9	Stanowisko 9 (szyna danych)

9.5.4)Przygotować taki zestaw instrukcji (maksymalnie 8), który będzie wczytywał liczbę 4 - bitową $A(a_3a_2a_1a_0)$ podawaną przez prowadzącego zajęcia, a następnie będzie wpisywał do rejestru wyjściowego **RWY** liczby będące o 2 większe niż poprzednia: A, A+2, A+4, A+6, A+8, etc. Program ma działać w nieskończonej pętli. Sprawdzić działanie w dwóch trybach pracy: CYKL i PROGRAM.

Do wpisanego numeru trzeba dodawać 2. Jak widać po tabeli obliczeń, jedyne możliwe dodawanie do R1 + R2, więc trzeba ustawić R2 = 2. Utworzenie liczby równej 2 hardcoded w RAM-ie jest niemozliwe, więc należy ją otrzymać operacjami podanymi w tabeli. Jednym z nich jest obrócenie 0 na 15 operacją negacji, dwukrotne odjęcie 1, obrócenie drugiego zera na 15 i wykonanie odejmowania R1 – R2 = 15 – 13.

Lp.	S_3	S_2	S_1	S_0	Operacje arytmetyczne M = 0	Operacje logiczne M = 1
1.	0	0	0	0	$Y = 0$	$Y = \overline{R1}$
2.	0	0	0	1	$Y = R1 + R2$	$Y = \overline{R1 \wedge R2}$
3.	0	0	1	0	$Y = R1 \times R2 - 1$	$Y = \overline{R1} \vee R2$
4.	0	0	1	1	$Y = R1 + R2$	$Y = 1$
5.	0	1	0	0	$Y = R1^2 - R2^2$	$Y = R1 \oplus R2$
6.	0	1	0	1	$Y = R1 \times 2$	$Y = \overline{R1 \Leftrightarrow R2}$
7.	0	1	1	0	$Y = \frac{R1 + R2}{2}$	$Y = R1 \Rightarrow R2$
8.	0	1	1	1	$Y = R1 \times R2$	$Y = R1 \vee R2$
9.	1	0	0	0	$Y = R1 \,\%\, R2$	$Y = \overline{R1 \Rightarrow R2}$
10.	1	0	0	1	$Y = R1 + 2$	$Y = \overline{R1 \wedge R2}$
11.	1	0	1	0	$Y = (R1 \times R2) - (R1 + R2)$	$Y = R1 \wedge R2$
12.	1	0	1	1	$Y = R1 - R2$	$Y = \overline{R1 \oplus R2}$
13.	1	1	0	0	$Y = (R1 - R2)^2$	$Y = \overline{R2}$
14.	1	1	0	1	$Y = R1$	$Y = \overline{R1 \vee R2}$
15.	1	1	1	0	$Y = R1^2$	$Y = R1 \Leftrightarrow R2$
16.	1	1	1	1	$Y = R1 - 1$	$Y = \overline{R1 \oplus R2}$

Należy rozpocząć od cyklu: 10100110
00111000
00000000
10001100. W tym cyklu w pierwszej linijce z inputu (101) otrzymana zostanie wartość A (001). Następnie A zostanie wypisane na output (110). Z kolei dwa ostatnie znaki w czterech linijkach wykonują negację R1 (M=1 s3s2s1s0=0000). Na koniec cyklu negacja R1 zostanie przypisana do C.
Drugi cykl to: 00000001
01110011
01101011
10001100. W tym cyklu do B zostanie przypisane C (czyli 15). Następnie wykonana zostanie operacja R1 = R1 -1 i nowe R1 (14) zostanie przypisane do C.
Trzeci cykl to 00000001
01110011
00000011
10001100, czyli kolejne odjęcie jedynki od R1 i zarazem C równego teraz 13.
Czwarty cykl to 00000001
01110101
01010011
10001100. W tym cyklu C, czyli 13 zostanie przypisane do R2, a B, czyli 15 – do R1. Zostanie również wykonana operacja R1 – R2, czyli 15 – 13, a jej wynik zostanie przypisany do C. C = 2, więc udało się otrzymać liczbę, która będzie dodawana.
Piąty cykl to 00000000

01110101
00110011
10000100. Jest to pierwsze dodawanie. Wartość C, czyli 2 zostanie przypisana do R2, a wartość A (input) – do R1. Po wykonaniu R1 + R2, A zmieni wartość na wynik.
Szósty cykl to dodawanie, które będzie zapętlone. Wygląda on następująco: 10011000
00110001
00001011
10000100. Najpierw wartość A po pierwszym dodawaniu zostanie wypisana na output, następnie powiększone o 2 A zostanie przypisane do R1, a B zaczyna przechowywać RI, dzięki czemu będzie możliwe cofnięcie. Operacja matematyczna w tym cyklu to kolejne R1 – R2.
Siódmy cykl wykonuje zapętlenie. Jego treść to: 10011000
00110001
01000011. Na początku wartość A zostanie wypisana i przypisana na nowo do R1, a później R1 cofa się o 1 przyjmuje wartość B. Nadanie R1 wartości A i wykonanie kolejnego działania R1 + R2 jest tutaj kluczowe, ponieważ w przeciwnym wypadku wartość A również cofałaby się i output byłby cały czas ten sam.

Cała treść pliku RAM.txt (linie z komentarzem # oddzielają cykle, żeby plik był czytelniejszy):
10100110
00111000
00000000
10001100
#
00000001
01110011
01101011
10001100
#
00000001
01110011
00000011
10001100
#
00000001
01110101
01010011
10001100
#
00000000
01110101
00110011
10000100
#
10011000
00110001
00001011
10000100
#
10011000
00110001
01000011