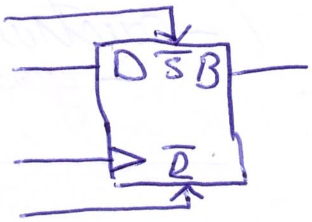
AC CB

3.3. Excitation and output equations
– derived from flowchart directly

Consider using D type flip-flop



$B(t+1) = D$

State variables: $B_0, B_1, -\,-, B_7$

State encoding:

State	B_7	B_6	B_5	B_4	B_3	B_2	B_1	B_0
S_0	0	0	0	0	0	0	0	1
S_1	0	0	0	0	0	0	1	0
S_2	0	0	0	0	0	1	0	0
S_3	0	0	0	0	1	0	0	0
S_4	0	0	0	1	0	0	0	0
S_5	0	0	1	0	0	0	0	0
S_6	0	1	0	0	0	0	0	0
S_7	1	0	0	0	0	0	0	0

Excitation equations

$D_0 = B_0 \cdot \overline{BEGIN}$ or B_7

$D_1 = B_0 \cdot BEGIN$

$D_2 = B_1$

$D_3 = B_2 \cdot \overline{Q[0]}$ or $B_4 \cdot \overline{COUNT7} \cdot Q[0]$

$D_4 = B_2 \cdot Q[0]$ or B_3 or $B_4 \cdot \overline{COUNT7} \cdot \overline{Q[0]}$

$D_5 = B_4 \cdot COUNT7$

$D_6 = B_5$

$D_7 = B_6$

Outputs equations

$END = B_0$

$C_0 = B_1$

$C_1 = B_2$

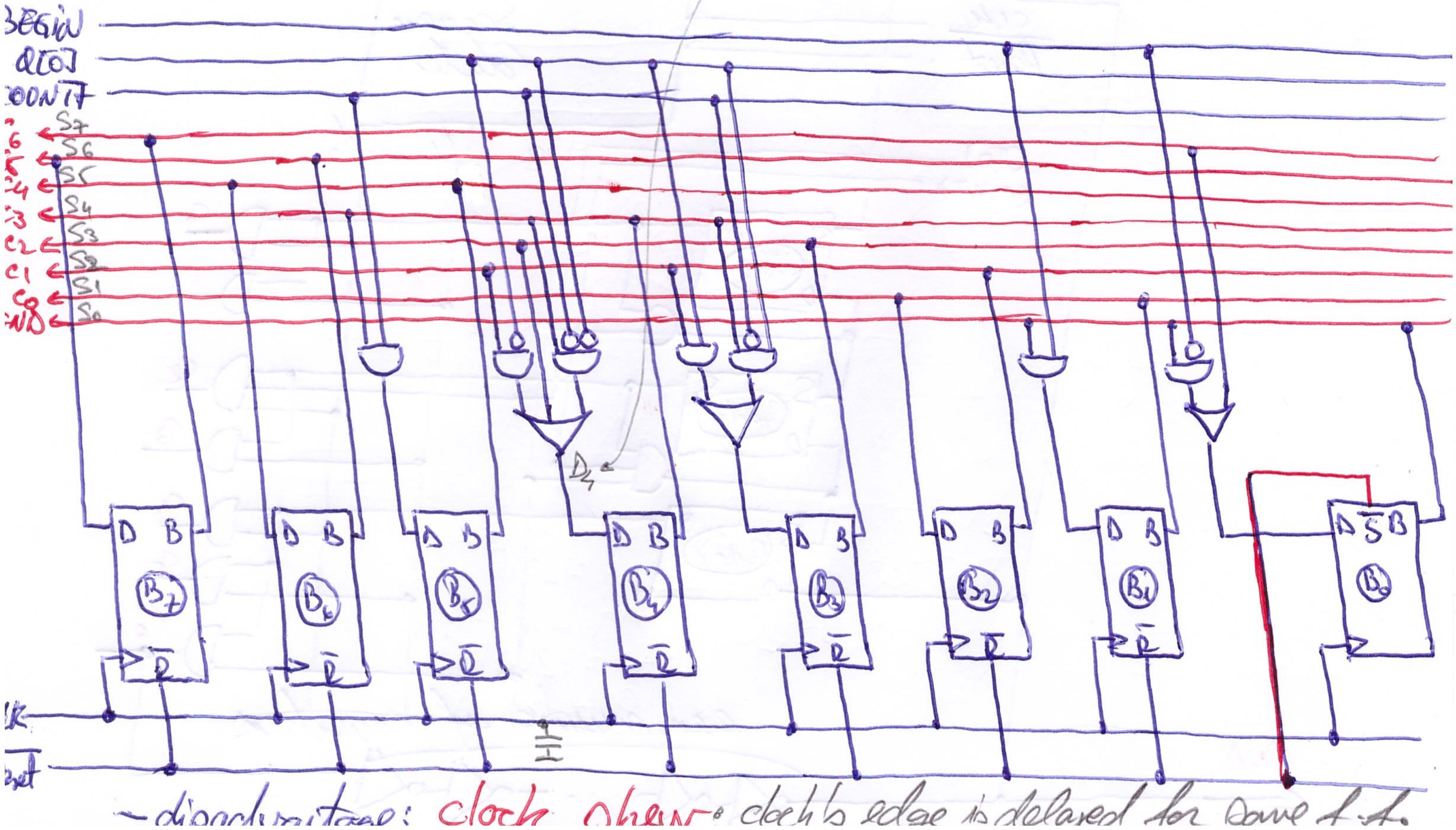
$C_2 = B_3$

$C_3 = B_4$

$C_4 = B_5$

$C_5 = B_6$

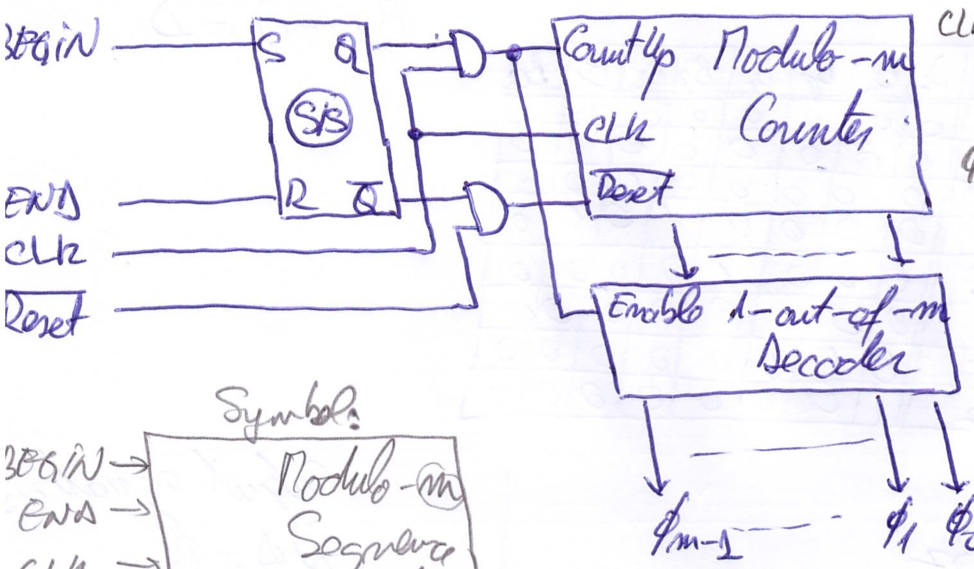
$C_6 = B_7$

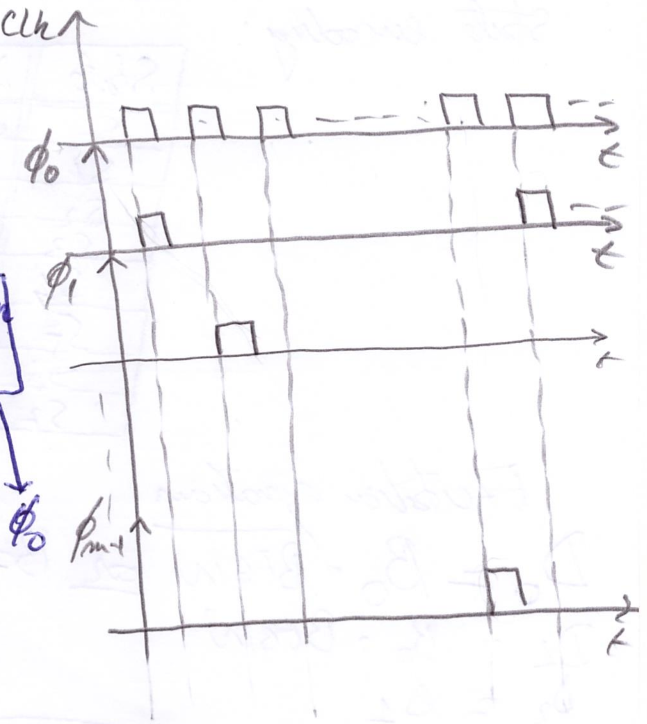


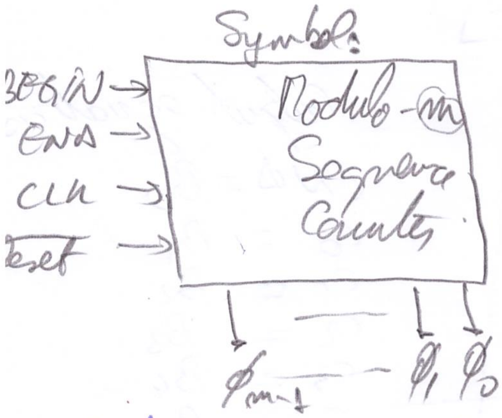
– disadvantage: clock skew: clock's edge is delayed for some f.f.

B) Sequence Counter

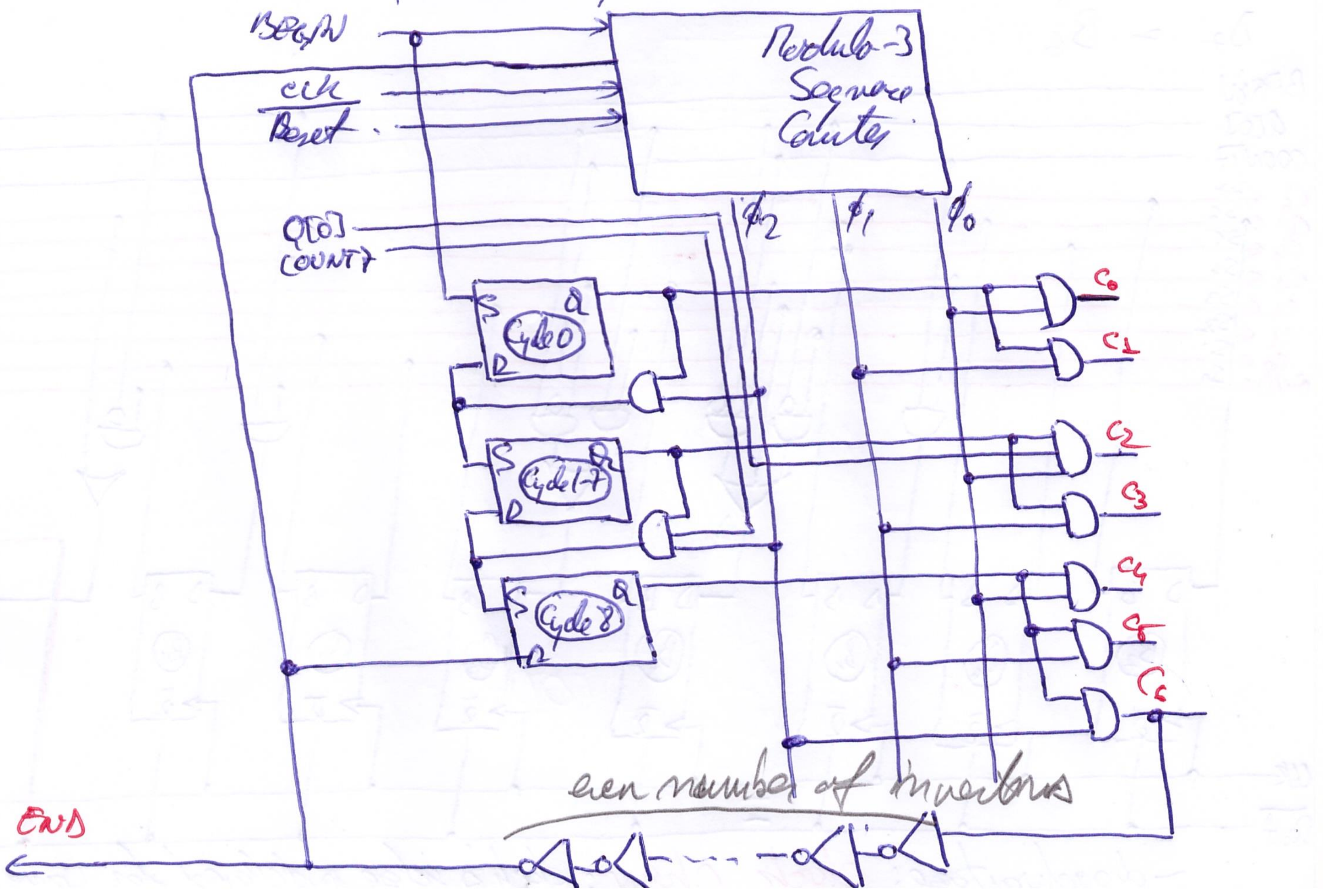
- constructed around a Sequence Counter: phase
- generates non-overlapping, synchronous pulses







Determine value of m: longest cycle : m = "3" (Cycle 8)
Provide one S/R latch for each cycle: active in the respective cycle.



3.4. Two's Complement Multiplication Based on Robertson's Procedure ②

Robertson's interpretation: the value of a number in C2 is equal with the value of the positive number obtained by clearing its sign bit from which is subtracted the weight associated with the sign bit

Example: X integer $X = \underset{2^3}{1}011_{C2} \rightarrow X = \overbrace{0011_{C2}}^{\text{positive SM}} - 1\cdot 2^3 = 3-8 = -5$

$1011_{C2} \rightarrow \overbrace{1010_{C1}}^{\text{complement}} \rightarrow 1101_{SM} = -5$

X integer: $X = \underset{2^3}{0}101_{C2} \rightarrow X = \underbrace{0101_{C2}}_{\text{positive SM}} - 0\cdot 2^3 = 5-0=5$

Multiplication of C2 can be obtained:
1) convert C2 → SM for X, Y (2×)
2) multiply X * Y = P in SM.
3) convert SM → C2 for P (1×), double weight

Let $X_{C2} = x_{n-1}.\, x_{n-2} \_\_\_ x_1 x_0.$ (fractional, integer)

for X, integer $X = \underbrace{-x_{n-1}\cdot 2^{n-1}}_{\text{CORRECTION}} + \underbrace{0\, x_{n-2} \_\_\_ x_1 x_0}_{\text{positive in C2, SM}}$

for X, fractional $X = \underbrace{-x_{n-1}\cdot 2^{0}}_{\text{CORRECTION}} + \underbrace{0\, x_{n-2} \_\_\_ x_1 x_0}_{\text{positive in C2, SM}}$

A positive in C2 ≡ A positive in SM
A negative in C2 ≡ A positive in SM + CORRECTION

$P = X_{C2} * Y = (\underbrace{0\, x_{n-2} \_\_ x_1 x_0}_{X^*_{SM}\text{-positive}} - \underbrace{x_{n-1}\cdot 2^0}_{\text{CORRECTION}}) * Y$ (for X, Y - fractionals)

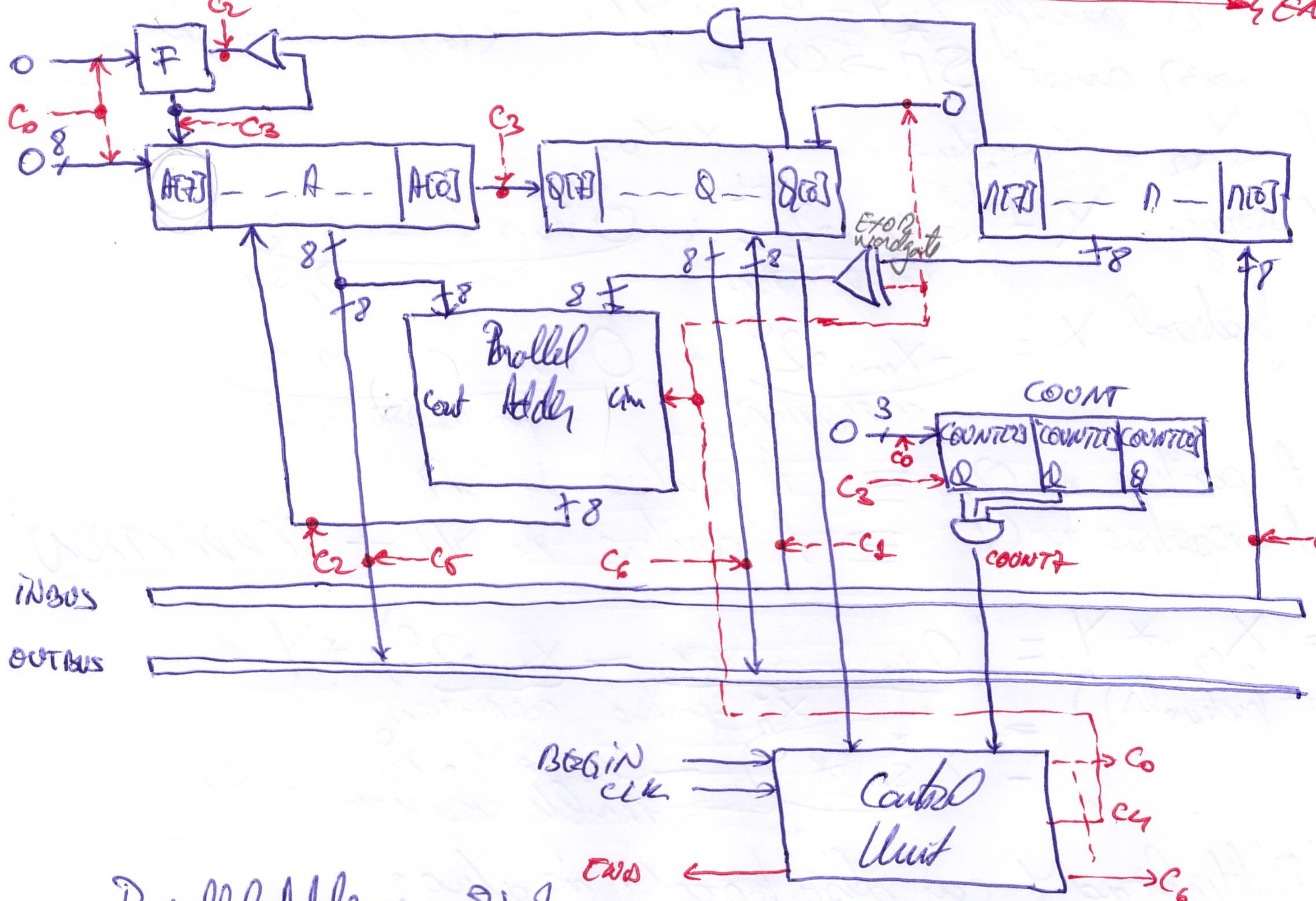
$= X^*_{SM} * Y - x_{n-1} * Y \cdot 2^0$ → multiplication in SM.

Multiplicand Y can also be a C2 negative:
⇒ all partial products have sign
— they have the sign of Y
→ accumulator A will now store the sign.

multiplier 3

declare registers A[7:0], Q[7:0], M[7:0], COUNT[2:0], F;
declare bus INBUS[7:0], OUTBUS[7:0];

BEGIN: A := 0, COUNT := 0, F := 0, {c_0}
INPUT: M := INBUS;
Q := INBUS; {c_1}
TEST1: if Q[0] = 0 then go to RShift;
ADD: A := A + M, F := F or (Q[0] and M[7]); {c_2}   sign of Y.
RShift: A[7] := F, A[6:0].Q := A.Q[7:1], {c_3}
INCREMENT: COUNT := COUNT + 1;
TEST2: if COUNT7 ≠ 1 then go to TEST1,
TEST3: if Q[0] = 0 then go to OUTPUT,
CORRECTION: A := A − M, Q[0] := 0; {c_2, c_4}
OUTPUT: OUTBUS := A; {c_5}
OUTBUS := Q; {c_6}
END {END}



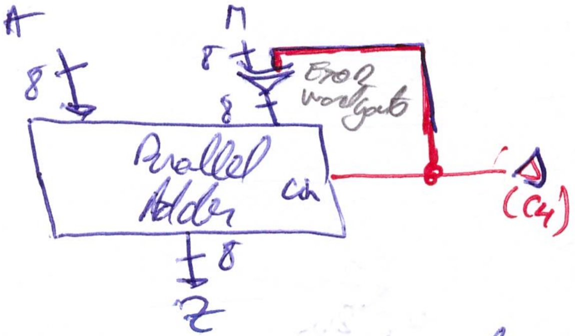
Parallel Adder: on 8 bits
– Cout: ignored!

Q: Can overflow occur?!? How is it treated?!?

A: At RShift, A is restoring its sign from flag F.

In flag F: store the correct sign of partial products.
– is part of a adder/subtractor.

$V = c_n \oplus c_{n-1}$



s: 0 : $Z = A + M$

1 : $Z = A - M$

Flag F: stores the sign of partial products.

Q: why not use M[7] (sign of Y)

A: as long as $P_i == 0$, $i \geq 0$
sign of those P_i must be 0, not M[7]
after P_i becomes $\neq 0$, F is set to M[7]

$\begin{cases} P_i := P_i + x_i \cdot Y \\ P_{i+1} : P_i \cdot 2^{-1} \end{cases}$

Q: which partial product, P_i, is $\neq 0$?!?
– the one for which x_i (stored in Q[0])
$x_i = 1$

Sign of result is not set:
– operating in C2 generates correct result sign.

Depending on the signs of the 2 operands, algorithm performs the following operations

x_7	y_7	CORRECTION STEP	ARITHMETIC RShift
0	0	NO	NO
0	1	NO	YES
1	0	YES	NO
1	1	YES	YES

Example: $X = -0.75 = 1.110_{SM} = 1.010_{C2}$
$Y = -0.875 = 1.111_{SM} = 1.001_{C2}$

F	A	Q	M	COUNT	Activated Control Signals
0	0000		1001	00	C_0
		1010			C_1
	0000	0101		01	C_3
	+1001				
1	1001				C_2
	1100	1010		10	C_3
	1110	0101		11 *COUNT3==1	C_3
	-1001				
	0101	0100			C_2, C_4
	$\boxed{0101}$				C_5
		$\boxed{0100}$			C_6

$P = 0.10101000 = +10101 \cdot 2^{-5} = +21 \cdot 2^{-5}$

$X = -3 \cdot 2^{-2}$ $Y = -7 \cdot 2^{-3}$ $P = (-3) \cdot 2^{-2} \cdot (-7) \cdot 2^{-3} = +21 \cdot 2^{-5}$