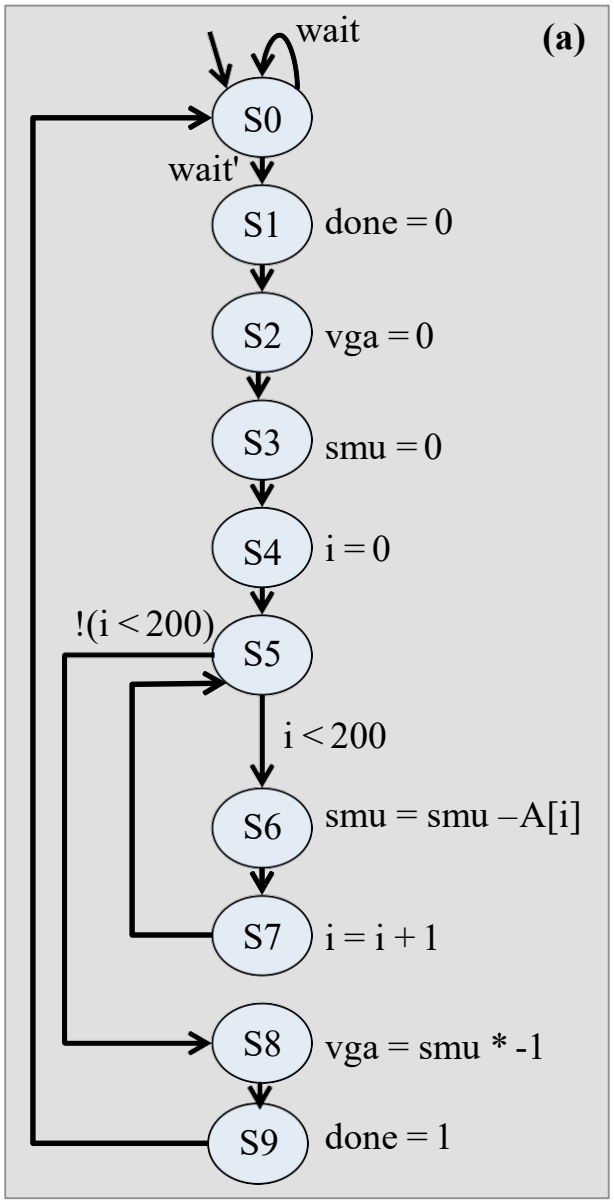
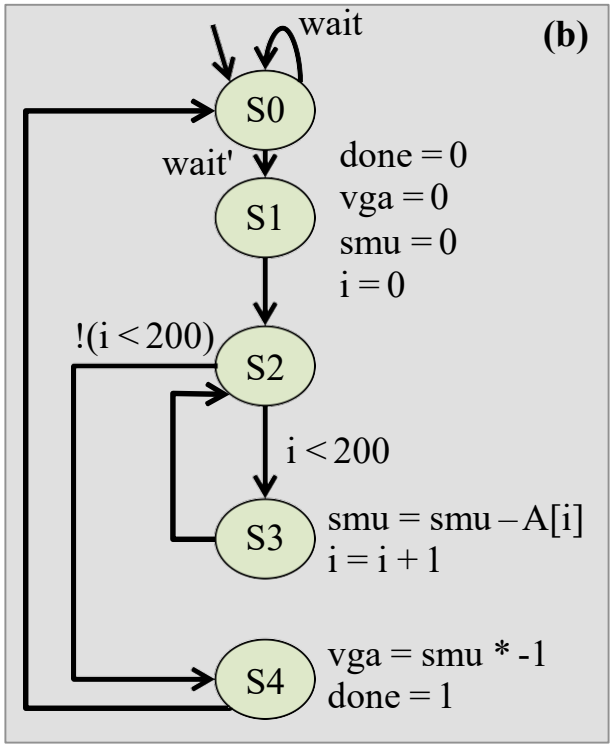
```
Inputs: Int8 A[256], Int1 wait
Outputs: Int8 vga, Int1 done
calc() {
   Int16 smu, i;

   while(1) {
       while(wait);
       done = 0;
       vga = 0;
       smu = 0;
       i = 0;
       while ( i < 200 ) {
          smu = smu - A[i];
          i = i + 1;
        }
        vga = smu * -1;
        done = 1;
    }
}
```



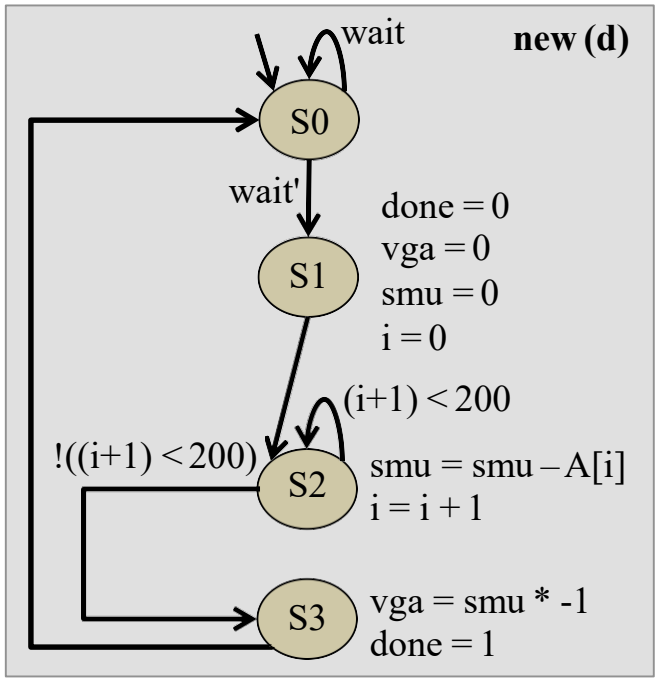

```
Inputs: Int8 A[256], Int1 wait
Outputs: Int8 vga, Int1 done
calc() {
    Int16 smu, i;

    while(1) {
        while(wait);
        done = 0;
        vga = 0;
        smu = 0;
        i = 0;
        while ( i < 200 ) {
            smu = smu - A[i];
            i = i + 1;
        }
        vga = smu * -1;
        done = 1;
    }
}
```




```
Inputs: Int8 A[256], Int1 wait
Outputs: Int8 vga, Int1 done
calc() {
   Int16 smu, i;

   while(1) {
       while(wait);
       done = 0;
       vga = 0;
       smu = 0;
       i = 0;
       while ( i < 200 ) {
          smu = smu - A[i];
          i = i + 1;
        }
        vga = smu * -1;
        done = 1;
    }
}
```




```
Inputs: Int8 A[256], Int1 wait
Outputs: Int8 vga, Int1 done
calc() {
    Int16 smu, i;

    while(1) {
        while(wait);
        done = 0;
        vga = 0;
        smu = 0;
        i = 0;
        while ( i < 200 ) {
            smu = smu - A[i];
            i = i + 1;
        }
        vga = smu * -1;
        done = 1;
    }
}
```

HLSM Tradeoffs: Determine latency (in cycles) and *minimum* datapath components/quantity for each implementation

- S0 to done = 1