Review Lecture

Last year's exam is not representative of what this year's will be like, but the other exam papers are pretty good.

In general proofs don't carry many marks. Exam questions with Karnaugh maps and optimisation tend to carry the most marks, so he recommends focusing on that.

Proofs

Distribution of XOR (`(+)`) with OR

Is the OR operation distributive with respect to XOR?

`A + (B (+) C (+) D (+) E …)` = `(A + B) (+) (A + C) (+) (A + D) (+) (A + E) …`?

Let `A = 1` and examine the result.

- The left-hand side is now 1, as one of the arguments for the OR is 1.
- On the right-hand side, each OR operation reduces to 1.
- So, we have a number of 1s XORed:
    - If we have an even number of 1s, the result is 0 and we have a contradiction.
    - If we have an odd number of 1s, the result is 1 as expected.

So, the property does not always hold.

Associative Law for OR

Is `(A + B + C)` = `A + (B + C)`?

We can generally see that this is true immediately, so he didn't provide a proof.

Associative Law for NOR

Is `(A + B + C)' = ((A + B)' + C)' = (A + (B + C)')'`?

First, remove the outer inverters from each expression.

```
A + B + C = (A + B)' + C = A + (B + C)'
```

Let `C = 1`, as this will simplify the middle expression hugely. Now the middle and right expressions become:

```
1 = A
```

So if `C = 1`, this associative law only holds if A is also 1, meaning it does not generally hold.

The idea here is to plug in a value to simplify the result, aiming to find a contradiction.

Grey-Coding

You can forget about the Quine-McClusky algorithm, it won't be examined. You just need to know the grey-coding ordering for the karnaugh maps.

Moore Machines and Mealy Machines

The course can be separated into combinational circuits and sequential circuits.

- For combinational circuits, the output is only a function of the inputs – you can always plug in the input values to determine the output of the circuit.
- For sequential circuits, the output is a function of input values and the state of the circuit.

Sequential Circuits

Sequential circuits can be broken into two categories:

- Continuous Time (rather than latches/flip-flops, these use an integrator)

- Discrete Time, which can be broken down further:
    - Asynchronous
    - Synchronous
        - Synchronous circuits have one clock, which is connected to everything that requires a clock input

Synchronous circuits can then be broken down into two categories:

- Moore machines
- Mealy machines

Mealy Machines

The output is always a function of the inputs and the current state, so if either changes, the output can change. Unlike Moore machines (below), the output can change at any time.

The next state (the state at the next clock step) is also a function of the inputs and the current state.

Moore Machines

Moore machines are a special case of Mealy machines, where the output is only dependant on the state, and not on the inputs.

Since the state changes only on the clock ticks, the output can only change on the clock ticks. This is the important property of Moore machines.

This Year's Exam

We've only ever looked at Moore and Mealy machines where the input and output are only one bit, so "it's reasonable to assume this year's exam will only contain similar problems."

Sample Question: 2012 Q4

(a)

Since this is a Moore machine, the output table contains the state and the

output. Each state has an output, and the binary value of the state corresponds to the input at last 3 clock steps.

z	Y
0	0
1	0
2	0
3	1
4	0
5	1
6	1
7	1

(b)

A state transition table is a truth table of the `f()` function which determines the next state. On the axes we have the current state and the value of the input.

z \ x	0	1
0	0	1
1	2	3
2	4	5
3	6	7
4	0	1
5	2	3
6	4	5
7	6	7

Notice that each state transition corresponds to a left-shift of the binary value of the state. If we are at `z = 3`, then the last three values we've recorded are 0, 1,

and 1. The next state will be 1, 1, and then whatever we've seen next.

So the left-ward shift corresponds to multiplying by 2 % 8.

Also note that this table makes it look like you can reduce these 8 states to 4 states, but as (e.g.) states 1 and 5 have different outputs. You can simplify it a bit (since 0 and 4 share output, as do 3 and 7), but we won't go into it.

(c)

An alternating input sequence, e.g. 0, 1, 0, 1…

(d)

A circuit realisation would contain 3 flip-flop circuits, as we have 8 states, and the state is represented by the binary number from combining the three circuits.

An alternate argument is that we need to record the value at 3 separate clock ticks, so we need a flip-flop to record the input at each of those 3 clock ticks.

Not part of the question, but the circuit realisation would be 3 flip-flops in series, with the output of each connected to a multiplexor.