

Lecture 25

Today's Topics:

Reduction of incompletely specified flow tables:

- a) Pair Table
- b) Compatibility
- c) Minimal Closed Covering

Example from last lecture:

$X = 0$	$X = 1$	
A, 0	B, 1	A Carry 0
A, 1	C, 0	B Carry 1
B, 0	C, 1	C Carry 2

Another State Assignment:

- A = 00
- B = 01
- C = 11 (was 10 in previous lecture)

Transition Table:

y_1y_2	$X = 0$	$X = 1$
	00	01
	00	11
	01	11
	dd	dd
	Y_1Y_2	

D Flip-Flop Implementation:

$d_1 = Y_1 = xy_2 \Rightarrow$ Simpler than what we had last time

$d_2 = Y_2 = x + y_1$

Let us now consider an incompletely specified machine and see how we may reduce the number of states:

	$X = 0$	$X = 1$
A	B, d	C, d
B	D, d	E, d
C	F, d	----
D	A, 0	A, 1
E	A, 0	A, 0
F	A, d	----

Consider rows D and F: F can be chosen to be the “same” as D
 (set next state/input under X = 0 to be A, 0 and under
 X = 1 to be A, 1)

D, F become “indistinguishable” (more precisely, compatible)

If so, then C can be made to be the same as B (set next state under X = 1 to be E)

This simple approach results in the following simplified machine:

		X = 0	X = 1
[A]	1	2, d	2, d
[B, C]	2	3, d	4, d
[D, F]	3	1, 0	1, 1
[E]	4	1, 0	1, 0

Is this the best that we can do? How can we formalize this procedure?

Consider a definition like the following:

Definition (Incomplete): States S1, S2 are compatible if for every input sequence, regardless of whether S1 or S2 is the initial state, the output sequence is the same **WHENEVER BOTH OUTPUTS ARE SPECIFIED**.

Note: For completely specified machines compatibility would be the same as indistinguishability.

Example: D and F are compatible because

Input		0	0	0	1	
State	F	A	B	D	A	
Output		d	d	d	1	
State	D	A	B	D	A	
Output		0	d	d	1	=> Compatible output sequences.

Important note: What makes this different for incompletely specified machines?
 Compatibility is NOT transitive(=> not an equivalence relation)

Compatible: [DF] and [FE] are compatible pairs but [DE] is a pair of incompatible states.

The above definition does not capture what happens when we have unspecified states.

Definition (Complete): States S1, S2 are compatible iff:

- 1) The outputs are the same whenever both are specified
- 2) The next states are the same whenever both are specified.

To determine all pairs of compatible states we can use array techniques.

B	BC DE				
C	BF	DF			
D	BA CA	AD AE	AF		
E	BA CA	AD AE	AF	X	
F	BA	AD	AF	*	*
	A	B	C	D	E

The entry in column A, row B means “A and B are compatible if [B,D] and [C,E] are compatible.”

A class of states [S1, S2, S3,.....] is a compatibility class if every pair of states in the class is compatible. For Example, [A,B,C] is a compatibility class.

A compatibility class is MAXIMAL if it is the largest possible (no other compatible class strictly contains it).

In our example, if we stare at our array for a bit we see that:

C1 = [A,B,C,D,F] and C2 = [A,B,C,E,F] are the maximal compatibility classes (MCC's). We will soon see how to systematically find the maximal compatibility classes.

Given the MCC's, we can obtain the following reduced machine:

Reduced Table

	X = 0	X = 1
C1	C1, 0	C2, 1
C2	C1, 0	C2, 0

How did we choose the entries in this table?

- Note that in general:
- 1) It is not necessary to use all MCC's.
 - 2) We need to use enough MCC's to cover all states.
 - Multiple tables reduced machines are possible
 - 3) We need to be careful to maintain compatibility of the next states and output. (closure)

Let us now consider the following example:

	00	01	11	10
A	C, 0	-----	B, d	-----
B	-----	D, 0	F, d	-----
C	E, 1	-----	-----	D, d
D	-----	A, 1	-----	C, d
E	A, d	-----	F, d	-----
F	D, d	E, d	F, d	-----

STEP 1: Construct the Pair Table

B	BF				
C	X	*			
D	*	X	*		
E	AC BF	*	AE	*	
F	CD BF	DE	DE	AE	AD
	A	B	C	D	E

Propagate X's:

B	BF				
C	X	*			
D	*	X	*		
E	X	*	X	*	
F	CD BF	DE	DE	X	AD
	A	B	C	D	E

STEP 2: Find maximal compatibility classes.
(see procedure on pages 408-409 of McCluskey)

Start at the rightmost column and move to the left, each time enlarging an MCC with the state of the current column if this state is compatible with all the states in a class; otherwise choose the largest subset of states that are compatible with the state that corresponds to the current column and form a new class.

Following the previously described procedure, in our example we get the following compatibility classes:

Column

F	F	
E	[EF]	
D	[ED] [EF]	
C	[CF] [CD] [ED] [EF]	
B	[BCF] [CD] [ED] [BEF]	
A	[AD] [CD] [ED] [ABF] [BEF] [BCF]	\leq MCC's

STEP3: Use MCC's to form reduced table.

Can be done by inspection or using a PI Table.

		00	01	11	10
[ABF]	1	2, 0	3, 0	1, d	----
[CD]	2	3, 1	1, 1	----	2, d
[DE]	3	1, d	1, 1	1, d	2, d

- Need: I) Every state is to be covered by at least one class.
 II) For each pair of states in a chosen compatibility class, each of its implied pairs has to be contained in some of the chosen compatibility classes.

E.g , [ABF] [CD] [DE] is a closed covering
 [AB] [F] [CD] [E] is a covering, but not closed.

At Home: Can you find another reduced machine with minimal (3) states?
 Why is this minimal?