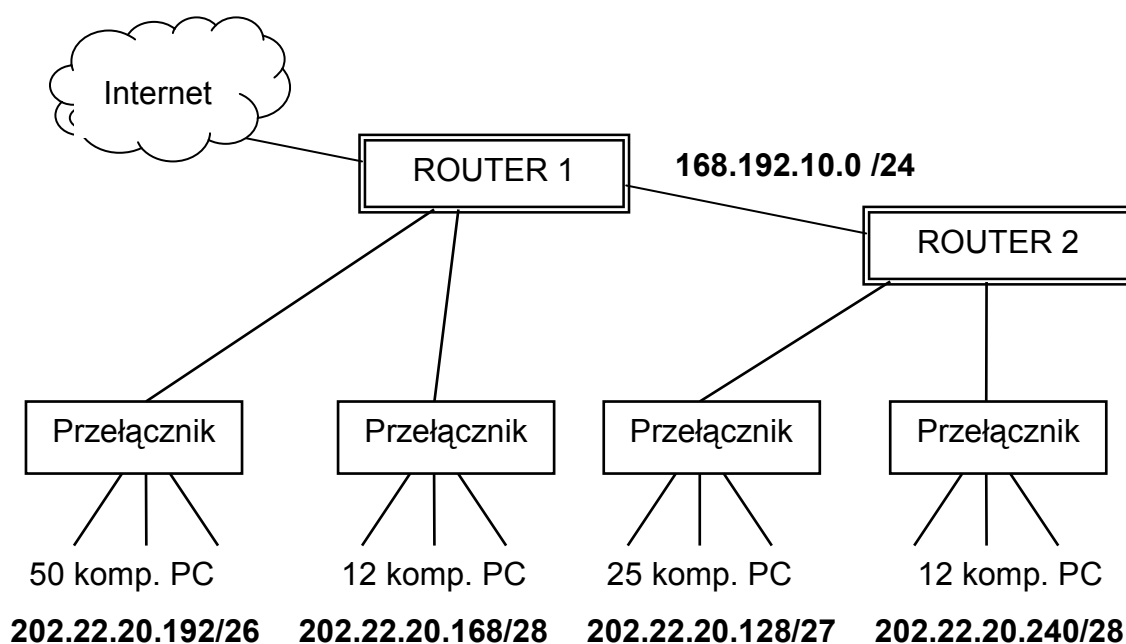


**„EUROELEKTRA”**  
**Ogólnopolska Olimpiada Wiedzy Elektrycznej i Elektronicznej**  
**Rok szkolny 2011/2012**  
**Odpowiedzi do zadań dla grupy teleinformatycznej na zawody III stopnia**

**Zad. 1**

Pewna firma postanowiła zbudować własną sieć LAN. Administrator otrzymał niewielką pulę publicznych adresów IP (202.22.20.128/25) i podzielił je pomiędzy podsieci w sposób przedstawiony na rysunku (postanowił użyć prywatnych adresów IP do połączeń pomiędzy routerami, aby zaoszczędzić publiczne adresy IP), a następnie włączył w routerach protokół routingu (zdecydował się na RIP w wersji 1). Niestety, okazało się, że sieć nie działa prawidłowo.

Wskaż błędy w działaniach administratora i zaproponuj sposób ich naprawienia.



**Rozwiązanie zadania nr 1**

Błędy:

- (1) Podsieć 168.192.10.0/24 nie należy do puli adresów prywatnych.
- (2) Podsieć 202.22.20.240/28 zawiera się w podsieci 202.22.20.192/26.
- (3) Podsieć 202.22.20.168/28 ma nieprawidłowy adres (powinien być wielokrotnością 16).
- (4) RIP wersja 1 nie obsługuje VLSM (masek o zmiennej długości).

Sposób naprawienia (najprostszy wariant): Użyj

- (1) 192.168.10.0/24 (zamiast 168.192.10.0/24),
- (2) 202.22.20.176/28 (zamiast 202.22.20.240/28),
- (3) 202.22.20.160/28 (zamiast 202.22.20.168/28),
- (4) RIP wersja 2 albo EIGRP albo OSPF (zamiast RIP wersja 1).

**Zad. 2**

Założmy, że odległość pomiędzy dwoma serwerami wynosi 500 km, a szybkość propagacji sygnału w kanale komunikacyjnym łączącym je ze sobą  $2 \cdot 10^8$  m/s. W warstwie łącza danych na

tych serwerach jest wykorzystywany kod detekcyjny, natomiast w warstwie transportowej, w celu korygowania przekłamań w ramach, w których wystąpiły błędy, algorytm zwany algorytmem stój-i-czekaj z okienkiem jednoramkowym i ustawionej maksymalnej liczbie powtórzeń przekłamanej ramki dwa. Niech szybkość transmisji w powyższym połączeniu wynosi 1 Mb/s, a tolerowane maksymalne opóźnienie odbioru ramki 30 ms. Oblicz, jaka może być wtedy maksymalna długość ramki (w bitach), przy której nie jest przekraczana podana powyżej maksymalna wartość opóźnienia. (Zaniedbać czasy przetwarzania na serwerach i długość ramki potwierdzenia. Ponadto, przyjąć, że ramka potwierdzenia zawsze dociera do nadawcy: albo jako ramka potwierdzenia pozytywnego albo negatywnego - ta ostatnia, gdy wystąpiło przekłamanie.)

### **Rozwiązanie zadania nr 2**

Oznaczmy sumę czasów propagacji sygnału ramki z danymi od nadawcy do odbiorcy oraz sygnału ramki potwierdzenia odbioru (ACK lub NACK, ang. acknowledgement or no acknowledgement) w kierunku odwrotnym (od odbiorcy do nadawcy) jako  $D_{prop}$ . Czas ten można obliczyć, korzystając z faktu, że sygnał w rozpatrywanym kanale komunikacyjnym rozchodzi się ze stałą prędkością. Wtedy obowiązuje następujący ogólny wzór

$$v = \frac{d}{t},$$

gdzie  $v$  oznacza prędkość,  $d$  przebytą drogą, a  $t$  jest czasem, w którym ta droga została pokonana. W naszym przypadku mamy:  $v = 2 \cdot 10^8$  m/s,  $d = 2 \cdot d_{na} = 2 \cdot 500$  km i  $t = D_{prop}$ ; ponadto, korzystając z powyższego wzoru, można wyznaczyć  $D_{prop}$ . Opóźnienie propagacji

$D_{prop}$  będzie równe  $\frac{2 \cdot d_{na}}{v}$ . Natomiast opóźnienie (związane z odbiorem całej ramki i dotarciem potwierdzenia jej odbioru do nadawcy, ACK lub NACK) obliczymy ze wzoru  $r = \frac{L}{t}$ , gdzie  $r$  oznacza szybkość transmisji (tutaj  $r = 1$  Mb/s), a  $L$  liczbę bitów przesłanych w czasie  $t$ . Oznaczmy powyżej zdefiniowane opóźnienie jako  $D_{ram}$ ; zatem, przy wykorzystaniu wzoru na  $r$ , możemy napisać

$$D_{ram} = \frac{L_1}{r} + \frac{L_2}{r} \cong \frac{L_1}{r},$$

gdzie  $L_1$  i  $L_2$  oznaczają odpowiednio długości (w bitach) ramek: wysłanej od nadawcy do odbiorcy i ACK lub NACK. Ponadto, przyjęto w treści zadania  $L_1 \gg L_2$ .

Korzystając z podanych wzorów na  $D_{prop}$  i  $D_{ram}$ , możemy teraz opóźnienie całkowite jednego obiegu w algorytmie stój-i-czekaj,  $D_c$ , będącego sumą  $D_{prop}$  i  $D_{ram}$ , wyrazić takim wzorem

$D_c = D_{prop} + D_{ram} = \frac{2 \cdot d_{na}}{v} + \frac{L_1}{r}$ . Z treści zadania (okienko jednoramkowe) wynika, że nadawca wysyła następną ramkę dopiero po odbiorze ACK lub NACK. Jeżeli jest odbierane NACK, to oznacza, że nastąpiło przekłamanie i nadawca wysyła ostatnio wysłaną przez siebie ramkę ponownie. Ale przyjęto w zadaniu, że ta operacja ponownego wysłania ramki może się odbyć maksymalnie tylko dwukrotnie. Jest najgorszy przypadek, dla którego musi być również spełniona nierówność

$$3 \cdot D_c = 3 \left( \frac{2 \cdot d_{na}}{v} + \frac{L_1}{r} \right) \leq 30 \text{ ms}.$$

Przekształcając powyższą nierówność, otrzymamy

$$L_1 \leq r \left( 10 \text{ ms} - \frac{2 \cdot d_{na}}{v} \right).$$

Podstawiając następnie wartości liczbowe podane w treści zadania, otrzymuje się ostatecznie  $L_1 \leq 5 \text{ kb}$ .

### Zad. 3

Tor transmisyjny składa się z kaskadowego połączenia kabli, pomiędzy które są włączone wzmacniaki, tak że tłumienia i wzmocnienia w tym torze, liczone od strony jego wejścia do wyjścia, wynoszą kolejno: tłumienie 5 dB, wzmocnienie 20 dB, tłumienie 18,5 dB, wzmocnienie 20 dB, tłumienie 16,5 dB, wzmocnienie 25 dB i tłumienie 12,2 dB. Oblicz, jaki jest poziom mocy sygnału na wejściu tego toru transmisyjnego, gdy na jego wyjściu wynosi on -6 dBm (minus 6 dBm). Następnie załóż, że ten tor ma długość 100 km, i że są w nim transmitowane dane z szybkością 1,544 Mb/s. Oblicz, jaka największa liczba bitów (jako elementów transmitowanego sygnału) może się znaleźć jednocześnie w tym torze. W obliczeniach przyjmij szybkość, z którą rozchodzi się sygnał w kablu, równą  $2 \cdot 10^8 \text{ m/s}$  i zerowe opóźnienia sygnału we wzmacniakach.

### Rozwiązanie zadania nr 3

Przy podanych danych, całkowite wzmocnienie mocy sygnału w torze będzie wynosić:  $20 + 20 + 25 = 65 \text{ dB}$ ; podobnie, całkowite tłumienie w torze będzie równe:  $-5 + (-18,5) + (-16,5) + (-12,2) = -52,2 \text{ dB}$ . Zatem całkowite wzmocnienie lub tłumienie (w zależności od wypadkowego znaku) w rozpatrywanym torze będzie wynosić:  $65 \text{ dB} + (-52,2) \text{ dB} = 12,8 \text{ dB}$  (wzmocnienie). W następnym kroku korzystamy z definicji wzmocnienia lub tłumienia mocy sygnału wyrażonego w mierze decybelowej oraz definicji tychże odniesionych do mocy 1 mW (to znaczy wyrażanych w jednostkach zwanych dBm). Pozwala to nam napisać

$$\begin{aligned} 12,8 \text{ dB} &= 10 \log_{10} \frac{P_{\text{wyj}}}{P_{\text{wej}}} = 10 \log_{10} \frac{\frac{P_{\text{wyj}}}{1 \text{ mW}}}{\frac{P_{\text{wej}}}{1 \text{ mW}}} = 10 \log_{10} \frac{P_{\text{wyj}}}{1 \text{ mW}} - 10 \log_{10} \frac{P_{\text{wej}}}{1 \text{ mW}} = \\ &= -6 \text{ dBm} - 10 \log_{10} \frac{P_{\text{wej}}}{1 \text{ mW}}, \end{aligned}$$

gdzie  $P_{\text{wyj}}$  oznacza moc sygnału na wyjściu toru, a  $P_{\text{wej}}$  na jego wejściu.

Po przekształceniu powyższego równania otrzymujemy

$$10 \log_{10} \frac{P_{\text{wej}}}{1 \text{ mW}} = -12,8 \text{ dB} - 6 \text{ dBm} = -18,8 \text{ dBm}.$$

Zatem odpowiedź brzmi tutaj następująco: poziom mocy sygnału na wejściu rozpatrywanego toru transmisyjnego wynosi -18,8 dBm.

Aby rozwiązać drugą część zadania, należy skorzystać ze wzoru określającego prędkość  $v$  w zależności od czasu  $t$  i przebytej drogi  $d$  - obowiązującego w ruchu jednostajnym bez przyspieszenia. Takim ruchem poruszają się elektrony w rozpatrywanym torze transmisyjnym, zatem tak dokonuje się w nim propagacja sygnału. Powyższy wzór ma postać

$v = \frac{d}{t}$ ; przekształcając go mamy  $t = \frac{d}{v}$ . Z drugiej strony wiemy, że wzór określający

szybkość transmisji to  $r = \frac{L}{t}$ , gdzie  $r$  oznacza szybkość transmisji, a  $L$  liczbę bitów

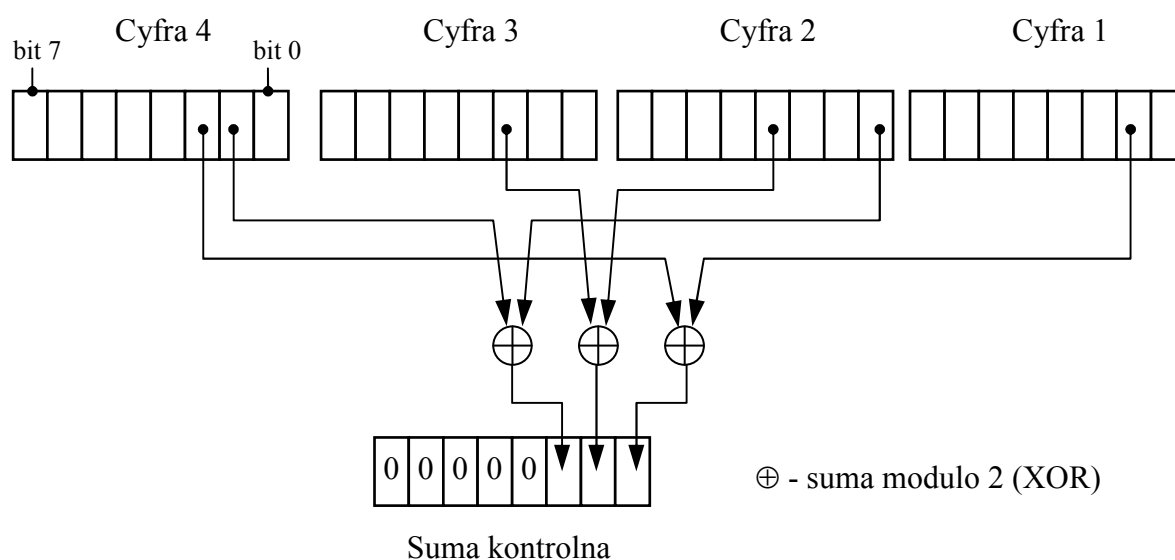
przesłanych w czasie  $t$ . Podstawiając  $t = \frac{d}{v}$  w tym ostatnim wzorze, otrzymuje się  $r = \frac{Lv}{d}$ ,

skąd po przekształceniu  $L = \frac{rd}{v}$ . Podstawienie wartości liczbowych daje  $L = \frac{1,544 \text{ Mb/s} \cdot 100 \text{ km}}{2 \cdot 10^8 \text{ m/s}} = 772 \text{ bitów}$ . Zatem w rozpatrywanym w tym zadaniu torze transmisyjnym może się znaleźć jednocześnie co najwyżej 772 bitów.

#### Zad. 4

Napisz podprogram w asemblerze mikrokontrolera AVR, obliczający sumę kontrolną czterocyfrowego identyfikatora, zgodnie z ilustracją na zamieszczonym poniżej rysunku. Przed wywołaniem podprogramu kolejne cyfry identyfikatora są zapisane w rejestrach od R21 do R24 (R21 – pierwsza cyfra, R22 – druga itd.) i wiadomo, że każda z nich ma wartość z przedziału 1 .. 9. Przed instrukcją powrotu z podprogramu wynik należy umieścić w rejestrze R20. Rejestrów R0-R19 nie wolno modyfikować, natomiast rejestry R25-R30 można używać do zapisania tymczasowych zmiennych pomocniczych.

Należy zadbać, aby podprogram działał możliwie szybko. Dla uproszczenia przyjmujemy, że wszystkie instrukcje są tak samo czasochłonne, zatem rozwiązanie jest tym lepsze, im zawiera mniej instrukcji.



Uwagi. (1) W rozwiązaniu NIE należy korzystać z instrukcji operujących na pojedynczych bitach (np. BST, BLD). (2) W rozwiązaniu należy podać wyłącznie kod podprogramu, bez elementów programu takich jak inicjalizacja wektorów przerwań czy stosu. (3) Przykładowy sposób wywołania podprogramu wygląda tak:

```
LDI R21, 1;
LDI R22, 9;
LDI R23, 7;
LDI R24, 4;
CALL sumaKontrolna;
// wynik, w tym przypadku 5, powinien być teraz w R20
```

#### Wybrane polecenia asemblera AVR:

Nazwa	Działanie	Przykład
ADD Rd, Rr	dodawanie bez przeniesienia, $Rd \leftarrow Rd + Rr$	ADD R21, R22
ADC Rd, Rr	dodawanie z przeniesieniem, $Rd \leftarrow Rd + Rr + C$	

SUB Rd, Rr	odejmowanie bez przeniesienia, $Rd \leftarrow Rd + Rr$	SUB R21, R22
SBC Rd, Rr	odejmowanie z przeniesieniem, $Rd \leftarrow Rd + Rr - C$	
INC Rd	inkrementacja, $Rd \leftarrow Rd + 1$	INC R20
DEC Rd	dekrementacja, $Rd \leftarrow Rd - 1$	
LSL Rd	przesunięcie w lewo, w praktyce $Rd \leftarrow Rd \cdot 2$	LSL R15
LSR Rd	przesunięcie w prawo, w praktyce $Rd \leftarrow Rd / 2$	
AND Rd, Rr	logiczne AND („i”), $Rd \leftarrow Rd \wedge Rr$	AND R28, R29
ANDI Rd, K	logiczne AND rejestru i stałej $Rd \leftarrow Rd \wedge K$	ANDI R21, 15
OR Rd, Rr	logiczna OR („lub”) $Rd \leftarrow Rd \vee Rr$	OR R28, R29
ORI Rd, K	logiczne OR rejestru i stałej, $Rd \leftarrow Rd \vee K$	ORI R21, 15
EOR Rd, Rr	logiczne XOR („modulo 2”), $Rd \leftarrow Rd \oplus Rr$	EOR R28, R29
CP Rd, Rr	porównanie wartości rejestrów, Rd-Rr	CP R21, R25
CPI Rd, K	porównanie wartości rejestru i stałej, Rd-K	CPI R22, 1
BREQ k	skok, jeżeli równe	BREQ gotowe
BRNE k	skok, jeżeli nie równe	
BRSH k	skok, jeżeli większe lub równe	
BRLO k	skok, jeżeli mniejsze	
JMP k	skok bezwarunkowy	JMP ponownie
RET	powrót z podprogramu	RET
MOV Rd, Rr	kopiowanie rejestru, $Rd \leftarrow Rr$	MOV R21, R22
LDI Rd, K	wpisanie wartości, $Rd \leftarrow K$	LDI R22, 1

Rd i Rr oznacza rejestr (od R0 do R31), K liczbę, natomiast k jest etykieta.

#### Rozwiązanie zadania nr 4

W rozwiązaniu należy zwrócić uwagę na kilka kwestii: (1) wyniki częściowe, uzyskane przez sumowanie modulo 2, powinny być poddane maskowaniu (przez operację AND), inaczej w wyniku mogą znaleźć się nadprogramowe bity; (2) operacja maskowania powinna być wykonana po sumie modulo 2, a nie przed (choć jest to również poprawne) – ponieważ wtedy wystarczy ją wykonać raz, zamiast dwa razy; (3) rejestry R22 (cyfra 2) i R24 (cyfra 4) trzeba przesuwąć po dwa razy – pierwsze przesunięcie zmienia położenie bitów, co trzeba uwzględnić przy drugim przesunięciu; alternatywnie można je skopiować do rejestrów R25-R30; użycie kopii zmniejsza liczbę potrzebnych przesunięć rejestrów, co zaoszczędza czas; (4) nie ma wymogu, aby dane wejściowe zachować niezmienione – wykonując obliczenia bezpośrednio na nich można zaoszczędzić czas.

Przykładowe rozwiązanie wygląda tak:

```
sumaKontrolna:
MOV R25, R22 // kopia cyfry 2 -> R25
MOV R26, R24 // kopia cyfry 4 -> R26

// bit 2
LSL R24 // cyfra 4, bit 1 -> bit 2
LSL R22 // cyfra 2, bit 0 -> bit 2
LSL R22
EOR R22, R24 // modulo 2
ANDI R22, 4 // istotny tylko bit 2, wyzerowanie pozostałych

MOV R20, R22 // bit 2 -> R20

// bit 1
LSR R23 // cyfra 3, bit 2 -> bit 1
```

```

LSR R25          // cyfra 2, bit 3 -> bit 1
LSR R25
EOR R23, R25     // modulo 2
ANDI R23, 2      // istotny tylko bit 1, wyzerowanie pozostałych

OR R20, R23      // bit 1 -> R20 (poprawne jest też ADD)

// bit 0
LSR R26          // cyfra 4, bit 2 -> bit 0
LSR R26
LSR R21          // cyfra 1, bit 1 -> bit 0
EOR R21, R26     // modulo 2
ANDI R21, 1      // istotny tylko bit 0, wyzerowanie pozostałych

OR R20, R21      // bit 0 -> R20

RET             // gotowe

```

### Zad. 5

W standardzie telewizyjnym HDTV 1080i ramki danych są transmitowane z szybkością 30 ramek/s; na ramkę danych w tym standardzie składa się 1080 pikseli wyświetlanych w pionie i 1920 pikseli wyświetlanych w poziomie. Jeżeli każdy z podstawowych kolorów (czerwony, niebieski i zielony) realizowanych w każdym pikselu jest zakodowany na 8 bitach, to jaka jest szybkość przesyłania tych nieskompresowanych danych video w Mb/s? Jaki stopień kompresji powyższych danych byłby wymagany, jeżeli miałyby być one przesyłane (bez buforowania), w sieci LAN w której transmitacja odbywa się z maksymalną szybkością 5 Mb/s? Czy obliczony stopień kompresji jest możliwy do uzyskania w praktyce?

#### Rozwiązanie zadania nr 5

Zacznijmy rozwiązywanie zadania od obliczenia liczby pikseli ( $L_R$ ) w ramce. Będzie ona wynosiła  $L_R = 1920 \cdot 1080 = 2073600$  pikseli. Następnie obliczmy liczbę bitów zawartych w ramce; będzie ona równa ( $L_R$ ) razy (3 kolory) razy (8 bitów na kolor) = 49766400 bitów = 49,7664 Mb. Szybkość transmisji ramek wynosi  $r = 30$  ramek/s, zatem korzystając z powyżej wyliczonej liczby bitów w ramce, otrzymujemy  $r = 1492,992$  Mb/s (jest to poszukiwana szybkość przesyłania nieskompresowanych danych video w Mb/s). Współczynnik  $k$  kompresji danych określa się takim wzorem

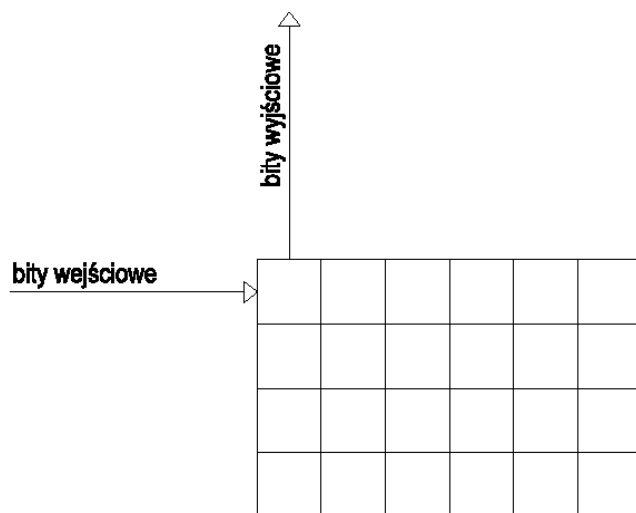
$$k = \frac{r_{we} - r_{wy}}{r_{we}},$$

gdzie  $r_{we}$  oznacza szybkość strumienia danych na wejściu układu (programu) dokonującego kompresji, a  $r_{wy}$  szybkość strumienia danych na jego wyjściu. W naszym przypadku mamy  $r_{we} = r = 1492,992$  Mb/s i  $r_{wy} = 5$  Mb/s. Podstawiając te dane do podanego powyżej wzoru, otrzymujemy  $k = 0,9965$  (99,65 %). Tak duży współczynnik kompresji, bliski prawie 100 %, nie jest możliwy do uzyskania w praktyce.

### Zad. 6

W bezprzewodowych kanałach z zanikami w celu zmniejszenia efektu grupowania się błędów binarnych w serie, stosuje się m.in. rozpraszanie błędów poprzez stosowanie przeplotu bitowego. Zapewnia to skuteczniejszą korekcję błędów w dekodерze kodu splotowego. Przeplot bitowy polega na nadawaniu bitów w kanał transmisyjny w innej kolejności niż są one generowane przez źródło bitów i jest realizowany przy użyciu tablicy przeplotu napełnianej np. wierszami a opróżnianej kolumnami. Na poniższym rysunku przedstawiono przykładową tablicę przeplotu. Wyznaczyć ciąg wyjściowy  $\{y\}$  tablicy, jeżeli ciąg wejściowy

$\{x\}$  ma postać  $\{x\}=(110100001001011101110100)$ . Bity wejściowe są podawane do tablicy począwszy od lewej strony ciągu wejściowego.



Tablica przeplotu bitowego

***Rozwiązanie zadania nr 6***

Algorytm realizacji przeplotu bitowego de facto został podany w treści zadania. Po jego zastosowaniu do sekwencji wejściowej otrzymujemy  $\{y\}=(100110110110101100000110)$ .