

---

Subject: Re: Non-Blocking I/O

Posted by [thompson](#) on Thu, 11 Feb 1999 08:00:00 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

ashmall@my-dejanews.com (Justin Ashmall) writes:

> Just a thought, but would an FSTAT on the unit number give you any information  
> as to whether there was data waiting to be read?

> Justin

Probably not. I've dealt with situations where we've had to read from a file which was open for write by another process. As I recall, the behavior of FSTAT was somewhat flakey under those conditions.

The situation we were dealing with was to read an incoming spacecraft telemetry stream. Since there already was a process (written in C) which was archiving the telemetry stream into data files, what we ended up doing was to simply read those files while they were still being written. That way, we avoided the whole pipe/fifo business. Sounds like that wouldn't help you, though.

Our original scheme was to use a two-way socket connection between IDL and a C process which was handling telemetry reception. IDL would send out a request for data to the socket, and the C process would either respond with a packet, or with a "no-data-yet" message. That way, IDL would always read back something.

> In article <36C2E662.F81D5072@Physik.Uni-Marburg.De>, Ruediger Kupper

> <Ruediger.Kupper@Physik.Uni-Marburg.De> wrote:

>> Hi!

>>

>> This is a question regarding Inter Process Communication

>> (IPC) in a UNIX environment:

>>

>> Is there any way to tell IDL not to wait for the next

>> incoming data when reading from a file?

>> Attempting to read from an IPC channel (a pipe or fifo)

>> using READF or READU will cause IDL to hang until this read

>> attempt is successful. (Pipes or fifos do not produce an

>> EOF-signal unless they are explicitly closed by all sending

>> processes.) This blocking behaviour is undesired if you need

>> to check more than one IPC channel for any waiting data, or

>> if you want to incorporate such a check into an event loop.

>>

>> The only solution I can think of is to provide support for

>> non-blocking I/O by a set of C-routines which could be

>> linked to IDL via CALL\_EXTAEARNAL, but I would prefer using

>> any "pure IDL" concept.  
>>  
>> If anyone out there ran into the same problem, this person  
>> could make a poor, frustrated IDL-programmer very happy by  
>> posting me a little hint...  
>>  
>> Best regards,  
>> Ruediger.  
>>  
>>

---

---

Subject: Re: Non-Blocking I/O  
Posted by [ashmall](#) on Thu, 11 Feb 1999 08:00:00 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

Just a thought, but would an FSTAT on the unit number give you any information as to whether there was data waiting to be read?

Justin

In article <36C2E662.F81D5072@Physik.Uni-Marburg.De>, Ruediger Kupper  
<Ruediger.Kupper@Physik.Uni-Marburg.De> wrote:

> Hi!  
>  
> This is a question regarding Inter Process Communication  
> (IPC) in a UNIX environment:  
>  
> Is there any way to tell IDL not to wait for the next  
> incoming data when reading from a file?  
> Attempting to read from an IPC channel (a pipe or fifo)  
> using READF or READU will cause IDL to hang until this read  
> attempt is successful. (Pipes or fifos do not produce an  
> EOF-signal unless they are explicitly closed by all sending  
> processes.) This blocking behaviour is undesired if you need  
> to check more than one IPC channel for any waiting data, or  
> if you want to incorporate such a check into an event loop.  
>  
> The only solution I can think of is to provide support for  
> non-blocking I/O by a set of C-routines which could be  
> linked to IDL via CALL\_EXTAEARNAL, but I would prefer using  
> any "pure IDL" concept.  
>  
> If anyone out there ran into the same problem, this person  
> could make a poor, frustrated IDL-programmer very happy by  
> posting me a little hint...  
>

> Best regards,  
> Ruediger.  
>  
>

---

Subject: Re: Non-Blocking I/O  
Posted by [Ruediger Kupper](#) on Fri, 12 Feb 1999 08:00:00 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

William Thompson wrote in response to ashmall@my-dejanews.com (Justin Ashmall):

>> Just a thought, but would an FSTAT on the unit number give you any information  
>> as to whether there was data waiting to be read?  
>  
>> Justin  
>  
> Probably not. I've dealt with situations where we've had to read from a file  
> which was open for write by another process. As I recall, the behavior of  
> FSTAT was somewhat flakey under those conditions.

Exactly. FSTAT -seems- to be just the IDL function that should do the job, but unfortunately it gives absolutely no hint in this case.  
FSTAT results do look like

```
** Structure FSTAT, 12 tags, length=36:
UNIT          LONG          100
NAME          STRING      '/homes/kupper/IPC/fifo'
OPEN          BYTE         1
ISATTY        BYTE         0
ISAGUI        BYTE         0
INTERACTIVE   BYTE         0
READ          BYTE         1
WRITE         BYTE         0
TRANSFER_COUNT LONG          1
CUR_PTR       LONG        -1
SIZE          LONG         0
REC_LEN       LONG         0
```

regardless of any waiting or not waiting data.

Good thought Justin, anyway!

> The situation we were dealing with was to read an incoming spacecraft telemetry  
> stream. Since there already was a process (written in C) which was archiving  
> the telemetry stream into data files, what we ended up doing was to simply read  
> those files while they were still being written. That way, we avoided the  
> whole pipe/fifo business. Sounds like that wouldn't help you, though.

>  
> Our original scheme was to use a two-way socket connection between IDL and a C  
> process which was handling telemetry reception. IDL would send out a request  
> for data to the socket, and the C process would either respond with a  
> packet, or with a "no-data-yet" message. That way, IDL would always read back  
> something.

Okay, so there seems to be no way around using some intermediary C-Routines which handle reception.

IDL just doesn't support Inter Process Communication...

Thank you both for your help.

Best regards,  
Ruediger.

---

Subject: Re: Non-Blocking I/O  
Posted by [ashmall](#) on Sat, 13 Feb 1999 08:00:00 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

In article <36C4195B.A3E6CF3C@Physik.Uni-Marburg.De>, Ruediger Kupper  
<Ruediger.Kupper@Physik.Uni-Marburg.De> wrote:  
> William Thompson wrote in response to ashmall@my-dejanews.com (Justin Ashmall):  
>  
>>> Just a thought, but would an FSTAT on the unit number give you any  
> information  
>>> as to whether there was data waiting to be read?  
>>  
>>> Justin  
>>  
>> Probably not. I've dealt with situations where we've had to read from a file  
>> which was open for write by another process. As I recall, the behavior of  
>> FSTAT was somewhat flakey under those conditions.  
>  
> Exactly. FSTAT -seems- to be just the IDL function that should do the job, but  
> unfortunately it gives absolutely no hint in this case.  
> FSTAT results do look like  
>

I thought as much! I actually posted a message a short while back about some trouble I was having with FSTAT and open files. I was hoping it might be peculiar to NT...

Justin

```

> ** Structure FSTAT, 12 tags, length=36:
>  UNIT      LONG      100
>  NAME      STRING    '/homes/kupper/IPC/fifo'
>  OPEN      BYTE      1
>  ISATTY    BYTE      0
>  ISAGUI    BYTE      0
>  INTERACTIVE  BYTE    0
>  READ      BYTE      1
>  WRITE     BYTE      0
>  TRANSFER_COUNT LONG    1
>  CUR_PTR   LONG      -1
>  SIZE      LONG      0
>  REC_LEN   LONG      0
>
> regardless of any waiting or not waiting data.
>
> Good thought Justin, anyway!
>
>> The situation we were dealing with was to read an incoming spacecraft
> telemetry
>> stream. Since there already was a process (written in C) which was archiving
>> the telemetry stream into data files, what we ended up doing was to simply
> read
>> those files while they were still being written. That way, we avoided the
>> whole pipe/fifo business. Sounds like that wouldn't help you, though.
>>
>> Our original scheme was to use a two-way socket connection between IDL and a
> C
>> process which was handling telemetry reception. IDL would send out a request
>> for data to the socket, and the C process would either respond with a
>> packet, or with a "no-data-yet" message. That way, IDL would always read
> back
>> something.
>
> Okay, so there seems to be no way around using some intermediary C-Routines
> which
> handle reception.
> IDL just doesn't support Inter Process Communication...
>
> Thank you both for your help.
>
> Best regards,
> Ruediger.
>
>

```

---



---

Subject: Re: Non-Blocking I/O  
Posted by [davidf](#) on Mon, 15 Feb 1999 08:00:00 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

Eric J. Korpela (korpela@albert.ssl.berkeley.edu) writes with respect to interprocess communication in IDL:

- > If the demand did exist someone like David
- > Fanning probably would have done it by now.

Uh, well, interprocess communication is not really my thing and if I learned any more about it I probably wouldn't have to ask Peter Mason and Mark Rivers how to do it, which would eliminate well over half of my most interesting e-mail conversations. :-)

And, truly, I don't think you would get rich writing about interprocess communication in IDL, no matter how badly a good book on the subject is needed.

Now...image processing in IDL. That's a topic that has an audience, I think. I'm finalizing details now, but I hope to announce a new initiative in this direction soon. :-)

Cheers,

David

P.S. I'm always interested in hearing what folks think is missing in IDL or what topics they think should be covered in more depth. Drop me a note any time.

--

David Fanning, Ph.D.  
Fanning Software Consulting  
Phone: 970-221-0438 E-Mail: [davidf@dfanning.com](mailto:davidf@dfanning.com)  
Coyote's Guide to IDL Programming: <http://www.dfanning.com/>  
Toll-Free IDL Book Orders: 1-888-461-0155

---

Subject: Re: Non-Blocking I/O  
Posted by [menakkis](#) on Mon, 15 Feb 1999 08:00:00 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

Ruediger Kupper <[Ruediger.Kupper@Physik.Uni-Marburg.De](mailto:Ruediger.Kupper@Physik.Uni-Marburg.De)> wrote:

<...>

- > Okay, so there seems to be no way around using some intermediary C-Routines
- > which handle reception.
- > IDL just doesn't support Inter Process Communication...

Frankly I'm surprised that this interesting thread didn't tease out more responses. I'm sure that there must be \*dozens\* of lurkers out there who have been down this road.

Now I know next to zero about IPC and the like, but I was wondering if you might have some luck with the /NOSTDIO switch to OPEN (on Unix platforms). Reportedly, if you READU from a LUN opened like this, IDL will only read the amount of data that is available. (Perhaps it will even read "zero bytes" if nothing's available?) And you can reportedly use the TRANSFER\_COUNT= switch to READU to find out how many \*elements\* (not necc. bytes) were transferred. Even if this works, I can imagine that reassembling data read in bits and pieces on the IDL side might be a bit tricky, but it might still be worth a try.

Peter Mason

-----== Posted via Deja News, The Discussion Network ==-----  
<http://www.dejanews.com/> Search, Read, Discuss, or Start Your Own

---

---

Subject: Re: Non-Blocking I/O  
Posted by [korpela](#) on Tue, 16 Feb 1999 08:00:00 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

In article <7a8f40\$qsj\$1@nnrp1.dejanews.com>,  
<menakkis@my-dejanews.com> wrote:  
> Ruediger Kupper <Ruediger.Kupper@Physik.Uni-Marburg.De> wrote:  
>  
> <...>  
>> Okay, so there seems to be no way around using some intermediary C-Routines  
>> which handle reception.  
>> IDL just doesn't support Inter Process Communication...  
>  
> Frankly I'm surprised that this interesting thread didn't tease out more  
> responses. I'm sure that there must be \*dozens\* of lurkers out there who  
> have been down this road.

Well, having the thread occur on a weekend probably doesn't help the response volume. I have been down this road before, and basically have come to the conclusion that if IDL doesn't have what you want, adding it, while not exactly trivial, is pretty damn easy.

Now no one is beating down the door to get to my web site and get VARRAY (which will add some shared memory support to IDL). No one has expressed

much interest in asking me to speed up publication of a multiprocessing library for IDL (see examples given on this group a couple months ago.) I've been using standard UNIX IPC mechanisms in IDL for a couple years now, but haven't taken the effort to make my software available. Nothing makes a 4 processor machine run like using all the processors.

Maybe the demand doesn't exist? If the demand did exist someone like David Fanning probably would have done it by now.

Eric

--

Eric Korpela | An object at rest can never be  
korpela@ssl.berkeley.edu | stopped.  
<a href="http://sag-www.ssl.berkeley.edu/~korpela">Click for home page.</a>

---