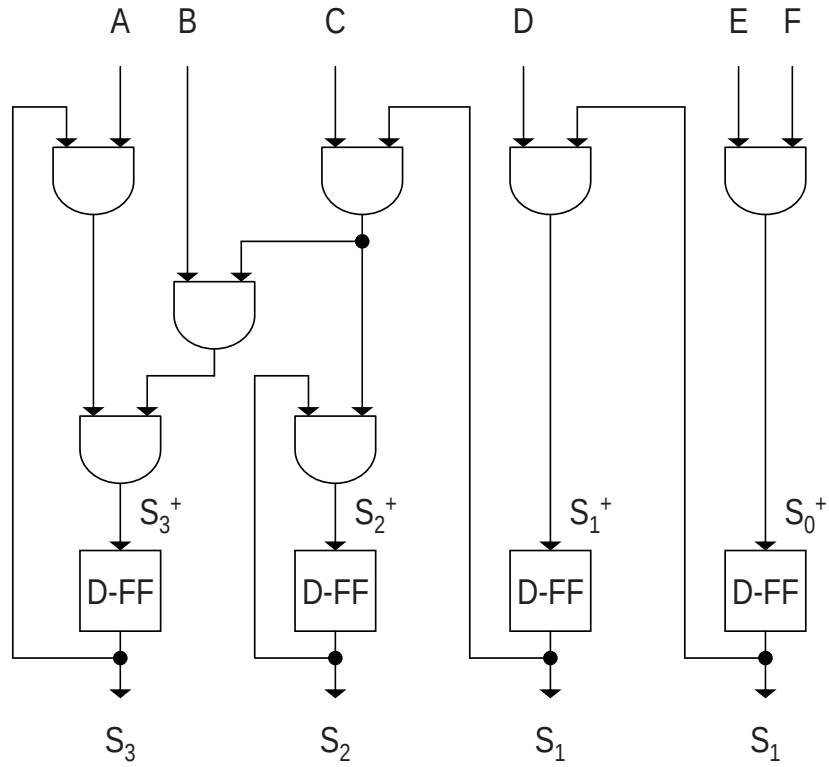


## 1) Delay Analysis

Assume **2 ns** gate delay for 2-AND. Assume the positive edge-triggered flip-flop delay is  $T_{delay_{FF}} = 1 \text{ ns}$  and **setup time** is also  $T_{setup_{FF}} = 1 \text{ ns}$ .



What are **final delays** for the following:

$$S_0^+ =$$

$$S_0 =$$

$$S_1^+ =$$

$$S_1 =$$

$$S_2^+ =$$

$$S_2 =$$

$$S_3^+ =$$

$$S_3 =$$

## 2) Number Representation

(a) For each binary vector below, what does it represent if it is interpreted as a Decimal number in conventional binary form (positive only), a Decimal number in 2's complement form (positive and negative), an Octal number, and a Hexadecimal number?

| Binary | Decimal           |                     | Octal | Hexadecimal |
|--------|-------------------|---------------------|-------|-------------|
|        | conventional form | 2's complement form |       |             |
| 000111 |                   |                     |       |             |
| 001111 |                   |                     |       |             |
| 011111 |                   |                     |       |             |
| 111110 |                   |                     |       |             |
| 111100 |                   |                     |       |             |
| 111000 |                   |                     |       |             |

(b) What is the **minimum** number of bits required to represent the following 2's complement numbers?

-27          num bits =

64          num bits =

### 3) Sequential Logic Implementation and Two-Level Minimization

Consider the following incompletely-specified Moore machine (“\*” indicates don’t cares), and the following state encoding.

State table

| $S_1S_0$ | $S_1^+S_0^+$ |    |    |    | $F$ |
|----------|--------------|----|----|----|-----|
|          | $XY = 00$    | 01 | 10 | 11 |     |
| A        | *            | B  | B  | *  | 1   |
| B        | A            | C  | C  | A  | 1   |
| C        | *            | A  | A  | *  | 0   |

State encoding

| State | $S_1S_0$ |
|-------|----------|
| A     | 00       |
| B     | 01       |
| C     | 10       |

For each function, fill in the K-map, identify the corresponding primes and essential primes, and find the minimum 2-level logic implementation.

|       |   | $F$   |   |
|-------|---|-------|---|
|       |   | $S_0$ |   |
| $S_1$ |   | 0     | 1 |
|       |   |       |   |
|       | 0 |       |   |
|       | 1 |       |   |

|          |    | $S_1^+$ |    |    |    |
|----------|----|---------|----|----|----|
|          |    | $XY$    |    |    |    |
| $S_1S_0$ |    | 00      | 01 | 11 | 10 |
|          |    |         |    |    |    |
|          | 00 |         |    |    |    |
|          | 01 |         |    |    |    |
|          | 11 |         |    |    |    |
|          | 10 |         |    |    |    |

|          |    | $S_0^+$ |    |    |    |
|----------|----|---------|----|----|----|
|          |    | $XY$    |    |    |    |
| $S_1S_0$ |    | 00      | 01 | 11 | 10 |
|          |    |         |    |    |    |
|          | 00 |         |    |    |    |
|          | 01 |         |    |    |    |
|          | 11 |         |    |    |    |
|          | 10 |         |    |    |    |

| $F$   |           |
|-------|-----------|
| Prime | Essential |
|       |           |

| $S_1^+$ |           |
|---------|-----------|
| Prime   | Essential |
|         |           |

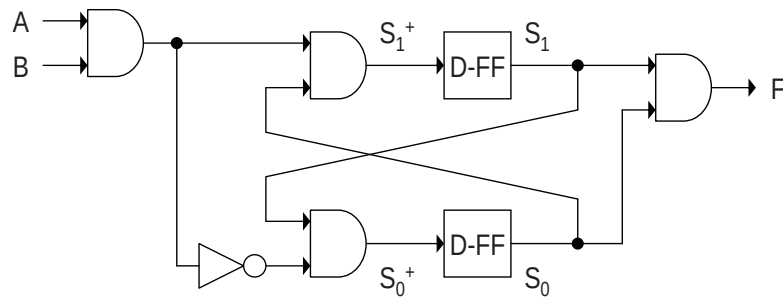
| $S_0^+$ |           |
|---------|-----------|
| Prime   | Essential |
|         |           |

$F =$

$S_1^+ =$

$S_0^+ =$

4) Reverse engineer sequential circuit to finite state machine



a) Fill out the following functions.

$$F =$$

$$S_1+ =$$

$$S_0+ =$$

b) With initial state = **00**, what states, if any, are **not** reachable?

c) Assume initial state is **00**, fill in the following state table. If a state is not reachable, indicate the appropriate **don't cares** (use "\*" or "X" to indicate don't cares).

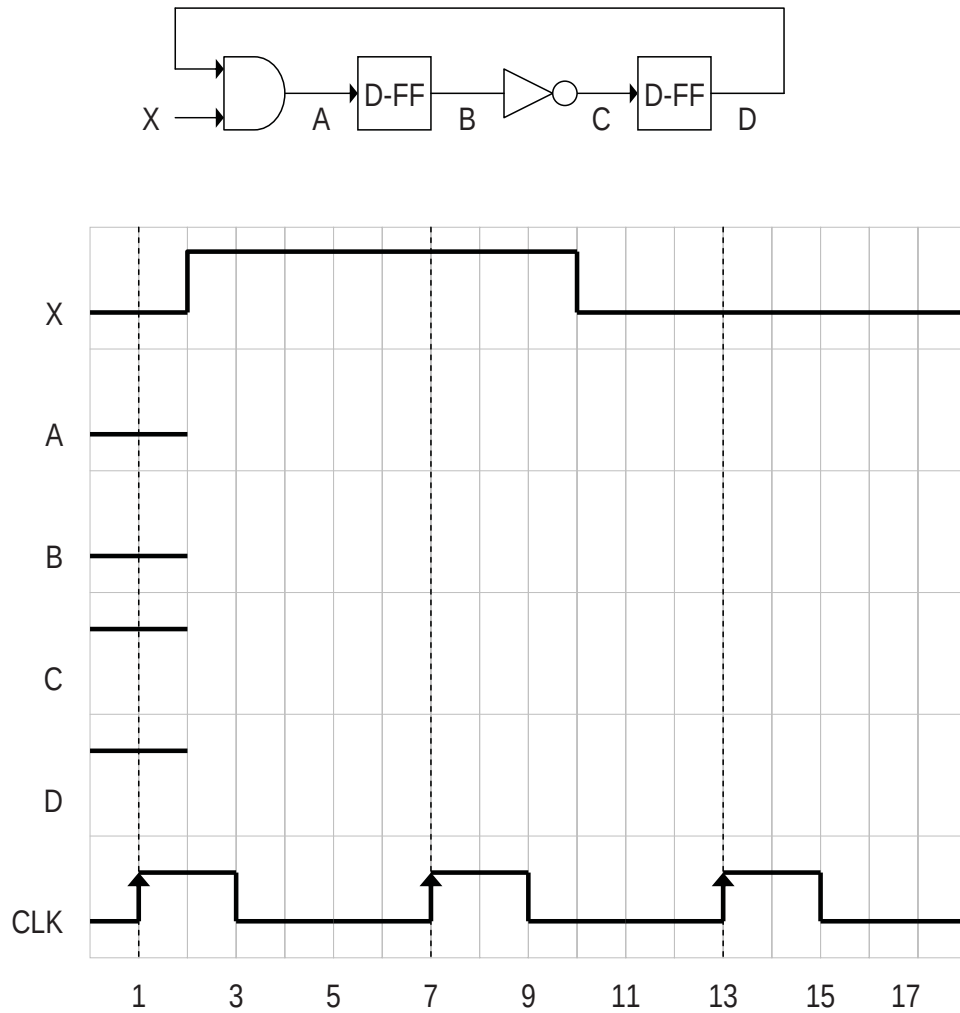
| $S_1S_0$ | $S_1^+S_0^+$ |    |    |    | $F$ |
|----------|--------------|----|----|----|-----|
|          | $AB = 00$    | 01 | 10 | 11 |     |
| 00       |              |    |    |    |     |
| 01       |              |    |    |    |     |
| 10       |              |    |    |    |     |
| 11       |              |    |    |    |     |

## 5) Timing analysis

Consider the following sequential logic diagram.

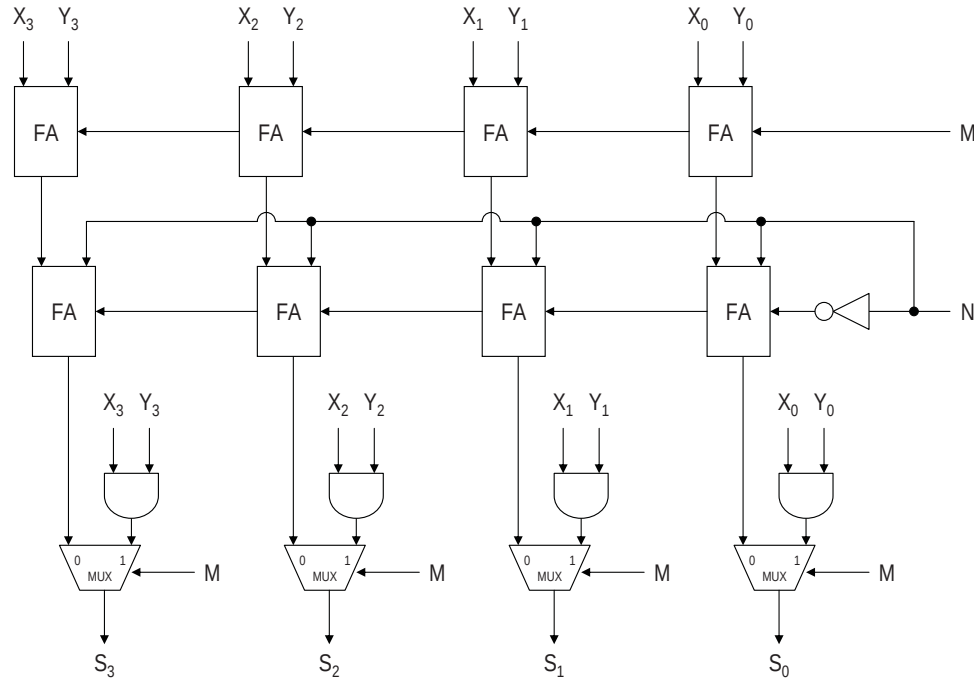
Assume the gate delay for 2-AND is **2 ns** and for INV is **1 ns**. Assume the positive edge-triggered flip-flop delay is  $T_{delayFF} = 1 \text{ ns}$ , and assume the positive edge-triggered flip-flop setup time is also  $T_{setupFF} = 1 \text{ ns}$ .

Fill out the rest of the timing diagram for  $A$ ,  $B$ ,  $C$ , and  $D$ .



## 6) Arithmetic/logic

Consider the following diagram. To avoid drawing many messy wires, the **name** of a wire is used in many cases instead.



Given the following inputs ( $M, N, X_3X_2X_1X_0, Y_3Y_2Y_1Y_0$ ), complete the following logic table.

| $MN$ | $X_3X_2X_1X_0$ | $Y_3Y_2Y_1Y_0$ | $S_3S_2S_1S_0$ |
|------|----------------|----------------|----------------|
| 00   | 0000           |                | 0001           |
| 00   |                | 0100           | 0111           |
| 01   | 0000           | 0001           |                |
| 01   |                | 0000           | 0011           |
| 10   | 1111           |                | 1111           |
| 10   | 0011           | 1101           |                |
| 11   | 0101           | 1010           |                |
| 11   | 0000           | 1111           |                |

## 7) Logic minimization

Minimize the number of literals for each question below. (You can introduce new intermediate logic equations if doing so helps to reduce the number of literals in total.)

a)  $F = \bar{A}BC + AC + \bar{C}$

b)  $F = \bar{A}\bar{B} + AC + BC + AD + BD + \bar{C}\bar{D}$

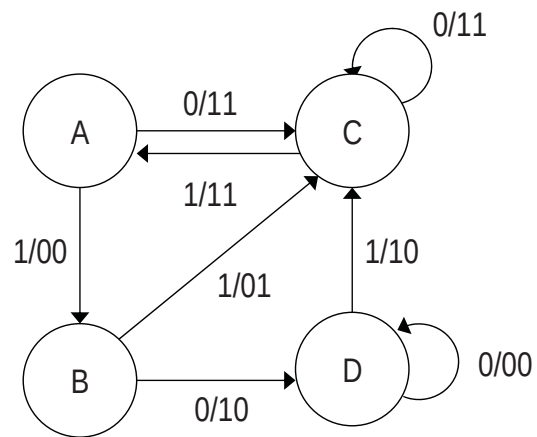
$$G = A\bar{C}\bar{D} + B\bar{C}\bar{D}$$

c)  $F = AB + \bar{A} + \bar{B}$

d)  $F = A(B + C) + \bar{B} + CD$

e)  $F = ABCD + ACD + BCD$

### 8) State machine



For the Mealy machine above, suppose the initial state is **A**. Starting from initial state **A**, what **outputs** are **NOT possible** (e.g. 1/00 means input is 1 and output is 00) when

- a) The input is always **0**:
- b) The input is always **1**: