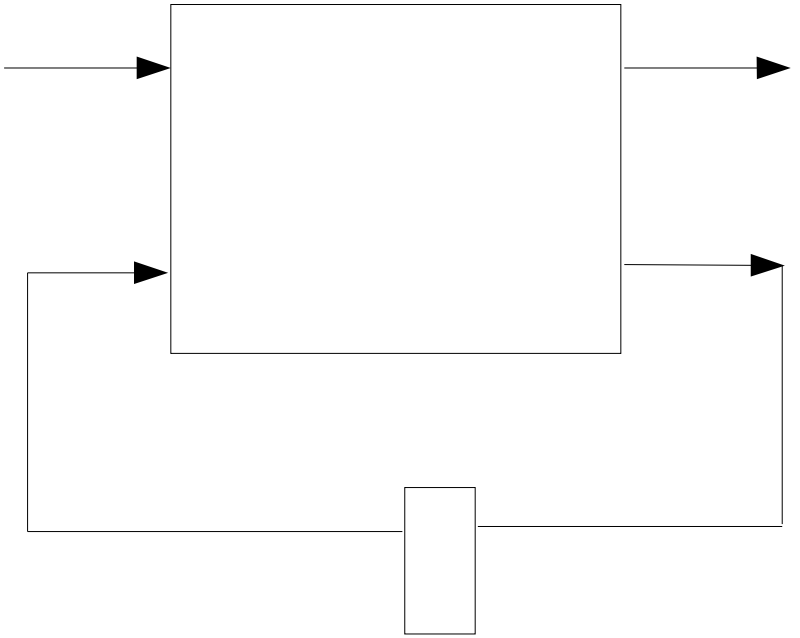
The Symbolic Encoding Problem

- Given a symbolic State machine, encode the symbolic states
- States are symbolic: S0, S1, S2.....
- 4 States: can encode with 2 bits (2 Ffs)
- What code to assign?
  - S0 = 00? 01? 10? 11?
- Encoding affects logic minimization.
- What if inputs & outputs also symbolic?



Inp	PS	NS	Out
I0	S0	S1	O0
I1	S0	S0	O1
I2	S0	S3	O2
.....	..	..	....

Encoding and Symbolic Minimization

- 3 Modes, use 2 bits
- 4 Opcodes, use 2 bits
- And so on...
- How about symbolic minimization?
- CNTA = Index * (AND + OR + JMP + ADD) =
- CNTB =
-
- Minimize symbolically & then do encoding?

_

Mode	Opcode	Output
Index	AND	CNTA
Index	OR	CNTA
Index	JMP	CNTA
Index	ADD	CNTA
Dir	AND	CNTB
Dir	OR	CNTB
Dir	JMP	CNTC
Dir	ADD	CNTC
Ind	AND	CNTB
Ind	OR	CNTD
Ind	JMP	CNTD
Ind	ADD	CNTC

Encoding and Symbolic Minimization

- Mode = 3 valued input
- Opcode = 4 valued input
- Multi-valued logic.
- Boolean algebra = set {0, 1} w/ operations
- True = U, False = NUL
- MV-var: {0, 1, 2, 3,...}
- U = {And, Or, Jmp, Add} => And+Or+Jmp+Add = 1
- Positional cube for MV minimization?

Mode	Opcode	Output
Index	AND	CNTA
Index	OR	CNTA
Index	JMP	CNTA
Index	ADD	CNTA
Dir	AND	CNTB
Dir	OR	CNTB
Dir	JMP	CNTC
Dir	ADD	CNTC
Ind	AND	CNTB
Ind	OR	CNTD
Ind	JMP	CNTD
Ind	ADD	CNTC

Positional Cube (Again?)

- NUL: 00
- 0: 10
- 1: 01
- DC: 11
- “1” in each position
- 11 = full literal

- 4-Valued Var: Opcode
- 4 positions
- NUL: 0: 0000
- AND: 1:
- OR: 2:
- JMP 3:
- ADD: 4:
- X ={JMP + ADD} =
- DC+ full literal =

Encoding and Symbolic Minimization

Mode	Opcode	Output	Mode	Opcode	Output
Index	AND	CNTA	100	1000	1000
Index	OR	CNTA	100	0100	1000
Index	JMP	CNTA	100	0010	1000
Index	ADD	CNTA	100	0001	1000
Dir	AND	CNTB	010	1000	0100
Dir	OR	CNTB	010	0100	0100
Dir	JMP	CNTC	010	0010	0010
Dir	ADD	CNTC	010	0001	0010
Ind	AND	CNTB	001	1000	0100
Ind	OR	CNTD	001	0100	0001
Ind	JMP	CNTD	001	0010	0001
Ind	ADD	CNTC	001	0001	0010

Symbolic Minimization

- Simplify Each output Separately
  - Keep Output bits fixed, simplify symbolic inputs
    - The INPUT ENCODING PROBLEM
    - Well understood….
  - Keep input bits fixed, simplify symbolic outputs
    - OUTPUT ENCODING Problem
    - “Somewhat understood”
  - Simultaneously simplify both symbolic inputs and symbolic outputs
    - Input-Output Encoding (Tough!)
  - Out focus: Input Encoding, dichotomy-based
    - DIET: General encoding problem, also applied to FSM. Relationship to logic minimization

Encoding and Symbolic Minimization

Mode	Opcode	Output
100	1000	1000
100	0100	1000
100	0010	1000
100	0001	1000
010	1000	0100
010	0100	0100
010	0010	0010
010	0001	0010
001	1000	0100
001	0100	0001
001	0010	0001
001	0001	0010

Mode	Opcode	Output
100	1111	1000
010	1100	0100
001	1000	0100
010	0011	0010
001	0001	0010
001	0110	0001

Encode Minimized MV-Cover

- Encoding Problem:
  - Each mv-literal implemented as a “bv-cube”
  - State “variable”, has 4 literals (s0, s1, s2, s3)
  - S0: !r1 !r2; S1: !r1 r2; S2: r1 !r2; S3: r1 r2
- Encoding constraints:
  - Preserve cardinality:
  - Each mvi-implicant should have a distinct encoding -> each mv-literal corresponds to only one bv-cube!
- Encoding constraint translates to:
- Assign Binary codes such that their super cubes DO NOT Intersect!
- Otherwise, incorrect circuit.