Floating Beacon (Concurrent) SynchSM Design Problem Solution

Problem

- A floating ocean beacon transmits data wirelessly to an on-shore receiver. Every 10 seconds, the beacon transmits a sensed temperature. Every 3 seconds, the beacon transmits the beacon's altitude.

- **Input:**
  - **Temperature** is just a 4-bit unsigned number input to **A3-A0.**
  - **Altitude** is a 4-bit unsigned number input to **A7-A4.**

- **Output:**
  - Transmitting the temperature is done by setting B1B0=01, placing the 4-bit temperature on B7-B4, and pulsing B2.
  - Transmitting the altitude is done by setting B1B0=10, placing the 4-bit altitude on B7-B4, and pulsing B2.
  - A pulse must be at least 300 ms high followed by at least 300 ms low.

Solution Overview

1. There are many different legitimate solutions with varying number of tasks. Among the many different solutions, none is more correct than the others.
2. At multiples of 30 seconds, both temperature and altitude must be transmitted. Since we are not trying to use parallel microcontrollers here, one of the two values will be transmitted before the others, as per the task scheduler
3. At most one task should write to a shared variable or output B; however, multiple tasks can write to a queue.
4. We need a 100ms period (GCD of 300ms, 3s, 10s).

Single-Task Solution

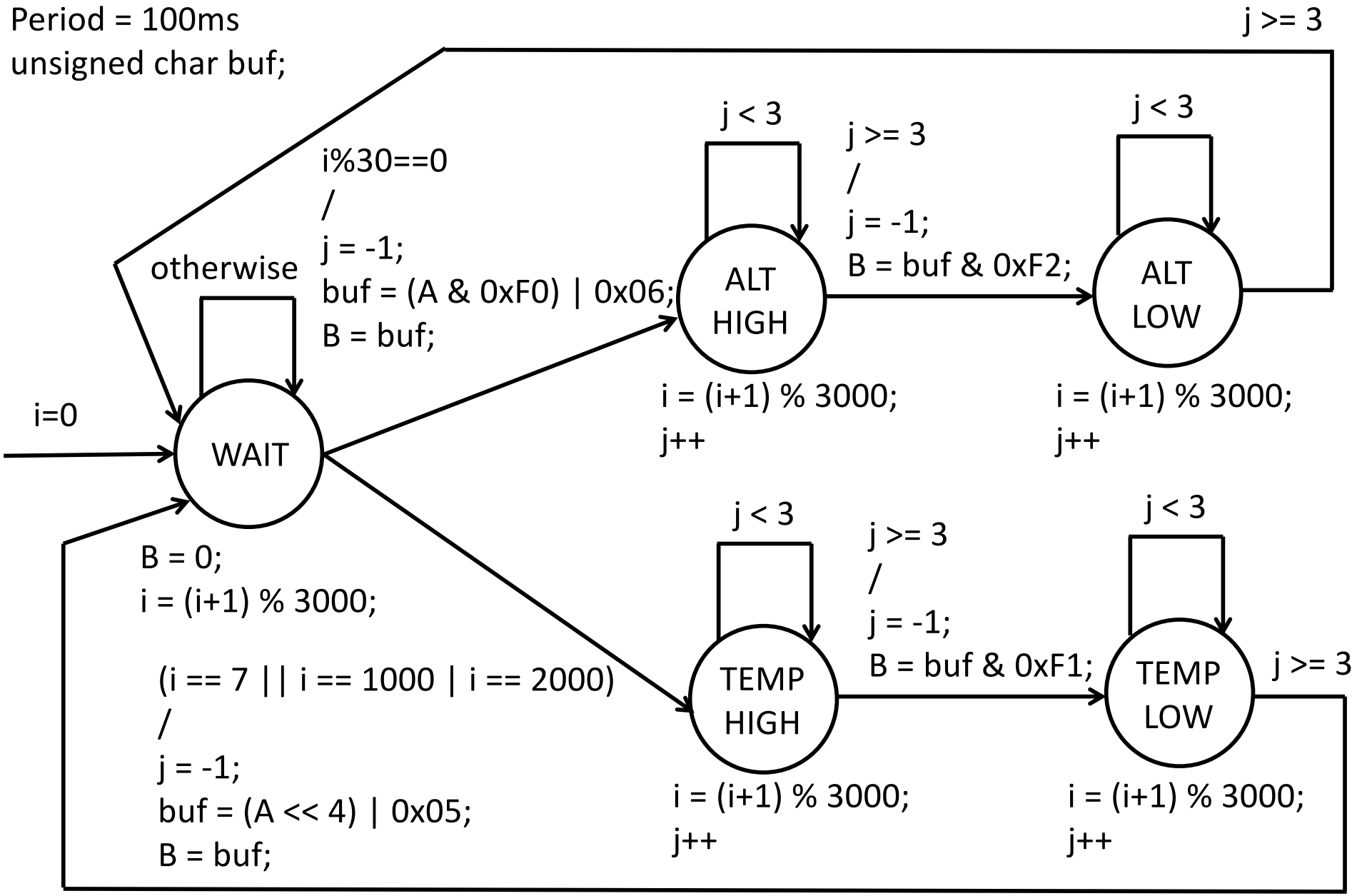
Overview

1. Transmit altitude at 0s, 3s, 6s, 9s, 12s, 15s, 18s, 21s, 24s, 27s, 30s, etc.
2. Transmit temperature at 0s, 10s, 20s, 30s
3. 600ms required to transmit data
4. So for the common time (multiples of 30s, inc. 0)

   4.1 Transmit altitude from 0s to 0.6s, 30s to 30.6s, etc.

   4.2 Transmit temperature from 0.7s to 1.2s, 30.7s to 31.2s, etc.
5. We count up by 100ms increments, so 3s means 30 ticks, 10s means 100 ticks, etc.

Single-Task Solution



Single-Task Solution (Notes)

Altitude

1. Transitioning into the ALT_HIGH state:
   B = (A & 0xF0) | 0x06;
   B7… B4 = A7…A4; B3…B0 = 0110

2. Transitioning into the ALT_LOW state:
   B = B & 0xF2;
   B7…B4 = A7…A4; B3…B0 = 0010

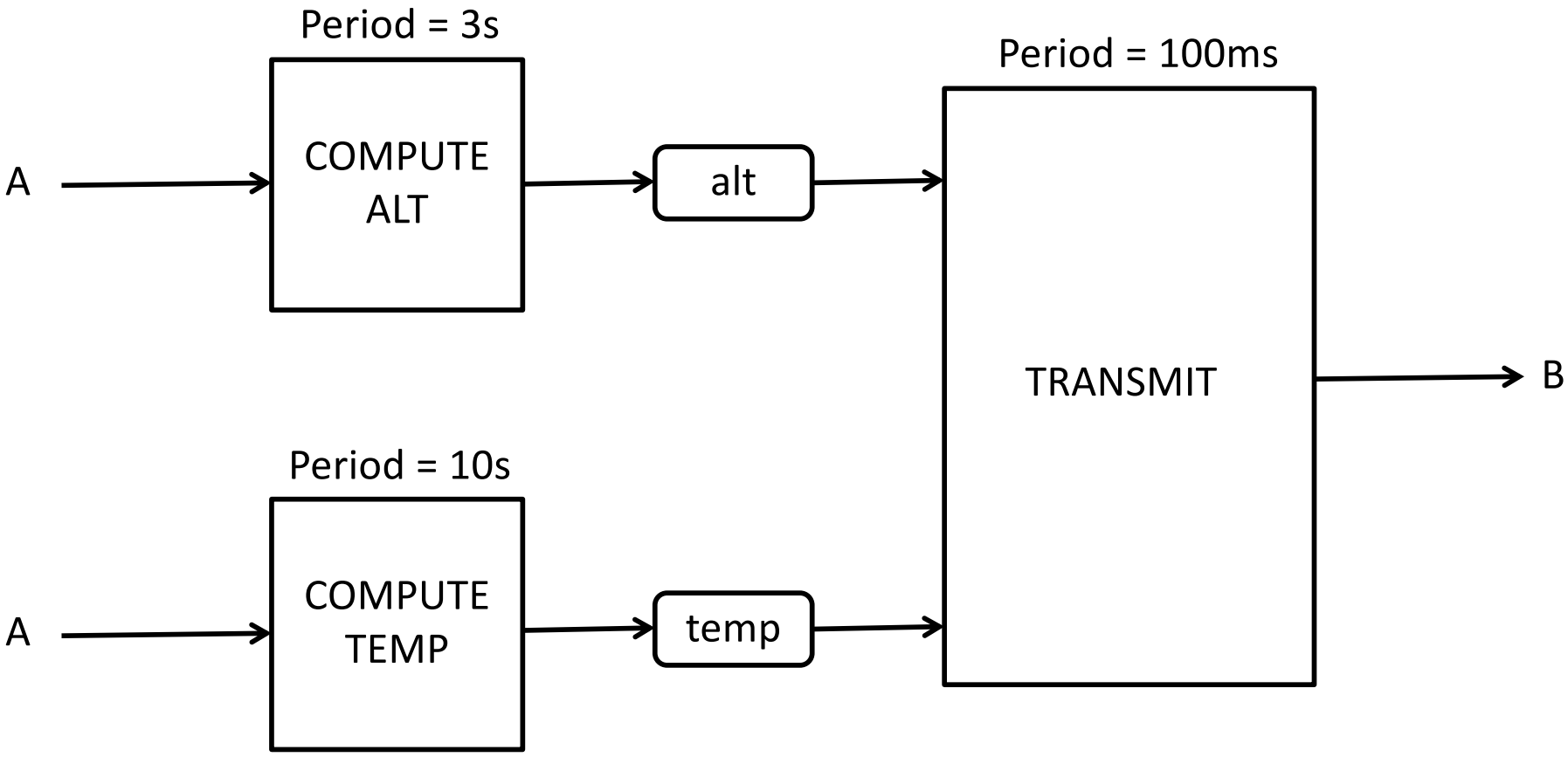
Single-Task Solution (Notes)

Temperature

1. Transitioning into the TEMP_HIGH state:
   B = (A << 4) | 0x05;
   B7… B4 = A3…A0; B3…B0 = 0101

2. Transitioning into the TEMP_LOW state:
   B = B & 0xF2;
   B7…B4 = A3…A0; B3…B0 = 0001

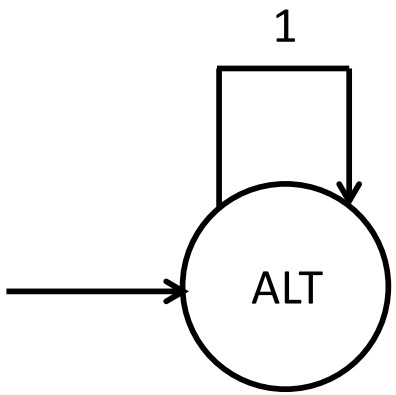
Concurrent SynchSM w/Shared Variables



unsigned char alt;
unsigned char temp;

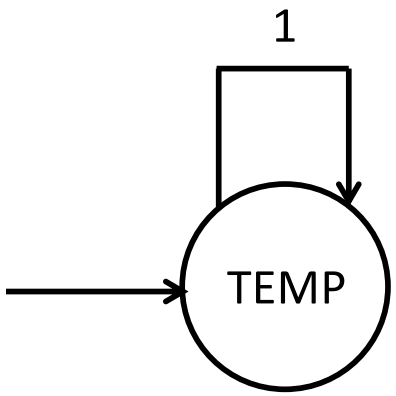
COMPUTE_ALT and COMPUTE_TEMP

Period = 3s



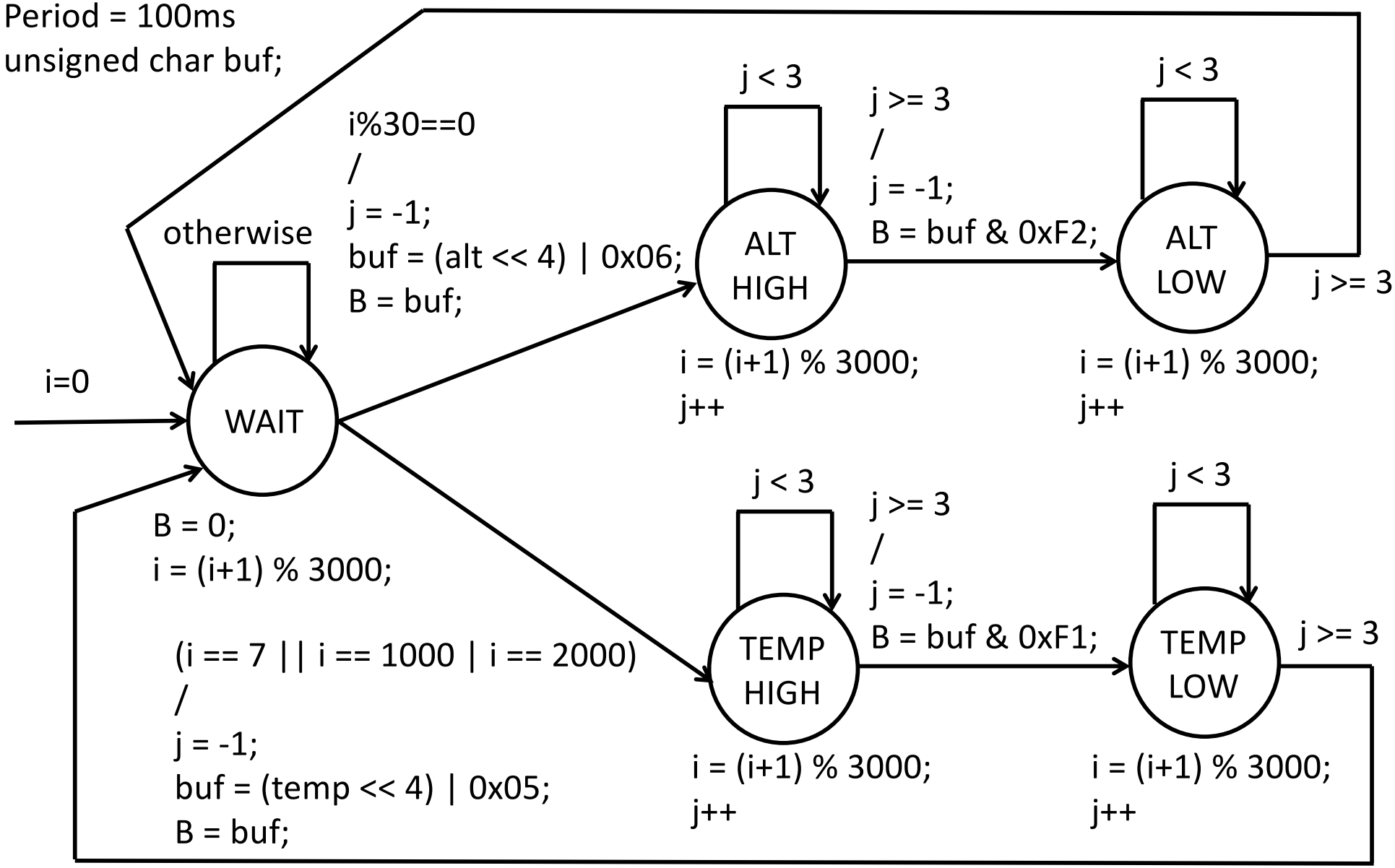
alt = A >> 4;

Period = 10s



temp = A & 0x0F;

TRANSMIT


Period = 100ms
unsigned char buf;
i=0
WAIT
otherwise
i%30==0
/
j = -1;
buf = (alt << 4) | 0x06;
B = buf;
ALT HIGH
j < 3
j >= 3
/
j = -1;
B = buf & 0xF2;
ALT LOW
j < 3
j >= 3
i = (i+1) % 3000;
j++
i = (i+1) % 3000;
j++
B = 0;
i = (i+1) % 3000;
(i == 7 || i == 1000 | i == 2000)
/
j = -1;
buf = (temp << 4) | 0x05;
B = buf;
TEMP HIGH
j < 3
j >= 3
/
j = -1;
B = buf & 0xF1;
TEMP LOW
j < 3
j >= 3
i = (i+1) % 3000;
j++
i = (i+1) % 3000;
j++

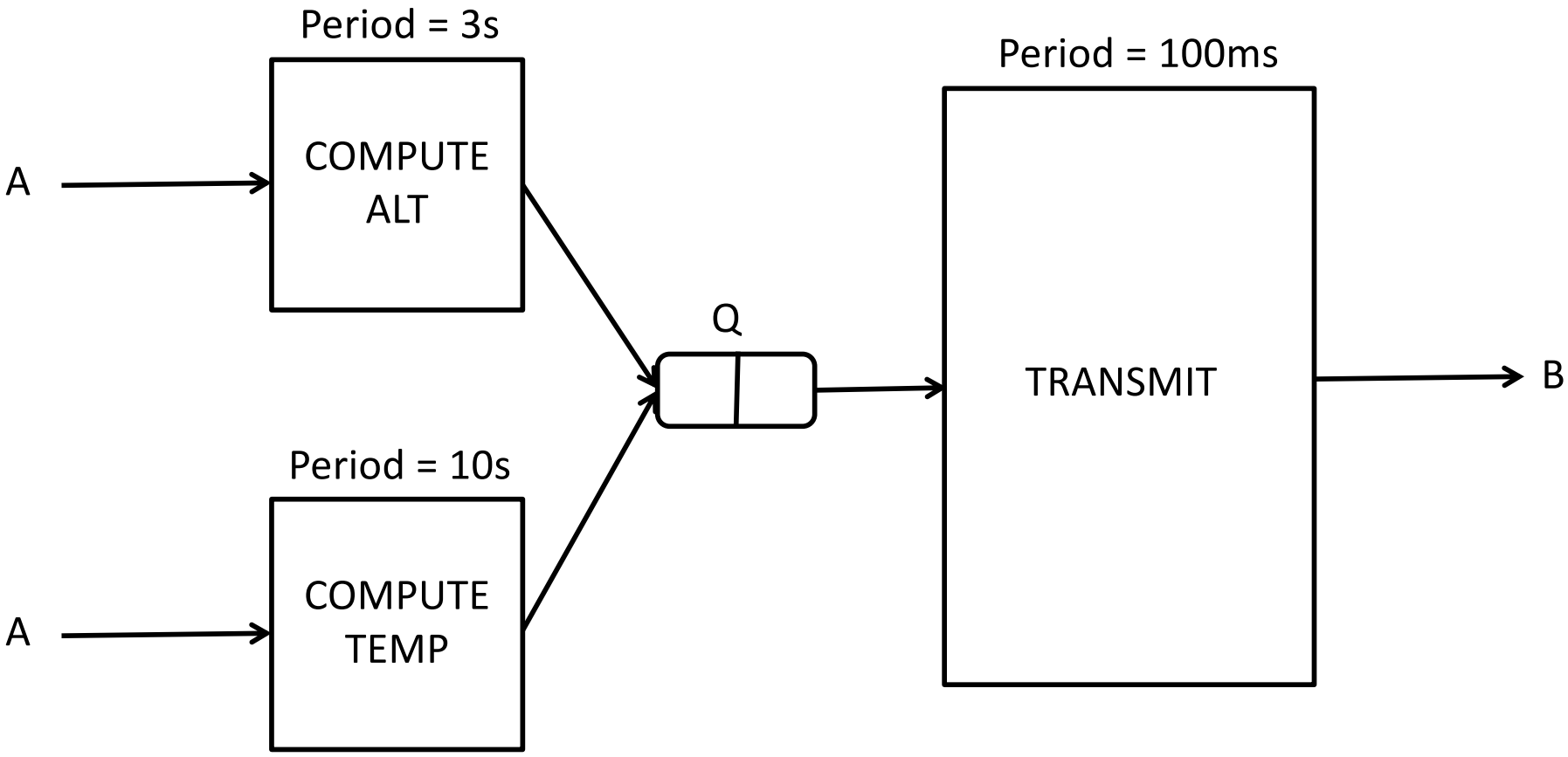
Concurrent SynchSM w/Shared Variables (Notes)

Minimal difference with single-task solution

Still have to remember exactly what time to transmit altitude and temperature

Probably not a good design

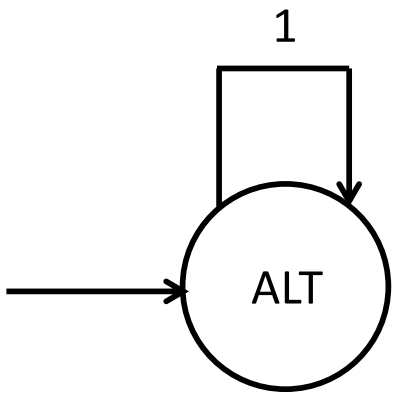
Concurrent SynchSM w/Queues



Q: Two unsigned chars

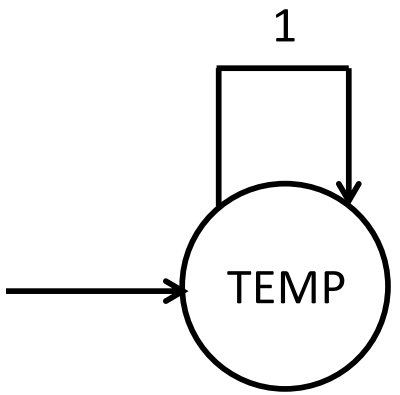
COMPUTE_ALT and COMPUTE_TEMP

Period = 3s



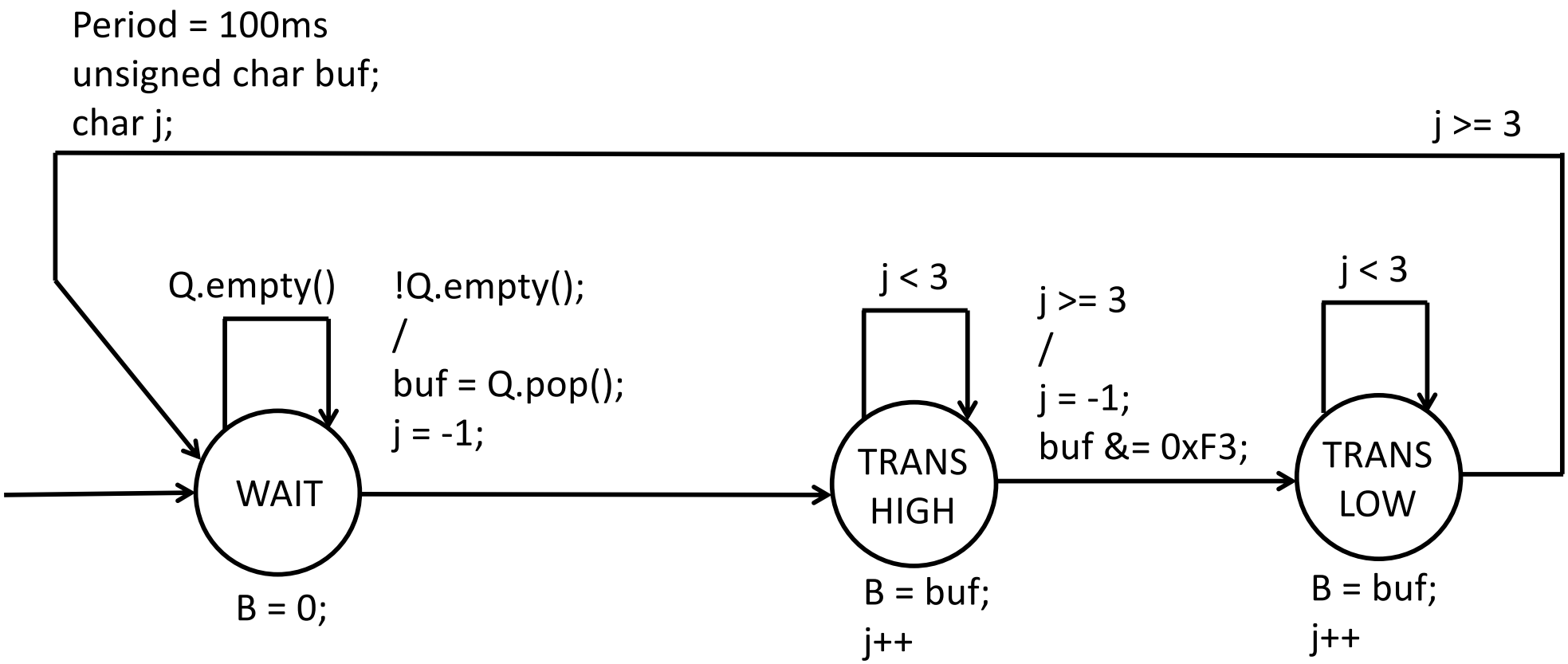
unsigned char alt = A & 0xF0;
Q.push(alt | 0x06)

Period = 10s



unsigned char temp = A << 4;
Q.push(temp | 0x05)

TRANSMIT


Period = 100ms
unsigned char buf;
char j;
j >= 3
Q.empty()
!Q.empty();
/
buf = Q.pop();
j = -1;
WAIT
B = 0;
j < 3
TRANS
HIGH
B = buf;
j++
j >= 3
/
j = -1;
buf &= 0xF3;
j < 3
TRANS
LOW
B = buf;
j++

Concurrent SynchSM w/Queues (Notes)

Queues lead to a simpler design!

Transmit if there is a data element in the queue

- Eliminates the need to track time explicitly
- Popping from a queue removes the element; reading a shared variable does not remove it

Multiple tasks may write to a queue!

x = x & 0xF3; sets bit B2 to 1 for both temperature and altitude