BINARY MULTIPLIER

➔ **To Multiply two unsigned binary numbers.**

➔ **Sequential Multiplier**

➔ **Uses one adder and a shift register.**

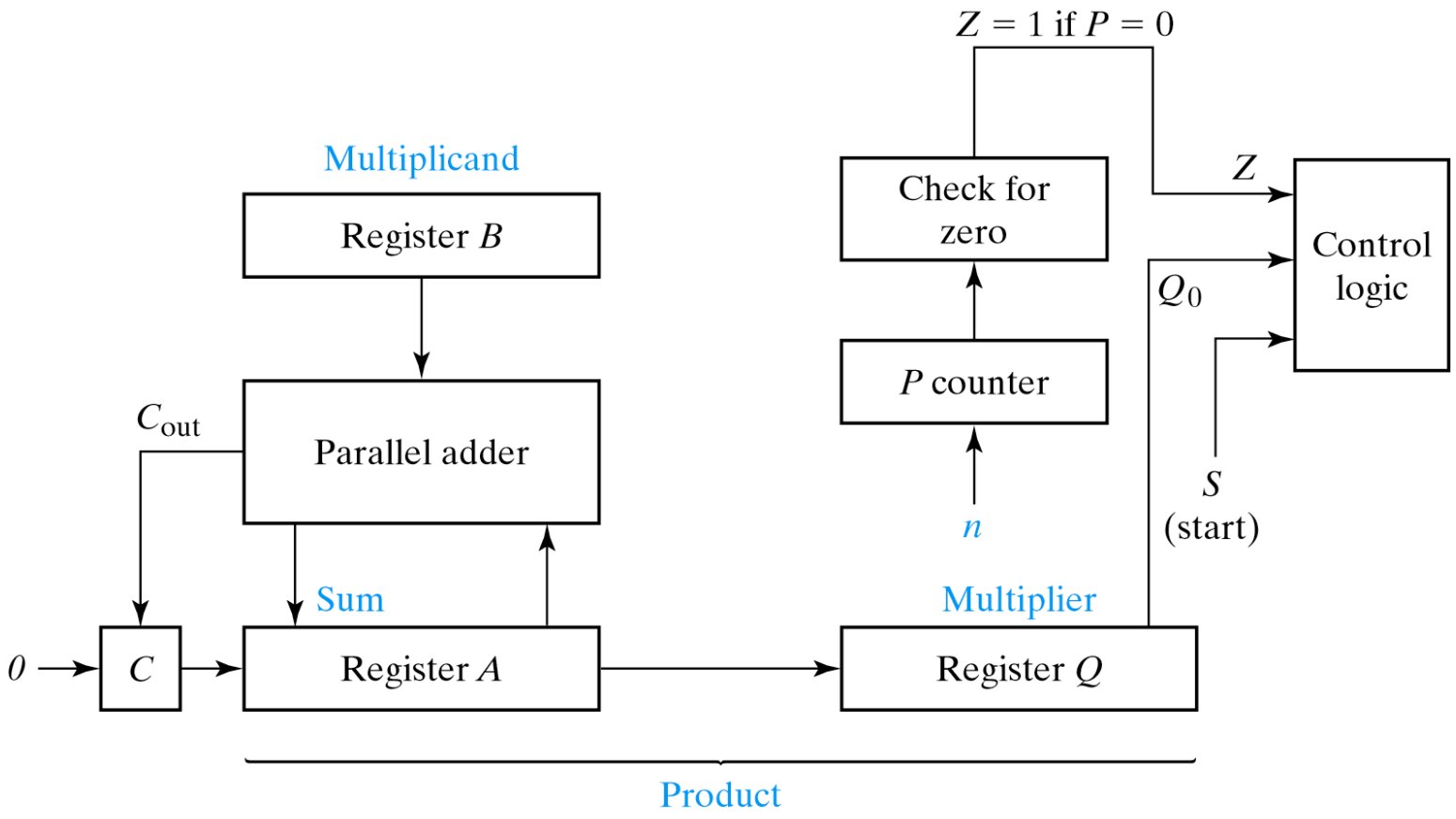


Fig. 8-13 Block Diagram of Binary Multiplier

➔ **Initially Multiplier in Q and Multiplicand in B**

➔ **With S = 0 no action and circuit is in state T0**

➔ **Multiplication process starts when S = 1 and control goes to T1.**

➔ **Register A and C set to 0 and counter P set to n, the number of bits in multiplier.**

➔ **System goes to T2.**

➔ **Multiplier bit Q0 is checked and if = 1, multiplicand B added to partial product A.**

➔ **Carry from addition transferred to C**

➔ **Partial product in A and C left unchanged if Q0 = 0**

➔ **P decremented, next state is T3.**

➔ **Registers CAQ shifted once to right**

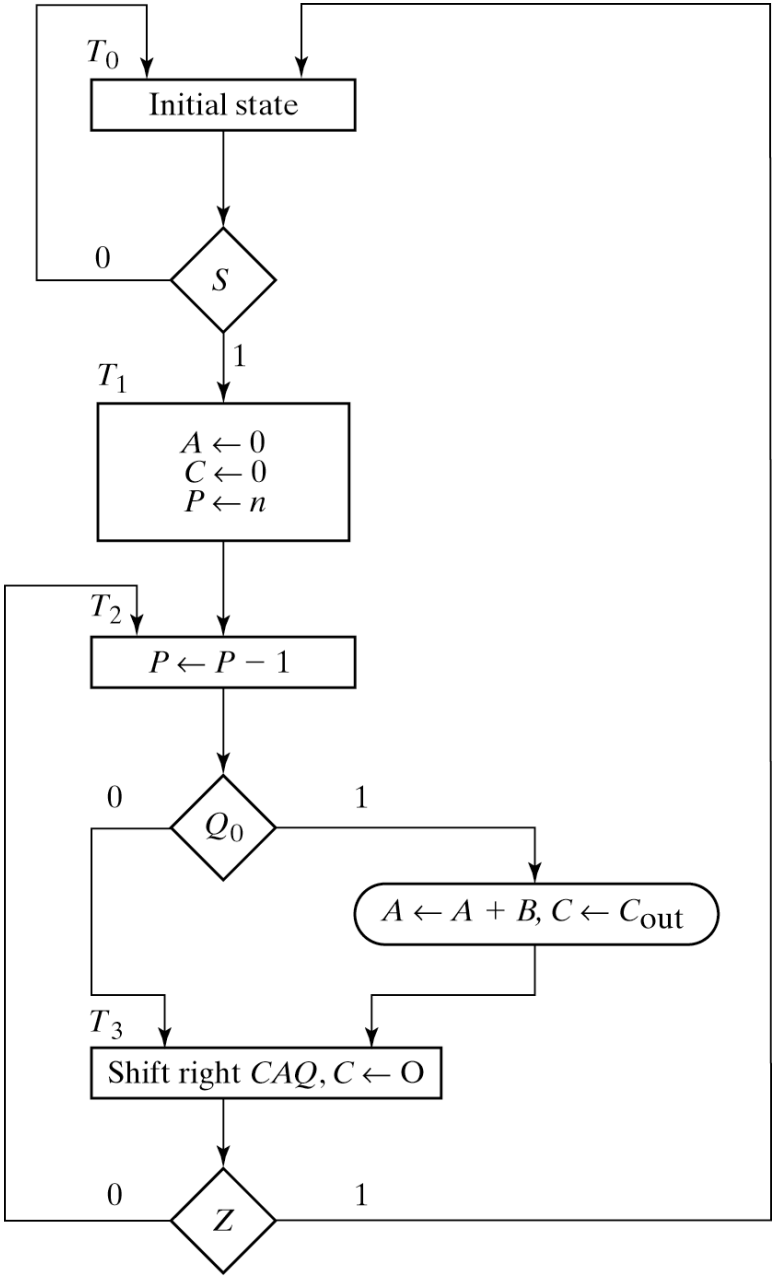


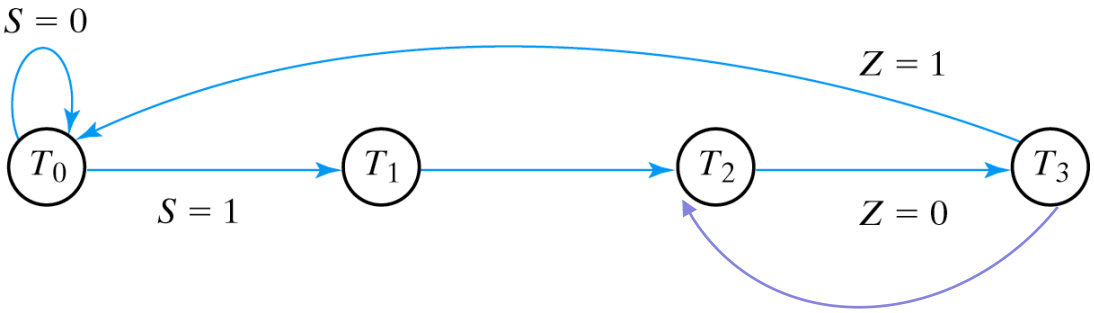
Fig. 8-14 ASM Chart for Binary Multiplier

Design steps:

➔Design of register operations in the datapath
➔Design of control logic

Diamond boxes determine the conditions for next state transition

Register transfer operations come from State and Conditional boxes



(a) State diagram

T_0: Initial state

T_1: $A \leftarrow 0, C \leftarrow 0, P \leftarrow n$

T_2: $P \leftarrow P - 1$

if $(Q_0) = 1$ then $(A \leftarrow A + B, C \leftarrow C_{out})$

T_3: shift right CAQ, $C \leftarrow 0$

(b) Register transfer operations

Fig. 8-15 Control Specifications for Binary Multiplier

In the design of control logic:

➔ **Establish required sequence of operations & provide signals to control register transfer operations.**

➔ **Sequence specified in state diagram**

➔**Signals for controlling register operations:**

T1 (for clearing A and C), T2 (for decrementing P)
T3(for shifting CAQ), Q0 to decide whether to add B or not

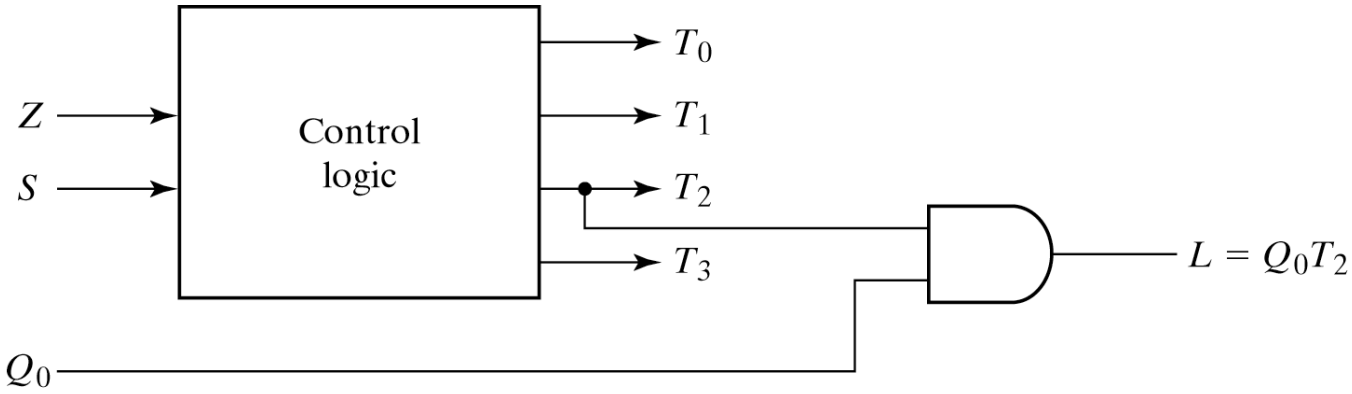


Fig. 8-16 Control Block Diagram

Inputs S and Z, outputs T0, T1, T2, T3
L = T2Q0 to load sum into A if Q0 = 1 while in T2.

State assignment for control

State	Binary	Gray code	One-Hot
T0	00	00	0001
T1	01	01	0010
T2	10	11	0100
T3	11	10	1000

Control logic design using sequence register and decoder

➔ An n bit sequence register is a circuit with n flip-flops

➔ Multiplier having four states and two inputs needs two flip-flops for the register and a 2:4 decoder for outputs

State Table:

Present state		Inputs		Next state		Outputs			
G1	G0	S	Z	G1	G0	T0	T1	T2	T3
0	0	0	X	0	0	1	0	0	0
0	0	1	X	0	1	1	0	0	0
0	1	X	X	1	0	0	1	0	0
1	0	X	X	1	1	0	0	1	0
1	1	X	0	1	0	0	0	0	1
1	1	X	1	0	0	0	0	0	1

➔ **Input columns have don't care entries whenever the input variable is not used to determine the next state**

➔ **Outputs are functions of present state, generated with a Decoder with two inputs (G1 G0) and four outputs (T0 –T3)**

➔**Next state of G1 is equal to 1**
when the present state is T1,
when the present state is T2,
when the present state is T3, provided Z = 0

So we write:

DG1 = T1 + T2 + T3Z'

Similarly:

DG0 = T0S + T2.

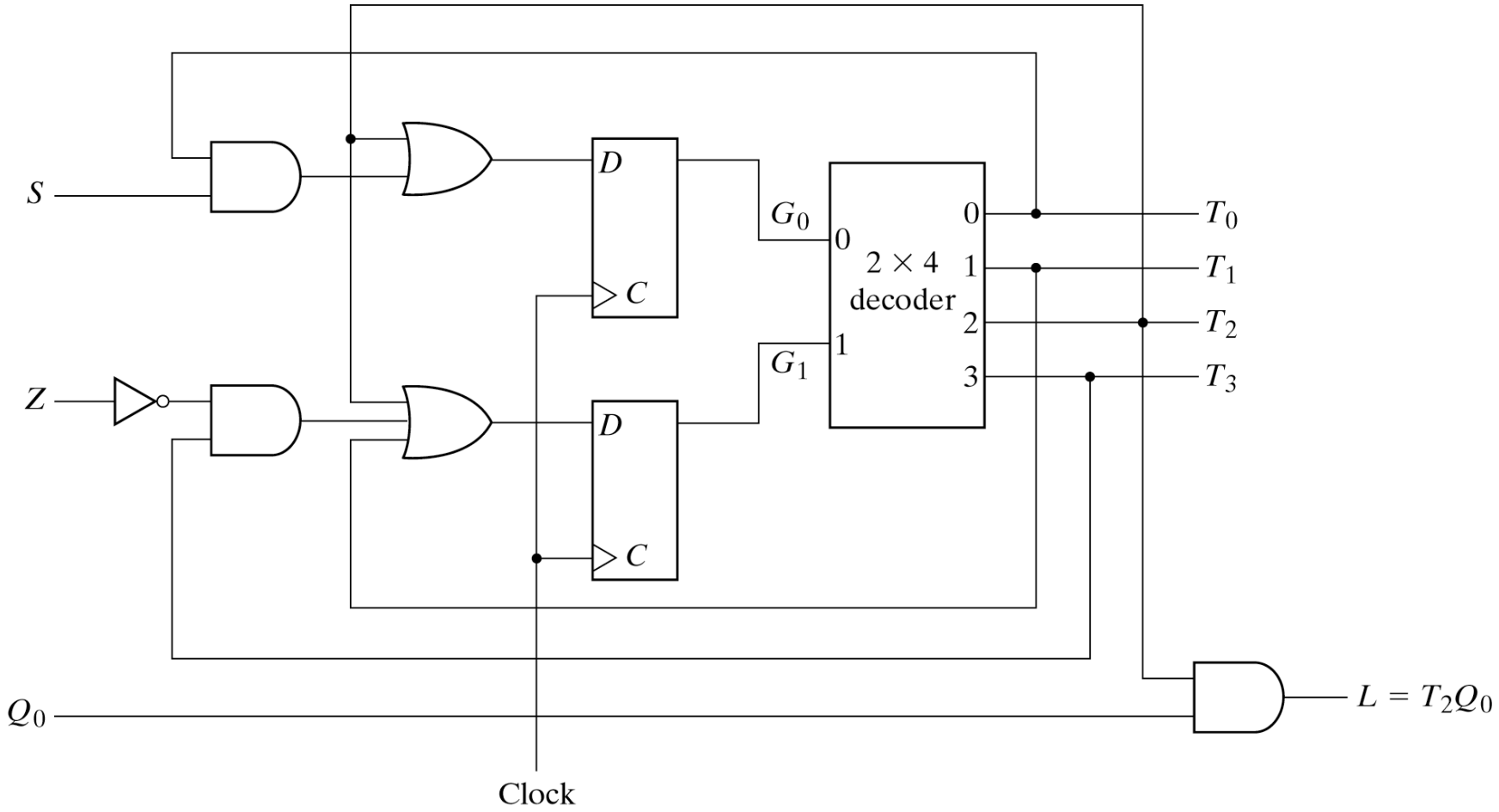


Fig. 8-17 Logic Diagram of Control for Binary Multiplier Using a Sequence Register and Decoder

Use of One Hot assignment which uses one Flip-flop per state

Single 1 propagates from one flip-flop to another under the control of decision logic.

DT0 = T0S' + T3Z
DT1 = T0S
DT2 = T1 + T3Z'
DT3 = T2.

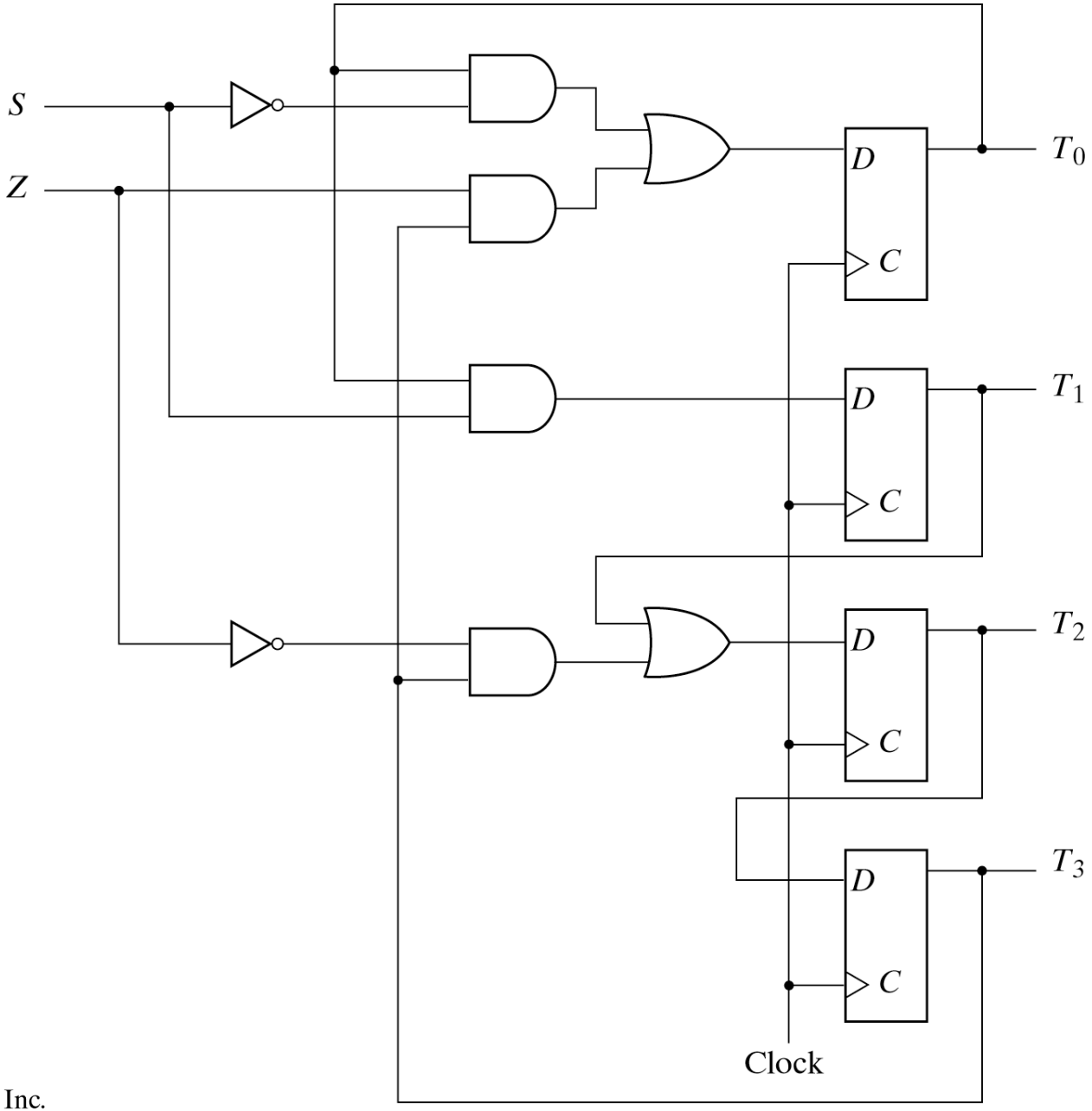


Fig. 8-18 One Flip-Flop Per State Controller

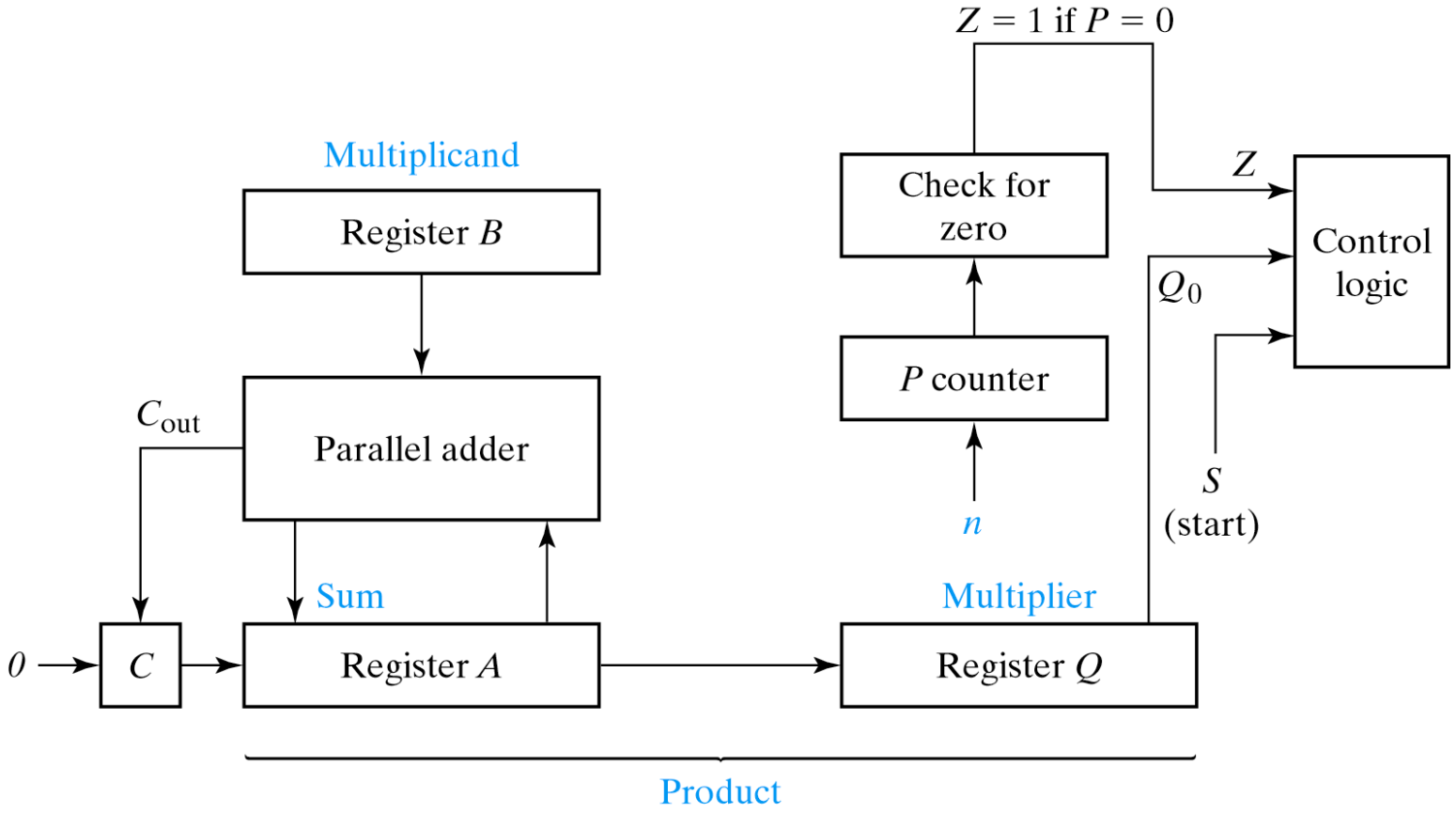


Fig. 8-13 Block Diagram of Binary Multiplier

T1 (for clearing A ,C and load P), T2 (for decrementing P)
T3(for shifting CAQ), T2Q0 to decide whether to Load A or not

Design with Multiplexers

➔**Combinational circuit can be implemented with Multiplexers instead of gates.**

➔**Results in regular pattern of three level of components.**

1. **Multiplexers that determine next state of register**
2. **Registers that hold the present state**
3. **Decoders that provide separate output for control state**

Predefined standard cell in many ICs.

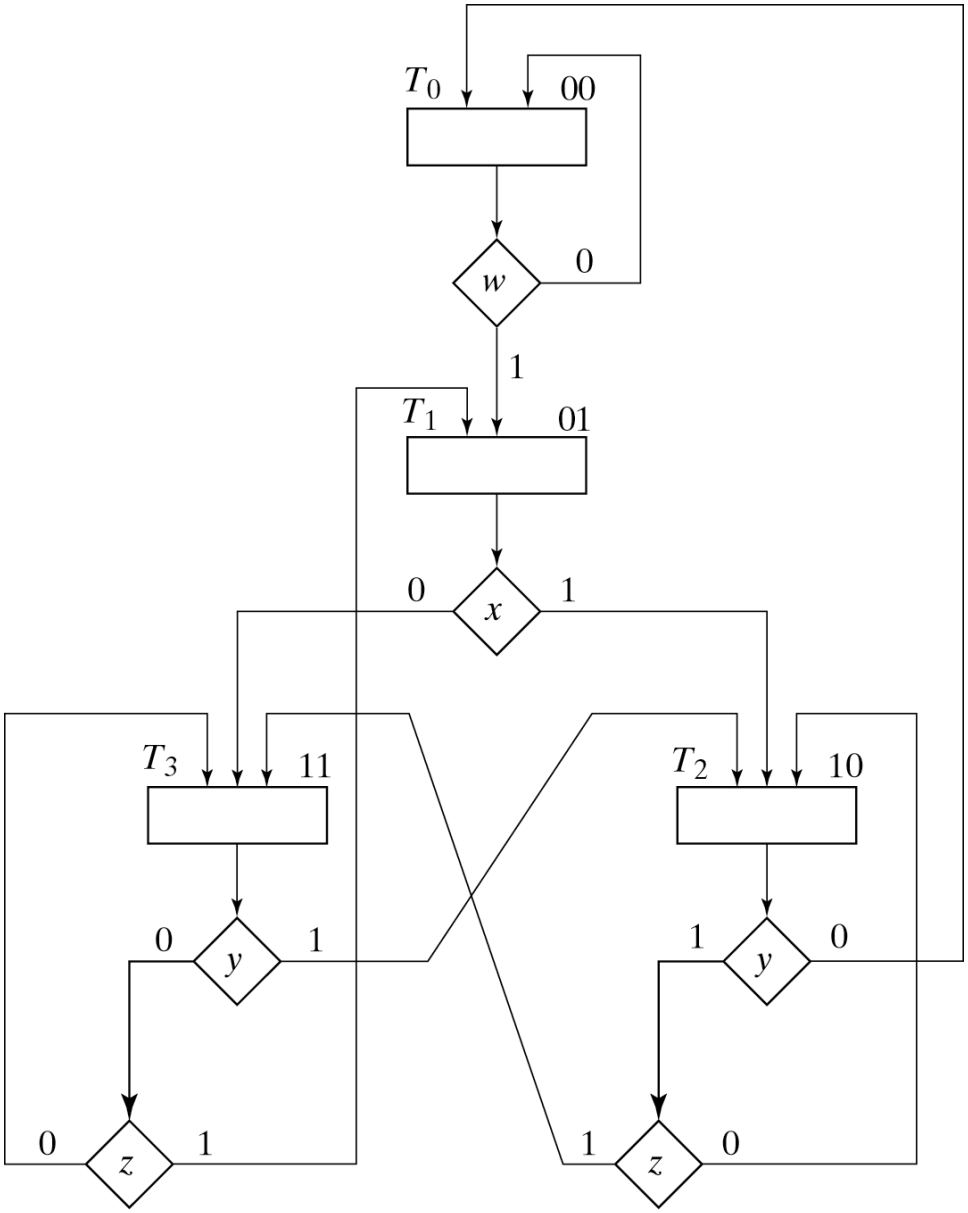


Fig. 8-19 Example of ASM Chart with Four Control Inputs

➔ **State boxes left empty as it is only to design Control sequence**

➔ **Binary assignments specified at upper right corner**

➔ **Decision boxes specify state transition as a function of four control inputs w, x, y,z**

➔ **Three level control implementation includes MUX, registers and decoders**

Multiplexer input conditions

Present state		Next state		Input	Inputs	
G1	G0	G1	G0	conditions	MUX1	MUX2
0	0	0	0	w’		
0	0	0	1	w	0	w
0	1	1	0	x		
0	1	1	1	x’	1	x’
1	0	0	0	y’		
1	0	1	0	yz’	yz’+yz = y	yz
1	0	1	1	yz		
1	1	0	1	y’z		
1	1	1	0	y		
1	1	1	1	y’z’	y + y’z’ = y + z’	y’z +y’z’ =y’

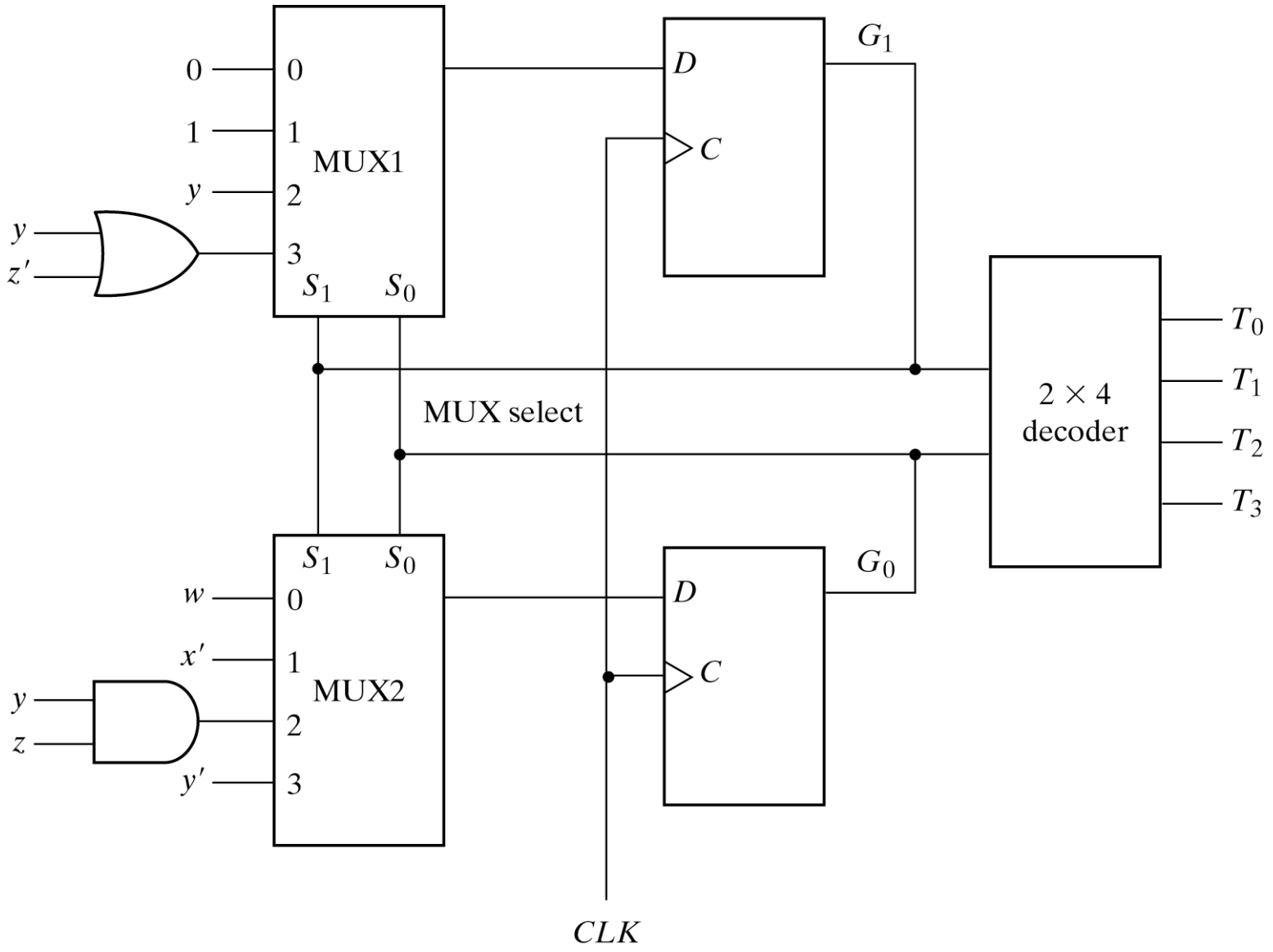


Fig. 8-20 Control Implementation with Multiplexers

Design Example:
Count number of 1's in a register

Two registers R1 and R2 and Flip-flop E

- **System counts number of 1's in R1 and sets R2 to that number.**
- **Done by shifting each bit in R1 one at a time into E, E checked by control and if 1 R2 is incremented.**
- **Control uses external input S to start and uses status input E and Z from datapath. Z = 1 when R1 = 0**

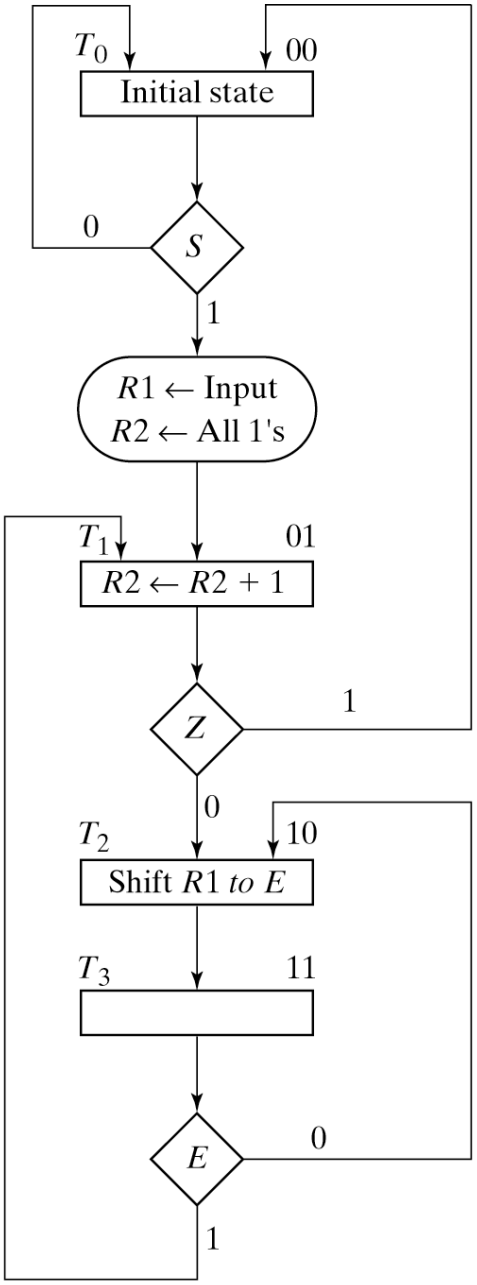


Fig. 8-21 ASM Chart for Count-of-Ones Circuit

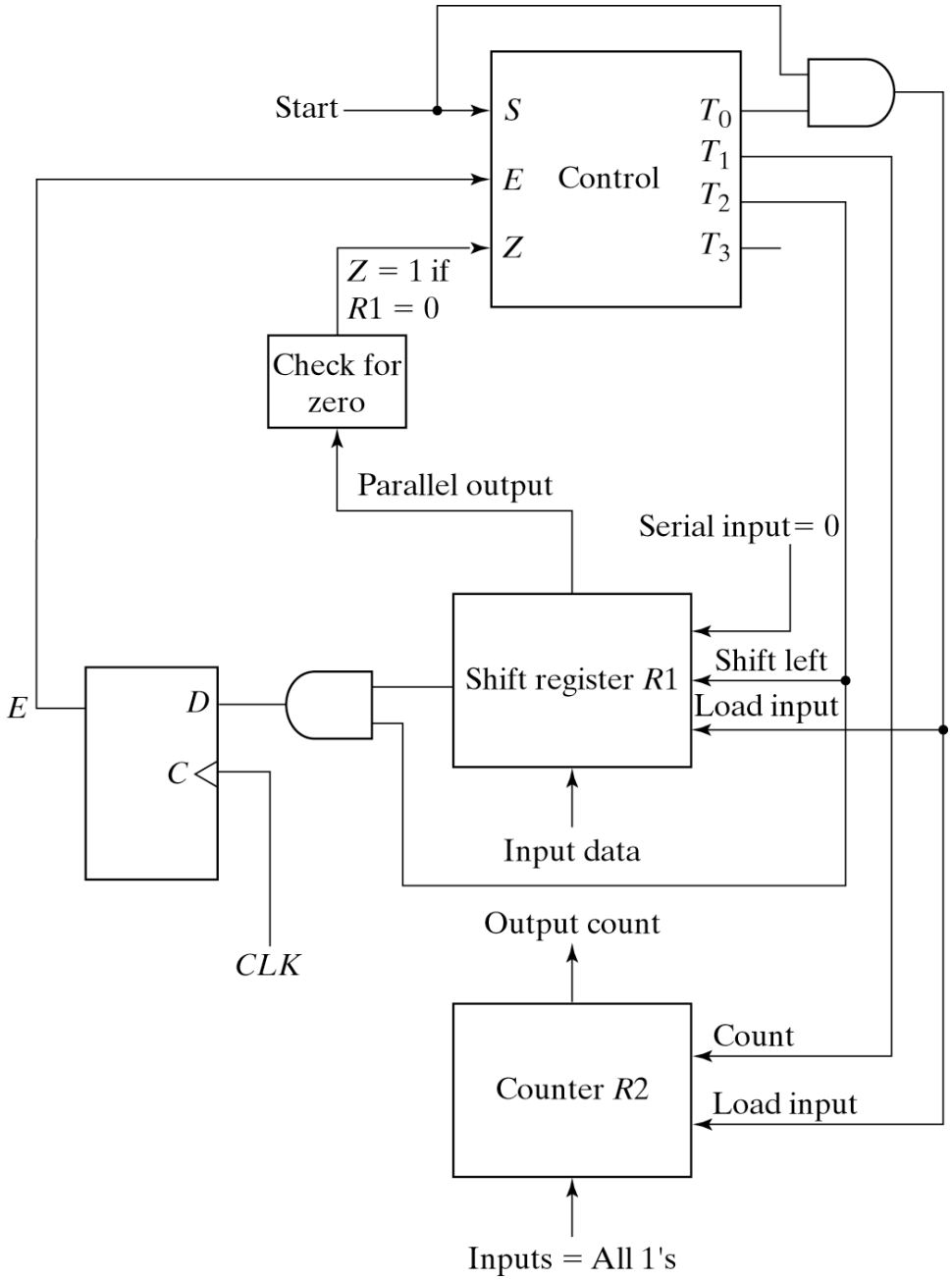


Fig. 8-22 Block Diagram for Count-of-Ones

Multiplexer input conditions

Present state		Next state		Input	Inputs	
G1	**G0**	**G1**	**G0**	**conditions**	**MUX1**	**MUX2**
0	0	0	0	S’		
0	0	0	1	S	0	S
0	1	0	0	Z		
0	1	1	0	Z’	Z’	0
1	0	1	1	None	1	1
1	1	1	0	E’		
1	1	0	1	E	E’	E

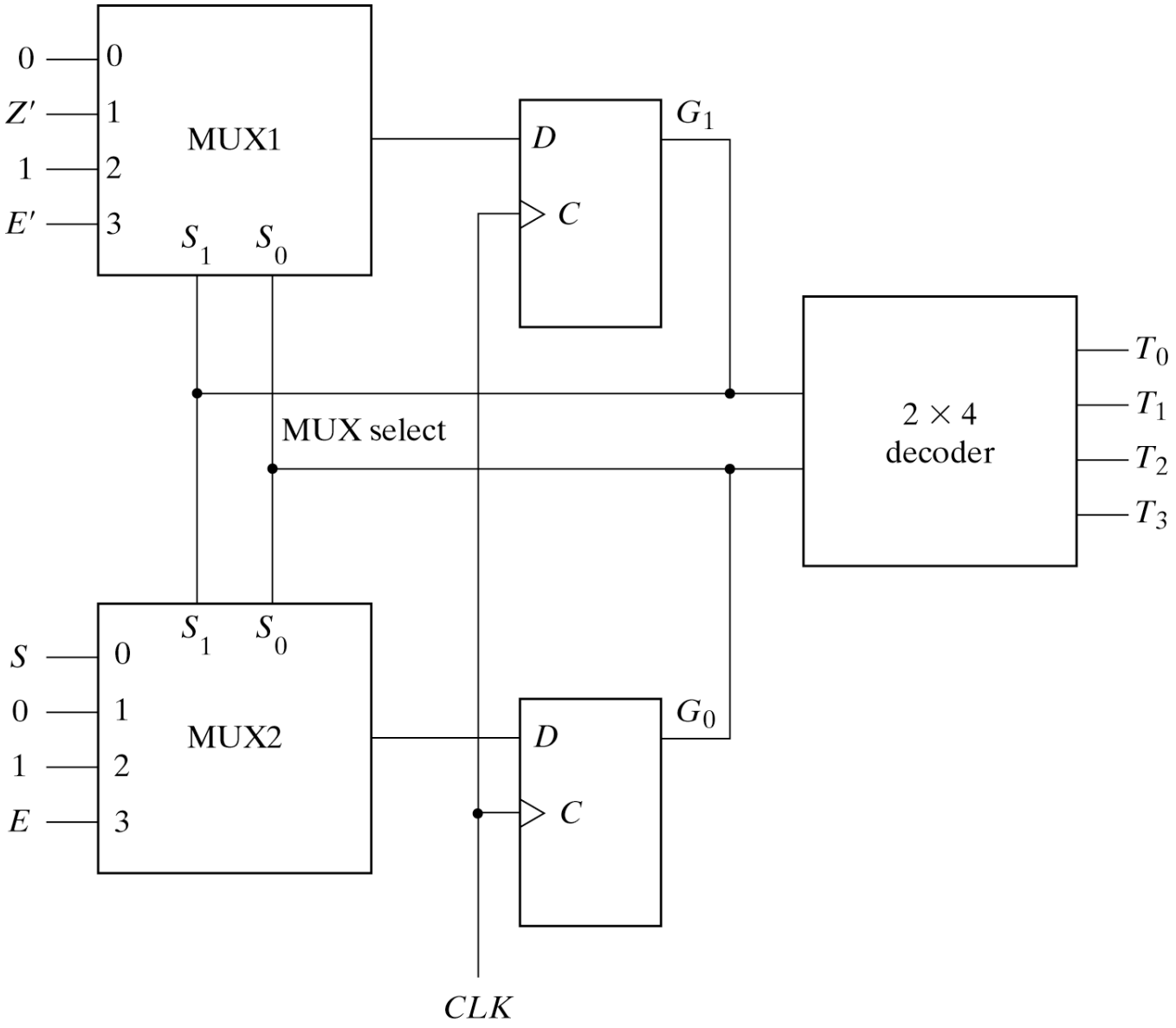


Fig. 8-23 Control Implementation for Count-of-Ones Circuit