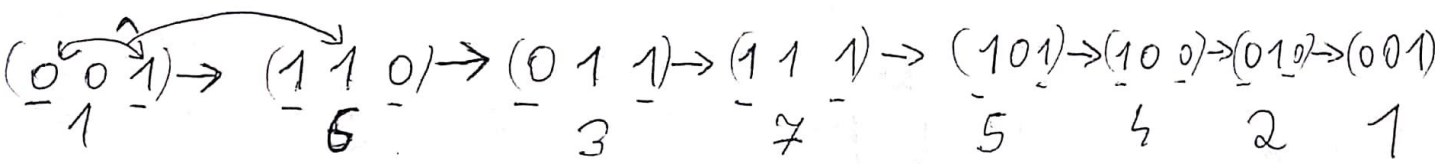
(1) $q_2, q_1, q_0 \rightarrow q_0', (q_0' \wedge q_2'), q_1'$



② $q_3, q_2, q_1, q_0 \rightarrow q_2', q_1', q_0', X$

$X = \sim(XOR(q_3, q_1))$

q_3	q_1	X
0	0	1
0	1	0
1	0	0
1	1	1

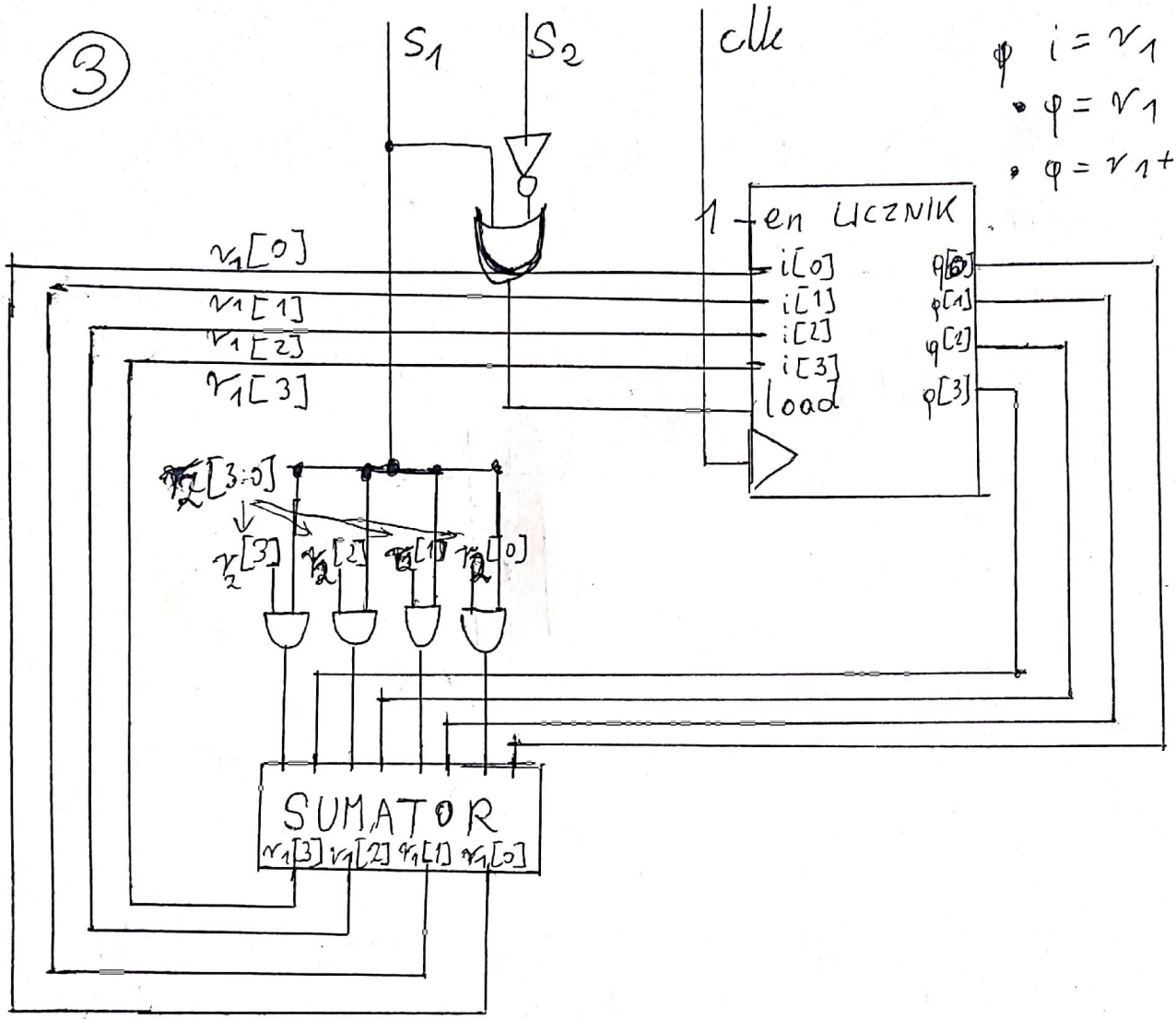
gdy !nrst t_0 (0 0 0 0)

Po zboczu nrst:

$(\underline{0}\,0\,\underline{0}\,0) \rightarrow (\underline{0}\,0\,\underline{0}\,1) \rightarrow (\underline{0}\,0\,\underline{1}\,1) \rightarrow (\underline{0}\,1\,\underline{1}\,0) \rightarrow (\underline{1}\,1\,\underline{0}\,0) \rightarrow (\underline{1}\,0\,\underline{0}\,0) \rightarrow (0\,0\,0\,0)$

0 1 3 6 12 8 0

(3)



- $q = i = r_1$
- $q = r_1$ gdy $s_1 \vee (!s_2)$
- $q = r_1 + 1$ gdy $(!s_1) \wedge s_2$

load = 0 <=> $s_1 = 0, s_2 = 1$

jeśli s_1 to wykonujemy obciążenie w.p.p jeśli s_2 to obciążamy 1, $(r_1 + r_2)$

s_1	s_2	$f(s_1, s_2)$
0	0	nic (poprzedni stan)
0	1	inkrementacja r_1
1	0	$r_1 <= r_1 + r_2$
1	1	$r_1 <= r_1 + r_2$

Sumator zawsze wykonuje obciążenie, ale gdy $s_1 = 1$ obciąża r_2, a gdy $s_1 = 0$ obciąża 0. Czyli nie zmienia stanu.

Gdy $s_1 = s_2 = 0$ ładujemy poprzedni stan do licznika aby nie zaszła inkrementacja. Dla $s_1 = 0, s_2 = 1$ pozwalamy, by licznik liczył.

(4). W pierwszym przypadku mamy przypisanie blokujące więc przypisania wyliczą się z góry na oślep, czyli

$rb = re - 1 = 0 \Rightarrow rb == 0$

i wtedy ($rb == 0$), więc $\underline{e = 1}$

• W drugim przypadku najpierw wyliczana jest prawa strona przypisania nieblokującego, a potem na zboczu sygnału zegara wykonywane są przypisania.
Układ sprawdza „poprzedni" stan, więc skoro $rb == 1$ to zachodzi komenda else i do $\underline{e \text{ zostanie przypisane } 0}$
(a do rb w tym cyklu 0, więc w kolejnym ~~zajdzie~~ zajdzie if)

(5) algorytm Euklidesa dla pary liczb a, b polega na odejmowaniu od większej mniejszej, aż do ich wyrównania. Otrzymane wyniki to NWD(a,b)

1/ Potrzebujemy

Wejście:
- 16-bitowa liczba – inA
- 16-bitowa liczba – inB
- 1-bit wejście zegara – clk
- 1-bit sygnał startu – init

Wyjście:
- 16-bitowa liczba – out – (w najgorszym przypadku liczby z których liczymy NWD są równe)
- 1-bitowe – fin – sygnalizuje koniec obliczeń

2/ 2 Rejestry 16-bitowe rejA, rejB, przechowujące obecny stan liczb a i b, na końcu jeden z nich zawiera wynik (zależny)
Potrzebny będzie też układ obliczający różnicę dwóch liczb, komparator i parę multiplekserów.

3/ Kiedy $init_{t-1} = 1$:

$rejA_t \leftarrow inA$

$rejB_t \leftarrow inB$

Kiedy $init_{t-1} = 0$ i $rejA_{t-1} \neq rejB_{t-1}$ ($fin = 0$):

1° $rejA_{t-1} > rejB_{t-1}$ ($cmp = 1$)

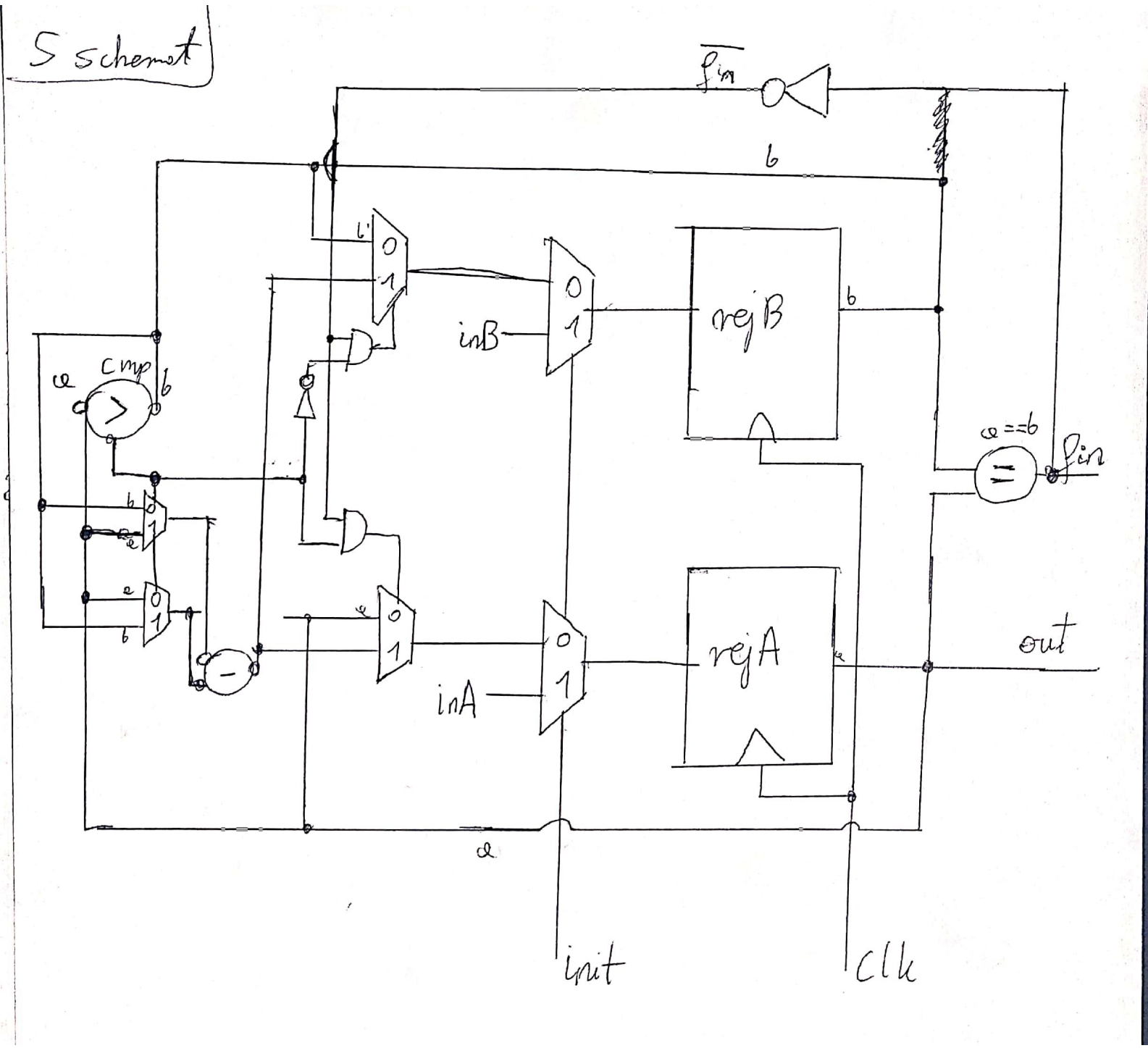
$rejA_t \leftarrow rejA_{t-1} - rejB_{t-1}$

$rejB_t \leftarrow rejB_{t-1}$ (bez zmian)

2° $rejA_{t-1} < rejB_{t-1}$ ($cmp = 0$)

$rejA_t \leftarrow rejA_{t-1}$

$rejB_t \leftarrow rejB_{t-1} - rejA_{t-1}$


5 schemat
fin
b
cmp
a
b
inB
regB
a==b
fin
inA
regA
out
a
init
clk