Arithmetic using Logic

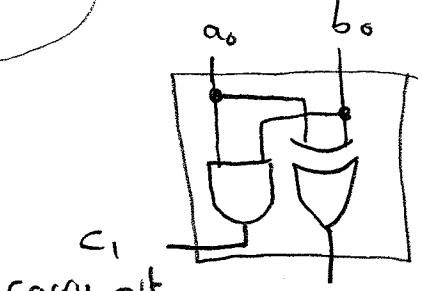
Half Adder

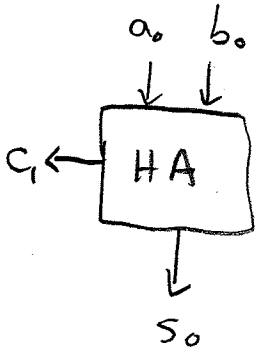
$$\begin{array}{r} 1 \\ +\,1 \\ \hline 10 \end{array} \qquad \begin{array}{l} a_0 \\ +\,b_0 \\ \hline c_1\ s_0 \end{array}$$

XOR

$$s_0 = a_0 \oplus b_0$$

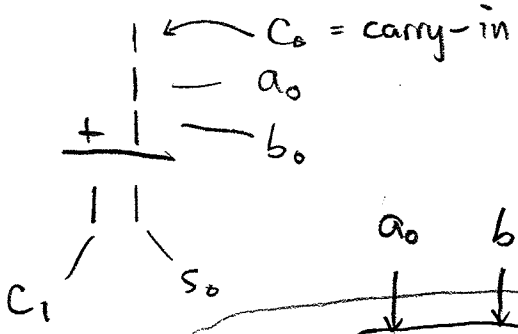
$$c_1 = a_0 \cdot b_0$$

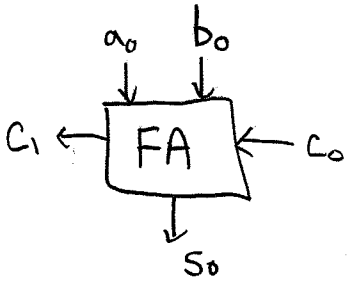


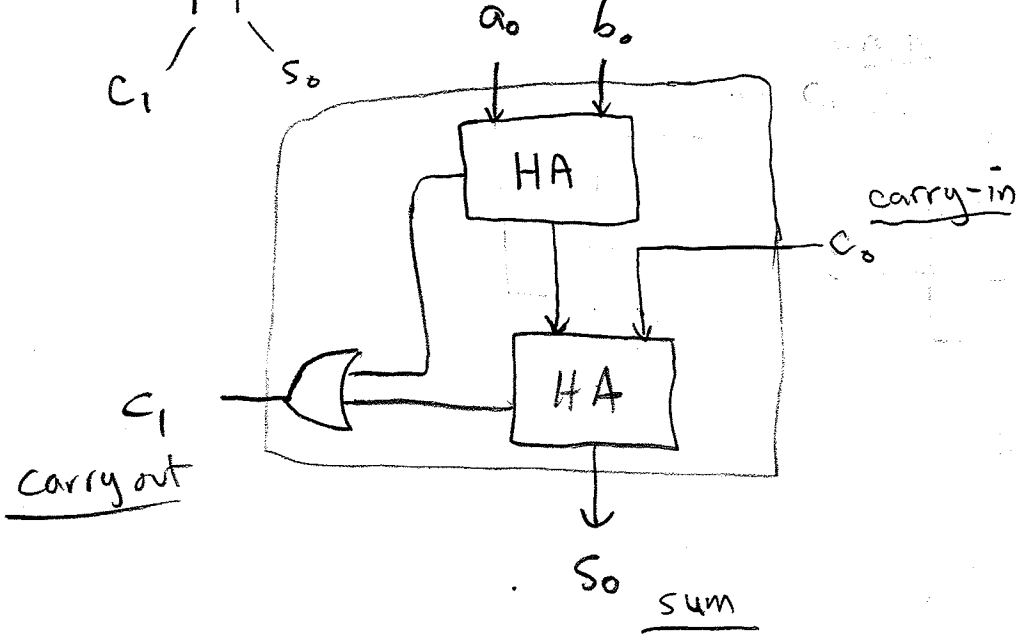


Full Adder

= 2 × Half Adders + ∫

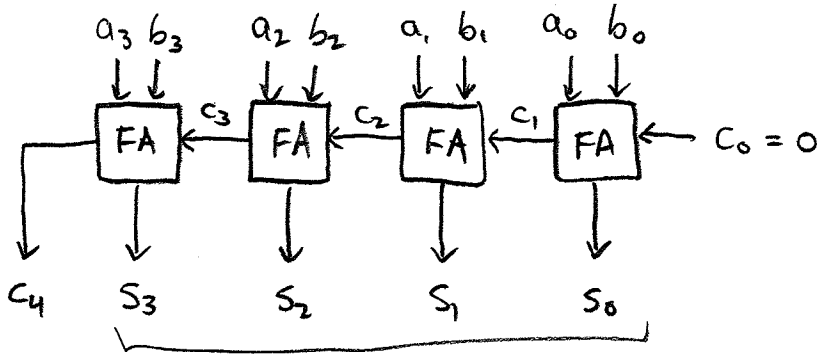




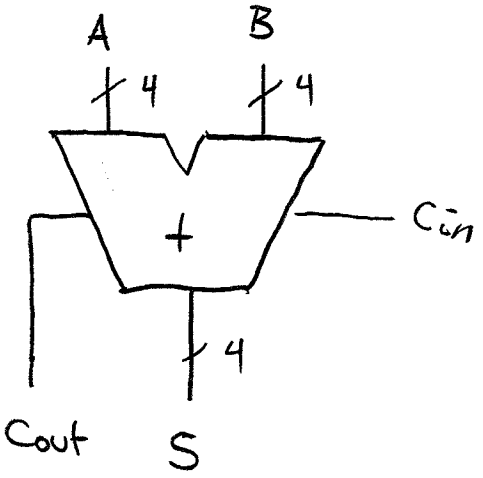


Verify: logic for carry-out c_1 is correct

4-bit adder



also:



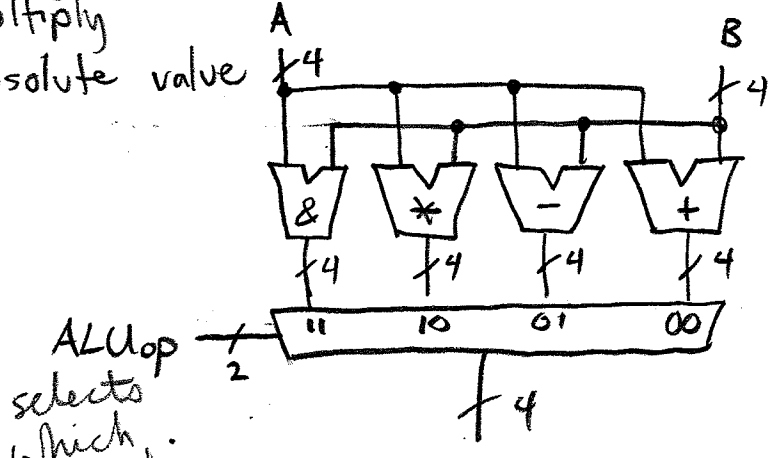
ALU: arithmetic logic unit — operates on integers

common operations
- add, subtract
- bitwise AND, OR, XOR
- shift
- multiply
- absolute value

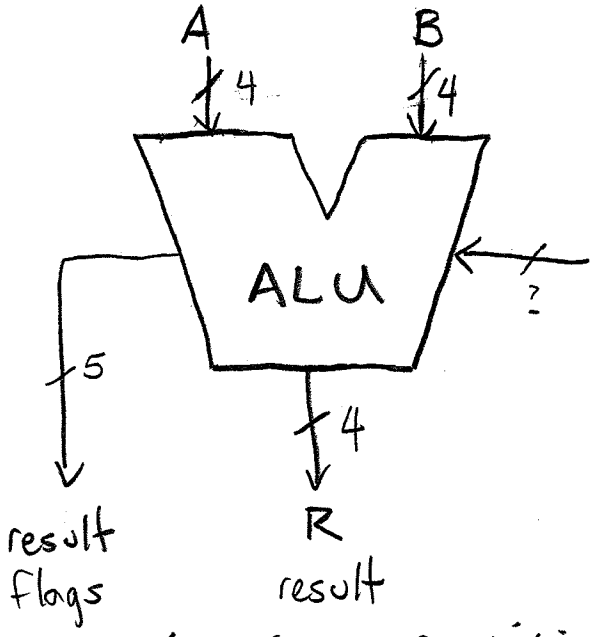
eg, bitwise AND:

a_3	a_2	a_1	a_0
& b_3	b_2	b_1	b_0
a_3b_3	a_2b_2	a_1b_1	a_0b_0

easy way to build



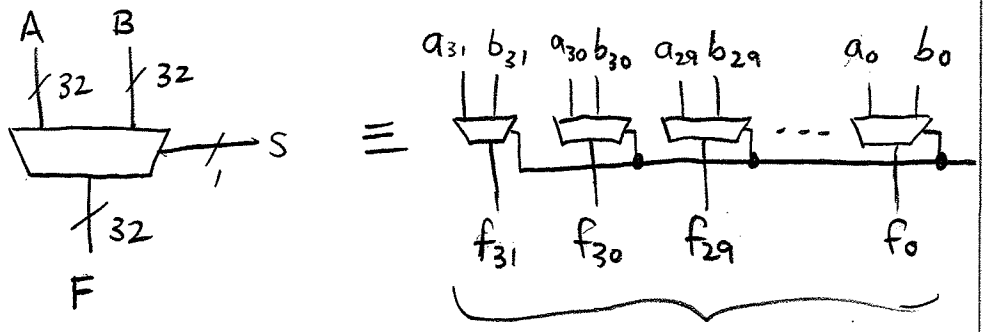
Symbol



1 bit each

- C carry out (C=1 if addition results in carry)
- B borrow (B=1 if subtraction requires a borrow)
- V overflow (V=1 if ALUop results in overflow)
- N negative (N=1 if MSB of R=1)
- Z zero (Z=1 if all bits in R=0)

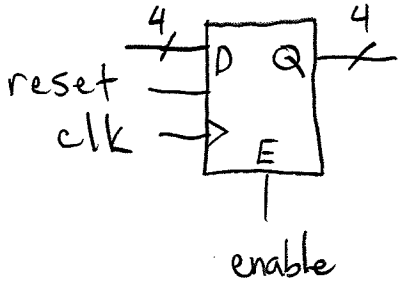
Note:-



one 32-bit multiplexer ≡ 32 multiplexers in parallel

≡ lots of gates!

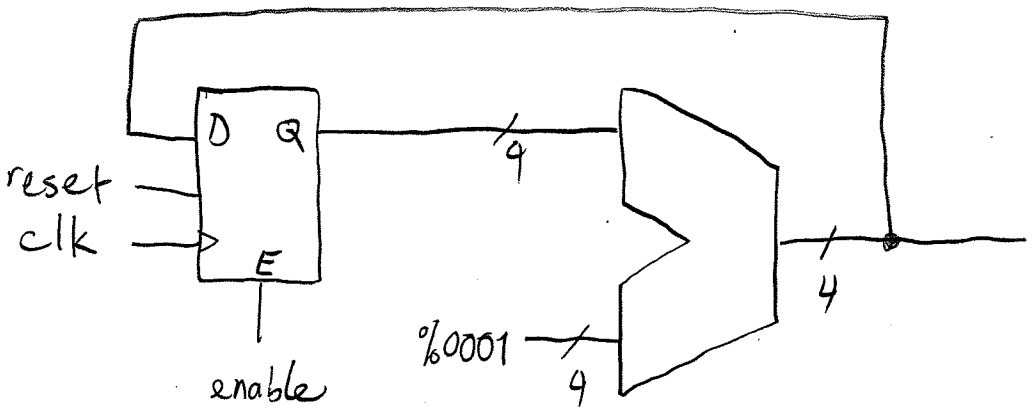
Registers a group of DFFEs with common **clock** and **enable** signals



reset clears all DFFEs (to Ø)

preset sets all DFFEs (to 1) immediately, no clock or enable needed

Accumulator - type of an adder/counter
- has memory (register)



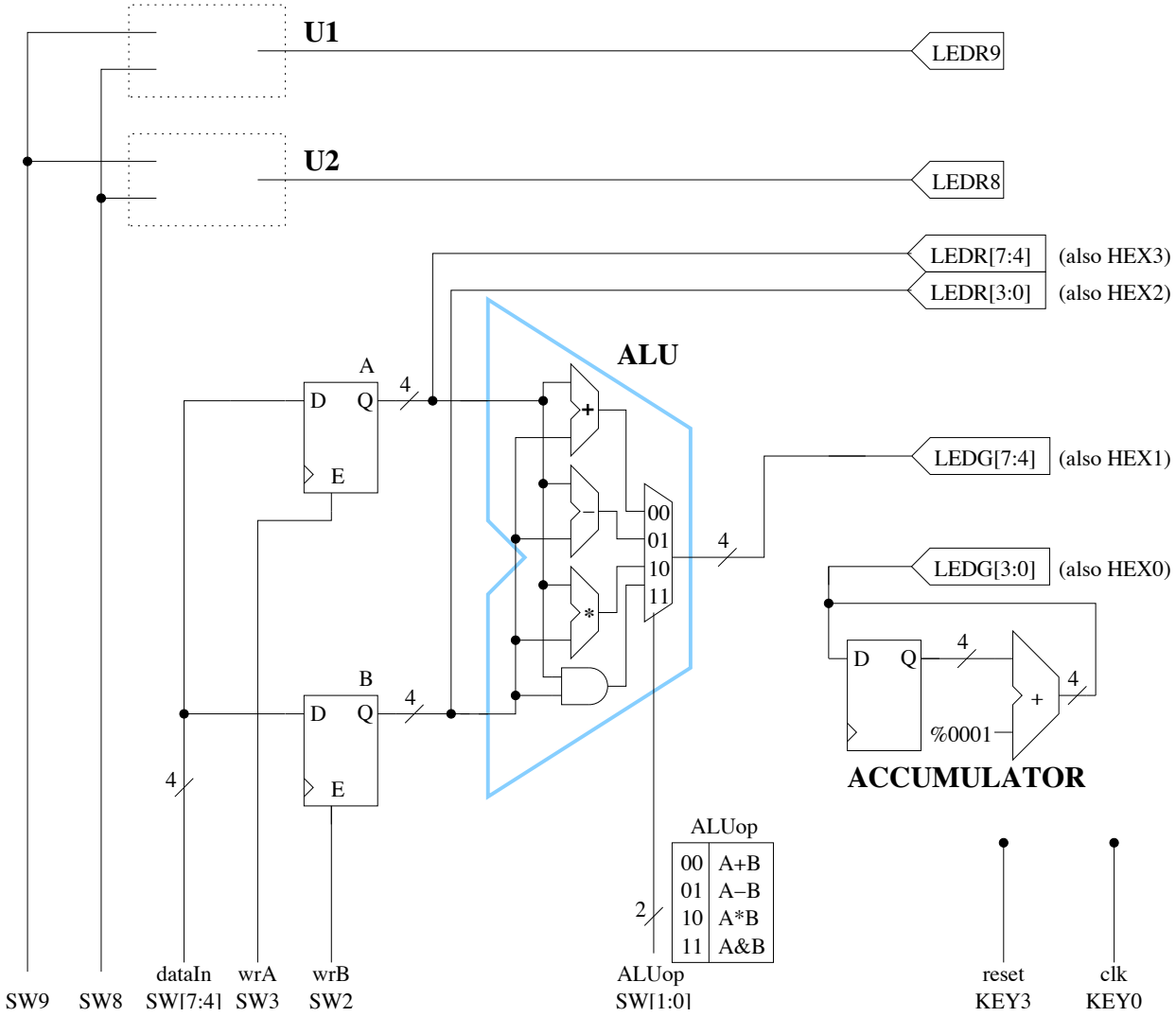
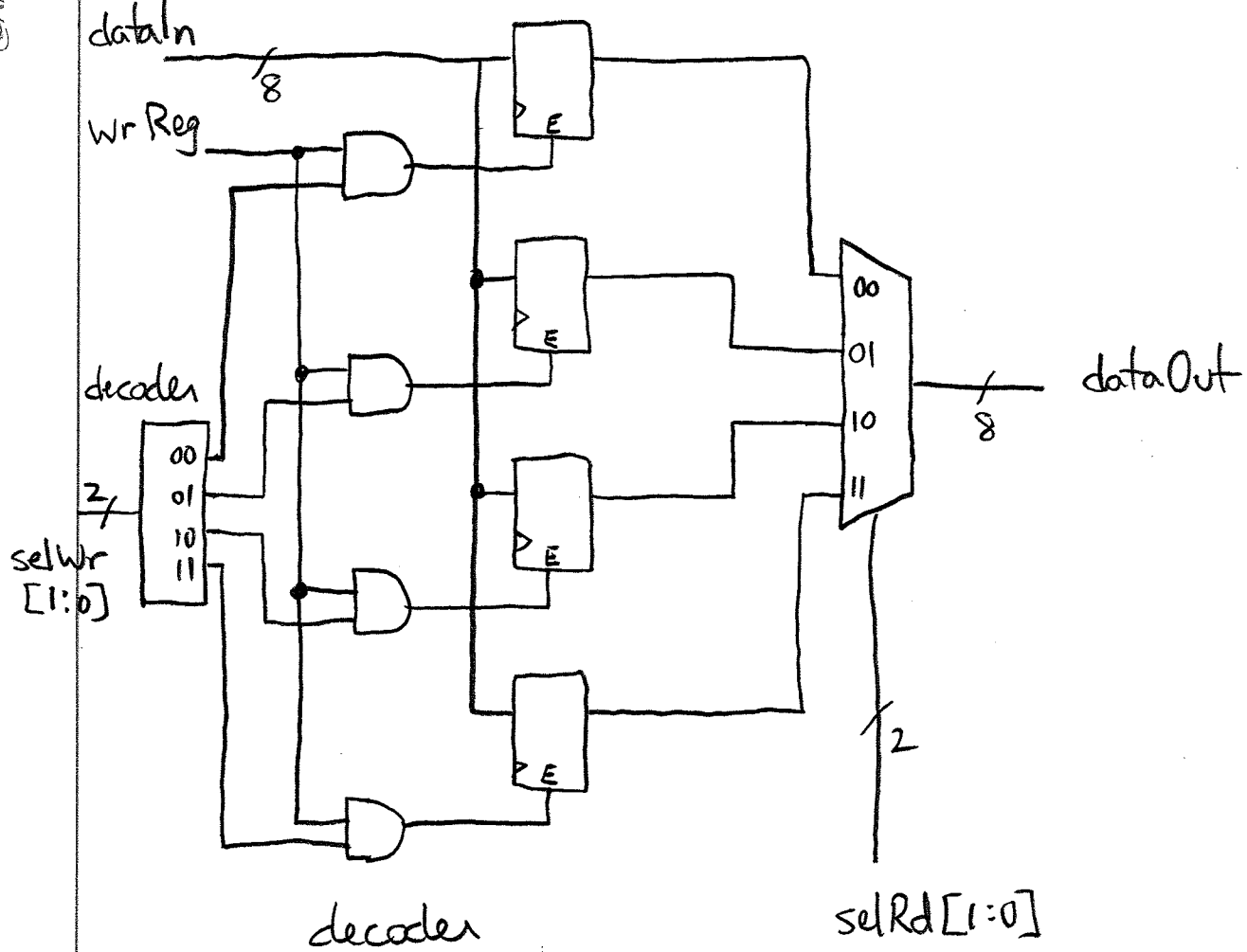
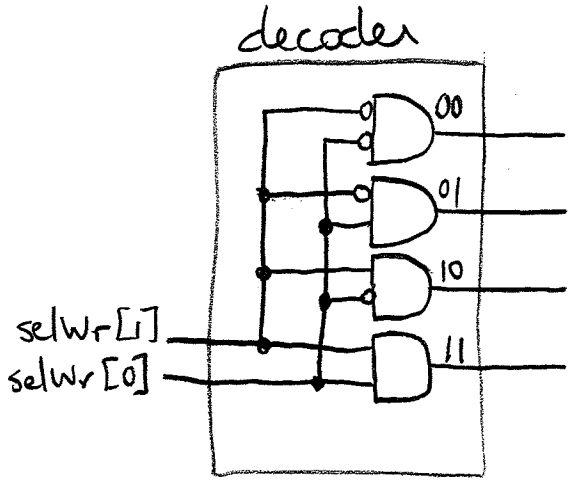


Figure A. Basic Logic Gates circuit.

Register File

- a set of M registers operating as a group
- we need 2 key operations
  1) read value from register #S, $0 \leq S < M$
  2) write a value to register #T, $0 \leq T < M$
- we can do <u>both</u> once per clock cycle





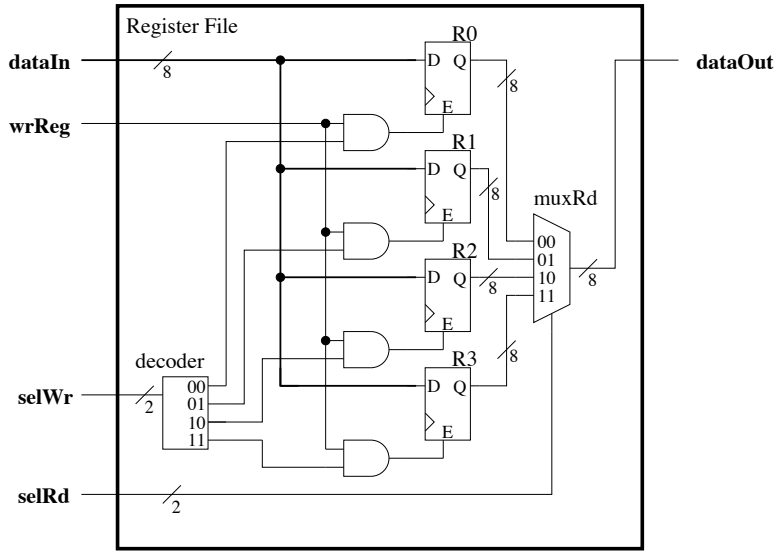


Figure B1. Register File details.

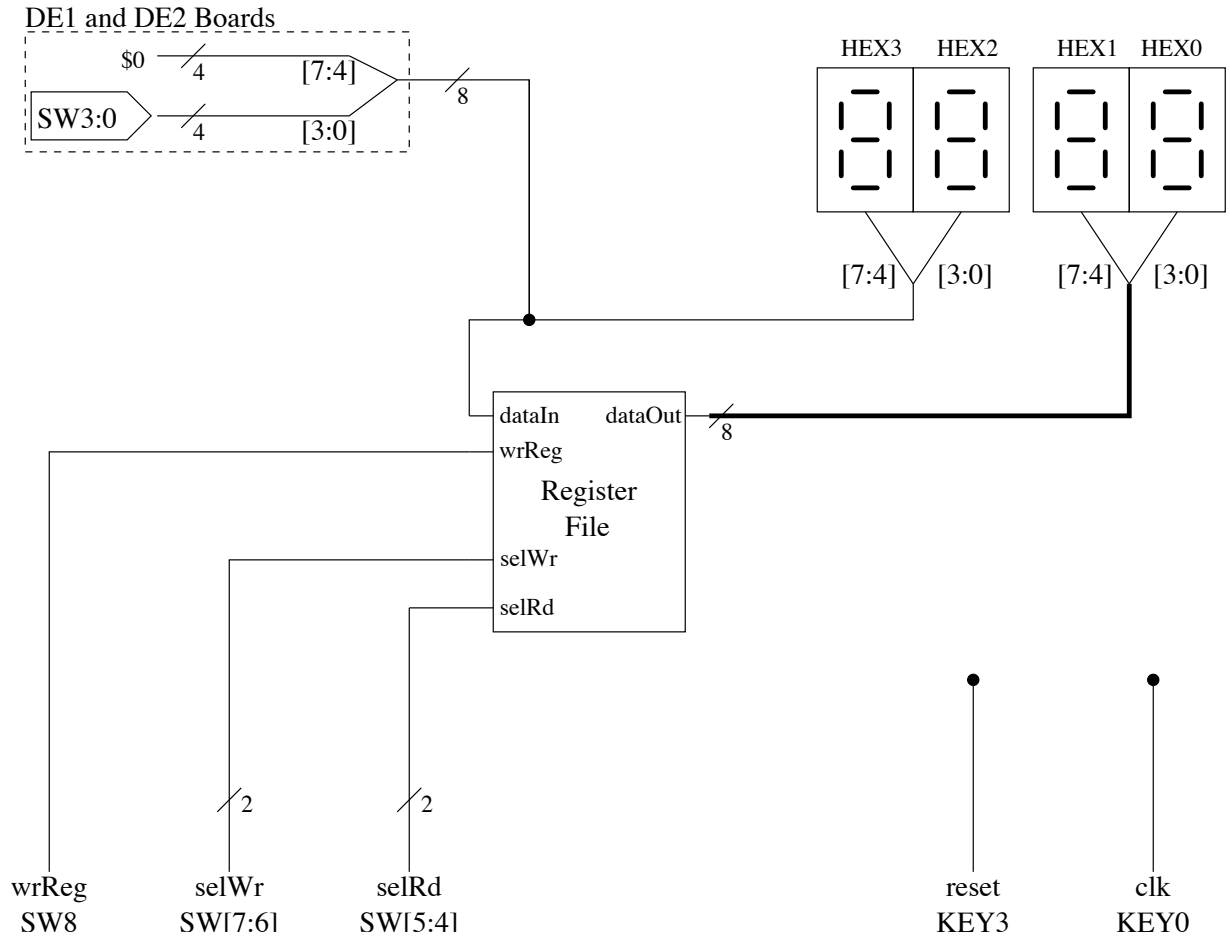


Figure B2. Register File circuit.