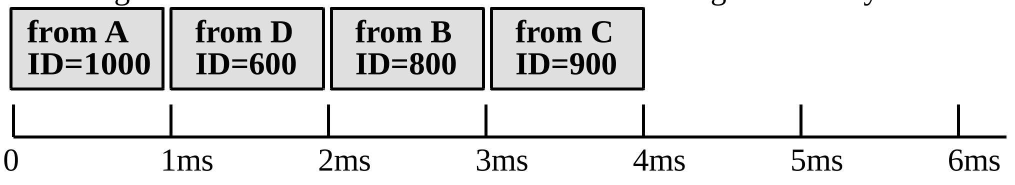
Jonathan W. Valvano
November 8, 2006, 1 to 1:50pm

(15) Question 1. The message from A goes first because no other messages are ready at time = 0. Once a message is started it is allowed to finish. During the 0 to 1 ms time, computers B,C,D all setup messages to be transmitted. Right at 1ms all three messages begin, but ID=600 exerts dominance over the other two because it has a smaller value (0 dominates over 1).At 2ms the messages B (ID=800) and C (ID=900) begin, but ID=800 exerts dominance over ID=900. Finally, at 3ms, the message from C can be send because no other messages are ready.



(10) Question 2. The CAN transfer rate of 100,000 bits/sec could be used to see if the distributed system works or not. It does work, because, the peak bandwidth

= 5 (bytes/frame)*100,0000 (bits/sec)/(11+5*8+36 (bits/frame))
= 5 (bytes/frame)*100,0000 (bits/sec)/(87 (bits/frame))
= 5 (bytes/frame)*100,0000 (bits/sec)/(87 (bits/frame)) = 5747 bytes/sec

However, the ***actual*** bandwidth of this network is determined by the rate at which messages are actually sent.

5 (bytes/msg) * 8(nodes) * 10 (msg/node/sec) = 400 bytes/sec

(25) Question 3.
Part a) One approach to this problem is to employ the existing two values from the CAN protocol, then add two more. First, 3.75/1.25 are chosen for 00, because 00 has ultimate dominance. 2.5/2.5 are chosen for 11 because 11 has ultimate recessive. The other two are selected in between (about every 0.42V)

Digital input	CANH	CANL	CANH+CANL
0 0	3.75	1.25	5.00 V
0 1	3.34	1.66	5.00 V
1 0	2.92	2.08	5.00 V
1 1	2.50	2.50	5.00 V

A second approach to this problem is to employ values equidistance in the 1 to 4V range.

Digital input	CANH	CANL	CANH+CANL
0 0	4.0	1.0	5.00 V
0 1	3.5	1.5	5.00 V
1 0	3.0	2.0	5.00 V
1 1	2.5	2.5	5.00 V

Part b) The values of CANH are 2.5, 2.92, 3.34 and 3.75 (the second protocol is 2.5, 3, 3.5 and 4). Dominance is implemented by setting the actual CANH signal to the highest value selected from the devices attempting to transmit. Similarly, dominance requires setting the actual CANL signal to the lowest value selected from the devices attempting to transmit.

Node A	CANH (A)	Node B	CANH (B)	CANH	
0 0	3.75	0 0	3.75	3.75	**A=B**
0 1	3.34	0 0	3.75	3.75	**B>A**
1 0	2.92	0 0	3.75	3.75	**B>A**
1 1	2.50	0 0	3.75	3.75	**B>A**
0 1	3.34	0 1	3.34	3.34	**A=B**
1 0	2.92	0 1	3.34	3.34	**B>A**
1 1	2.50	0 1	3.34	3.34	**B>A**
1 0	2.92	1 0	2.92	2.92	**A=B**
1 1	2.50	1 0	2.92	2.92	**B>A**
1 1	2.50	1 1	2.50	2.50	**A=B**

(25) Question 4.

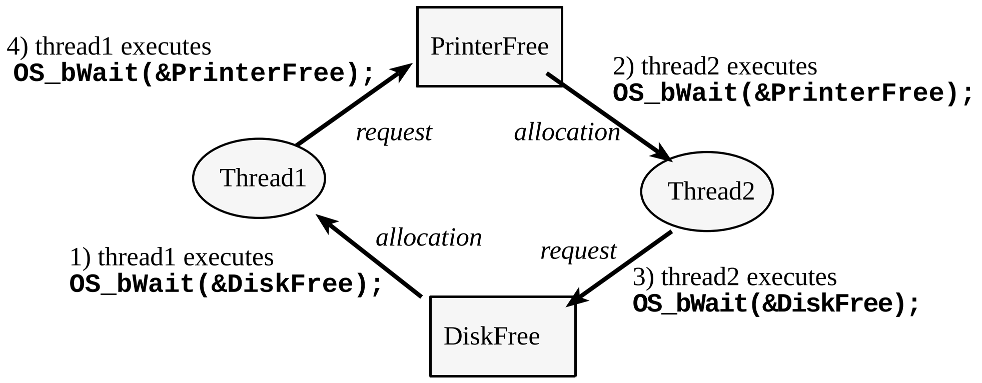
Part a) There are no critical sections, because the read modify write sequence is atomic.

Part b) *The first thread to execute the* ___ `minm` ___ *instruction will be the one to return from* `OS_bWait`*, while the other thread(s) will spin in the* `minm/bcc` *loop.*

Part c) This is better because it does not disable interrupts. Disabling interrupts increases latency of real-time interfaces that use interrupts to service requests.

(25) Question 5.

Part a)



Part b) A closed path in the shape of a **circle** in the resource allocation graph defines a deadlock.

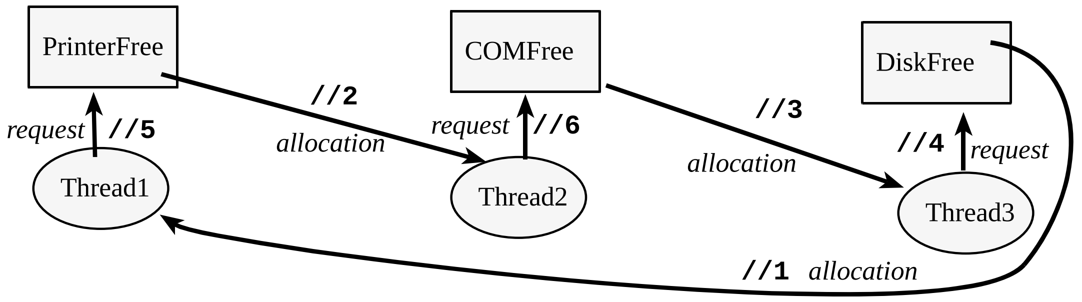
"There is a deadlock if and only if the resource allocation graph contains a shape in the form of a **_cycle_***".*

Part c)

```
void thread1(void){
  OS_bWait(&DiskFree);    //1
  OS_bWait(&PrinterFree); //5
// use disk and printer
  OS_bSignal(&DiskFree);
  OS_bSignal(&PrinterFree);
}
```

```
void thread2(void){
  OS_bWait(&PrinterFree); //2
  OS_bWait(&COMFree);     //6
// use printer and COM port
  OS_bSignal(&PrinterFree);
  OS_bSignal(&COMFree);
}
```

```
void thread3(void){
  OS_bWait(&COMFree);  //3
  OS_bWait(&DiskFree); //4
// use disk and COM port
  OS_bSignal(&COMFree);
  OS_bSignal(&DiskFree);
}
```



PrinterFree
COMFree
DiskFree
request
//5
//2
allocation
request
//6
//3
allocation
//4
request
Thread1
Thread2
Thread3
//1 allocation