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;
}
```
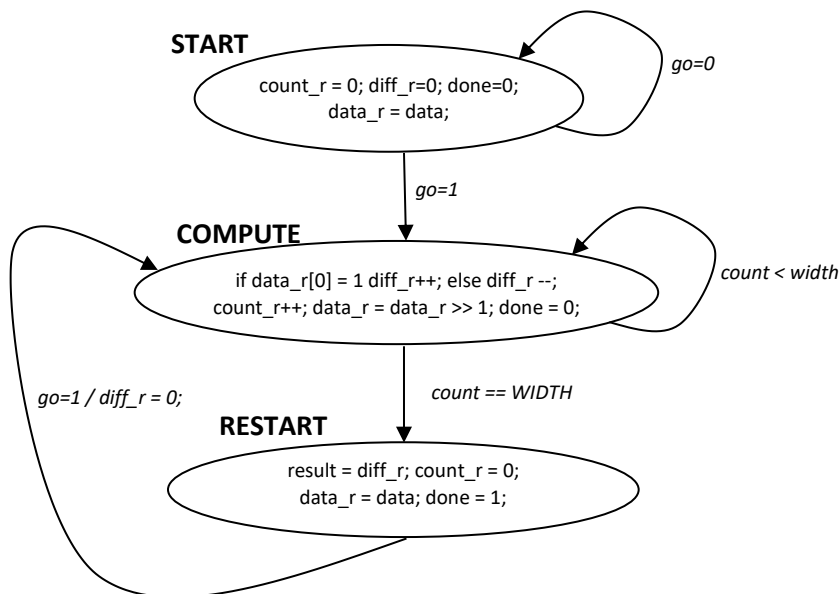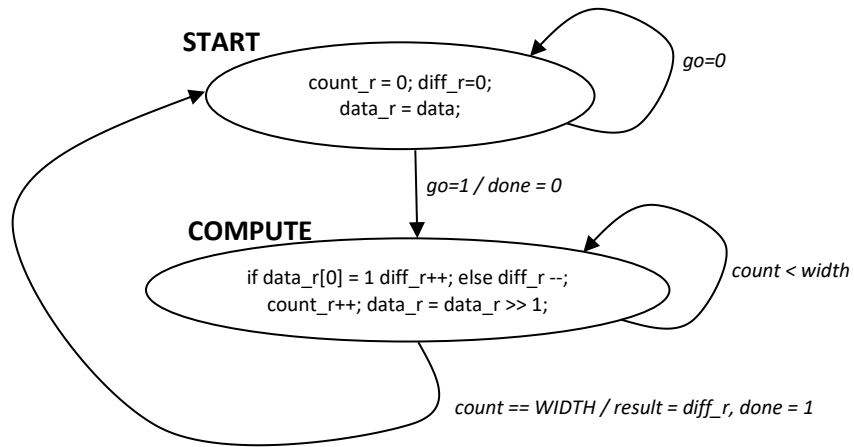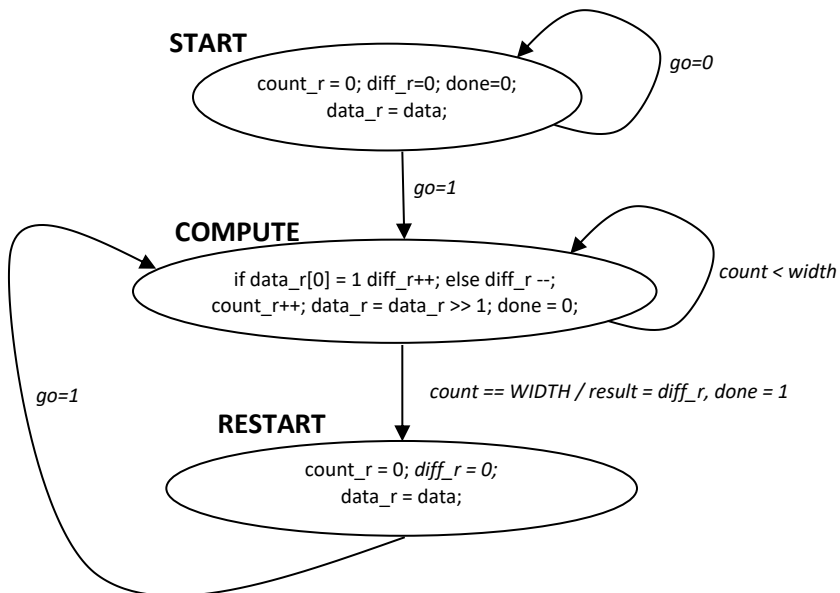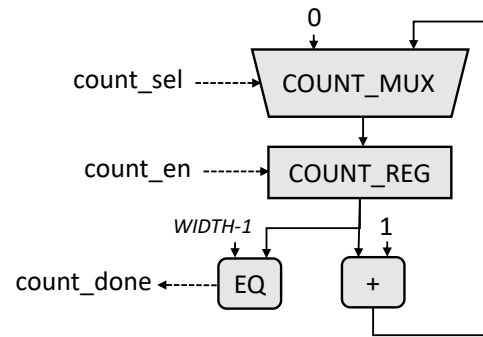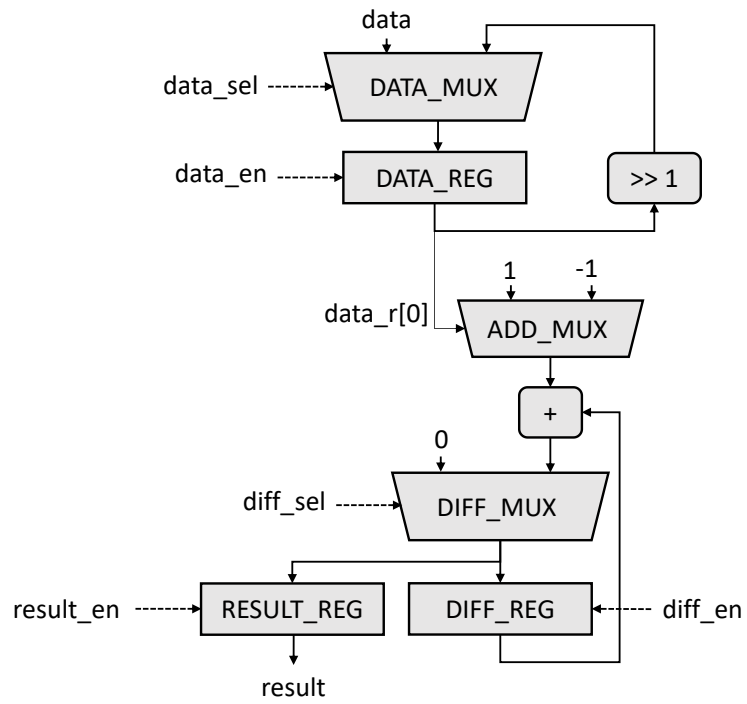
One possible FSMD (used by the bit_diff_fsmd_1p module):

An alternative FSMD with 2 states (used by the bit_diff_fsmd_1p_2 module):

START

count_r = 0; diff_r=0;
data_r = data;

go=0

go=1 / done = 0

COMPUTE

if data_r[0] = 1 diff_r++; else diff_r --;
count_r++; data_r = data_r >> 1;

count < width

count == WIDTH / result = diff_r, done = 1

A third possible FSMD (used by the other modules):

START

count_r = 0; diff_r=0; done=0;
data_r = data;

go=0

go=1

COMPUTE

if data_r[0] = 1 diff_r++; else diff_r --;
count_r++; data_r = data_r >> 1; done = 0;

count < width

go=1

count == WIDTH / result = diff_r, done = 1

RESTART

count_r = 0; *diff_r = 0;*
data_r = data;

DATAPATH 1 & 2

DATAPATH 3