
**Faculty of Engineering and Technology**
**Computer Systems Engineering Department**

# ADVANCED DIGITAL SYSTEMS DESIGN
# ENCS3310

## Homework 2
## Designing sequential circuits

**Prepared by:**
**Abdel Rahman Shahen 1211753**

**Instructor:**
**Dr. Abdallatif Abuissa**

**Section:** 2
**Date:** 6/7/2024

**1)** Given the following primitive flow table, go through design procedure steps and implement the circuit using SR latches. **[20 points]**

| Stable State | Inputs | | output | Notes |
|---|---|---|---|---|
| | x1 | x2 | Q | |
| a | 1 | 1 | 1 | After c |
| b | 0 | 1 | 0 | After e |
| c | 0 | 1 | 1 | After a, f |
| d | 1 | 0 | 0 | After a, e, f |
| e | 1 | 1 | 0 | After b, d |
| f | 0 | 0 | 1 | After b, c, d |

Step 1: translating primitive flow table

| State/input | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| a | _,_ | c, _ | a,1 | d, _ |
| b | f, _ | b,0 | e, _ | _,_ |
| c | f, _ | c,1 | a, _ | _,_ |
| d | f, _ | _,_ | e, _ | d,0 |
| e | _,_ | b, _ | e,0 | d, _ |
| f | f,1 | c, _ | _,_ | d, _ |

step 2: finding how many state variables we have:

count (a, b, c, d, e, f) = 6 → ceil (ln6/ln2) = 3. So, we need 3 state variables to implement this logic

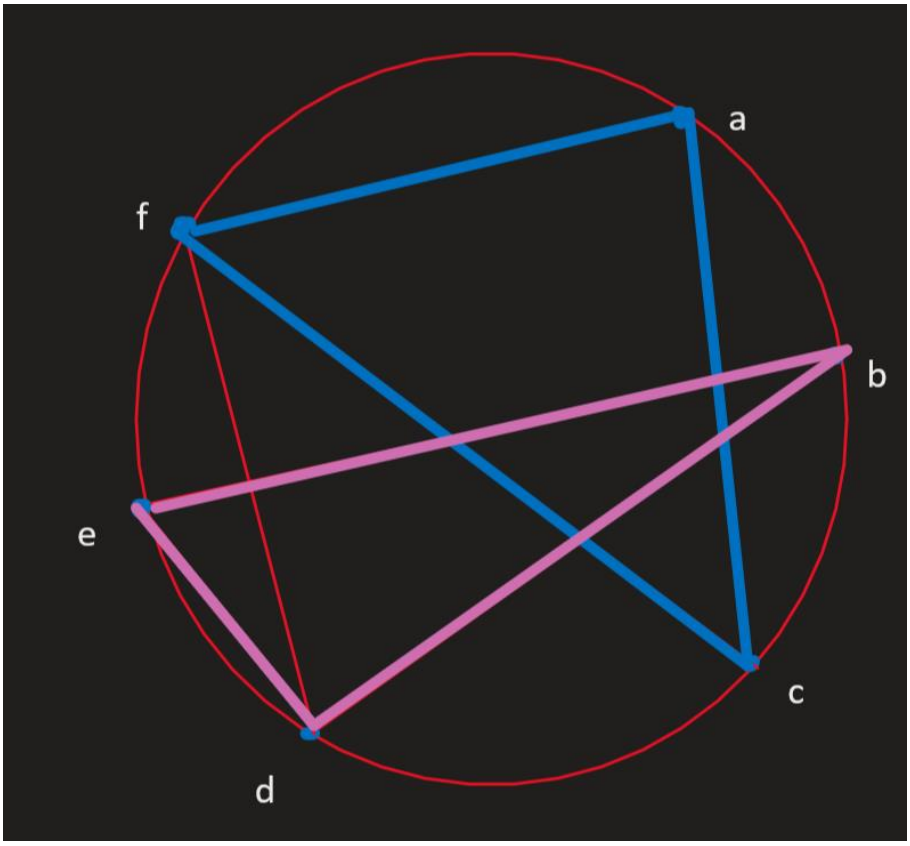
step 3: now let's construct the table of reduction to reduce our state variables

| | | | | | |
|---|---|---|---|---|---|
| b | c,b<br>a,e | | | | |
| c | ✓ | ✓ | | | |
| d | a,e | ✓ | a,e | | |
| e | b,c<br>a,e | ✓ | c,b<br>a,e | ✓ | |
| f | ✓ | b,c | ✓ | ✓ | b,c |
| states | a | b | c | d | e |

compatible pairs:

(a, c) (a, f) (b, d) (b, e) (c, f) (d, e) (d, f)

step 4: now we use merger diagram to see what pairs we still have:




we can take (b, e, d) and (a, f, c) and (d, f)

since we already cover a and f in the first two states we don't need them. Now we remain with states:

(b, e, d) let's call it A

(a, f, c) let's call it B

step 5: creating new table with new states

| state/input | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| b, e, d | f,_ | b,0 | e,0 | d,0 |
| a, f, c | f,1 | c,1 | a,1 | d,_ |

using this table: we can fill the above table:

| State/input | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| a | _,_ | c, _ | a,1 | d, _ |
| b | f, _ | b,0 | e, _ | _,_ |
| c | f, _ | c,1 | a, _ | _,_ |
| d | f, _ | _,_ | e, _ | d,0 |
| e | _,_ | b, _ | e,0 | d, _ |
| f | f,1 | c, _ | _,_ | d, _ |

new table with replaced state names:

| state/input | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| A | B, _ | A,0 | A,0 | A,0 |
| B | B,1 | B,1 | B,1 | A, _ |

our table reduced flow table is now ready

step 6: state assignment and table separating output and next State

output Q table:

| state/input | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | _ | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | _ |

1 → 0: x | 0 → 1: x | 0 → 0: 0 | 1 → 1: 1

| state/input | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | X | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | X |

Q = y

next state table Y:

| state/input | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 |

Y = (x1` x2`) + (y x2)

step 7: SR tables for Y

we need to use SR excitation table:

| y | Y | S | R |
|---|---|---|---|
| 0 | 0 | 0 | x |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | x | 0 |

S

| state/input | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 0 |
| 1 | x | x | x | 0 |

R

| state/input | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 0 | x | x | x |
| 1 | 0 | 0 | 0 | 1 |

S = (X1` X2`)

R = (X1 X2`)

Finally, we have:

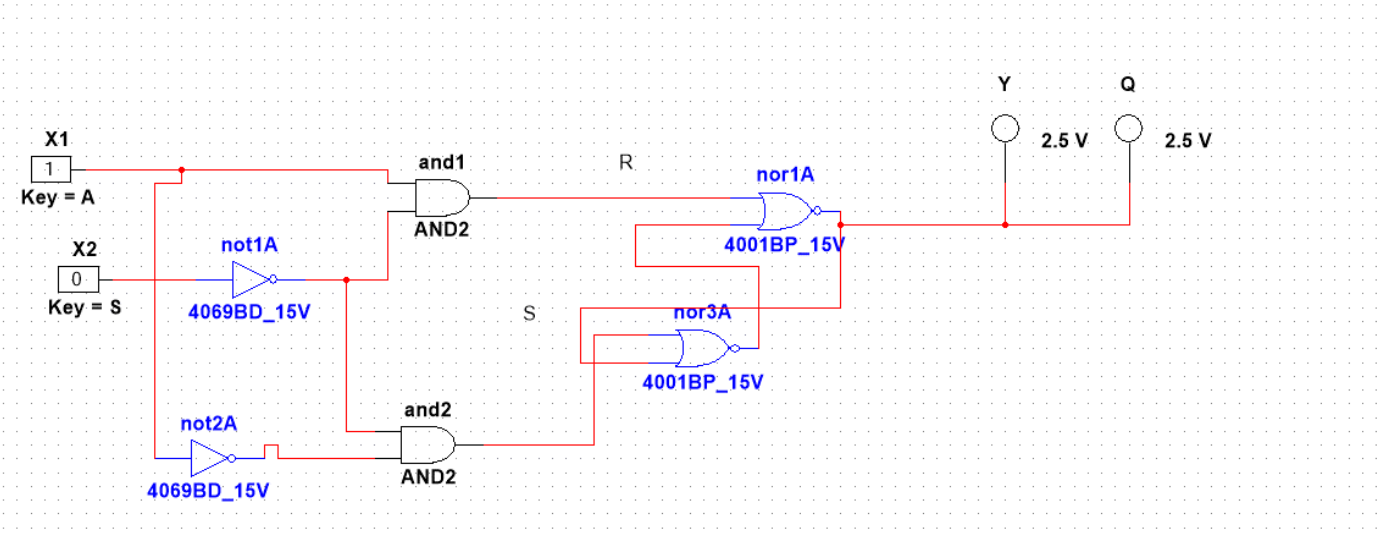
Q = y

SY = (X1` X2`)

RY = (X1 X2`)

step 8: designing circuit

personally, I prefer using NI Multisim 14.1



2) Show the primitive flow table for positive edge T-FF **[10 points]**

| State | Inputs | | Output | Comments |
|---|---|---|---|---|
| | *T* | *C* | *Q* | |
| *a* | 1 | 0 | 0 | After state d or f |
| *b* | 1 | 1 | 1 | After state a or g |
| *c* | 1 | 0 | 1 | After state b or h |
| *d* | 1 | 1 | 0 | After state c or e |
| *e* | 0 | 1 | 0 | After state d or f |
| *f* | 0 | 0 | 0 | After state a or e |
| *g* | 0 | 1 | 1 | After state b or h |
| *h* | 0 | 0 | 1 | After state c or g |

2 laws:

- No more than 1 input change.
- Q(t) = Q(t-1) ` @ C: 0 → 1 else Q(t) = Q(t-1) (no change)

final primitive flow table:

| State/input | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| a | f, _ | _, _ | b, _ | a, 0 |
| b | _, _ | g, _ | b, 1 | c, _ |
| c | h, _ | _,_ | d, _ | c,1 |
| d | _, _ | e, _ | d, 0 | a, _ |
| e | f, _ | e, 0 | d, _ | _, _ |
| f | f,0 | e, _ | _,_ | a, _ |
| g | h, _ | g, 1 | b, _ | _, _ |
| h | h, 1 | g, _ | _, _ | c, _ |