## IPC – Winter 2012

Refer slide deck 05\_nto\_ipc\_r11.pdf for answers.

### General

1. List three IPC’s unique to QNX. (slide 3)  
   message passing, pulses, and Persistent Publish and Subscribe
2. List three POSIX/UNIX IPC’s (slide 3)

Semaphores, message queues, shared memory

### Message Passing

1. With help of a simple illustration briefly explain the IPC using message passing. Clearly label the three calls used in message passing. (refer slide 5)

MsgSend

MsgReceive

MsgReply

1. Explain what is meant by the term **blocked,** when referring to message passing.

Blocked means that execution stops and waits for either a message to be sent (server) or waits for a message reply (client)

1. Consider this scenario. At time t1 the MsgReceive call is used, later at time t2   
   (t2 > t1), MsgSend is used at client. Explain the status of the server when MsgReceive is issued. (refer slide 7)

MsgReceieve is blocked until the it receives a message from the client.

1. Explain what is meant by the term send blocked, reply blocked and receive blocked when referring to message passing. Support your answer with a diagram which includes a time domain. Refer Slide 8.

Send Blocked – When the MsgSend is waiting for a connection to the server

Reply Blocked – After sending the message, the client is blocked and waits for the server to send a response

Receive Blocked – When the server is waiting for the client to send a msg.

1. True/False. A server can connect to only one client.  
   False
2. True/False. A client can connect to only one server.  
   False
3. Explain how errno works in older Unix functions and “modern” POSIX functions. Refer slide 21.

In older Unix Functions, errno would be set with an error value, and -1 returned in the function to indicate an error. Modern POSIX functions will return 0 on success, and an errno value on fail.

1. True/False. During MsgSend the kernel passes the address of the first byte of the message to the server. (slide 24)

False, the whole message is copied

1. A client sends a message using MsgSend(), the server replies with no data. When would a server use this feature? (slide 25)

In cases when only a status is required by the client

1. “QNX Neutrino messaging is synchronous”. Explain.  
     
   Messages will cause the calling thread to block until it has been successfully completed. This is synchronous.
2. If you used MsgSend() in the main() function of a client or used MsgReceive() in the main() function in server, it would block; until the client received a reply it would remain blocked. How would you solve this situation? (slide 34)

Send the message via a new thread.

1. Explain the purpose of name\_open() and name\_attach()? (slide 46)  
     
   name\_open() and name\_attach() are ways of connecting to a server without knowing its PID.

Pulses

1. Sending a pulse is non-blocking for the sender. Explain

When sender sends the pulse, it does not wait for a response from the server, and so execution is not blocked.

1. A pulse can have
2. True/False. There is no MsgReply() for a pulse.  
   true
3. What is the return value from MsgReceive() when (refer slide 37)

0 for pulses

Greater then 0 for messages

1. True/False. Pulses are asynchronous.

true

1. How is MsgReceivePulse() different from MsgReceive()? Would you use both? Describe a situation when you would use both.

MsgReceivePulse() will only receive pulses. You might use both if you wanted both regular messages, and critical changes in state from the client. You might have a separate thread monitoring for critical state changes via a pulse;

Shared Memory

Refer : Slides 132 to 140

1. Explain how shared memory works.  
     
   Shared memory is a virtual address space that maps to physical memory, and can be accessed by multiple processes. There is no guarantee that the memory would be contiguous .
2. What is an advantage of shared memory? What are two disadvantages of shared memory?  
     
   An advantage of shared memory is that both processes have direct access to the data, and so no data is being needlessly copied. Unfortunately, this also means that the process writing to the memory doesn’t know when it is safe to write, and the one reading doesn’t know when it is safe to read.
3. Explain the synchronization problem in shared memory. Suggest a possible solution to the problem.   
     
   Mentioned above. A solution would be to use semaphores or pulses to control access to the block of memory. Atomic function would work as well, or mutexes.
4. Give an example of an application where shared memory is used effectively.

### Event Delivery

Refer

* slide deck 05\_nto\_ipc\_r11.pdf, Slides 117 to 118
* QNX help on structure sigevent, MsgDeliverEvent()

1. What is meant by an *event* in QNX? Give examples of events. Give examples of entities that deliver events?  
     
   An event in QNX is a form of notification. Pulses, signals, message passing. Interrupt, timer, application.
2. Give examples of how events are delivered.   
     
   messages, pulses
3. What does the function InterruptWait() do?   
     
   Waits for a hardware interrupt
4. Name the structure that stores properties of an event?

sigevent

1. Identify the entity that would usually initialize the event structure
   1. **Client**
   2. Server
   3. Kernel
   4. File System
   5. procnto
2. Identify two ways the event structure can be initialized.

Manually or using macros

1. Draw a simple fence diagram with two entities that would send and receive events.
2. Draw a fence diagram between a client and server to illustrate *and* explain how an event is delivered. Indicate the sigevent structure’s origination and delivery, MsgSend(), its subsequent replies and MsgDeliverEvent(). How does the server associate MsgDeliverEvent() to the initial request?
3. Event notifications are delivered through
   1. A **channel**
   2. A pipe
   3. A message queue
   4. A socket
   5. A connection ID