OpenRTM-aist (Java) - バグ #2327

RingBuffer関連の問題

2012/01/16 10:29 - 匿名ユーザー

ステータス:	終了	**開始日:**	2012/01/16
優先度:	通常	**期日:**	
担当者:	ga	**進捗率:**	100%
カテゴリ:		**予定工数:**	0.00時間
対象バージョン:			

説明

KEK仲吉さん安さんから指摘。

とりあえずの対処として read()の前に InPortに対し isEmpty()を
行っていました。この件を安さんが調査し次のような問題を見つけ
られましたので報告致します（以下、安さんからのメールの抜粋）。

```
RingBuffer.hのwriteメソッドで、
従来は

if   (empty_)
{
        Guard   eguard(m_empty.mutex);
        m_empty.cond.signal();
}
advanceWptr(1);

となっていました。
これだと、readでバッファがなく待っていた状態で、writeが入り、
cond.signalでシグナルを出しますが、   advanceWptr(1)をする前に、
シグナルを出しているため、writeのバッファポインタがインクリ
メントされる前に、readがwake   upする可能性が出てきてしまいます。
そこで、下記のように変更しました。

if   (empty_)
{
            Guard   eguard(m_empty.mutex);
            advanceWptr(1);
            m_empty.cond.signal();
}   else
            advanceWptr(1);

これで、昨日の昼から走らせていますが、数秒で止まってしまう場合
があったり数十分で止まってしまったありする場合があるのに比べて、
いまだ走っていますので、つじつまがあう点を考えても、問題は解決
したものと思います。
```

関連するチケット:

関連している OpenRTM-aist (C++) - バグ #1426: 1.0.0のRingBuffer関連の問題	**終了**	**2010/04/22**

関係しているリビジョン

リビジョン 641 - 2012/01/16 10:33 - fsi-katami

RingBuffer\'s bug was fixed. refs #2327

リビジョン 684 - 2013/02/03 13:17 - ga

The bugs of RingBuffer class have been fixed. refs #2327

履歴

#1 - 2012/01/16 10:33 - 匿名ユーザー

- ステータス を 新規 から 解決 に変更

- 担当者 を 匿名ユーザー にセット

#2 - 2013/01/17 01:41 - ga

- 担当者 を 匿名ユーザー から ga に変更

- 進捗率 を 0 から 80 に変更

#3 - 2013/02/03 13:30 - ga

- 進捗率 を 80 から 100 に変更

1. write()メソッドの問題修正

- m_rtimeoutをm_wtimeoutに修正
- block modeにおいて、m_full.mutex.wait()からの復帰がnotify()によるものか、タイムアウトによるものなのかにかかわらず"return ReturnCode.TIMEOUT;"となっていたため、"full()"にて確認したうえで "return ReturnCode.TIMEOUT;"するように修正。

2. read()メソッドの問題修正

- m_rtimeoutをm_wtimeoutに修正
- block modeにおいて、m_full.mutex.wait()からの復帰がnotify()によるものか、タイムアウトによるものなのかにかかわらず"return ReturnCode.TIMEOUT;"となっていたため、"full()"にて確認したうえで "return ReturnCode.TIMEOUT;"するように修正。

3. 本チケットに関する実装の見直し

- advanceRptr(1)の直前でfull()をコールするように変更。
- advanceWptr(1)の直前でempty()をコールするように変更。

上記の変更は、 r684 にて対応しました。

#4 - 2013/06/12 00:35 - n-ando

- ステータス を 解決 から 終了 に変更