Logic Design Example
or
How To Fly a Drone in 5 Easy Steps

ENCE260: Computer Architecture Topic 6

Specificaitons

- Drone has a safety switch: motors won't turn without S = 1.
- To bank left, drive motors 2 & 4 (M2 & M4)
- To fly forward, drive motors 3 & 4 (M3 & M4)

Ciaran Moore
ciaran.moore@canterbury.ac.nz

Design Process

1. Define the function using a **truth table**
2. Convert the truth table to **Karnaugh maps**
3. Derive a **simplified Boolean expression** from each K-map
4. Map Boolean operators to **logic gates**

Ciaran Moore
ciaran.moore@canterbury.ac.nz

Define the Truth Table

State	Inputs	Outputs
	S A B C	M1 M2 M3 M4
Left	1000	
Right	1001	
Forward	1010	
Backward	1011	
Up	1100	
Down	1101	
Safety	0XXX	
Unused States	111X	

Ciaran Moore
ciaran.moore@canterbury.ac.nz

Define the Truth Table

State	Inputs	Outputs
	S A B C	M1 M2 M3 M4
Left	1000	0101
Right	1001	1010
Forward	1010	0011
Backward	1011	1100
Up	1100	1111
Down	1101	0000
Safety	0XXX	0000
Unused States	111X	0000

Ciaran Moore
ciaran.moore@canterbury.ac.nz

Convert to K-Maps

State	Inputs	Outputs
	S A B C	M1 M2 M3 M4
Left	1000	0101
Right	1001	1010
Forward	1010	0011
Backward	1011	1100
Up	1100	1111
Down	1101	0000
Safety	0XXX	0000
Unused States	111X	0000

	$\overline{s}\overline{a}$	$\overline{s}a$	sa	$s\overline{a}$
$\overline{b}\overline{c}$				
$\overline{b}c$				
bc				
$b\overline{c}$				

Ciaran Moore
ciaran.moore@canterbury.ac.nz

Convert to K-Maps

State	Inputs	Outputs
	S A B C	M1 M2 M3 M4
Left	1000	0101
Right	1001	1010
Forward	1010	0011
Backward	1011	1100
Up	1100	1111
Down	1101	0000
Safety	0XXX	0000
Unused States	111X	0000

M1	$\overline{s}\overline{a}$	$\overline{s}a$	sa	$s\overline{a}$
$\overline{b}\overline{c}$	0	0	1	0
$\overline{b}c$	0	0	0	1
bc	0	0	0	1
$b\overline{c}$	0	0	0	0

Ciaran Moore
ciaran.moore@canterbury.ac.nz

Derive Expression

State	Inputs	Outputs
	S A B C	M1 M2 M3 M4
Left	1000	0101
Right	1001	1010
Forward	1010	0011
Backward	1011	1100
Up	1100	1111
Down	1101	0000
Safety	0XXX	0000
Unused States	111X	0000

M1	$\bar{s}\bar{a}$	$\bar{s}a$	sa	$s\bar{a}$
$\bar{b}\bar{c}$	0	0	1	0
$\bar{b}c$	0	0	0	1
bc	0	0	0	1
$b\bar{c}$	0	0	0	0

$$M1 = s\bar{a}c + sa\bar{b}\bar{c}$$

$$M1 = s\left(\bar{a}c + a\bar{b}\bar{c}\right)$$

Ciaran Moore
ciaran.moore@canterbury.ac.nz

Convert to K-Maps

State	Inputs	Outputs
	S A B C	M1 M2 M3 M4
Left	1000	0101
Right	1001	1010
Forward	1010	0011
Backward	1011	1100
Up	1100	1111
Down	1101	0000
Safety	0XXX	0000
Unused States	111X	0000

M2	$\overline{s}\overline{a}$	$\overline{s}a$	sa	$s\overline{a}$
$\overline{b}\overline{c}$	0	0	1	1
$\overline{b}c$	0	0	0	0
bc	0	0	0	1
$b\overline{c}$	0	0	0	0

Ciaran Moore
ciaran.moore@canterbury.ac.nz

Derive Expression

State	Inputs	Outputs
	S A B C	M1 M2 M3 M4
Left	1000	0101
Right	1001	1010
Forward	1010	0011
Backward	1011	1100
Up	1100	1111
Down	1101	0000
Safety	0XXX	0000
Unused States	111X	0000

M2	$\bar{s}\bar{a}$	$\bar{s}a$	sa	$s\bar{a}$
$\bar{b}\bar{c}$	0	0	1	1
$\bar{b}c$	0	0	0	0
bc	0	0	0	1
$b\bar{c}$	0	0	0	0

$$M2 = s\bar{b}\bar{c} + s\bar{a}bc$$

$$M2 = s\left(\bar{b}\bar{c} + \bar{a}bc\right)$$

Ciaran Moore
ciaran.moore@canterbury.ac.nz

Convert to K-Maps

State	Inputs	Outputs
	S A B C	M1 M2 M3 M4
Left	1000	0101
Right	1001	1010
Forward	1010	0011
Backward	1011	1100
Up	1100	1111
Down	1101	0000
Safety	0XXX	0000
Unused States	111X	0000

M3	$\overline{s}\overline{a}$	$\overline{s}a$	sa	$s\overline{a}$
$\overline{b}\overline{c}$	0	0	1	0
$\overline{b}c$	0	0	0	1
bc	0	0	0	0
$b\overline{c}$	0	0	0	1

Ciaran Moore
ciaran.moore@canterbury.ac.nz

Derive Expression

State	Inputs	Outputs
	S A B C	M1 M2 M3 M4
Left	1000	0101
Right	1001	1010
Forward	1010	0011
Backward	1011	1100
Up	1100	1111
Down	1101	0000
Safety	0XXX	0000
Unused States	111X	0000

M3	$\bar{s}\bar{a}$	$\bar{s}a$	sa	$s\bar{a}$
$\bar{b}\bar{c}$	0	0	1	0
$\bar{b}c$	0	0	0	1
bc	0	0	0	0
$b\bar{c}$	0	0	0	1

$$M3 = s\bar{a}\left(b \oplus c\right) + sa\bar{b}\bar{c}$$

$$M3 = s\left(\bar{a}\left(b \oplus c\right) + a\bar{b}\bar{c}\right)$$

Ciaran Moore
ciaran.moore@canterbury.ac.nz

Convert to K-Maps

State	Inputs	Outputs
	S A B C	M1 M2 M3 M4
Left	1000	0101
Right	1001	1010
Forward	1010	0011
Backward	1011	1100
Up	1100	1111
Down	1101	0000
Safety	0XXX	0000
Unused States	111X	0000

M4	$\overline{s}\overline{a}$	$\overline{s}a$	sa	$s\overline{a}$
$\overline{b}\overline{c}$	0	0	1	1
$\overline{b}c$	0	0	0	0
bc	0	0	0	0
$b\overline{c}$	0	0	0	1

Ciaran Moore
ciaran.moore@canterbury.ac.nz

Derive Expression

State	Inputs	Outputs
	S A B C	M1 M2 M3 M4
Left	1000	0101
Right	1001	1010
Forward	1010	0011
Backward	1011	1100
Up	1100	1111
Down	1101	0000
Safety	0XXX	0000
Unused States	111X	0000

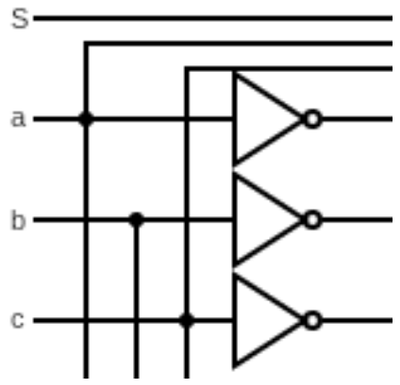
M4	$\bar{s}\bar{a}$	$\bar{s}a$	sa	$s\bar{a}$
$\bar{b}\bar{c}$	0	0	1	1
$\bar{b}c$	0	0	0	0
bc	0	0	0	0
$b\bar{c}$	0	0	0	1

$$M4 = s\bar{b}\bar{c} + s\bar{a}\bar{c}$$

$$M4 = s\bar{c}\left(\bar{b} + \bar{a}\right)$$

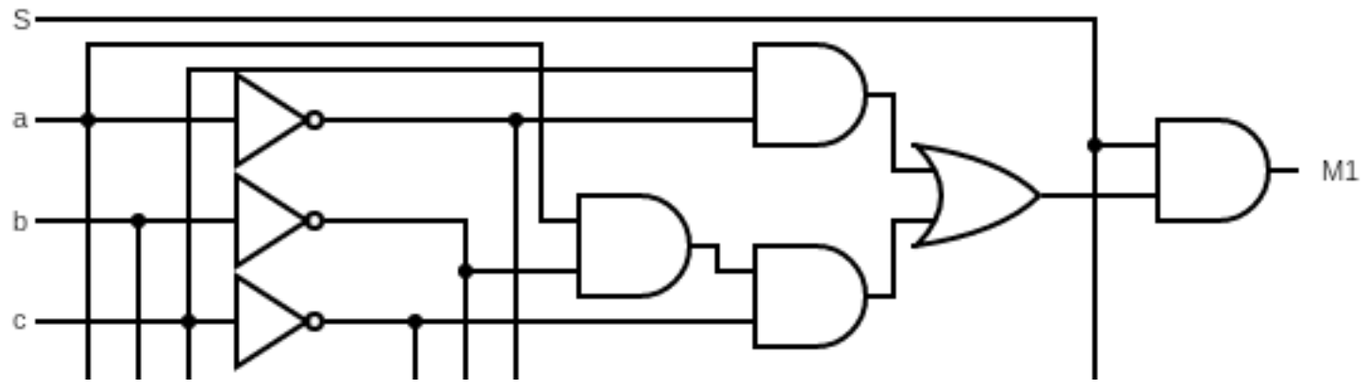
Ciaran Moore
ciaran.moore@canterbury.ac.nz

Build logic circuit from expressions



Ciaran Moore
ciaran.moore@canterbury.ac.nz

Build logic circuit from expressions

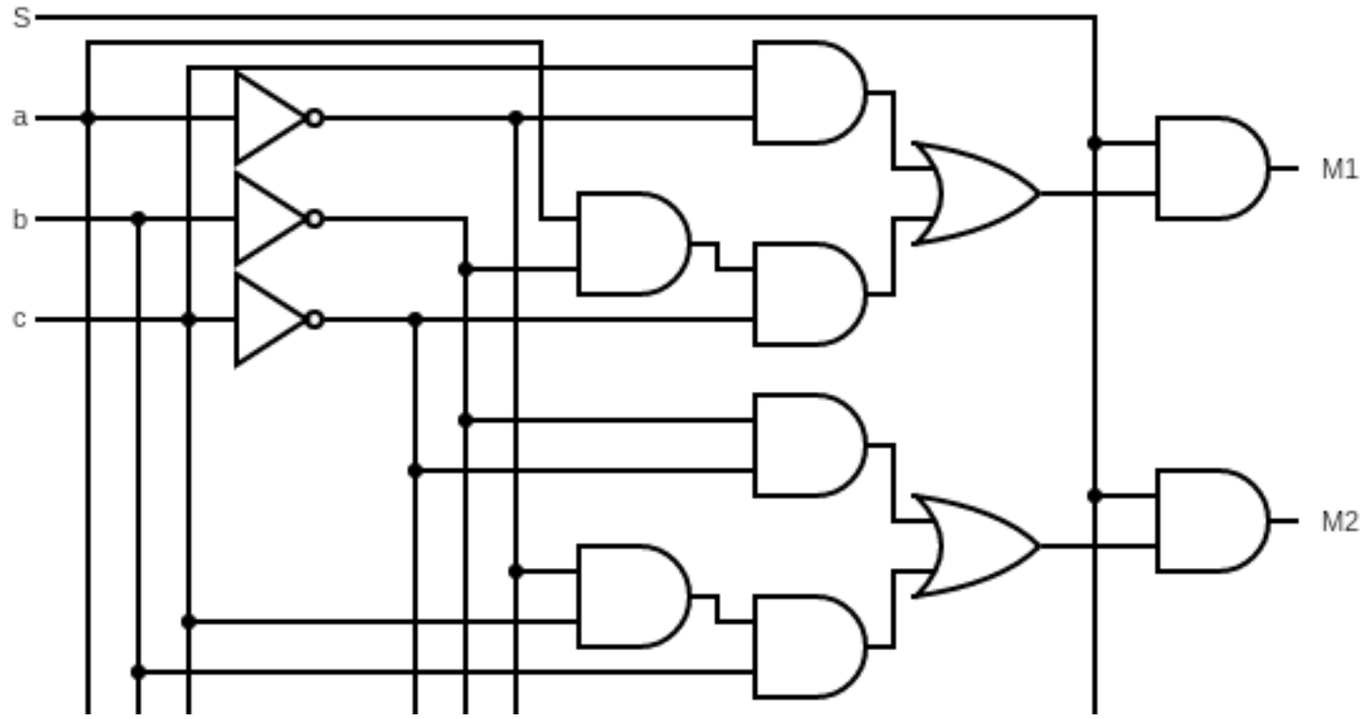


$$M1 = s\left(\overline{a}c + a\overline{b}\overline{c}\right)$$

Ciaran Moore
ciaran.moore@canterbury.ac.nz

Build logic circuit from expressions

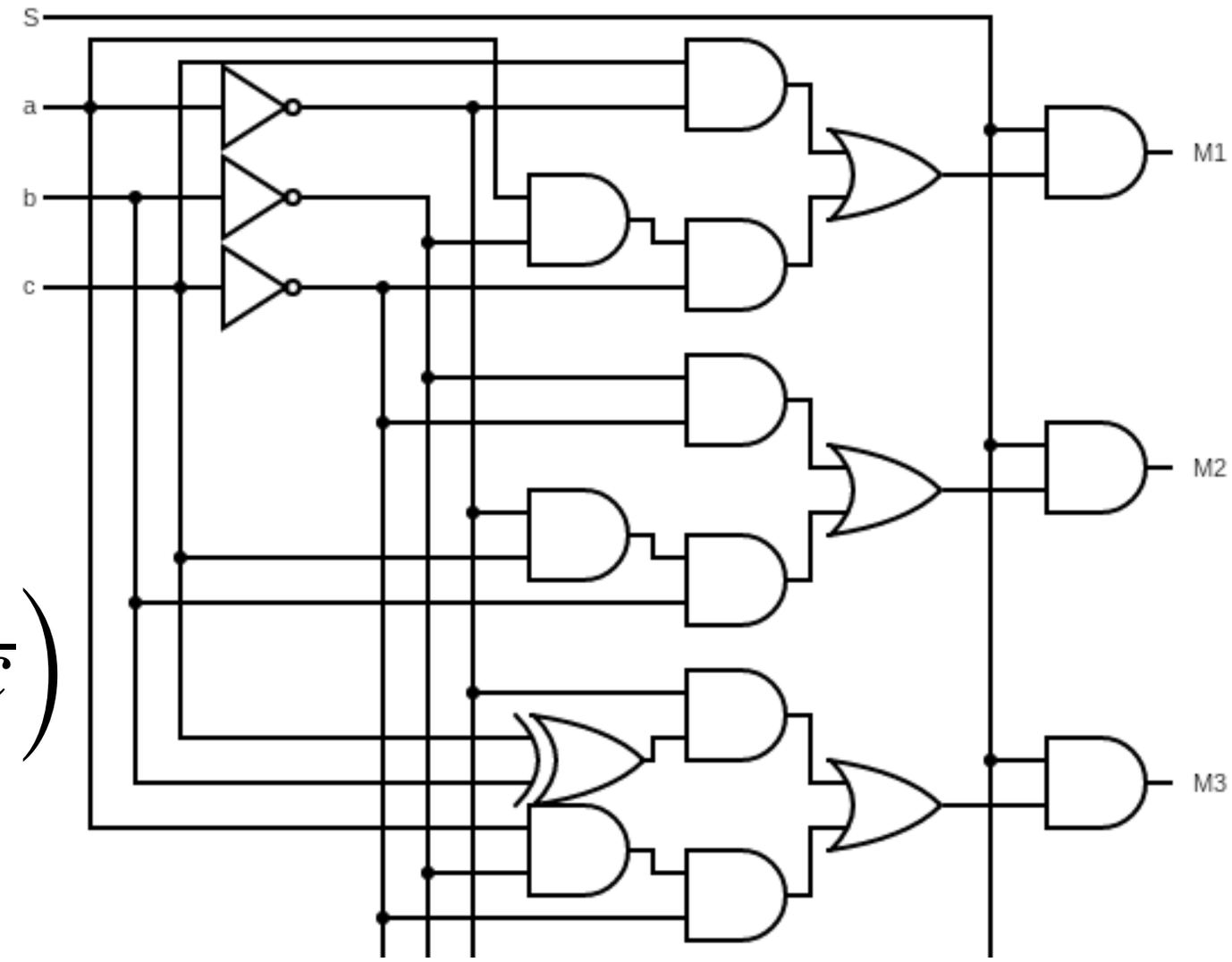
$$M1 = s\left(\overline{a}c + a\overline{b}\overline{c}\right)$$
$$M2 = s\left(\overline{b}\overline{c} + \overline{a}bc\right)$$



Ciaran Moore
ciaran.moore@canterbury.ac.nz

Build logic circuit from expressions

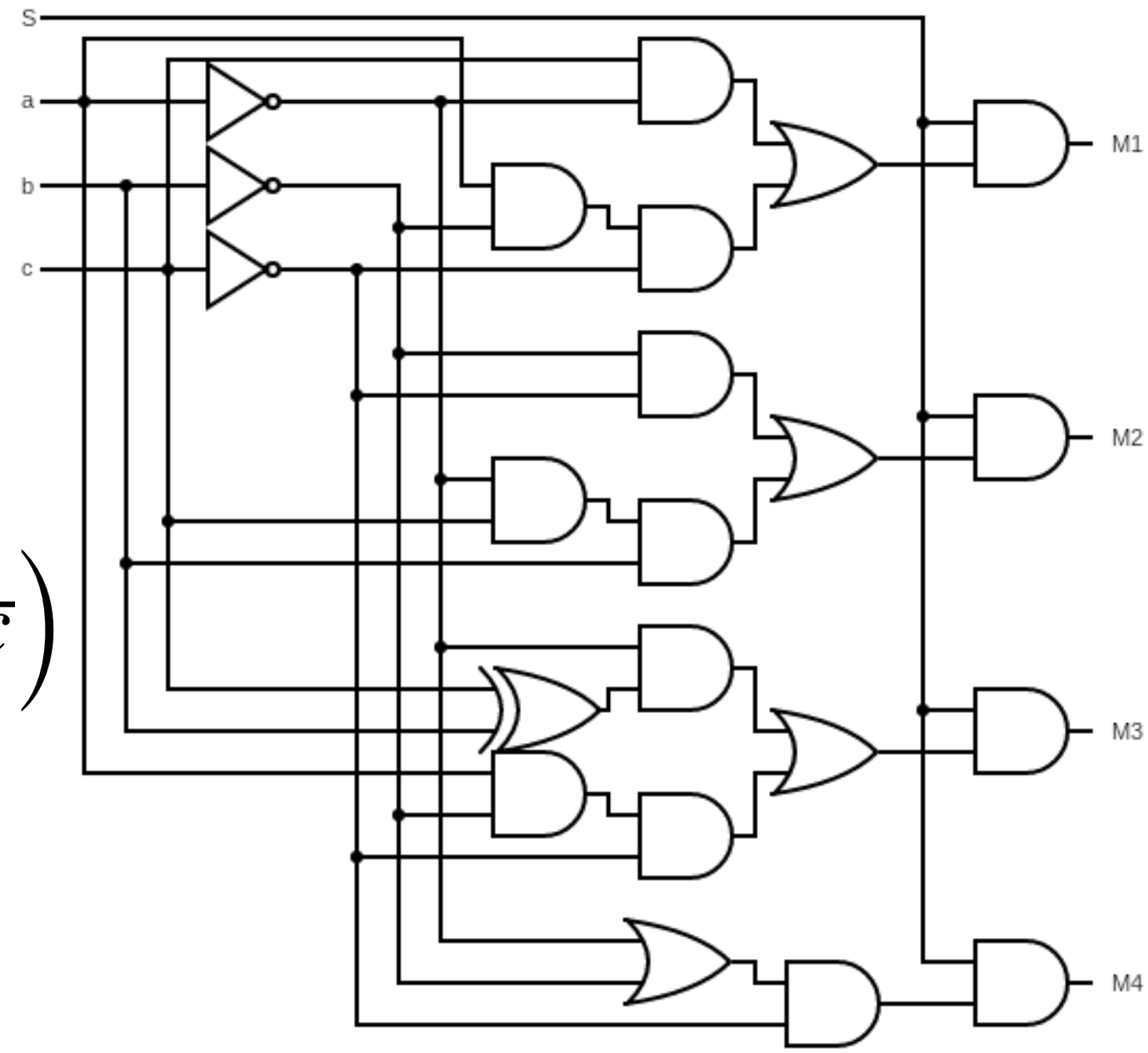
$$M1 = s\left(\overline{a}c + a\overline{b}\overline{c}\right)$$
$$M2 = s\left(\overline{b}\overline{c} + \overline{a}bc\right)$$
$$M3 = s\left(\overline{a}\left(b \oplus c\right) + a\overline{b}\overline{c}\right)$$



Ciaran Moore
ciaran.moore@canterbury.ac.nz

Build logic circuit from expressions

$$M1 = s\left(\overline{a}c + a\overline{b}\overline{c}\right)$$
$$M2 = s\left(\overline{b}\overline{c} + \overline{a}bc\right)$$
$$M3 = s\left(\overline{a}\left(b \oplus c\right) + a\overline{b}\overline{c}\right)$$
$$M4 = s\overline{c}\left(\overline{b} + \overline{a}\right)$$



Ciaran Moore
ciaran.moore@canterbury.ac.nz

Summary

1. Define the function using a **truth table**
2. Convert the truth table to **Karnaugh maps**
3. Derive a **simplified Boolean expression** from each K-map
4. Map Boolean operators to **logic gates**

Ciaran Moore
ciaran.moore@canterbury.ac.nz