Solutions

(a) Note that a 3-to-8 decoder and a single AND gate can implement any Boolean fuction of 3 variables. So the strategy is to find an analytic expression for the given function (of 5 variables) and try to simplify it, so that it can be implemented using a decoder, using additional flexibility provided by Enable input (of the decoder).
Hence, solution to this problem requires two parts:
(1) Deriving an analytic expression for the Boolean function F(A,B,C,D,E) shown on K-map, in a min SOP or min POS form.
(2) Implementing this Boolean function using the decoder.

Part (1). The decoder specified in this problem has 3 inputs and inverted outputs; therefore it can implement any Boolean function of 3 variables in a maxterm canonical form, where selected decoder outputs (maxterms) are used as inputs to a single AND gate. Hence, a reasonable solution strategy is to find a simple expression for the Boolean function in a minimum Product-of-Sums (POS) form.
The 5-variable function shown on K-map is (almost) symmetric with respect to C, i.e. all maxterms in the right half (for C=1) are also present in the left half of the map (for C=0). In addition, the left half has an extra maxterm (A+B+C+D+E').
So we can first simplify the symmetric part in a min POS form, and then add an extra product term (the extra maxterm), to form the function F(A,B,C,D,E).
Note that the symmetric part of the K-map is a function of 3 variables, because its maxterms are symmetric with respect to both C and A variables. So its min POS form is a 3-variable function G(B,D,E) = (B+D'+E')(B'+E) – can be found from K-map.
The original 5-variable function can be decomposed as
F(A,B,C,D,E) = (A+B+C+D+E') G(B,D,E) or, even better/simpler as
F(A,B,C,D,E) = (A+B+C+E') G(B,D,E)

Part (2). Function G(B,D,E) can be implemented using a decoder with inputs B,D,E and a single AND gate. In order to decide which outputs (maxterms) should be ANDed together, the function needs to be represented in the maxterm canonical form, i.e.

G(B,D,E) = (B+D'+E')(B'+E) = (B+D'+E')(B'+D'+E))(B'+D+E) = Π M(3,4,6)

In order to generate the required function F(A,B,C,D,E) = (A+B+C+E') G(B,D,E), the additional product term is fed to Enable (EN) input as EN' = (A+B+C+E') = (A'B'C'E)'. This requires a second AND gate.

Grading: due to a typo in the original K-map specification for this problem, implementation of the Boolean function F(A,B,C,D,E) requires two AND gates rather than one. So full credit (for this problem) will be given to students who showed correct algebraic/analytic representation of F(A,B,C,D,E), in either min POS or SOP form, - this corresponds, roughly, to Part (1) in the above solution.

(b) For the original counter (with negative edge-triggered flip-flops):

Flip-Flop operation table
Q_1 complemented on negative edge of CP
Q_2 complemented if [Q_8=0 and Q_1: $1 \rightarrow 0$]
Q_2 cleared if [Q_8=1 and Q_1: $1 \rightarrow 0$]
Q_4 complemented if [Q_2: $1 \rightarrow 0$]
Q_8 complemented if [Q_4Q_2=11 and Q_1: $1 \rightarrow 0$]
Q_8 cleared if [(Q_4 =0 or Q_2=0) and Q_1: $1 \rightarrow 0$]

These conditions result in the following count sequence, starting from initial state 0000
This is a BCD or modulo 10 counter 0,1,2,…,9 and repeat.

Q_8	Q_4	Q_2	Q_1
0	0	0	0
0	0	0	1
0	***0***	***0***	***0***
0	0	1	0
0	0	1	1
0	***0***	***1***	***0***
0	***0***	***0***	***0***
0	1	0	0
0	1	0	1
0	***1***	***0***	***0***
0	1	1	0
0	1	1	1
0	***1***	***1***	***0***
1	***1***	***0***	***0***
1	0	0	0
1	0	0	1
1	***0***	***0***	***0***

Transition states are bold and italicized

ANALYSIS of UNUSED STATES
Present $Q_8Q_4Q_2Q_1$: 1010 1011 1100 1101 1110 1111

Next state : 1011 0100 1101 0100 1111 0000

→ So this counter is self-starting.

For the counter with positive edge-triggered flip-flops:

Flip-Flop operation table

Q_1 complemented on positive edge of CP

Q_2 complemented if [Q_8=0 and Q_1: $0 \rightarrow 1$]

Q_2 cleared if [Q_8=1 and Q_1: $0 \rightarrow 1$]

Q_4 complemented if [Q_2: $0 \rightarrow 1$]

Q_8 complemented if [Q_4Q_2=11 and Q_1: $0 \rightarrow 1$]

Q_8 cleared if [(Q_4=0 or Q_2=0) and Q_1: $0 \rightarrow 1$]

These conditions result in the following count sequence, assuming initial state 0000.

Q_8	Q_4	Q_2	Q_1	State
0	0	0	0	0
0	***0***	***0***	***1***	
0	***0***	***1***	***1***	
0	1	1	1	7
0	1	1	0	6
0	***1***	***1***	***1***	
1	1	0	1	13
1	1	0	0	12
1	***1***	***0***	***1***	
0	1	0	1	5
0	1	0	0	4
0	***1***	***0***	***1***	
0	***1***	***1***	***1***	
0	0	1	1	3
0	0	1	0	2
0	***0***	***1***	***1***	
0	0	0	1	1

Transition states are bold and italicized

This counter also cycles through a sequence of 10 states:
0, 7, 6, 13, 12, 5, 4, 3, 2, 1, and repeat.