Monitor Pseudo Code:

monitor monitor\_name {

//declaring shared variables

variables\_declaration;

condition\_variables;

procedure p1(...) {

...

};

procedure p2(...){

...

};

...

procedure pn(...){

..

};

{

Initailisation Code();

}

}

Producer-Consumer using Monitors

# define N 3000000

semaphore mutex = 1 ;

semaphore empty = N;

semaphore ful l = 0 ;

void main ( void )

{

// create four producer threads

// create four consumer threads

}

void producer ( void )

{

while ( 1 )

{

down(&empty) ;

down(&mutex) ;

insert(’X’) ;

up(&mutex) ;

up(&full) ;

}

}

void consumer ( void )

{

while( 1 )

{

down(&full) ;

down(&mutex) ;

remove( ) ;

up(&mutex) ;

up(&empty) ;

}

}

# define N 3000000

/ / arena

semaphore mutex = 1 ;

cond empty ;

cond full ;

int count;

char buf[N] ;

// the five variables above should be created as area in moninit

void main ( void )

{

//call moninit( )

//create six producer threads

//create six consumer threads

}

void producer ( void )

{

while( 1 )

{ moninsert ( ’X’ ) ; }

}

void consumer ( void )

{

while ( 1 )

{ mon remove ( ) ; }

}

/ / monitor . c

void monenter ( )

{ down(&mutex ) ; }

void monexit ( )

{ up(&mutex ) ; }

void moninsert( char a l p h a )

{

monenter ( ) ;

if ( count == N) wait ( full ) ;

insertitem ( alpha ) ; // insert alpha into buf

count = count+1;

if (count==1) signal (empty);

monexit ( ) ;

}

void monremove ( )

{

monenter ( ) ;

if (count==0) wait(empty);

removeitem( ) ; / / remove an item from buf

count=count-1=;

if ( count == N − 1) signal ( full ) ;

monexit( ) ;

}