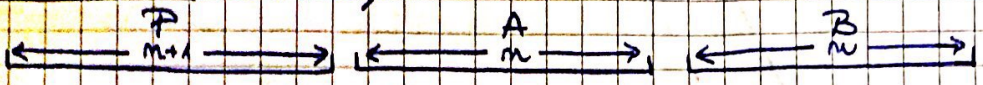
Algoritmul lui SRT: (radix 2)

2^k Divident 2^k Divisor



1. If B has k leading 0's (zeros), shift P.A.B k positions left
2. For $i=0$ to $n-1$
   If top 3 bits of register P are equal then $q_i=0$; Shift P.A left
   If top 3 bits of P are not equal and $P>0$ then:
   $q_i=1$; Substract B from P; Shift P.A left
   If top 3 bits of P are not equal and $P<0$ then:
   $q_i=\bar{1}$; Add P to B; Shift P.A left
3. If $P<0$ Add B to P; Substract 1 from A
4. Shift P right k positions

CURS 5 — 26.03.2019

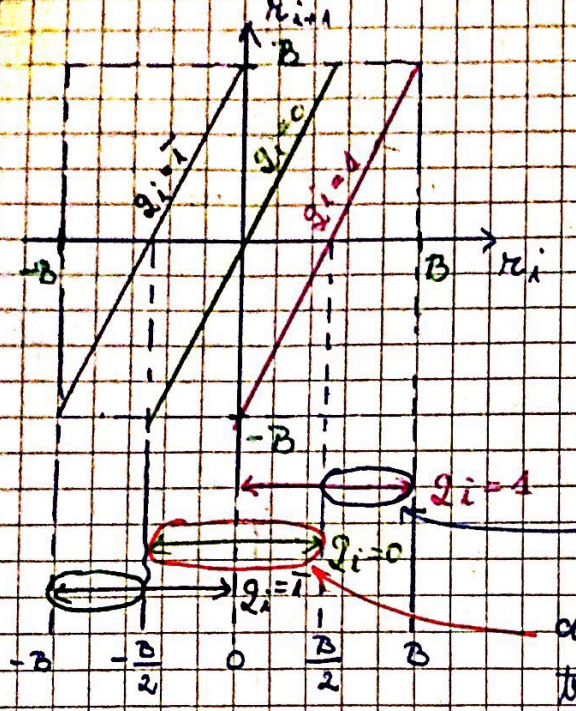
Exemplu S.R.T

$229 : 12 = 19$ $r=1$

COUNT	P	A	B
000	00000000000	11100101	00001100
	00000011100	10101000	11000000 — shift
	$q_0=0$		
	00001110010	10100000	
001	$q_1=0$		⊛ → 1 nu poate fi reținut în calc, deci folosim codificarea:
	00011100101	00000000	
010	$q_2=0$		00010100 −
	00111001010	00000000	00000001
011	$q_3=1$		00010011
	01110010101	00000001	Quotient = 19 (10)
	− 11000000		
	00010010101		
100	$q_4=0$		
	00100101010	00000010	
101	$q_5=1$		
	01001010100	00000101	
	− 11000000		
	11101010000		
110	$q_6=0$		
	11010100000	00001010	
111	$q_7=\bar{1}$		
	10101000000	00010101	
	+ 11000000		
	! 00010000		⊛

Shiftăm la dreapta cu 4 poziții

Remainder: 0000000001 = 1 (10)



r_i = rest partial

$r_{i+1} = 2r_i - q_i \cdot B$; $q_i \in \{\bar{1}, 0, 1\}$

pt. $q_i = 1$ luăm regm. asta pt $r_i \in [\frac{B}{2}, B]$
dar pt. $r_i \in [0, \frac{B}{2}]$ e mai ok $q_i = 0$
cea mai favorabilă situație pt. că nu trie să faci nimic

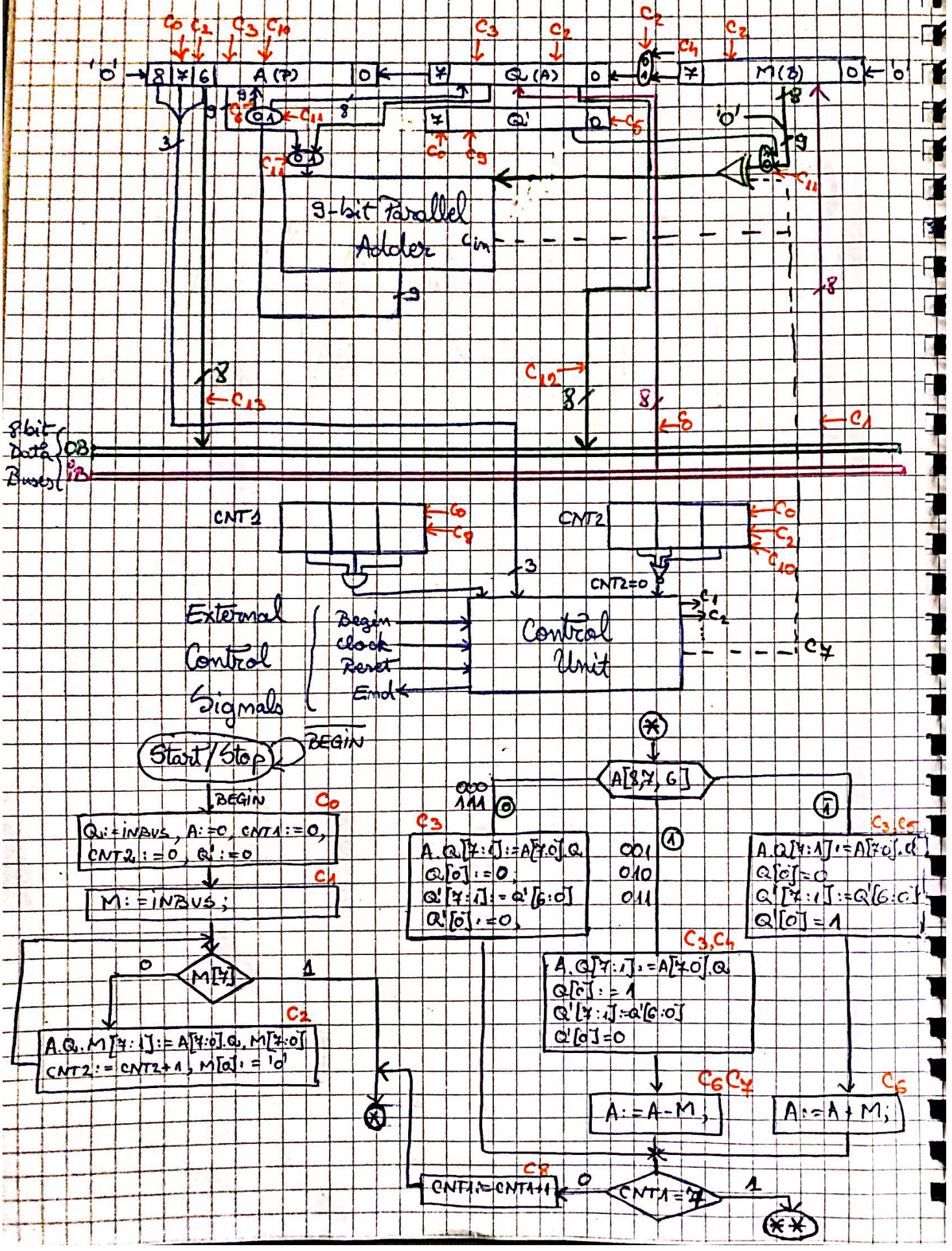
⊙ Exemplu S.R.I.T:

COUNT	P	A	B
000	0 0 0 0 0 0 0 0 0 0	1 1 1 0 1 0 0 1	0 0 0 0 0 1 0 0 1
	0 0 0 0 0 0 1 1 1 0	1 0 0 1 0 0 0 0	1 0 0 1 0 0 0 0
	$q_0=0$		
	0 0 0 0 0 1 1 1 0 1	0 0 1 0 0 0 0 0	
001	$q_1=0$		
	0 0 0 0 1 1 1 0 1 0	0 1 0 0 0 0 0 0	
010	$q_2=0$		
	0 0 0 1 1 1 0 1 0 0	1 0 0 0 0 0 0 0	
011	$q_3=1$		
	0 1 1 1 0 1 0 0 1	0 0 0 0 0 0 0 1	
	- 1 0 0 1 0 0 0 0		
	0 0 1 0 1 1 0 0 1		
100	$q_4=1$		
	0 1 0 1 1 0 0 1 0	0 0 0 0 0 0 0 1 1	
	- 1 0 0 1 0 0 0 0		
	0 0 0 1 0 0 0 1 0		
101	$q_5=0$		
	0 0 1 0 0 0 1 0 0	0 0 0 0 0 1 1 0	
110	$q_6=1$		
	0 1 0 0 0 1 0 0 0	0 0 0 0 1 1 0 1	
	- 1 0 0 1 0 0 0 0		
	1 1 1 1 1 1 0 0 0		
111	$q_7=0$		
	1 1 1 1 1 1 0 0 0 0	0 0 0 1 1 0 1 0 - 1	
Correction (pt. că 1)	+ 1 0 0 1 0 0 0 0	0 0 0 1 1 0 0 1	
	0 1 0 0 0 0 0 0 0	Quotient = $25_{(10)}$	
	(4 pași)		
Shift	0 0 0 0 0 1 0 0 0		

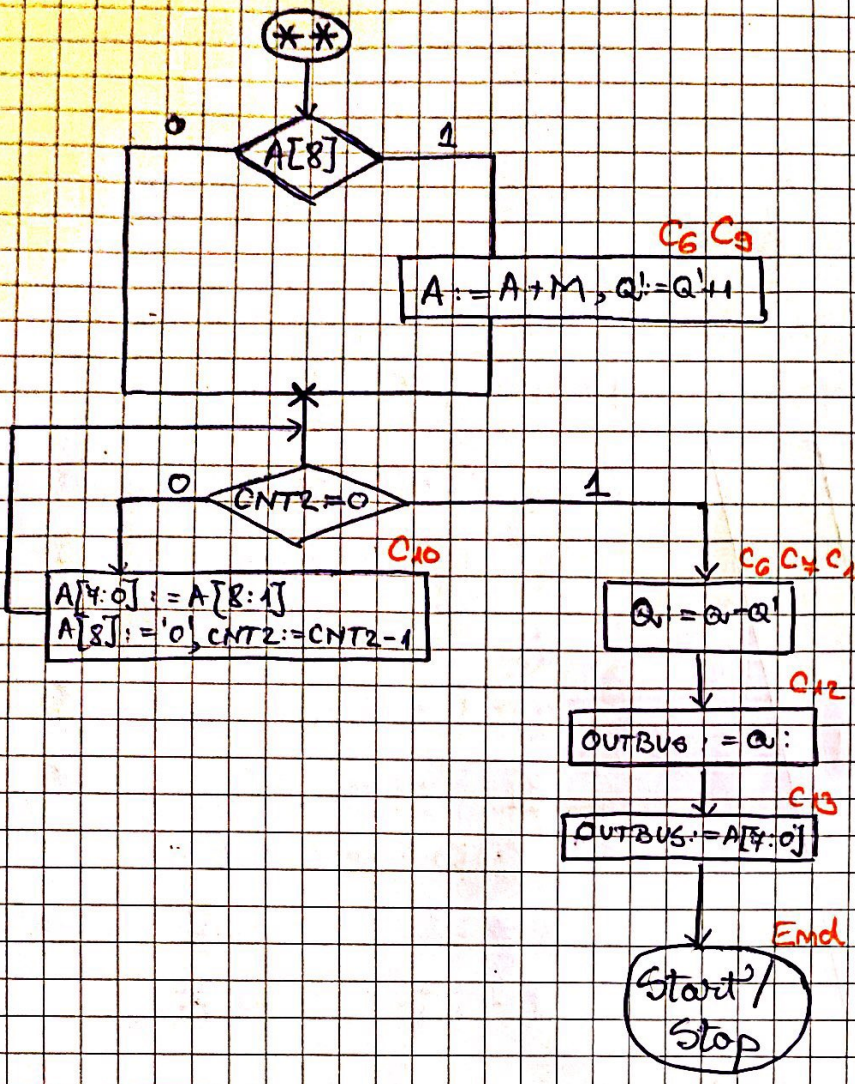
Remainder = $8_{(10)}$

CURS 5 (continuare)

1.4.4. Hardware implementation for radix-2 SRT



// CNT 2 trebuie să fie un registru capabil și de incrementare și de decrementare



1.5 Radix-4 SRT

$q_i \in \{\bar{2}, \bar{1}, 0, 1, 2\}$

$$r_{i+1} = 4r_i - q_i \cdot B$$

- $|r_i| < B \Rightarrow |r_{i+1}| < B$
  - insuficient

Fp. $r_i = B \Rightarrow r_{i+1} = 4B - q_i B = 4B - 2B = 2B$ (X)

- $|r_i| < \frac{2}{3} B \Rightarrow |r_{i+1}| < \frac{2}{3} B$

Fp. că $r_i = \frac{2}{3} B \Rightarrow r_{i+1} = \frac{8}{3} B - \frac{6}{3} B = \frac{2}{3} B$ (✓)