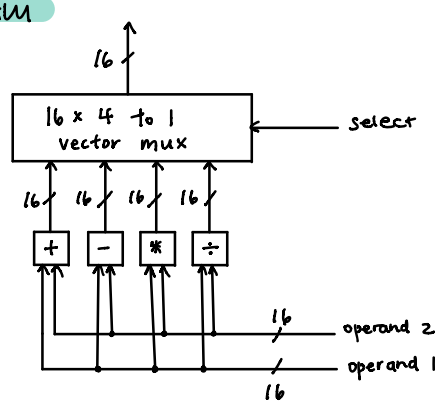
I have completed this assignment entirely on my own except for discussion with Janice Tsai.

- Behavior:
  1. enter first operand
     - A. encode numeric keypad buttons into BCD digits
     - B. shift BCD digits into a shift register (display)
     - C. hit 'e'
     - D. push operand into stack
     - E. clear display register
  2. enter second operand
     - A. encode numeric keypad buttons into BCD digits
     - B. shift BCD digits into a shift register (display)
     - C. hit 'e'
     - D. push operand into stack
     - E. clear display register
  3. enter an operator (+, −, *, ÷)
     - A. encode operator key press
     - B. pop off 2 elements from stack
     - C. perform operation
     - D. push result onto stack
  4. '=' Equals
     - A. pop off first element from stack
     - B. display the number
  5. 'e' enter key continues calculation

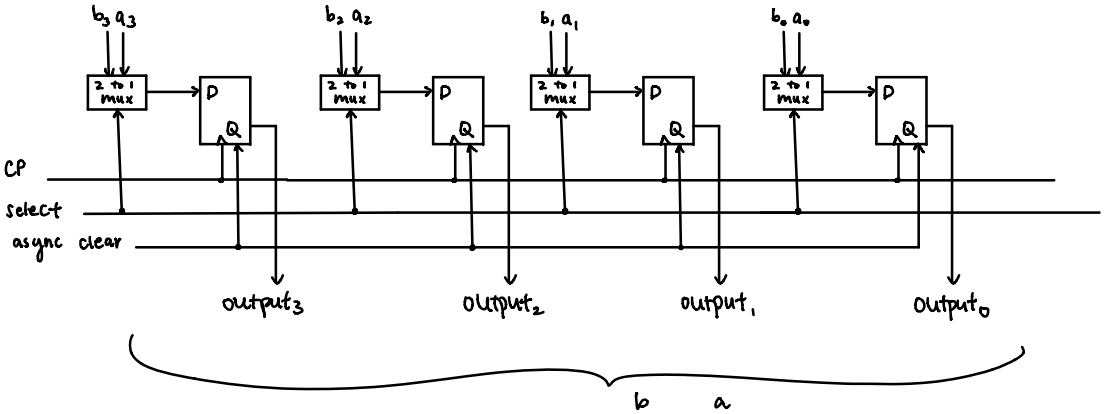
Example:
12e34e+56e78e+*=
- display 1 then 12
- push 12 onto stack
- clear display
- display 3 then 34
- push 34 onto stack
- clear display
- pop off 34 and 12
- add 12+34 and push sum onto stack
- display 5 then 56
- push 56 onto stack
- clear display
- display 7 then 78
- push 78 onto stack
- clear display
- pop off 78 and 56
- add 56+78 and push sum onto stack
- pop off 134 and 46
- multiply 46 x 134 and push product onto stack
- pop off from stack + display it

Define Circuit Components:

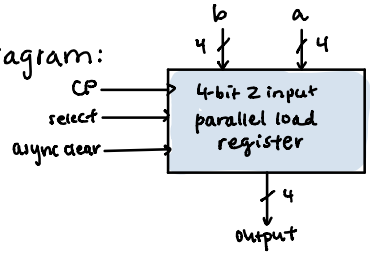
- 4-digit BCD ALU



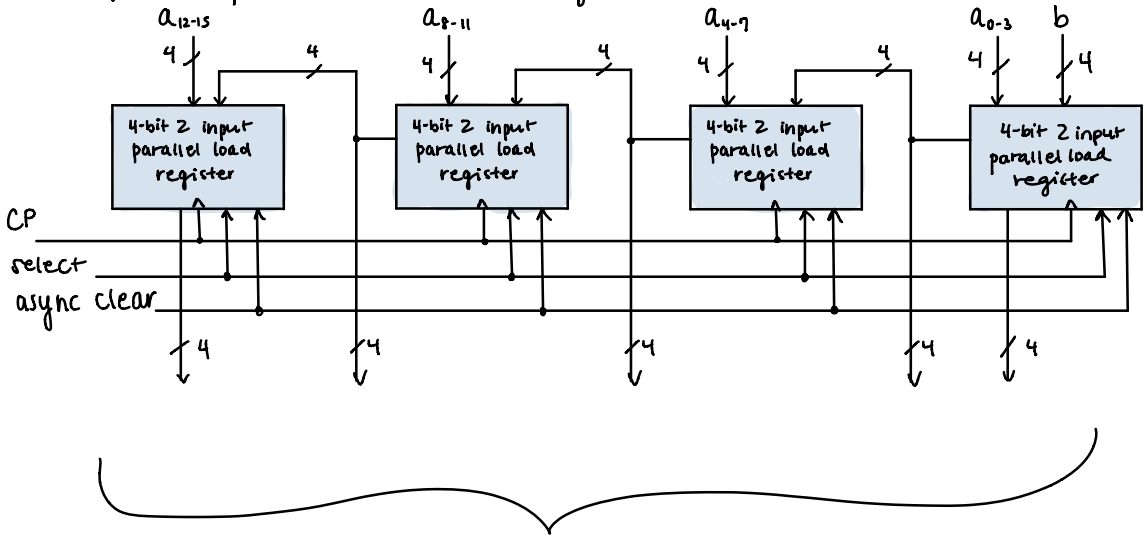
- 4-bit 2 input parallel load register



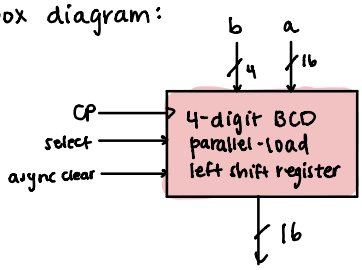
blackbox diagram:



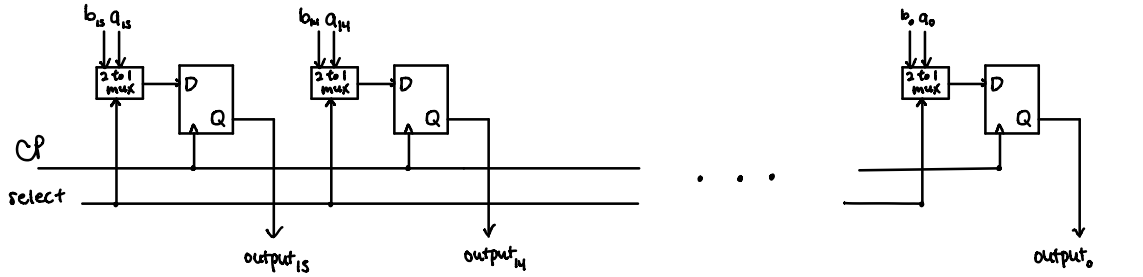
- 4-digit BCD parallel-load left shift register

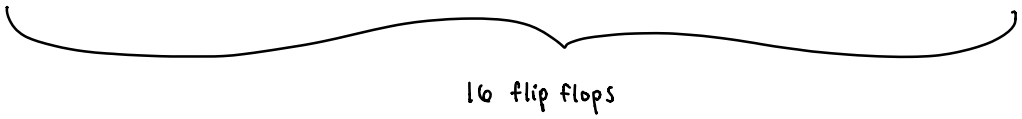


blackbox diagram:

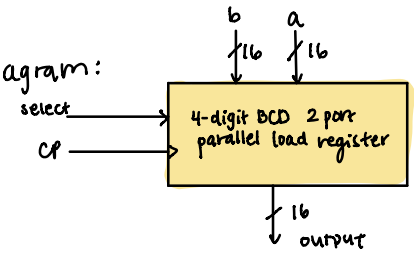


- 4-digit BCD 2 port parallel load register (stack register)

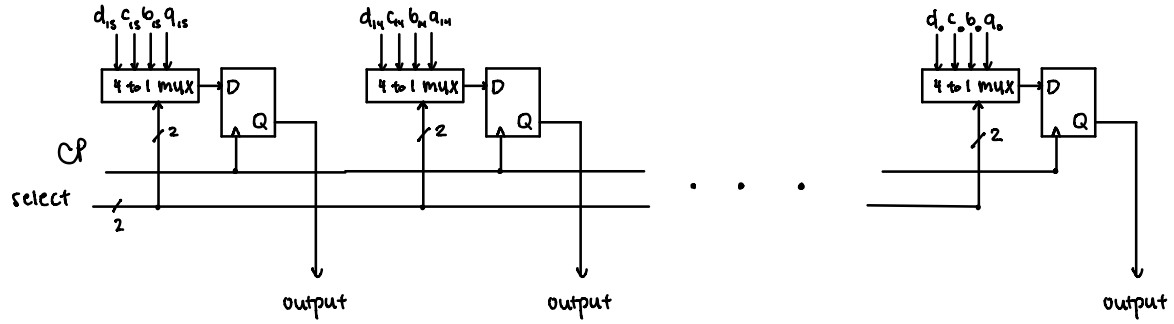




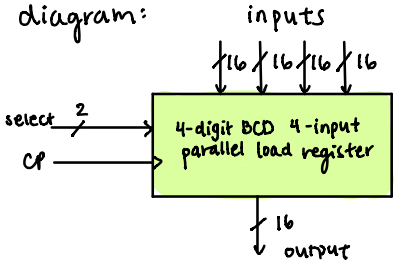
Blackbox diagram:



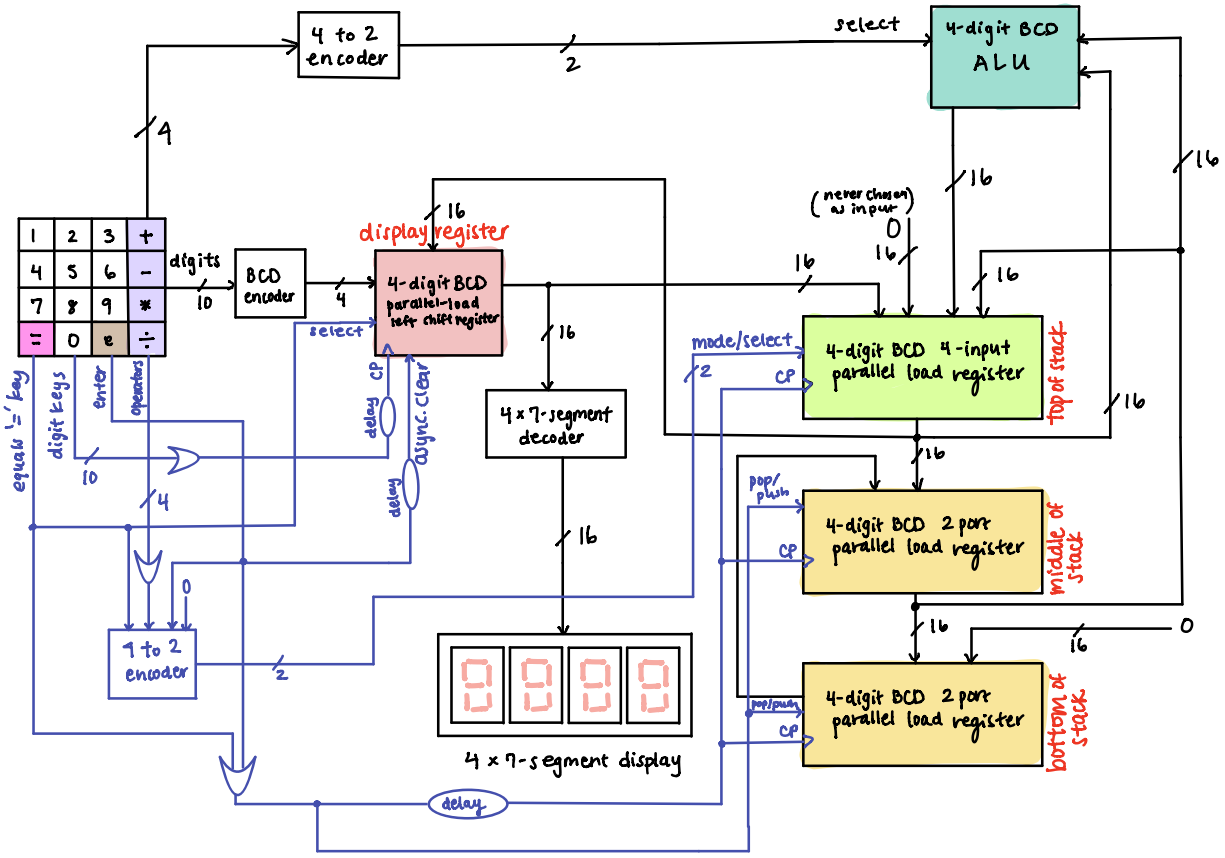
- 4-digit BCD 4-input parallel load register



Blackbox diagram:



Data Paths



Control circuit

- display register clock
  - whenever a digit is pressed (delay b/c of encoder and select lines)
- display register mux select/mode
  - any digit keys → shift load
  - '=' equals key → parallel load
- display register asynchronous clear
  - whenever enter key is hit (delay to push onto stack, longer than CP)
- stack registers clock
  - whenever '=' equals or enter key is pressed (delay b/c of mode/select)
- top of stack register select/mode using 4 to 2 encoder
  - operator key → select output of ALU as parallel input
  - enter key → select output of display register as parallel input
  - '=' key → select output of middle of stack register as parallel input
- middle & bottom of stack register select/mode
  - operator key → pop select
  - '=' key or enter key → push select

circuit description:

1) When the first operand is entered, each digit key press is encoded into 4-digit BCD and entered into the 4-digit BCD parallel-load shift register with the mode/select set to serial input
   - each digit is displayed in the 7 segment display
2) When 'e' is pressed, the value in the 4-digit BCD parallel-load shift register is pushed onto the 3 element stack register
   - the select line for the top of the stack register is set to receive input from the output of the display register
   - the select lines for the middle + bottom stack registers is set to 'push' so that the number in the top register is loaded into the middle register as the number currently in the middle register is loaded into the bottom register.
   - the display gets cleared
3) steps 1 + 2 are repeated for every digit(s) followed by 'e'
4) when an operator is entered, it is encoded into a 2 bit select line for the 4-digit BCD ALU
   - the top 2 elements in the stack are sent to the ALU
   - the select line for the top stack register is set to receive input from the ALU
   - the select lines for the middle + bottom stack registers is set to 'pop' so the contents of the bottom register moves up to the middle stack register as the Ø is loaded into the bottom stack register.
5) when the '=' is pressed, the mode/select for the display register is set to receive parallel input from the top of the stack
   - the mode/select for the top of stack register is set to receive input from the middle register
   - the mode/select of the middle + bottom stack registers is set to pop so the element in the bottom register moves up to the middle register and a Ø moves into the bottom register.