

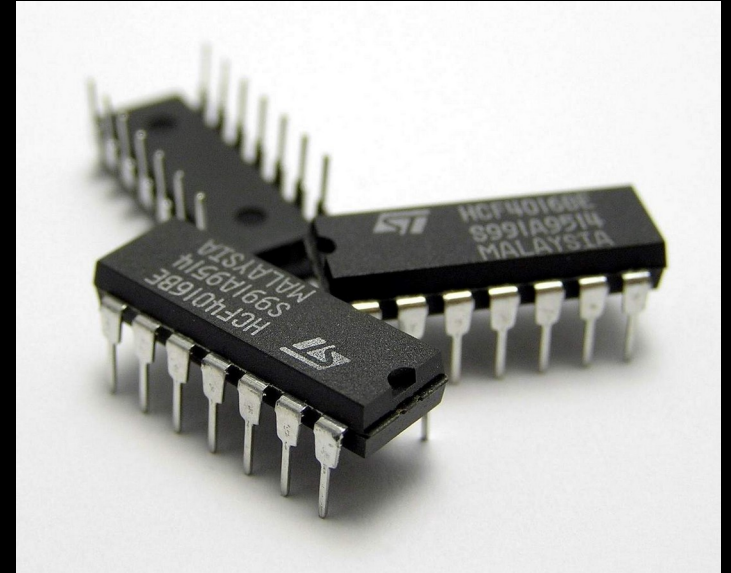
NandSim

Digital Logic Circuit Simulator

Svendeprøve
Simon From Jakobsen
Juni 2026

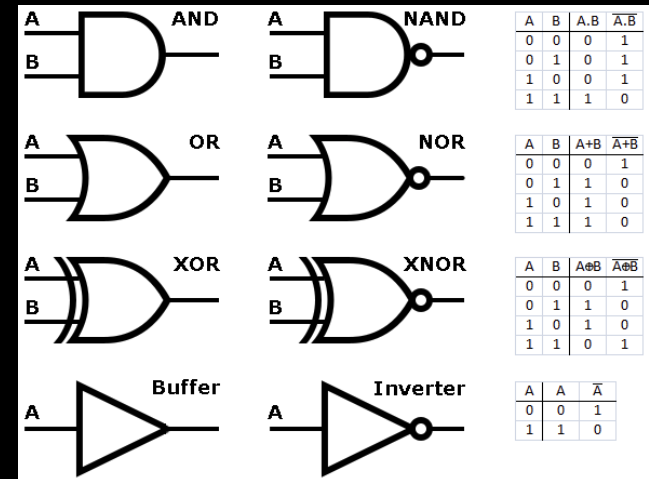
Integrerede kredsløb

- Elektroniske kredsløb
 - ledninger, komponenter, strøm, spænding
- Fotolitografi
- Formål, fx digital-logik
- Samlet pakke
 - inkluderer ofte software



Digital-logik

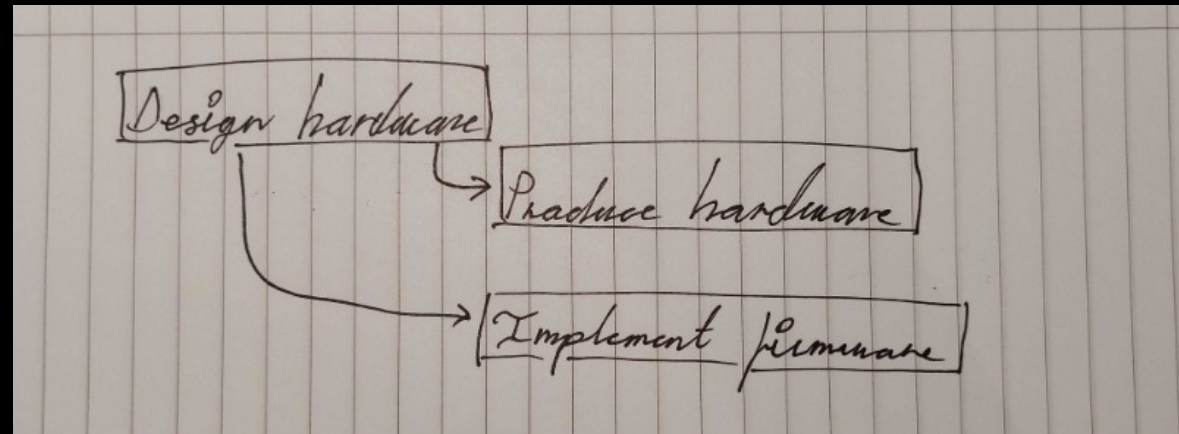
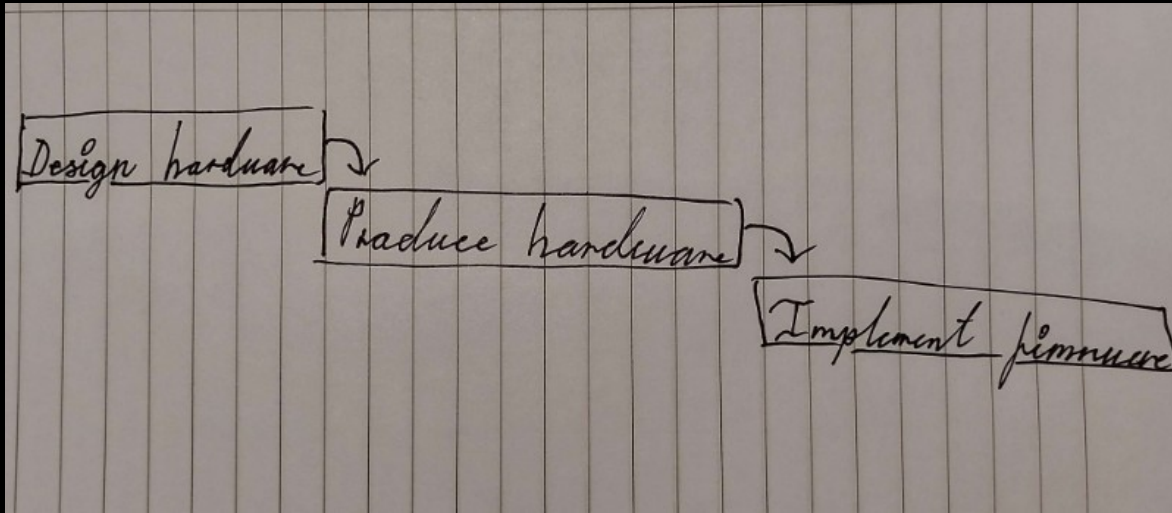
- Digital elektronik
- Binære værdier (1 eller 0, sandt eller falsk...)
- Logik-gates: *AND*, *OR*, *NOT*, ...
- Boolesk algebra: \wedge , \vee , \neg , ...
- Komponering
- Computing



Integreret software

- Firmware
- Drivers
- HAL
- RTOS
- Microcode
- Dataset





nandsim

nandsim — Mozilla Firefox Private Browsing

Private browsing

nandsim.sjfa.dk

150%

Save

Rename

Close

Toolbar

input

output

and

or

not

(Unnamed)

+

input (off)

input (off)

and

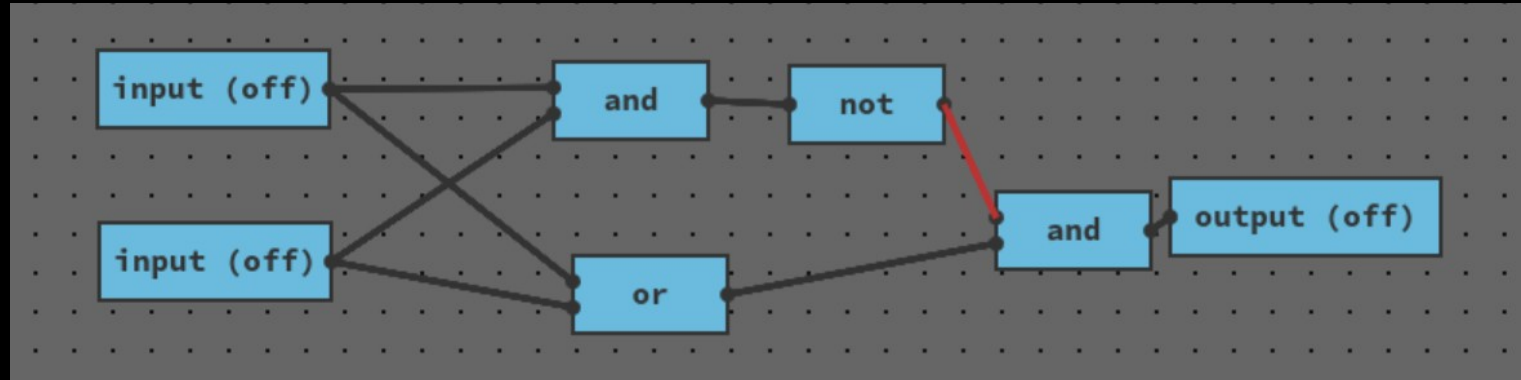
output (off)

```
graph LR; I1[input (off)] --> A[and]; I2[input (off)] --> A; A --> O[output (off)];
```

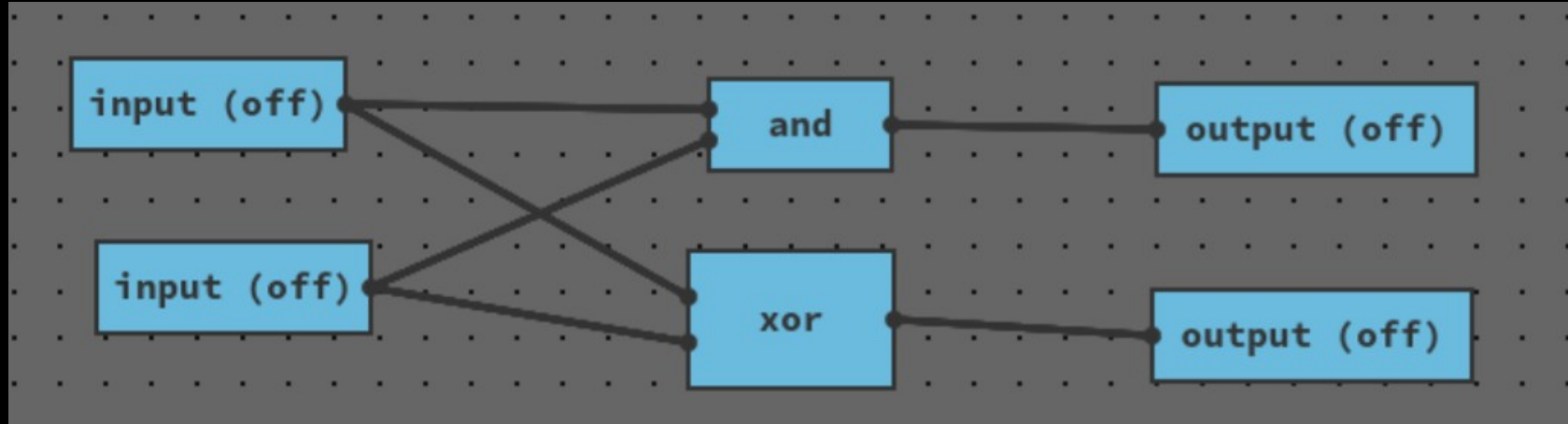
<https://nandsim.sfja.dk/>

Video af demonstration

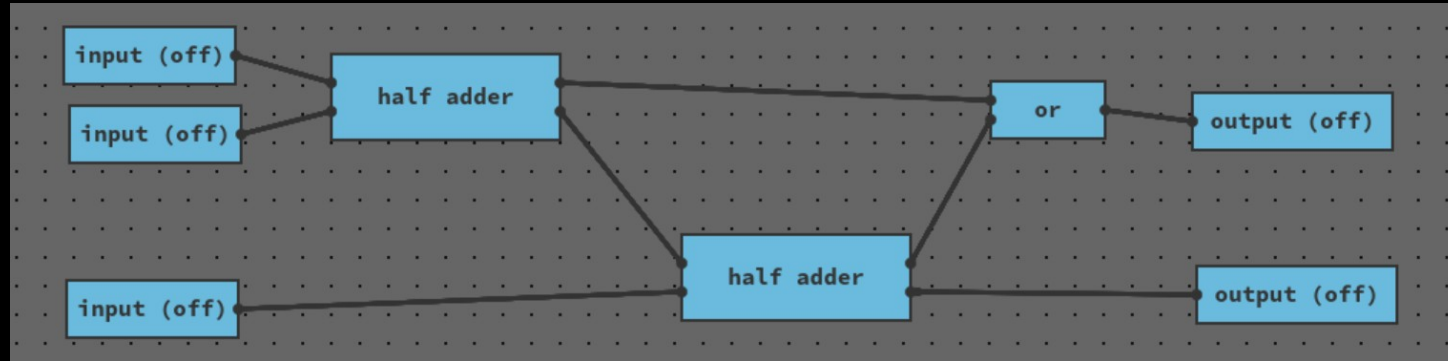
XOR



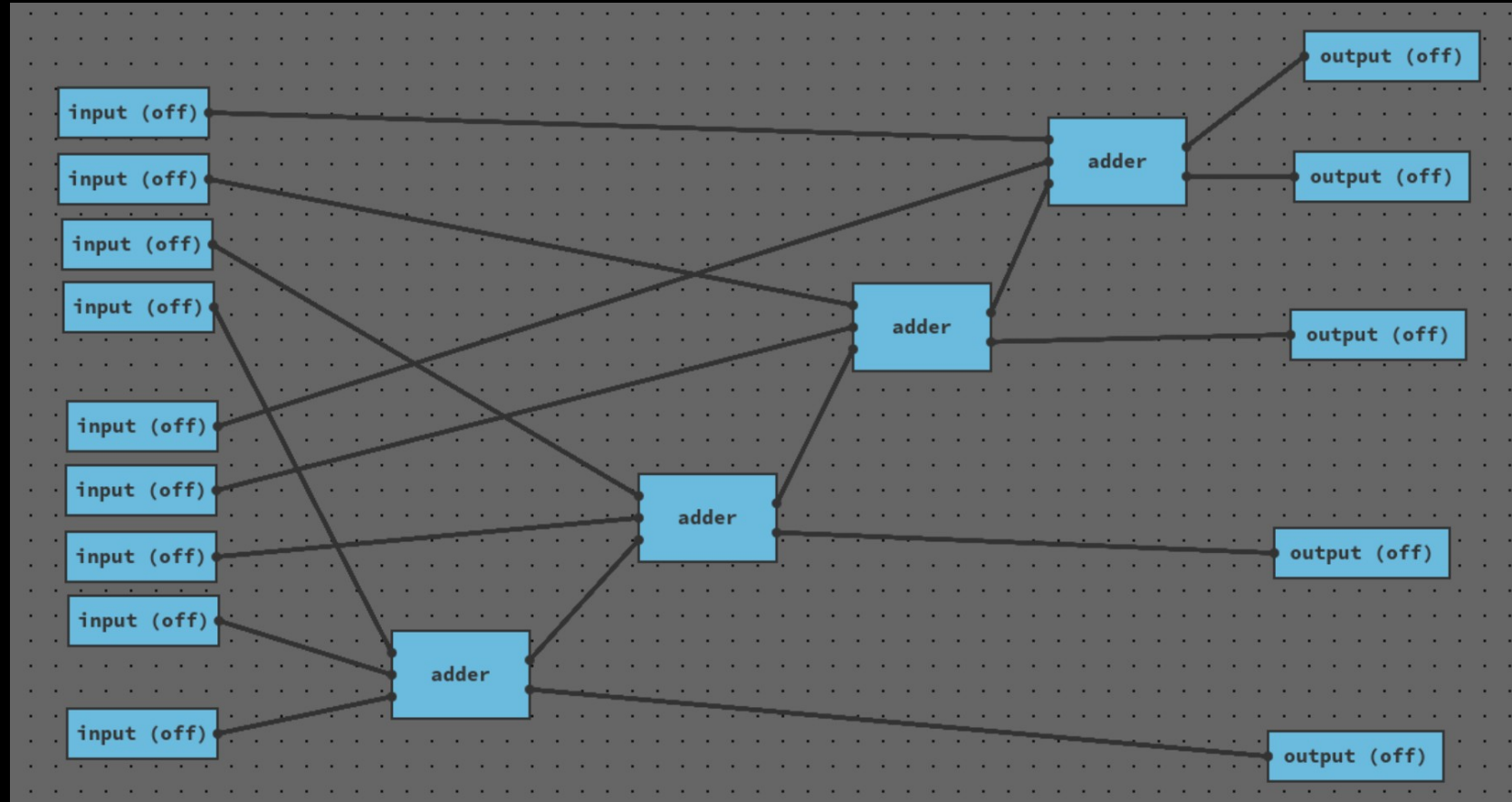
HALF ADDER

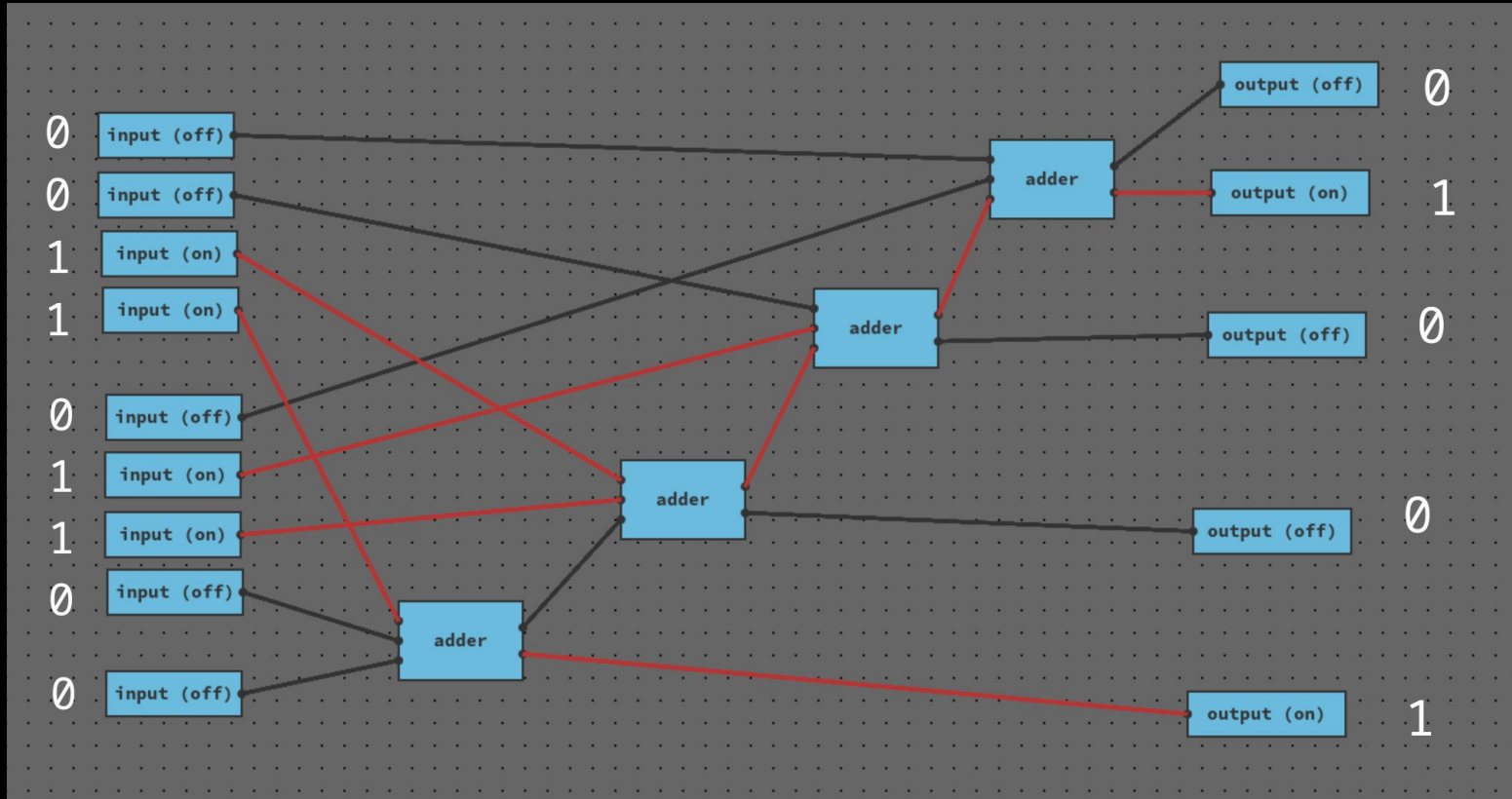


FULL ADDER



4-bit ADDER





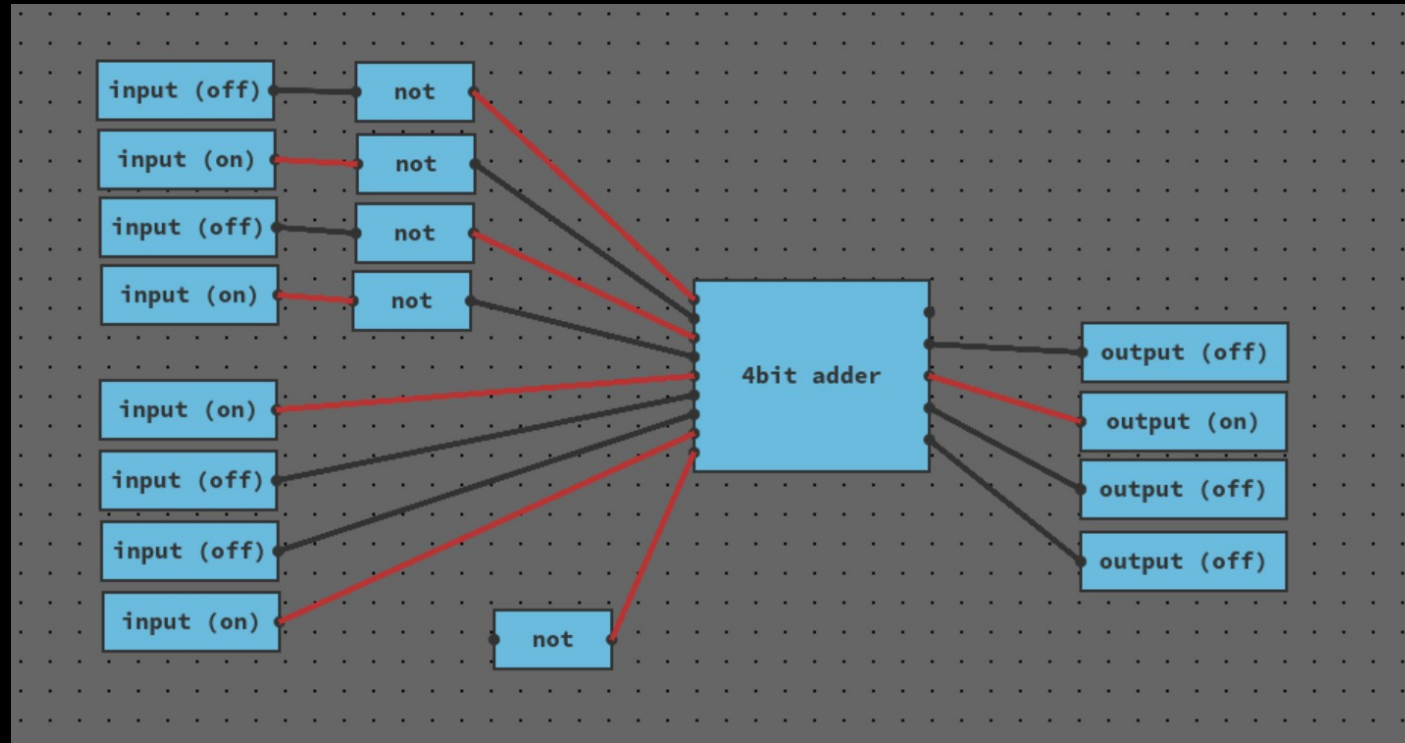
$$3 = 0011$$

$$6 = 0110$$

$$\begin{array}{r} 0011 \\ + 0110 \\ \hline = 1001 \end{array}$$

$$1001 = 9$$

$$3 + 6 = 9$$



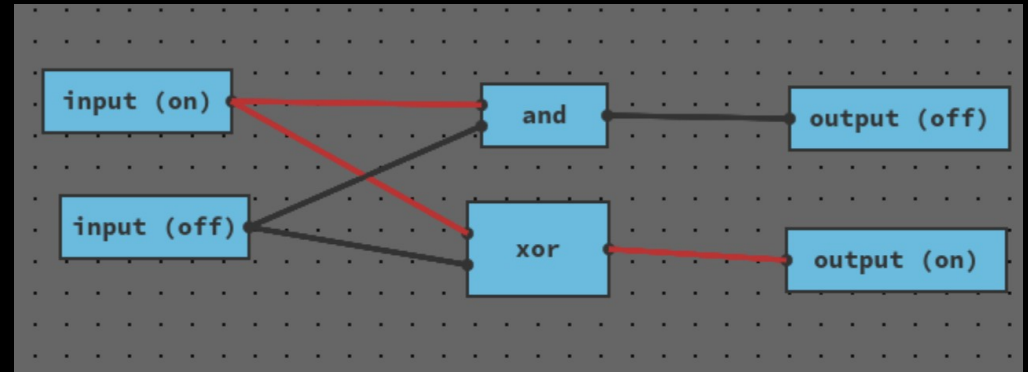
5 = 0101
9 = 1001

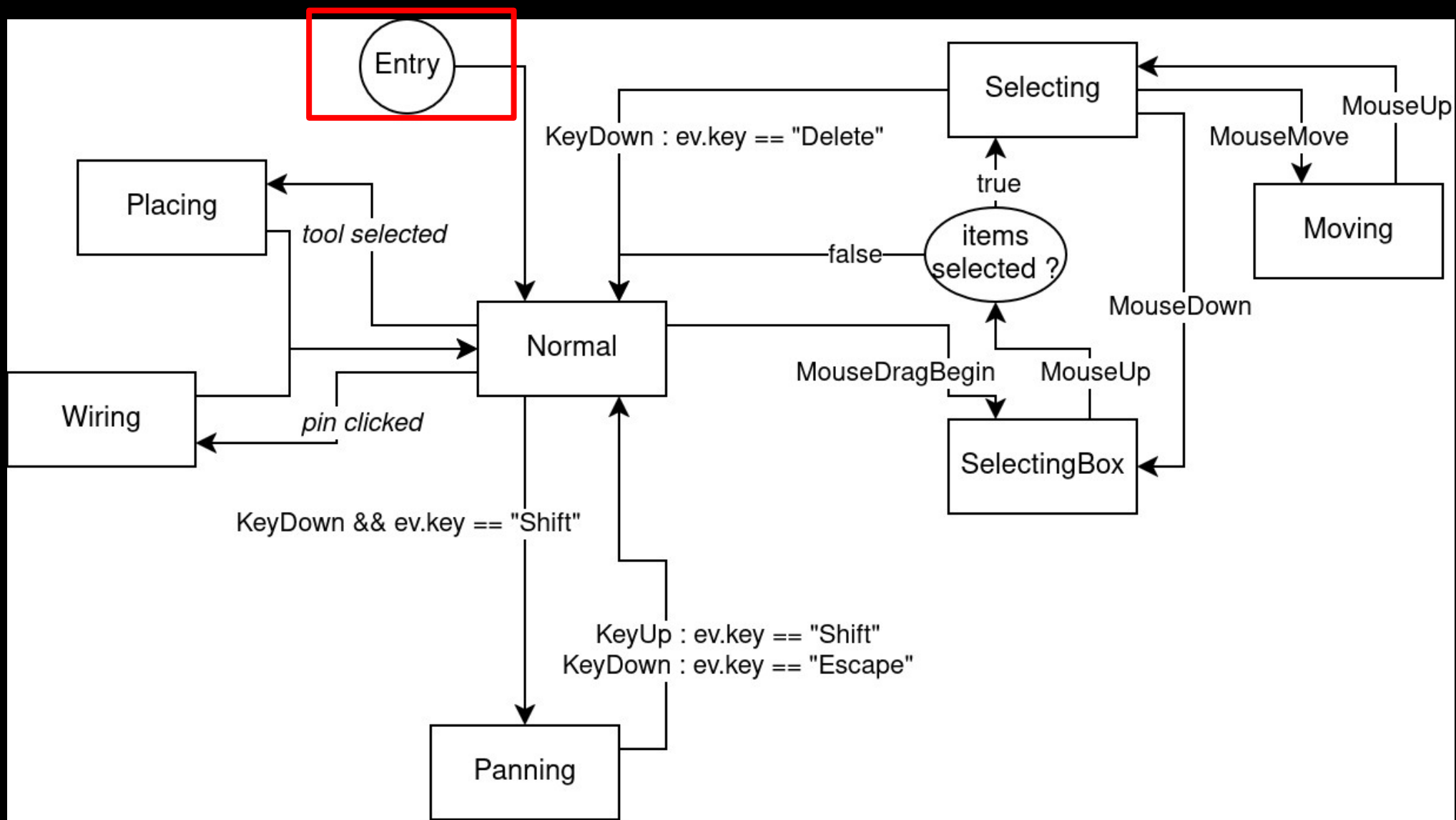
9 - 5 = 4

4 = 0100

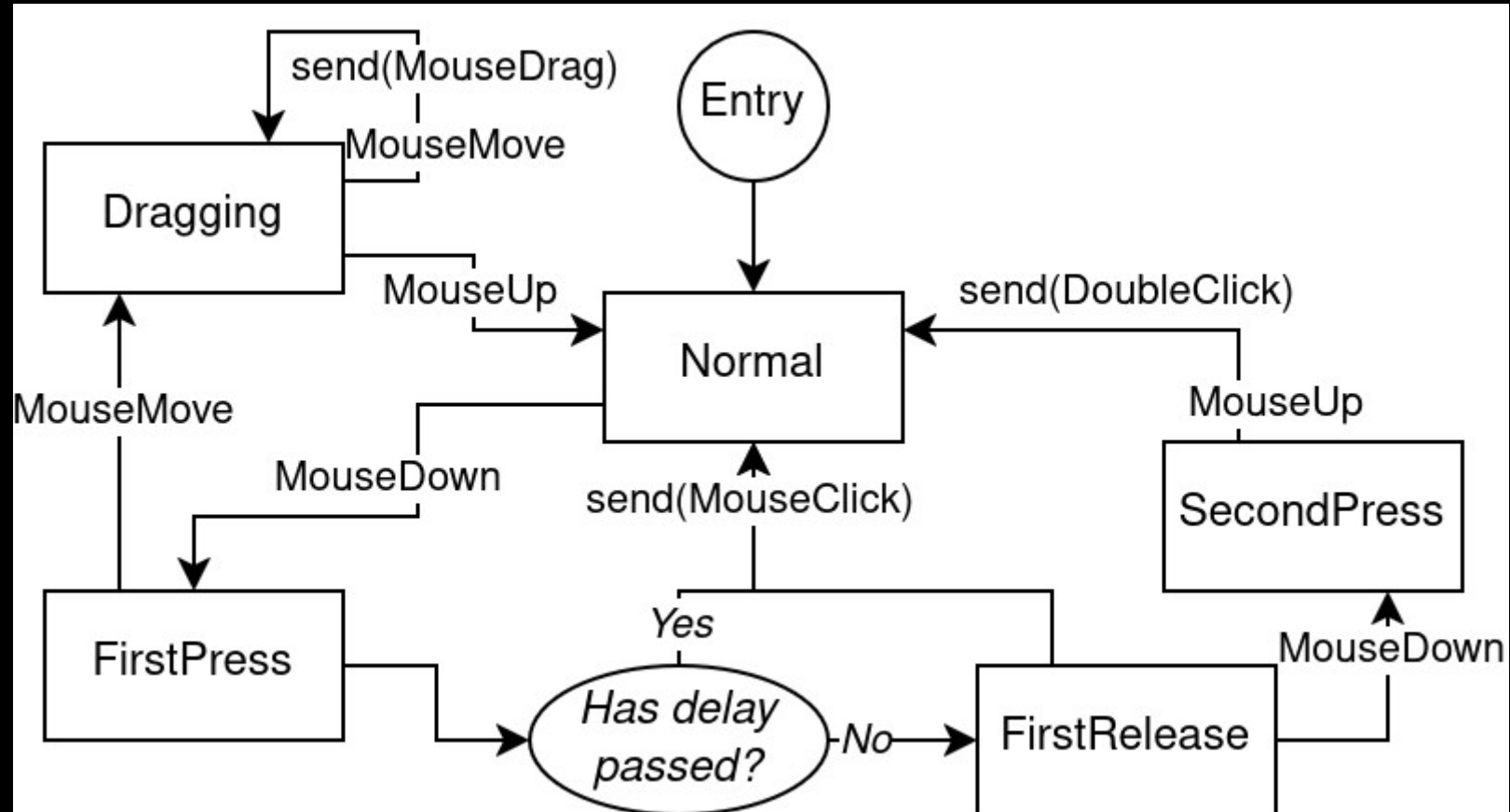
Editor'en

- Visuel
- Kamera
- Selection
- Interaktion
- Flere modes

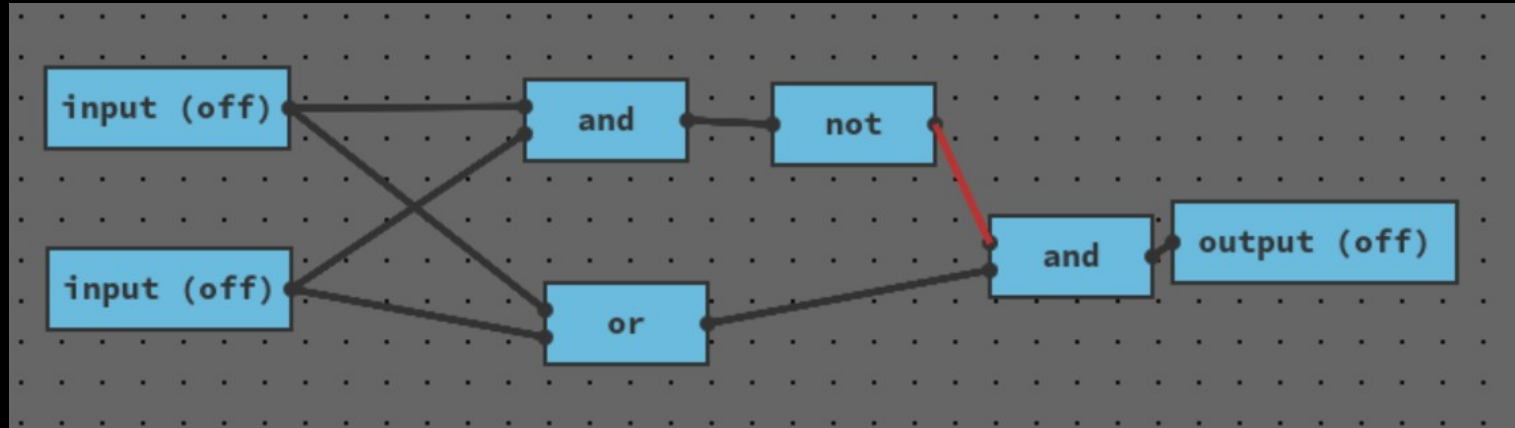




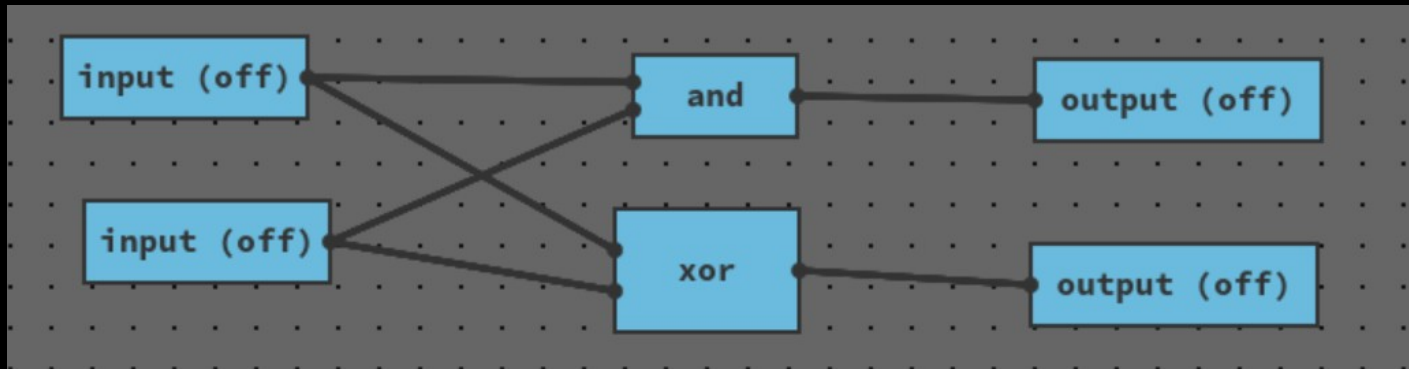
Mouse



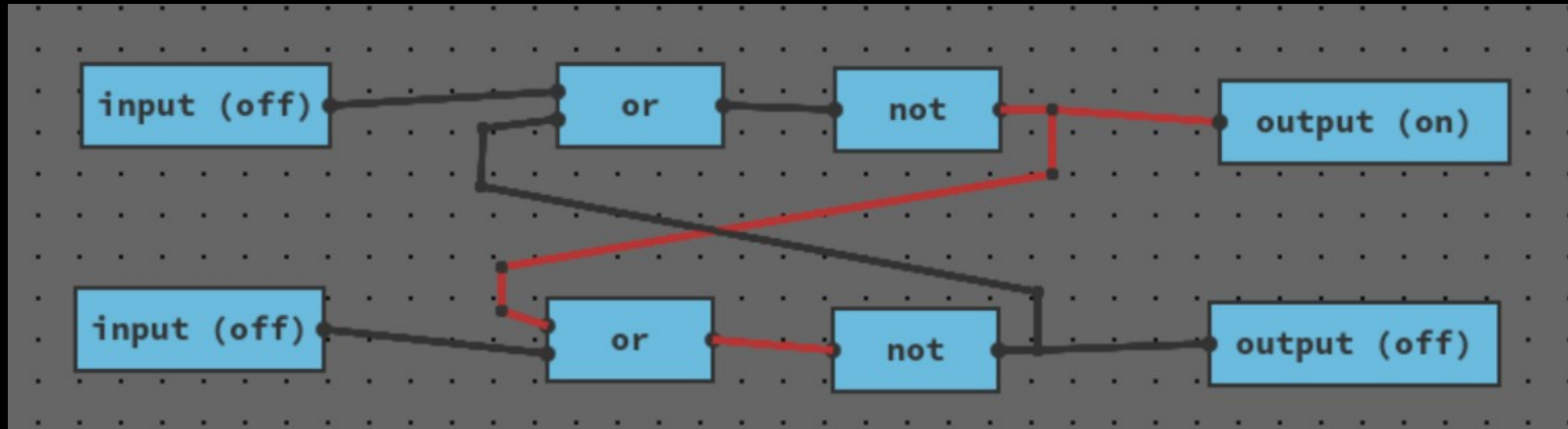
IR / Simulation



```
component xor 2 1 {  
  state [ ]  
  %0 = Input 0  
  %1 = Input 1  
  %2 = Nand %0, %1  
  %3 = Or %0, %1  
  %4 = And %2, %3  
  Output 0, %4  
}
```

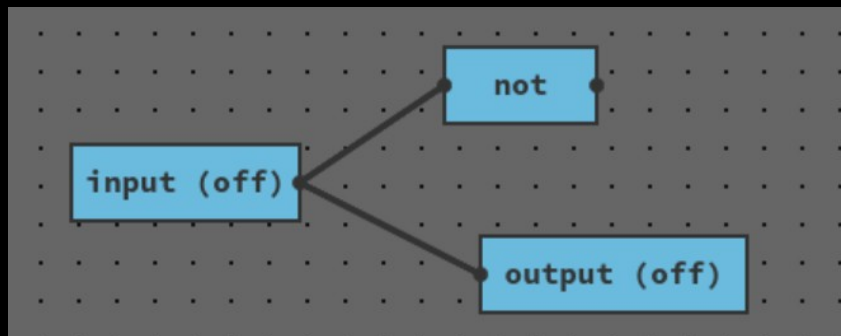


```
component half adder 2 2 {  
    state [ ]  
    %0 = Input 0  
    %1 = Input 1  
    %2 = Call xor (%0, %1)  
    %3 = Elem %2, 0  
    Output 1, %3  
    %4 = And %0, %1  
    Output 0, %4  
}
```



```
component rs latch 2 2 {  
    state [ #0 ]  
    %0 = Input 0  
    %1 = Input 1  
    %2 = GetState #0  
    %3 = Nor %0, %2  
    %4 = Nor %3, %1  
    SetState #0, %4  
    Output 1, %4  
    Output 0, %3  
}
```

```
412  
413 eliminateUnusedStmts() {  
414     const mut = new StmtMutater(this.comp, this.replacedStates);  
415  
416     const useCount = new Map<Stmt, number>();  
417     for (const stmt of mut) {  
418         for (const src of stmt.sources()) {  
419             useCount.set(src, (useCount.get(src) ?? 0) + 1);  
420         }  
421     }  
422  
423     const nonEffectfulStmts = new Set<StmtKind["tag"]>([  
424         "And", "Or", "Nand", "Nor", "Elem", "Input", "GetState", "Null",  
425     ]);  
426  
427     for (const stmt of mut) {  
428         if (!useCount.get(stmt) && nonEffectfulStmts.has(stmt.kind.tag)) {  
429             mut.removeStmt(stmt);  
430         }  
431     }  
432 }  
433
```



```
component <main> 1 1 {  
  state [ #0 ]  
  %0 = Input 0  
  %1 = Not %0  
  Output 0, %0  
}
```

An orange arrow points to the line `%1 = Not %0` in the code block.

```
component <main> 1 1 {  
  state [ #0 ]  
  %0 = Input 0  
  Output 0, %0  
}
```

```

26   for (let i = 0; i < comp.stmts.length; ++i) {
27       const stmt = comp.stmts[i];
28       const k = stmt.kind;
29       switch (k.tag) {
30           case "Null":
31               regs[i] = false;
32               break;
33           case "Input":
34               regs[i] = inputs[k.i];
35               break;
36           case "Output":
37               outputs[k.i] = regs[stmtIds.get(k.src)!];
38               break;
39 >       case "GetState": ...
49 >       case "SetState": { ...
59           }
60       case "Not":
61           regs[i] = operation((v) => !v, k.op);
62           break;
63       case "And":
64           regs[i] = operation((a, b) => a && b, k.lhs, k.rhs);
65           break;
66       case "Or":
67           regs[i] = operation((a, b) => a || b, k.lhs, k.rhs);
68           break;
69       case "Nand":
70           regs[i] = operation((a, b) => !(a && b), k.lhs, k.rhs);
71           break;
72       case "Nor":
73           regs[i] = operation((a, b) => !(a || b), k.lhs, k.rhs);
74           break;
75 >       case "Call": { ...
85           }
86 >       case "Elem": { ...
93           }

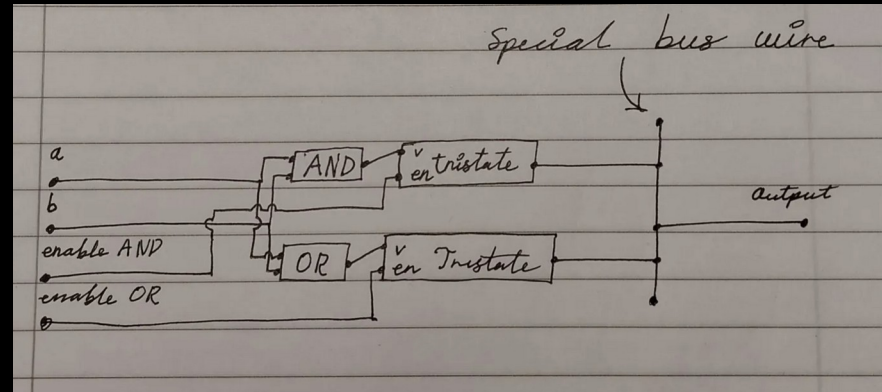
```


Ting der kan forbedres

- Musefunktioner virker suboptimalt
- Man kan ikke zoome
- Tabs og save-funktionalitet har huller
- Wire-stilen er grim
- Event-systemet er rodet

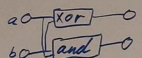
Ting jeg ikke nåede

- Importering af data
- Cloud-miljø
- CI-setup
- Native-afvikling
- Low impedance/Tri-state

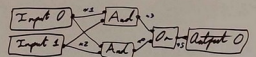
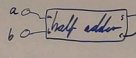


Konklusion

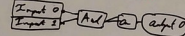
half adder:



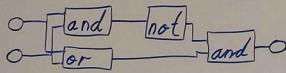
full adder:



In this case collapse the And's.



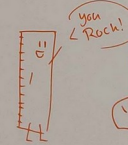
XOR:



input a, b, c, d
e = and a, b
f = and c, d

$\{e, f\} = \text{and}\{a, b\}, \{c, d\}$

ac
0000
b d
0000
e f
0000



#KARNAUGH

component a

input x1, x2, x3, x4
x5, x6 = xor x1, x2, x3, x4
x7 = xor x5, x6
output x7

component xor2

input x1, x2, x3, x4
x5 = xor x1, x3
x6 = xor x2, x4
output x5, x6

TOPSDAG:

- STOLE OP
- LEDNINGER VÆK FRA GULVE K.A.T.

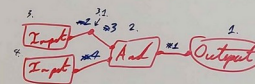
Mags-000

Merc1234!

SPS koordination, Katrine

Kaso (Skriver gerne på Teams)

MIS.



Forward traverse

1. x1 = Input 0
2. x2 = Input 1
3. x3 = Input 2
4. x4 = Input 3
5. x5 = Input 4
6. x6 = Input 5
7. x7 = Input 6
8. x8 = Input 7
9. x9 = Input 8
10. x10 = Input 9
11. x11 = Input 10
12. x12 = Input 11
13. x13 = Input 12
14. x14 = Input 13
15. x15 = Input 14
16. x16 = Input 15
17. x17 = Input 16
18. x18 = Input 17
19. x19 = Input 18
20. x20 = Input 19
21. x21 = Input 20
22. x22 = Input 21
23. x23 = Input 22
24. x24 = Input 23
25. x25 = Input 24
26. x26 = Input 25
27. x27 = Input 26
28. x28 = Input 27
29. x29 = Input 28
30. x30 = Input 29
31. x31 = Input 30
32. x32 = Input 31
33. x33 = Input 32
34. x34 = Input 33
35. x35 = Input 34
36. x36 = Input 35
37. x37 = Input 36
38. x38 = Input 37
39. x39 = Input 38
40. x40 = Input 39
41. x41 = Input 40
42. x42 = Input 41
43. x43 = Input 42
44. x44 = Input 43
45. x45 = Input 44
46. x46 = Input 45
47. x47 = Input 46
48. x48 = Input 47
49. x49 = Input 48
50. x50 = Input 49
51. x51 = Input 50
52. x52 = Input 51
53. x53 = Input 52
54. x54 = Input 53
55. x55 = Input 54
56. x56 = Input 55
57. x57 = Input 56
58. x58 = Input 57
59. x59 = Input 58
60. x60 = Input 59
61. x61 = Input 60
62. x62 = Input 61
63. x63 = Input 62
64. x64 = Input 63
65. x65 = Input 64
66. x66 = Input 65
67. x67 = Input 66
68. x68 = Input 67
69. x69 = Input 68
70. x70 = Input 69
71. x71 = Input 70
72. x72 = Input 71
73. x73 = Input 72
74. x74 = Input 73
75. x75 = Input 74
76. x76 = Input 75
77. x77 = Input 76
78. x78 = Input 77
79. x79 = Input 78
80. x80 = Input 79
81. x81 = Input 80
82. x82 = Input 81
83. x83 = Input 82
84. x84 = Input 83
85. x85 = Input 84
86. x86 = Input 85
87. x87 = Input 86
88. x88 = Input 87
89. x89 = Input 88
90. x90 = Input 89
91. x91 = Input 90
92. x92 = Input 91
93. x93 = Input 92
94. x94 = Input 93
95. x95 = Input 94
96. x96 = Input 95
97. x97 = Input 96
98. x98 = Input 97
99. x99 = Input 98
100. x100 = Input 99
101. x101 = Input 100
102. x102 = Input 101
103. x103 = Input 102
104. x104 = Input 103
105. x105 = Input 104
106. x106 = Input 105
107. x107 = Input 106
108. x108 = Input 107
109. x109 = Input 108
110. x110 = Input 109
111. x111 = Input 110
112. x112 = Input 111
113. x113 = Input 112
114. x114 = Input 113
115. x115 = Input 114
116. x116 = Input 115
117. x117 = Input 116
118. x118 = Input 117
119. x119 = Input 118
120. x120 = Input 119
121. x121 = Input 120
122. x122 = Input 121
123. x123 = Input 122
124. x124 = Input 123
125. x125 = Input 124
126. x126 = Input 125
127. x127 = Input 126
128. x128 = Input 127
129. x129 = Input 128
130. x130 = Input 129
131. x131 = Input 130
132. x132 = Input 131
133. x133 = Input 132
134. x134 = Input 133
135. x135 = Input 134
136. x136 = Input 135
137. x137 = Input 136
138. x138 = Input 137
139. x139 = Input 138
140. x140 = Input 139
141. x141 = Input 140
142. x142 = Input 141
143. x143 = Input 142
144. x144 = Input 143
145. x145 = Input 144
146. x146 = Input 145
147. x147 = Input 146
148. x148 = Input 147
149. x149 = Input 148
150. x150 = Input 149
151. x151 = Input 150
152. x152 = Input 151
153. x153 = Input 152
154. x154 = Input 153
155. x155 = Input 154
156. x156 = Input 155
157. x157 = Input 156
158. x158 = Input 157
159. x159 = Input 158
160. x160 = Input 159
161. x161 = Input 160
162. x162 = Input 161
163. x163 = Input 162
164. x164 = Input 163
165. x165 = Input 164
166. x166 = Input 165
167. x167 = Input 166
168. x168 = Input 167
169. x169 = Input 168
170. x170 = Input 169
171. x171 = Input 170
172. x172 = Input 171
173. x173 = Input 172
174. x174 = Input 173
175. x175 = Input 174
176. x176 = Input 175
177. x177 = Input 176
178. x178 = Input 177
179. x179 = Input 178
180. x180 = Input 179
181. x181 = Input 180
182. x182 = Input 181
183. x183 = Input 182
184. x184 = Input 183
185. x185 = Input 184
186. x186 = Input 185
187. x187 = Input 186
188. x188 = Input 187
189. x189 = Input 188
190. x190 = Input 189
191. x191 = Input 190
192. x192 = Input 191
193. x193 = Input 192
194. x194 = Input 193
195. x195 = Input 194
196. x196 = Input 195
197. x197 = Input 196
198. x198 = Input 197
199. x199 = Input 198
200. x200 = Input 199

Can I handle the SubStates?