Concurrent Programming Project

1. Introduction

This project is designed to be submitted before **midnight of November 26th (Monday), 2007**. You will obtain experience with concurrent programming through the BACI concurrent interpreter. Some of the learning objectives and goals are the following:
(a) to learn and implement the Producer Consumer Problem with a shared buffer using semaphores,
(b) to implement the varied rate Producer Consumer program to observe the outcome for different rates.

2. Project Description

You will implement the basic producer consumer problem using BACI by maintaining a shared, bounded buffer. You will need to use semaphores to implement the exclusion operation to enforce the appropriate access of the producer and consumer. The following functions need to be implemented:

void producer(int NullLoop, int NumberOfItems)

The function simulates the producer. **NumberOfItems** is the total data items that the producer should produce. The producer generates a random integer and then puts it into the shared buffer. You will then do the following to display output to the screen:

```
cout << "producer enter a new item " << i << " , value: " << value << endl;
```

i is the index of the new item in the total NumberOfItems items, not the index of its position in the shared buffer. value is the value of the new item.
The **NullLoop** is used to control the speed of the producer. Since you will need a way to change the speed of the producer relative to the consumer, you need to create an empty 'for' loop that will iterate NullLoop times. This will allow you to simulate varied rate producer. The bigger NullLoop, the slower the producer runs.
When the producer has produced all the items, it will need to set up another shared variable to indicate that the producer is finished so that the consumer will terminate after the producer terminates and it consumes all the items.

void consumer(int NullLoop, int ConsumerID)

The function simulates the consumer. The **ConsumerID** is the ID for the consumer procedure (using 1, 2 if there are 2 consumers). While you will only have one producer, you may have several consumers, and this variable provides a way to differentiate between multiple consumers.
The consumer will add up all the values that the produced put in the buffer. The consumer will first check whether the producer terminates and whether there are available items in the buffer. If there are available items, it will pick up the items from the buffer (you could use the First In First Out to pick up the items). The consumer will add the items and then output to the screen using:

cout << "consumer " << ConsumerID << " remove an item " << value << endl;
NullLoop is used to control the speed of the consumer just like for the producer() function. You just put the NullLoop numbers of none-operation loops in the iteration. This will allow you to simulate the varied rate for the consumer.

Do the following simulations to test your program:
(a) One producer and one consumer, producer runs the same speed as the consumer, producer runs faster than the consumer or producer runs slower than the consumer. For example

producer(0, 20); consumer(0, 1);
producer(100, 10); consumer(0, 1);

(b) One producer and two consumers, For examples:

producer(100, 20); consumer(0, 1); consumer(0, 2);
producer(0, 10); consumer(100, 1); consumer(100, 2);

3. Project Direction

You will write the program based on the BACI interpreter. Its homepage is located at http://www.mines.edu/fs_home/tcamp/baci/. Read through the user guide at http://www.mines.edu/fs_home/tcamp/baci/bacidocpdf.tar.gz. Follow these instructions to complete your work:
(a) Download and install a copy of BACI to your own computer. Or you can use BACI already installed on Olympus, which is located in /export/home/c/cop46001/basunxe (add /export/home/c/cop46001/basunxe to your PATH)
(b) Download the **obj5.cm** (a framework to help you begin coding) from http://www.eecs.ucf.edu/courses/cop4600/fall2007/files/obj5.cm. Write all your code in obj5.cm.
(c) Using ***bacc obj5.cm*** to compile your program
(d) Using ***bainterp obj5.pco*** to run your program
(e) After you finish all testing, using following command to submit your code:

rm Makefile
cp /export/home/e/emontagn/project/Makefile3 Makefile
make submitcc

4. Project Submission

Please submit your source code using “make submitcc” on olympus. Make sure you update “Makefile” (using the command ***cp /export/home/e/emontagn/project/Makefile3 Makefile***) before submit. The command “make submitcc” will copy your obj5.cm to the correct location.