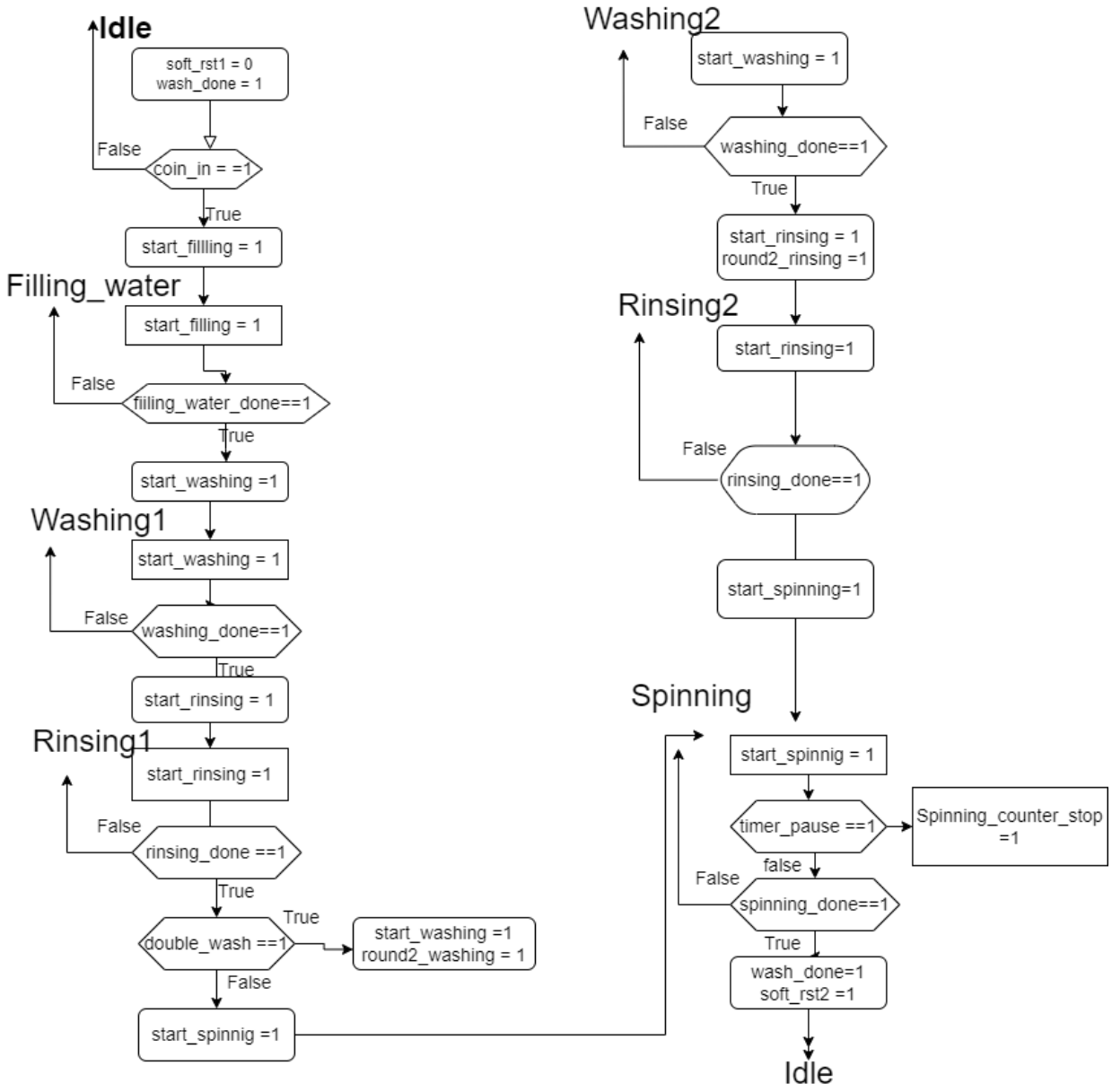
Mixel Assignment

Name	**Mohamed Sayed**
Phone	01110064792
Email	mohamed.sayed991@eng-st.cu.edu.eg

ASMD Chart


Idle
soft_rst1 = 0
wash_done = 1
False
coin_in = =1
True
start_fillling = 1
Filling_water
start_filling = 1
False
fiiling_water_done==1
True
start_washing =1
Washing1
start_washing = 1
False
washing_done==1
True
start_rinsing = 1
Rinsing1
start_rinsing =1
False
rinsing_done ==1
True
True
double_wash ==1
start_washing =1
round2_washing = 1
False
start_spinnig =1
Washing2
start_washing = 1
False
washing_done==1
True
start_rinsing = 1
round2_rinsing =1
Rinsing2
start_rinsing=1
False
rinsing_done==1
start_spinning=1
Spinning
start_spinnig = 1
timer_pause ==1
Spinning_counter_stop
=1
false
False
spinning_done==1
True
wash_done=1
soft_rst2 =1
Idle

Design Flow

- The system has seven states (idle, Fallingwater, washing1, Rinsing1, washing2, Rinsing2, spinning)
- Needed counters (Timers) in my design to achieve the period of every state
- Separated FSM and counters to different modules for (error detection and correction)
- At first the system will be in idle state and all counters are zeros
- When coin in is asserted for one clock cycle the system starts to work
- In my design every counter has start signal which comes from FSM to start calculate the duration for every state
- I had two ideas about this signal the first one is to assert the start signal when entering the state but that will make a clock cycle delay for every state, so I avoided do that and asserted start signal one cycle before entering any state, so I had no wasted clocks but unfortunately, I needed two soft reset signals
- When entering Filling state, the counter of filling start to count until reach the end and assert flag done to FSM which change the state in the next cycle(washing)
- When washing1 state, the counter of filling start to count until reach the end and assert flag done to FSM which change the state in the next cycle(rinsing)
- When rinsing1 state, the counter of filling start to count until reach the end and assert flag done to FSM which change the state in the next cycle(spinning) and finally assert wash done signal
- In case of double wash, the flow will be washing1->rinsing1->washing2->rinsing2->spinning

- Notes about the duration of timers

- There are four frequencies and every state should take specific duration
- that reflects on the number of counts that every counter should which is calculated by formula -> counts = frequency *Time in seconds
- So, I calculated the number of counts (clock cycles) for every counter at different frequencies
- Numbers are in hexadecimal format because they are large
- Chose the width of counter (32 bit) because the multiplication of max frequency and max time can be represented in 32 bit and not smaller
- The design has two soft resets the first one is asserted at idle state to clear all counters except(filling) and the second one is asserted at spinning state to clear Filling water counter all this for the sake of not wasting any clock cycle) so I decided to assert the enable of any counter one clock before entering any state

Testbench

Tests are done at four frequencies but I chose in my testbench (1MHZ)

My test bench contains four tasks

- Initialize to initialize signals
- Reset to make hard reset
- Coin generation to assert coin in for one clock cycle
- Do_read_check to check the output

I made four tests on the design

- Single wash check after 600 seconds (10 minutes)
- Single wash check with timer pause (asserted 20) after 600 (failed because of pause) seconds and after 620 seconds (pass because we waited same amount of time as delay to observe)
- Double wash check after 1020 seconds (17 minutes)
- Double wash check with timer pause (asserted 20) after 1020 (failed because of pause) seconds and after 1040 seconds (pass because we waited same amount of time as delay to observe)
- Made one task to test all these cases using two additional signals (single_double, time_pause)
- When entering the task by inputs (1'b1, 1 , 0)
  That means check if wash done for single wash and no pause
- When entering the task by inputs (1'b1, 1 ,)
  That means check if wash done for single wash and pause
- When entering the task by inputs (1'b1, 0 , 0)
  That means check if wash done for double wash and no pause
- When entering the task by inputs (1'b1, 0 , 1)
  That means check if wash done for single wash and pause
- The scenario made is to wait 10 minutes for single wash (passed) and go to idle
- Generate coin and wait 10 minutes but using delay so will be failed then wait the amount of delay will pass and go to idle
- Generate coin and wait 17 minutes for double wash (passed) and go to idle
- Generate coin and wait 17 minutes but using delay so will fail until wait the amount of delay so than can pass
  Simulation Results

```
VSIM 3> run -all
# READ Operation IS Passed
# READ Operation IS failed
# READ Operation IS Passed
# READ Operation IS Passed
# READ Operation IS failed
# READ Operation IS Passed
# ** Note: $stop    : C:/Users/20111/Desktop/mixel/tstbench.v(58)
#    Time: 3280000020 us  Iteration: 0  Instance: /tstbench
# Break in Module tstbench at C:/Users/20111/Desktop/mixel/tstbench.v line 58
```