CoCoALib - Feature #714

Interrupt mechanism

19 May 2015 10:04 - John Abbott

Status:	Closed	**Start date:**	19 May 2015
Priority:	Normal	**Due date:**	
Assignee:	John Abbott	**% Done:**	100%
Category:	New Function	**Estimated time:**	4.51 hours
Target version:	CoCoALib-0.99536 June 2015	**Spent time:**	4.45 hours

Description

Add an interrupt mechanism to CoCoALib.

Related issues:

Related to CoCoA-5 - Bug #713: External libs: interrupting not easy	**Closed**	**18 May 2015**
Related to CoCoALib - Feature #718: Insert calls to CheckForInterrupt	**Closed**	**21 May 2015**
Related to CoCoA-5 - Bug #345: Interpreter interrupt delayed confusingly	**Closed**	**24 Apr 2013**
Related to CoCoALib - Feature #385: Design new errors using inheritance	**In Progress**	**08 Jul 2013**
Related to CoCoALib - Feature #638: Time limit: let user specify time limit f...	**Closed**	**27 Oct 2014**
Related to CoCoALib - Bug #971: CheckForInterrupt does not work in the expect...	**Closed**	**14 Nov 2016**
Related to CoCoALib - Bug #1458: Redesign interrupt mechanism?	**Rejected**	**10 May 2020**

History

#1 - 19 May 2015 10:09 - John Abbott

As far as CoCoA-5 is concerned CoCoALib is "external software", and the problem noted in #713 applies as much to CoCoALib as to other external libraries.

I suggest the following design:

- there is a global flag CoCoA::InterruptFlag which is normally false
- there is a function CoCoA::SignalInterrupt which sets the flag
- there is a function CoCoA::CheckForInterrupt which tests the flag; if false, the function simply returns; otherwise it resets the flag, and throws a CoCoA::interrupted exception

CoCoALib authors would then be obliged to call CheckForInterrupt every so often, so that an interrupt will actually take effect within a reasonable time.

#2 - 19 May 2015 10:16 - John Abbott

There is a slight risk in my proposal above. If SignalInterrupt is called asynchronously while CheckForInterrupt is running, and has just reset the flag but not yet thrown the exception, then the flag will be set while the exception is thrown. This could be rather unexpected.

I suggest that a new class CoCoA::InterruptException be the type thrown. Its destructor could reset the interrupt flag. This should ensure that the flag is reset once the (interrupt) exception handler exits scope. This should have the effect of ignoring any further signals while the exception is propagating and being handled.

Maybe even: the ctor for CoCoA::InterruptException could block all signals (and then reset the flag), and the dtor unblocks them. This fits in well with ctor-dtor corresponding to resource acquisition-release.

#3 - 19 May 2015 11:41 - John Abbott

Here is a further complication: how to handle several libraries together?
Let us suppose that CoCoALib has a mechanism similar to that outlined above, and that libnormaliz has an analogous system.

Suppose we have CoCoALib+libnormaliz working on a problem, and we want to interrupt the computation. From outside we do not know whether the currently running code is part of CoCoALib or part of libnormaliz; so we want to set both flags to be certain that the running code will see the interrupt.

We must also make sure that all flags are reset when the interrupt is handled. Perhaps the simplest solution is to have a single shared flag; this implies that CoCoALib and libnormaliz must work with references to some common global flag. How to tell them where this flag is? In CoCoALib, this could probably be done via GlobalManager.

#4 - 19 May 2015 16:56 - John Abbott

- Status changed from New to In Progress

- % Done changed from 0 to 20

I have implemented a first version following the outlines given above.

It seems to pass a quick test -- I have made RingBase::mySequentialPower interruptible, and have verified that it works.

Matters still to resolve:

- **(A)** check that the new catch statement in the interpreter (Interpreter.C:2678) is all that is needed
- **(B)** print out some message about being interrupted
- **(C)** is there some way to put in context information? (*e.g.* convert CoCoA::InterruptReceived into InterruptException with context info)
- **(D)** insert numerous calls to CheckForInterrupt() throughout CoCoALib

(D) will be long and tedious, but probably not too hard -- make a new issue?
(C) requires studying the interpreter; maybe Giovanni can help?
(B) probably becomes irrelevant if **(C)** can be done well
(A) again Giovanni's help would be a boon!

#5 - 19 May 2015 17:55 - John Abbott

Also seems to work with the C5 GUI now :-)

#6 - 20 May 2015 12:21 - John Abbott

I am undecided about which ways to offer for telling the GlobalManager which interrupt flag to watch:

- **(A)** specify the flag in GlobalSettings which is passed as ctor param to GlobalManager
- **(B)** specify the flag by calling a mem fn of GlobalManager

Approach **(B)** is more powerful than approach **(A)** in that approach **(A)** can always be replaced by approach **(B)** but not vice versa (as I discovered when modifying src/CoCoA-5/Main.C)

Is there any sense in offering approach **(A)**? Perhaps it is best to offer just **(B)**, and add approach **(A)** later if there is good justification?

#7 - 20 May 2015 12:38 - Anna Maria Bigatti

John Abbott wrote:

> Is there any sense in offering approach **(A)**? Perhaps it is best to offer just **(B)**, and add approach **(A)** later if there is good justification?

(I admit I'm not understanding all the implications) my preference was for **(B)** even before reading John's last comment ;-)

#8 - 21 May 2015 09:41 - John Abbott

- Status changed from In Progress to Resolved

- Assignee set to John Abbott

- % Done changed from 20 to 80

I have checked in the impl with both **(A)** and **(B)**.
Then I deleted **(A)** and checked in again. This way I hope **(A)** might still be recoverable from CVS if we should ever want it.

I'll discuss the design with Christof; and if all goes well, we can proceed to feedback status.

PS I'm hoping Christof will add a similar feature to libnormaliz.

#9 - 26 Jun 2015 14:56 - John Abbott

- Status changed from Resolved to Feedback

- % Done changed from 80 to 90

I've been using the current code for a little while (not really proper testing), and no problems have come up. So changing status to **feedback**.

I am just slightly uneasy about the "clever trick" of making InterruptReceived dtor reset the interrupt flag... but I cannot see how it can be harmful.

#10 - 26 Jun 2015 16:14 - Anna Maria Bigatti

- Status changed from Feedback to Closed

- % Done changed from 90 to 100

- Estimated time set to 4.51 h

Tested on my computer: pending interrupt and caught interrupt, they both work! Closing

#11 - 28 Jun 2015 20:46 - John Abbott

- Target version changed from CoCoALib-0.99540 Feb 2016 to CoCoALib-0.99536 June 2015

#12 - 14 Nov 2016 11:12 - John Abbott

- Related to Bug #971: CheckForInterrupt does not work in the expected way added

#13 - 10 May 2020 12:07 - John Abbott

- Related to Bug #1458: Redesign interrupt mechanism? added