Chapter 7
Simplification of Sequential Circuits

- Tabular Method for State Reduction
- Partitions (OC and SP)
- State Reduction Using Partition
- Choosing a State Assignment

Two states of a sequential system are *equivalent* if, starting in either state, any one input produces the same output and equivalent next states.

- If two states are equivalent, we can remove one of then and have a system with fewer states.
- Usually, systems with fewer states are less expensive to implement.

- Occasionally, we can tell states are equivalent by just inspecting the state table.

A state table

q	q*		z	
	x=0	x=1	x=0	x=1
A	C	B	0	0
B	E	D	0	0
C	A	D	0	1
D	A	B	0	1
E	A	B	0	1

(Back to the example)

Reduced state table

q	q*		z	
	x=0	x=1	x=0	x=1
A	C	B	0	0
B	E	D	0	0
C	A	D	0	1
D	A	B	0	1

Tabular Method for State Reduction

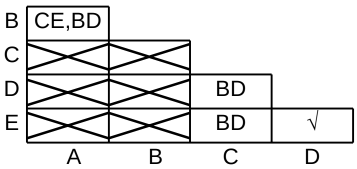
- A technique using a chart with one square for each possible pairing of states.
- Enter in that square
  - ❶ an *X* if those states cannot be equivalent because the outputs are different,
  - ❷ a √ if the states are equivalent (because they have the same output and go to the same state or to each other for each input), and
  - ❸ otherwise the conditions that must be met for those two states to be equivalent.

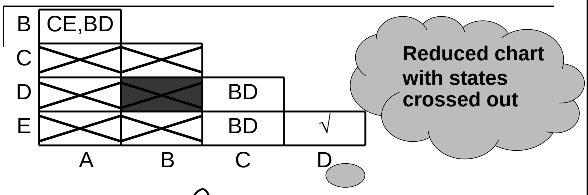
For example (from the previous state table):

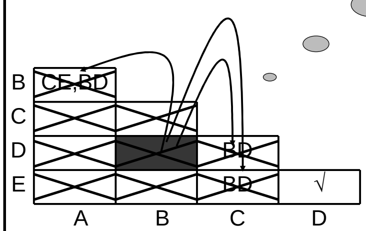
In the AB square, in order for states A and B to be equivalent, they must have the same output for both x=0 and x=1 (which they do) and must go to equivalent states. Thus C must be equivalent to E and B must be equivalent to D.

Those squares contain ***X*** because states A and B have a 0 output for x=1 and states C,D and E have a 1 output.

Finally, the DE square contains √ since both states have the same output and the next state for each input.







q	q*		z	
	x=0	x=1	x=0	x=1
A	C	B	0	0
B	E	D	0	0
C	A	D	0	1
D	A	B	0	1

Example

q	q*		z	
	x=0	x=1	x=0	x=1
A	F	B	0	0
B	E	G	0	0
C	C	G	0	0
D	A	C	1	1
E	E	D	0	0
F	A	B	0	0
G	F	C	1	1

	A	B	C	D	E	F
B	EF,BG					
C	BG,CF	CE				
D	X	X	X			
E	EF,BD	DG	DG	X		
F	√	AE,BG	AC,BG	X	AE,BD	
G	X	X	X	AF	X	X

Example (CONT.)

	A	B	C	D	E	F
B	~~EF,BG~~					
C	~~BG,CF~~	√ CE				
D	X	X	X			
E	~~EF,BD~~	√ DG	√ DG	X		
F	√	~~AE,BG~~	~~AC,BG~~	X	~~AE,BD~~	
G	X	X	X	√ AF	X	X

q	q*		z	
	x=0	x=1	x=0	x=1
A-F	A	B	0	0
B-C-E	B	D	0	0
D-G	A	C	1	1

Partitions

A *partition* on the state of a system is a grouping of the states of that system into one or more blocks. Each state must be in one and only one block.

- For a system with 4 states; A, B, C, and D, the complete list of partition is:

$P_0 = (A)(B)(C)(D)$ $P_4 = (A)(BC)(D)$ $P_7 = (AB)(CD)$ $P_{11} = (ABD)(C)$

$P_1 = (AB)(C)(D)$ $P_5 = (A)(BD)(C)$ $P_8 = (AC)(BD)$ $P_{12} = (ACD)(B)$

$P_2 = (AC)(B)(D)$ $P_6 = (A)(B)(CD)$ $P_9 = (AD)(BC)$ $P_{13} = (A)(BCD)$

$P_3 = (AD)(B)(C)$ $P_{10} = (ABC)(D)$ $P_N = (ABCD)$

State assignments for four states

Any partition with two blocks can be used to assign one of the state variables. Those states in the first block would be assigned 0 and those in the second block 1 (or vice versa).

P_7 through P_{13} meet that requirement.

$P_7 = (AB)(CD)$
$P_8 = (AC)(BD)$
$P_9 = (AD)(BC)$
$P_{10} = (ABC)(D)$
$P_{11} = (ABD)(C)$
$P_{12} = (ACD)(B)$
$P_{13} = (A)(BCD)$

An adequate state assignment

$P_7 = (AB)(CD)$
$P_8 = (AC)(BD)$
$P_9 = (AD)(BC)$

q	q_1	q_2
A	0	0
B	0	1
C	1	0
D	1	1
	P_7	P_8

(a)

q	q_1	q_2
A	0	0
B	0	1
C	1	1
D	1	0
	P_7	P_9

(b)

q	q_1	q_2
A	0	0
B	1	1
C	0	1
D	1	0
	P_8	P_9

(c)

An unsuccessful assignment

- If we try any other pair of two-block partitions, we do not have an adequate state assignment.

$P_7 = (AB)(CD)$
$P_8 = (AC)(BD)$
$P_9 = (AD)(BC)$
$P_{10} = (ABC)(D)$
$P_{11} = (ABD)(C)$
$P_{12} = (ACD)(B)$
$P_{13} = (A)(BCD)$

q	q_1	q_2
A	0	0
B	1	0
C	0	1
D	1	0
	P_8	P_{11}

Identical!

Output consistent (OC)

- Another useful class of partitions for which all of the states in each block have the same output for each of the inputs.
- P_0 (=(A)(B)(C)(D)) is always OC.

P_2 = (AC) (B) (D)

P_5 = (A) (BD) (C)

P_8 = (AC) (BD)

q	q*		z
	x=0	x=1	
A	C	A	1
B	D	B	0
C	A	B	1
D	B	A	0

- Knowing the block of an OC partition and the input is enough info to determine the output (without having to know which state within a block).

Substitution Property (SP) Partition

- For SP partitions, knowing the block of the partition and the input is enough info to determine the block of the next state.

•P_N (=(ABCD)) is always SP since all states are in the same block.

•P_0 (=(A)(B)(C)(D)) is always SP since knowing the block is the same as knowing the state.

q	q*		z
	x=0	x=1	
A	C	A	1
B	D	B	0
C	A	B	1
D	B	A	0

P_7 = (AB) (CD)

P_9 = (AD) (BC)

- If a partition other than P_0 is both SP and OC, then the system can be reduced to one having just one state for each block of that partition.
- That should be obvious since knowing the input and the block of the partition is all we need to know to determine the output, since it is **OC**, and to determine the next state, since it is **SP**).

Properties of Partitions

- **Greater than or equal (≥)**

  $P_a \geq P_b$ iff all states in the same block of P_b are also in the same block of P_a.

P_{10} = (ABC)(D) ≥ P_2 = (AC)(B)(D)

Properties of Partitions (cont.)

- **Product**

  Two states are in the same block of the product P_c iff they are in the same block of *both* P_a and P_b.

$P_{12}P_{13}$ ={(ACD)(B)}{(A)(BCD)} = (A)(B)(CD) = P_6

$P_aP_0 = P_0$ and $P_aP_N = P_a$

***P_0* = (*A*)(*B*)(*C*)(*D*)**

***P_N* = (*ABCD*)**

Properties of Partitions (cont.)

- **Sum**

  Two states are in the same block of the sum P_d if they are in the same block of *either* P_a or P_b or both.

$P_2 + P_5$ = {(AC)(B)(D)} + {(A)(BD)(C)}
= P_8 = (AC)(BD)

$P_a + P_0 = P_a$ and $P_a + P_N = P_N$

***P_0* = (*A*)(*B*)(*C*)(*D*)**

***P_N* = (*ABCD*)**

Finding SP Partitions

- Step 1: สำหรับ state แต่ละคู่ หา SP Partition ที่เล็กที่สุดที่ทำให้ states คู่นั้นอยู่ในกลุ่มเดียวกัน

q	q*	
	x=0	x=1
A	C	D
B	C	D
C	B	D
D	C	A

(AB) → √ ⇒ **P_1 = (AB)(C)(D)**

(AC) → (BC), (BC) → ok ⇒ **P_2 = (ABC)(D)**

(AD) → √ ⇒ **P_3 = (AD)(B)(C)**

(BC) → √ ⇒ **P_4 = (A)(BC)(D)**

(BD) → (AD) → (ABD) ⇒ **P_5 = (ABD)(C)**

(CD) → (BC), (AD) ⇒ **P_N**

Step 1 produces 5 SP partitions.

Finding SP Partitions (cont.)

- Step 2: หา<u>ผลรวม</u>ของ SP Partitions ทั้งหมดจาก step 1 และทำซ้ำกับ SP Partitions ใหม่ที่เกิดขึ้นด้วย

P_1 + $\mathbf{P_2}$ = (ABC)(D) ⇒ P_2 *not needed*
P_1 + P_3 = (ABD)(C) ⇒ P_5
P_1 + P_4 = (ABC)(D) ⇒ P_2
P_1 + $\mathbf{P_5}$ = (ABD)(C) ⇒ P_5 *not needed*
$\mathbf{P_2}$ + P_3 ⇒ P_N *not needed*
$\mathbf{P_2}$ + P_4 = (ABC)(D) ⇒ P_2 *not needed*
$\mathbf{P_2}$ + $\mathbf{P_5}$ = ⇒ P_N *not needed*
P_3 + P_4 = (ABC)(D) ⇒ $\mathbf{P_6}$ ***= (AD)(BC)*** *only one new SP from step 2*
P_3 + $\mathbf{P_5}$ = (ABD)(C) ⇒ P_5 *not needed*
P_4 + $\mathbf{P_5}$ = ⇒ P_N *not needed*

Step 2 here really only requires 3 sums.

Blues are two-block and thus never produce anything new!

Example SP-1

q	q*	
	x=0	x=1
A	C	A
B	D	B
C	A	B
D	B	A

(AB) → (CD) ⇒ **(AB)(CD) = P_1**

(AC) → (AB) → (CD) ⇒ (ABCD) = P_N

(AD) → (BC) ⇒ **(AD)(BC) = P_2**

(BC) → (AD) ⇒ (AD)(BC) = P_2

(BD) → (AB) ⇒ = P_N

(CD) → (AB) ⇒ (AB)(CD) = P_1

Only step 1! No need for step 2.

Example SP-2

Step 1:

q	q* x=0	q* x=1	z
A	C	D	0
B	D	A	0
C	E	D	0
D	B	A	1
E	C	D	1

(AB)→(CD)(AD)=(ACD)→(BCE) ⇒P_N
(AC)→(CE) ⇒**(ACE)(B)(D) = P_1**
(AD)→(BC)→(DE) ⇒**(ADE)(BC) = P_2**
(AE)→ √ ⇒**(AE)(B)(C)(D) = P_3**
(BC)→(ADE) ⇒P_2
(BD)→ √ ⇒**(A)(BD)(C)(E) = P_4**
(BE)→(ACD) →(BCE) ⇒P_N
(CD)→(BE)(AD) →(BC) ⇒P_N
(CE)→ √ ⇒**(A)(B)(CE)(D) = P_5**
(DE)→(BC)(AD) →(ADE) ⇒P_2

Step 1 produces 5 SP partitions.

Example SP-2 (cont.)

Step 2:

From step 1:

P_1=(ACE)(B)(D)
P_2=(ADE)(BC) → 2-block partition (not needed to produce new SP)
P_3=(AE)(B)(C)(D)
P_4=(A)(BD)(C)(E)
P_5=(A)(B)(CE)(D)

P_1+ P_4= **(ACE)(BD) = P_6** → 2-block partition (not needed)
P_3+ P_4= **(AE)(BD)(C) = P_7**
P_3+ P_5= (ACE)(B)(D) = P_1
P_4+ P_5= **(A)(BD)(CE) = P_8**
Now add pairs of these new partitions:
P_7+ P_8= (ACE)(BD) = P_6
If there were new partitions of more than 2 blocks → repeat!

From this example, there are 8 nontrivial SP partitions, of which two are 2-blcok and none are OC.

State reduction using partitions

Any partition that is both **OC** and **SP** can be used to reduce the system to one with one state for each block of that partition.

Just as there is always a unique largest SP partition (P_N), there is always a unique largest OC SP partition. That is the one with the fewest blocks and thus corresponds to the reduced system with the fewest number of states.

(It is possible that P_N is OC; but that is a combinational system, where the output does not depend on the state.)

Example OC/SP-1

q	q*		z
	x=0	x=1	
A	C	D	1
B	C	D	0
C	B	D	1
D	C	A	0

There are 6 SP partitions in this example:
P_1 = (AB)(C)(D)
P_2 = (ABC)(D)
P_3 = (AD)(B)(C)
P_4 = (A)(BC)(D)
P_5 = (ABD)(C)
P_6 = (ABD)(C)
The only SP partition that is OC is P_3.

Thus, this state table can be reduced to one with 3 states (one for each block of P_3).

q	q*		z
	x=0	x=1	
A-D	C	A-D	1
B	C	A-D	0
C	B	A-D	1

Example OC/SP-2

q	q*		z
	x=0	x=1	
A	C	D	0
B	D	A	1
C	E	D	0
D	B	A	0
E	C	D	0

(a)

q	q*		z
	x=0	x=1	
A	C	D	0
B	D	A	1
C	E	D	0
D	B	A	1
E	C	D	0

(b)

The set of SP partitions for these two state tables:

P_1 = (ACE)(B)(D) P_2 = (ADE)(BC)
P_3 = (AE)(B)(C)(D) P_4 = (A)(BD)(C)(E)
P_5 = (A)(B)(CE)(D) P_6 = (ACE)(BD)
P_7 = (AE)(BD)(C) P_8 = (A)(BD)(CE)

Example OC/SP-2 (cont.)

q	q*		z
	x=0	x=1	
A	C	D	0
B	D	A	1
C	E	D	0
D	B	A	0
E	C	D	0

In (a), P_1, P_3 and P_5 are the only OC partitions. Since

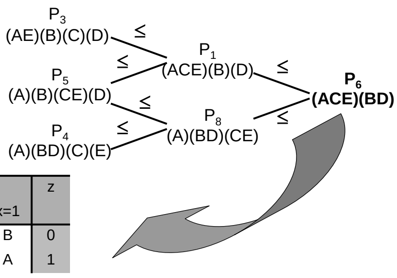
P_3 = (AE)(B)(C)(D) ≤ **P_1 = (ACE)(B)(D)**
P_5 = (A)(B)(CE)(D) ≤ **P_1 = (ACE)(B)(D)**

q	q*		z
	x=0	x=1	
A-C-E	A-C-E	D	0
B	D	A-C-E	1
D	B	A-C-E	0

Example OC/SP-2 (cont.)

q	q* x=0	q* x=1	z
A	C	D	0
B	D	A	1
C	E	D	0
D	B	A	1
E	C	D	0

In (b), P_1, P_3,P_4, P_5, P_6 and P_8 are the only OC partitions. Since



q	q* x=0	q* x=1	z
A	A	B	0
B	B	A	1

Choosing a state assignment

- 2 states ⇒ P_0
- 3-4 states ⇒ 3 state assignments
- 5 states ⇒ 140 state assignments
- 6 states ⇒ 420 state assignments
- 7-8 states ⇒ 840 state assignments
- ...

Example

q	q* x=0	q* x=1	Z
A	B	C	0
B	A	D	1
C	A	D	0
D	B	C	1

SP Partitions: P_1 = (AB)(CD) ⇒ can be used
P_2 = (AD)(B)(C)
P_3 = (A)(BC)(D)
P_2+P_3= P_4 = (AD)(BC) ⇒ can be used

OC: (AC)(BD)

Example (cont.)

q	q_1	q_2
A	0	0
B	0	1
C	1	1
D	1	0

$q_1 \Leftarrow$ P1 (AB)(CD)
$q_2 \Leftarrow$ P4 (AD)(BC)

$z = q_1'q_2+q_1q_2'$
$D_1 = x$
$D_2 = q_2'$

q	q_1	q_2
A	0	0
B	0	1
C	1	0
D	1	1

$q_1 \Leftarrow$ P1 (AB)(CD)
$q_2 \Leftarrow$ OC (AC)(BD)

$z = q_2'$
$D_1 = x$
$D_2 = x'q_1'q_2' + x'q_1q_2 + xq_1'q_2 + xq_1q_2'$

q	q_1	q_2
A	0	0
B	1	1
C	0	1
D	1	0

$q_1 \Leftarrow$ OC (AC)(BD)
$q_2 \Leftarrow$ P4 (AD)(BC)

$z = q_1$
$D_1 = x'q_2' + xq_2$
$D_2 = q_2'$

Conclusions

- The choice of state assignment is more an art than a science.
- Use two-block SP partitions when possible
- When run out of those, OC partition
- And the grouping suggested by other SP partitions (if there are any).