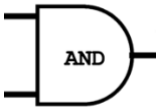
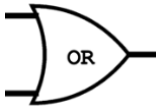
CSE 390B Midterm Reference Sheet

Fundamental Combinational Logic Gates

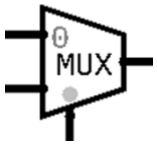


a	b	out
0	0	0
0	1	0
1	0	0
1	1	1



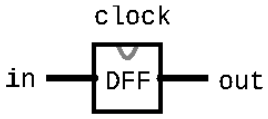
a	b	out
0	0	0
0	1	1
1	0	1
1	1	1

in	out
0	1
1	0



a	b	sel	out
0	0	0	0
0	1	0	0
1	0	0	1
1	1	0	1
0	0	1	0
0	1	1	1
1	0	1	0
1	1	1	1

Fundamental Sequential Logic Gate



`out(t) = in(t-1)`

- Triangle indicates implicitly connected to hardware clock
- **`out`** only changes on clock signal boundaries

HDL

Syntax:

- Basic Format: **`ChipName (in1=w1, in2=w2, ..., out=w3);`**
- Example: **`Mux (a=w1, b=w2, sel=w3, out=w4);`**
- Multiple wires connected to single output:
  `Mux (a=w1, b=w2, sel=w3, out=w4, out=w5);`

Multi-Bit Buses:

- Accessing Single Bit: **`w1[2]`**
- Slicing Multiple Bits: **`w1[0..3]`** (indices inclusive)
- Multi-Bit Input / Output Declaration: **`IN a[16];`**

Special Values:

- **`true`** is an any-width bus of all 1s, **`false`** of all 0s

Hack Assembly Language

Machine Characteristics:

- Two physical registers: **`D`**, **`A`**
- Pseudoregister **`M`** accesses memory at address **`A`**
- **`RAM`** and **`ROM`** have different, 0-indexed address spaces

Existing Symbols:

- **`R0...R15`** are "virtual registers": symbols bound to addresses 0 … 15 of **`RAM`**
- **`SCREEN`** is symbol bound to address at start of screen memory map
- **`KBD`** is bound to address of keyboard memory map (immediately after screen)

Label: **(*`LABELNAME`*)**

- Binds symbol ***`LABELNAME`*** to line number of instruction after it

A-Instructions: ***`@VALUE`***

- Loads ***`VALUE`*** into A register

C-Instructions: **`DEST=COMP;JUMP`**

- **`DEST`** or **`JUMP`** optional
- Performs **`COMP`**, result is stored in **`DEST`**, and if the result satisfies **`JUMP`** the PC jumps to address in A register

C-Instructions: Options for Fields

`COMP`

`0`
`1`
`-1`
`D`
`A`
`!D`
`!A`
`-D`
`-A`
`D+1`
`A+1`
`D-1`
`A-1`
`D+A`
`D-A`
`A-D`
`D&A`
`D|A`
`M`
`!M`
`-M`
`M+1`
`M-1`
`D+M`
`D-M`
`M-D`
`D&M`
`D|M`

`DEST`

`(empty)`
`M`
`D`
`A`
`MD`
`AM`
`AD`
`AMD`

`JUMP`

`(empty)`	`No jump`
`JGT`	`Jump if out > 0`
`JEQ`	`Jump if out = 0`
`JGE`	`Jump if out >= 0`
`JLT`	`Jump if out < 0`
`JNE`	`Jump if out != 0`
`JLE`	`Jump if out <= 0`
`JMP`	`Always jump`