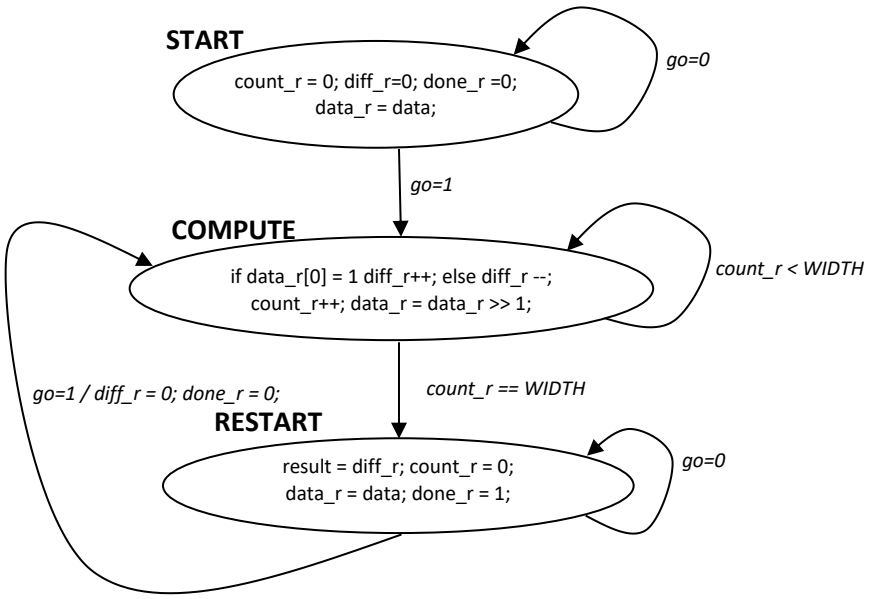
The following pseudo-code illustrates the behavior of the bit difference calculator. If there are X more 1s than 0s in the input, then the output is X. If there are Y more 0s than 1s, then the output is –Y.

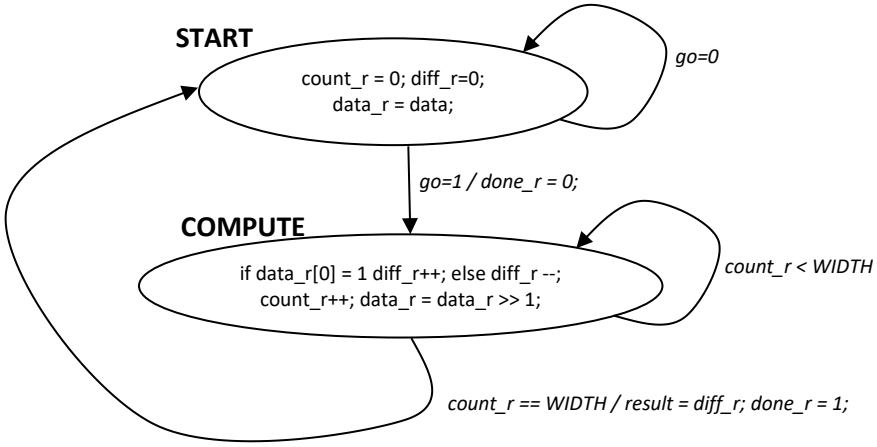
```
Inputs: go, data (WIDTH bits)
Outputs: result (clog2(2*WIDTH+1) bits), done
Reset values: done = 0, result = 0, diff = 0, data_r = 0

while(1) {
      while (go == 0);
      done = 0;
      data_r = data;  // Store input in a register.
                      // This ensures that the code will still
                      // work if the input changes during the loop.
      diff = 0;
      for width iterations {
            if data_r[0] == 1
                  diff ++;
            else
                  diff --;
            data_r = data_r >> 1; // Shift right by 1
      }
      result = diff;
      done = 1;
}
```

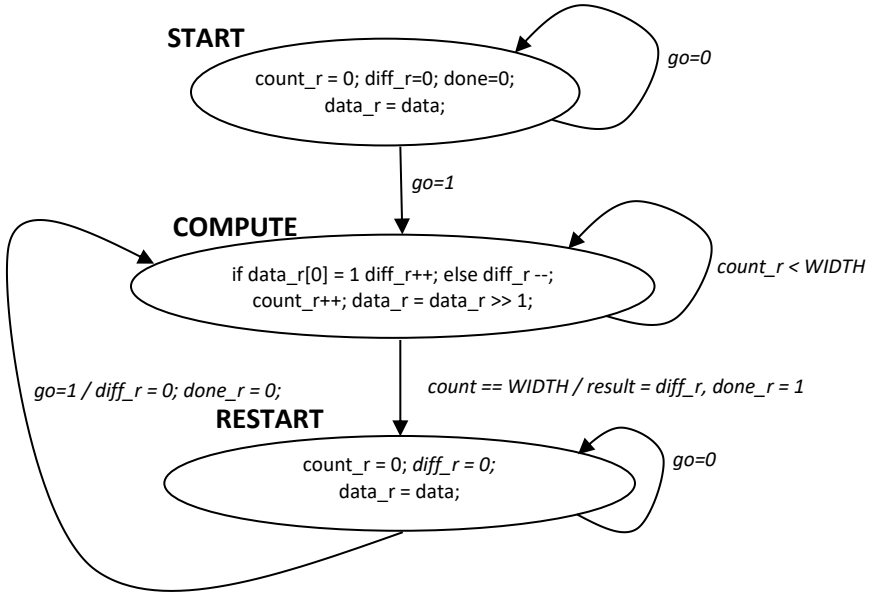
One possible FSMD (used by the fsmd_1p architecture):



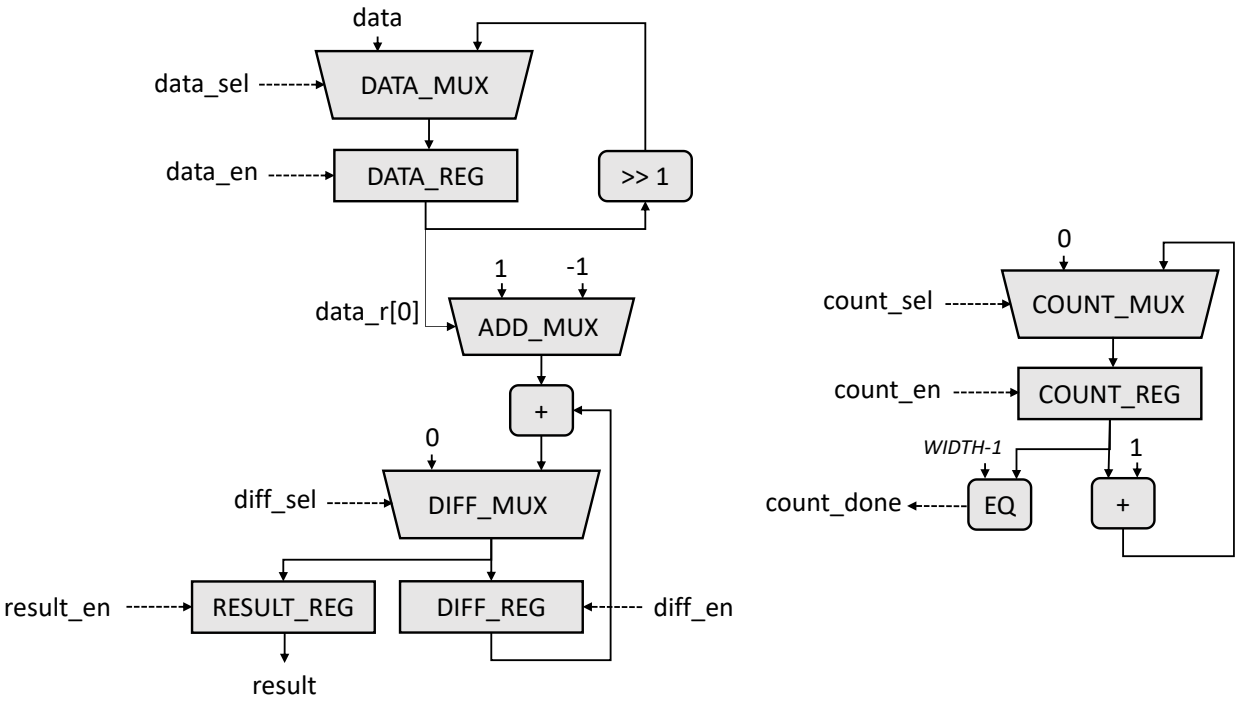
An alternative FSMD with 2 states (used by the fsmd_1p_2 architecture):



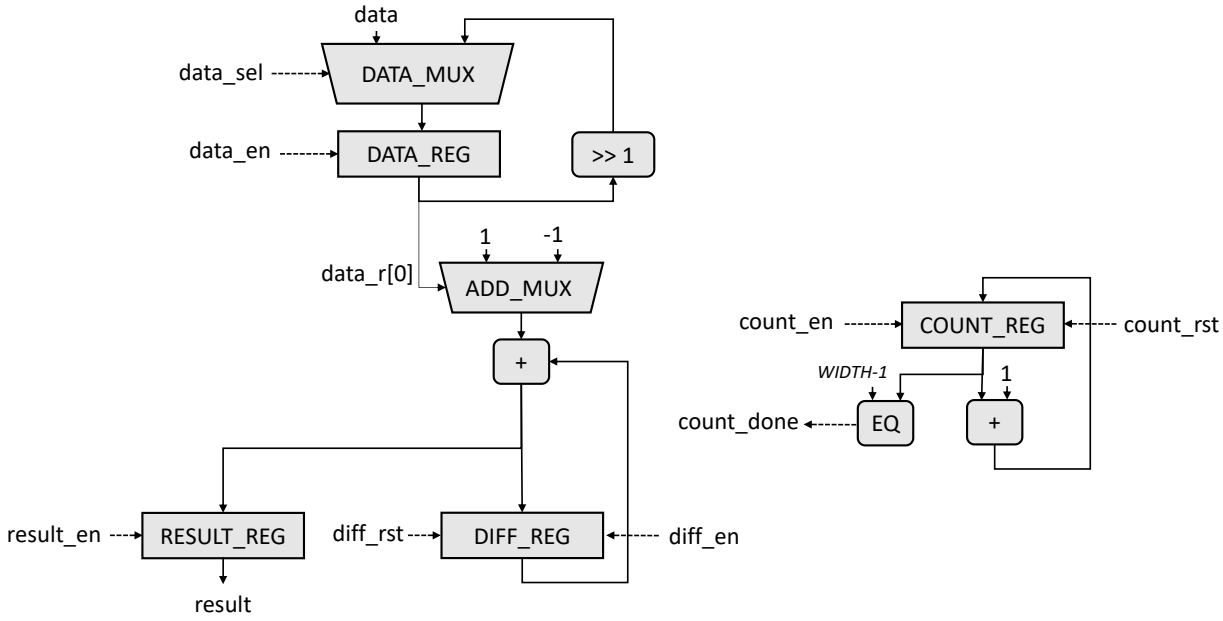
A third possible FSMD:



DATAPATH 1 & 2


data
data_sel
DATA_MUX
data_en
DATA_REG
>> 1
1
-1
data_r[0]
ADD_MUX
+
0
diff_sel
DIFF_MUX
result_en
RESULT_REG
DIFF_REG
diff_en
result
0
count_sel
COUNT_MUX
count_en
COUNT_REG
WIDTH-1
1
count_done
EQ
+

DATAPATH 3


data
data_sel
DATA_MUX
data_en
DATA_REG
>> 1
1
-1
data_r[0]
ADD_MUX
+
count_en
COUNT_REG
count_rst
WIDTH-1
1
count_done
EQ
+
result_en
RESULT_REG
result
diff_rst
DIFF_REG
diff_en