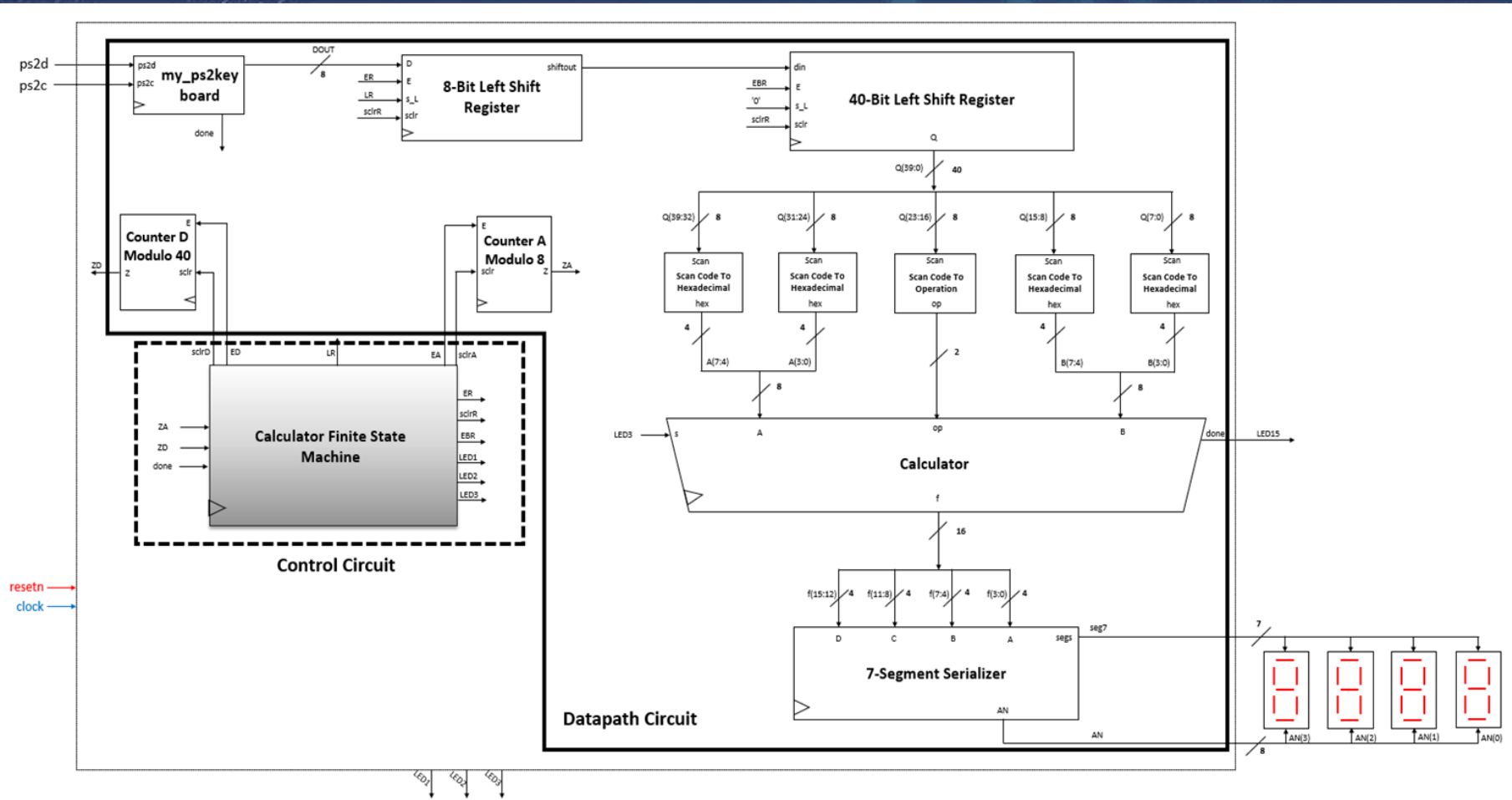
8 Bit Signed Calculator

Giuseppe Rizzo, Evan Kornhaus, David Kosa, France Zaytouna

Overview of the Calculator

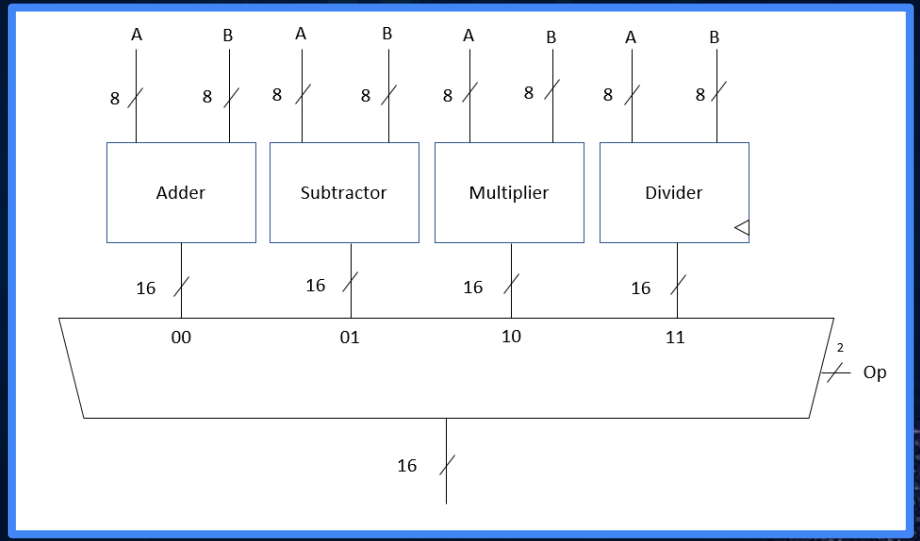
- This calculator takes two 8-bit signed numbers as inputs. The inputs are entered through a USB keyboard as hexadecimal numbers
- The operator is chosen from one of the four keys displayed below
- The output is a 16-bit signed number, which is displayed as a hexadecimal number on a seven-segment display
- To enter data, first enter the first number in hexadecimal, choose an operator, and then enter the second number

How to use calculator


ps2d
ps2c
my_ps2key board
DOUT
8
done
8-Bit Left Shift Register
shiftout
40-Bit Left Shift Register
Q(39:0)
40
Counter D Modulo 40
Counter A Modulo 8
ZD
ZA
Scan Code To Hexadecimal
Scan Code To Operation
Calculator Finite State Machine
Control Circuit
Calculator
LED3
done
LED15
resetn
clock
7-Segment Serializer
seg7
segs
AN
Datapath Circuit
AN(3)
AN(2)
AN(1)
AN(0)

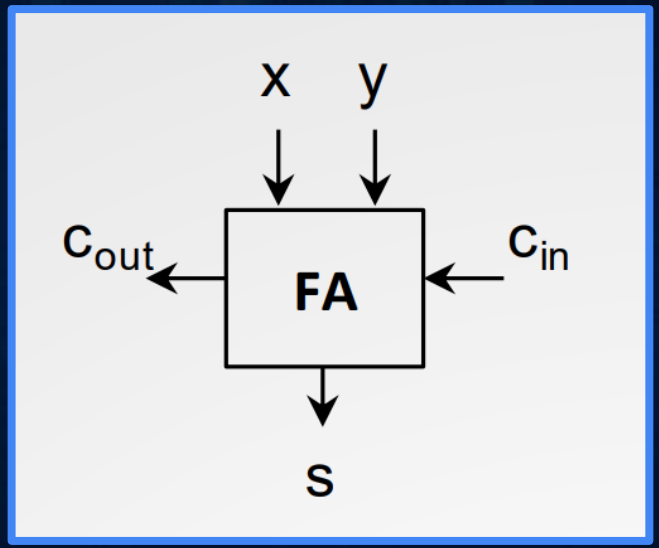
A Closer Look At the Calculator Component

- The computational side of the design was shown in the previous block diagram as a component named “calculator”
- This component consists of the four arithmetic circuits connected through a multiplexor
- The output is determined by the “op” signal that is received from the keyboard input

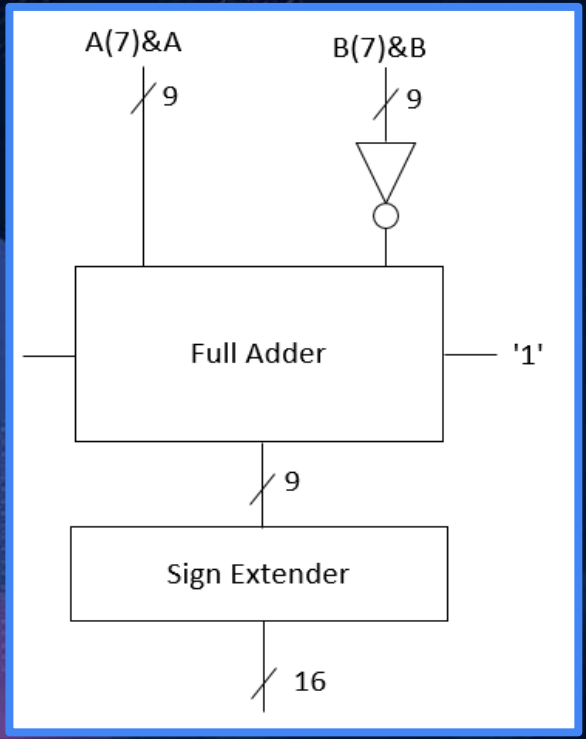


Adder

- In order to avoid overflow, the two 8-bit inputs were sign extended to 9-bits before entering the adder
- 9 full-adders were connected in series in order to create the final adder circuit
- The sum output signal was then sign extended to 16-bits
- In order to use this function the “+” button should be pressed on the keyboard (NOTE: it is not necessary to hold shift while hitting the “+” key)



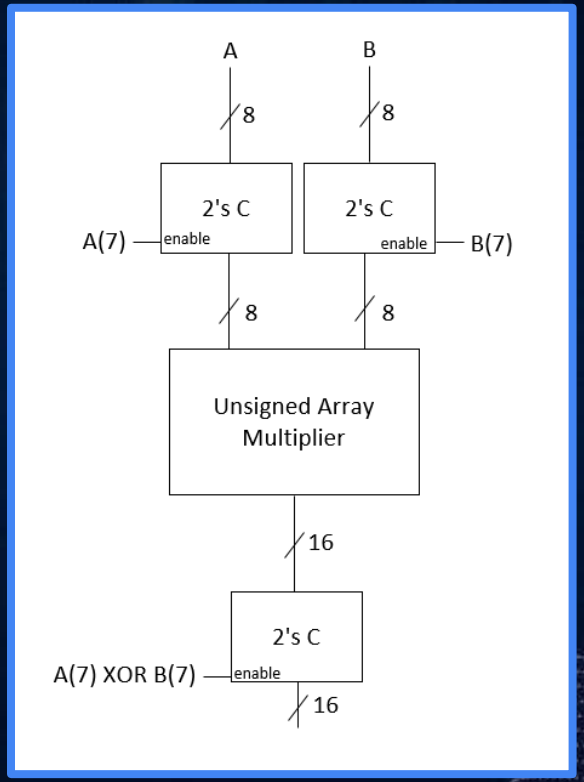
Subtractor



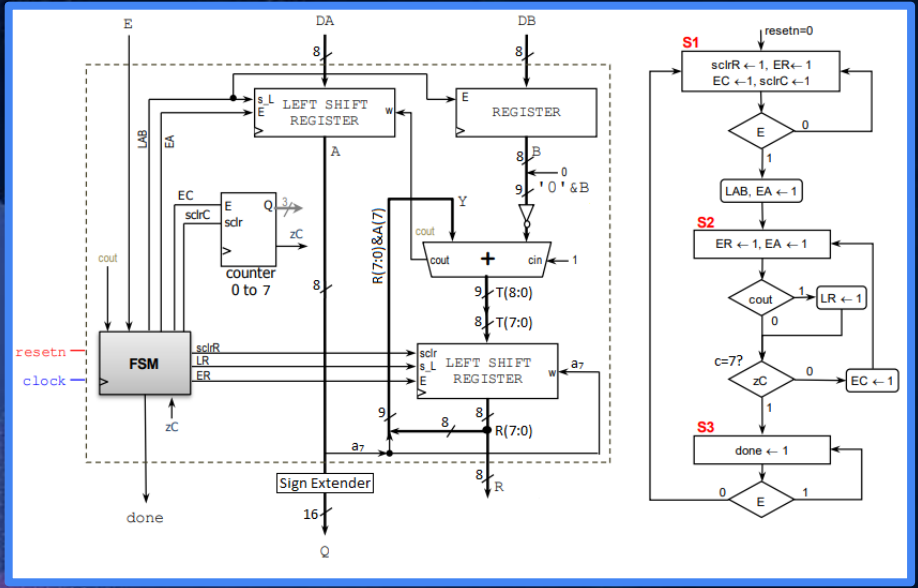
- The subtractor was created using a similar design to the adder with a few small alterations
- The inputs were sign extended to 9-bits to avoid overflow
- The carry in signal is set as a ‘1’, while the concatenation of B(7) and B is inverted to perform the two's-complement operation on the the sign extended “B” signal
- The user must press the keyboard's "-" key to turn on this feature.

Multiplier

- The multiplier component was initially implemented as an unsigned multiplier
- Rather than modifying the array multiplier, another component was created to convert it to a signed multiplier
- The two’s complement component works by performing the two’s complement operation on the incoming signal if enable = ‘1’. If enable is ‘0’ then the original signal is passed through
- The multiply function is used with the “.” key.



Divider



- The divider used a clock, shift registers, a counter, and an FSM to control the circuit
- Similar to the multiplier, the divider was initially unsigned, but was altered using the two's complement with enable component
- To use this function, the divider user must use the keyboard's "/" key. The input of the circuit requires two eight-bit inputs. The output is sixteen bits.

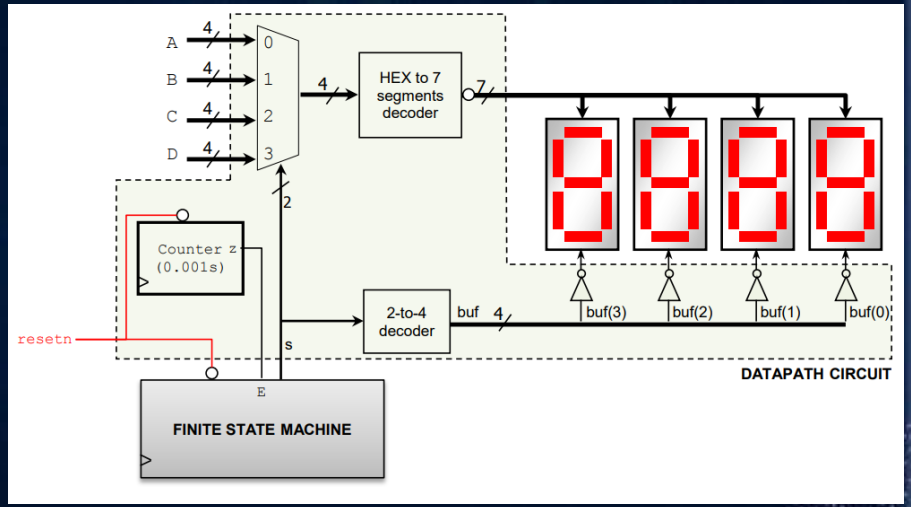
7-Segment Serializer

From Professor Llamocca's Unit 7 notes

The component allows four 7-Segment displays to appear lit; each 7-Segment is on for 1 ms every 4 ms

The output of the calculator was split into four busses, each bus used as inputs to the serializer

This component allows for the display of the 16-bit calculation



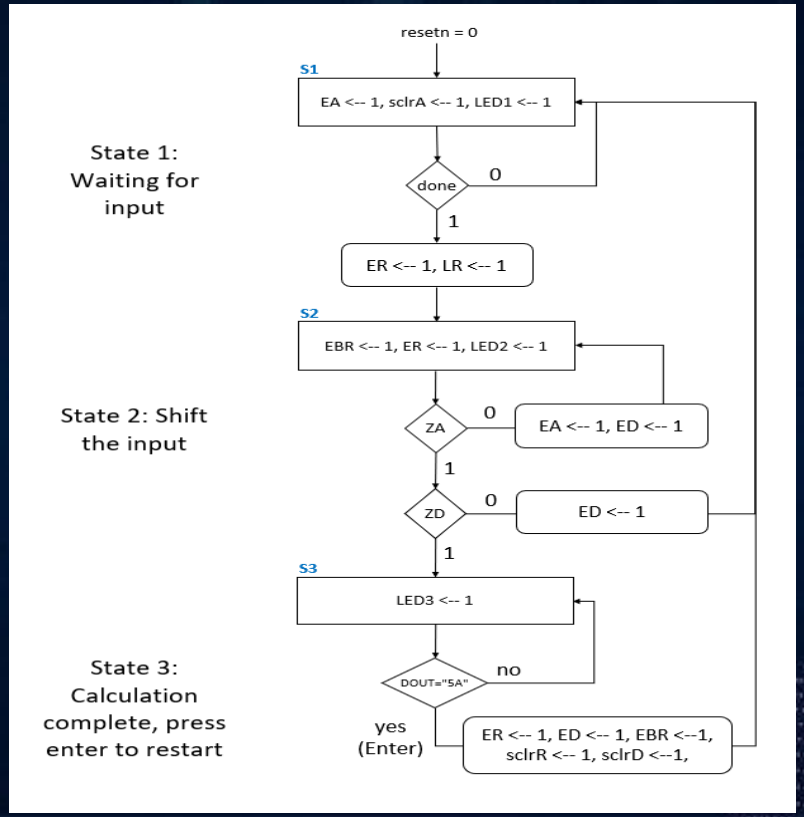
FSM For Calculator

The FSM has three different states:

State 1: Waits for an input; if detected, load 8-bit register with scan code

State 2: Shift scan code data into 40-bit register & return to state 1; if all data is input, then go to state 3

State 3: Outputs calculation and then returns to state 1 when user inputs “Enter”



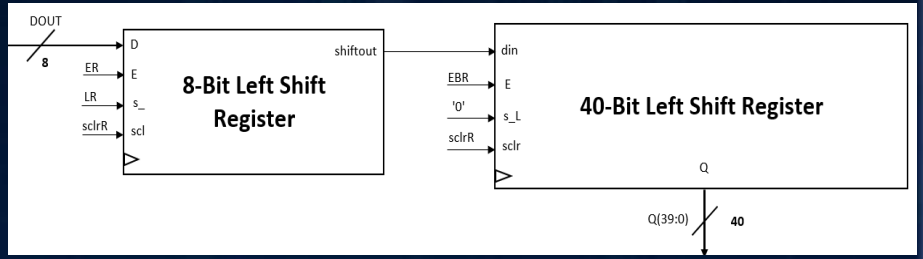
Counters & Registers Explained…

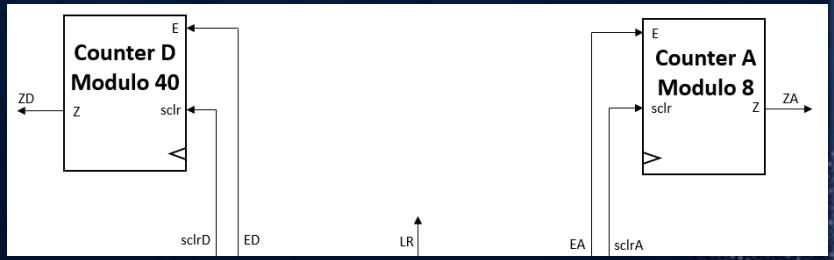
The scan code data (DOUT) is handled through the counters & registers

This is primarily done during state 2 of the FSM

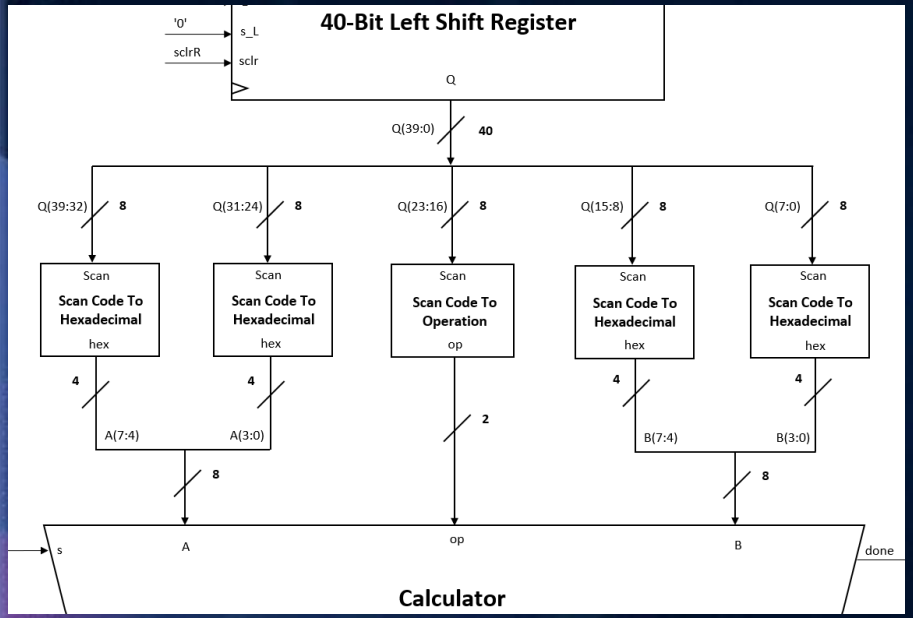
Each input is loaded into the 8-bit register; the input is then left shifted into the 40-bit register (MSB first)

Only 5 inputs allowed: two hexadecimal values, an operation, and two more hexadecimal values





Scan Code Encoders



The 40-bit register stores all the input data as a string of scan code bits

These components encode each scan code as either a hexadecimal value or operation

The outputs act as inputs to the calculator

Thank you!
Any Questions?