MULTIPROGRAMMING OPERATING SYSTEM (MOS) PROJECT

Third Phase

ASSUMPTIONS (Added):

- Multiprogramming and virtual memory added
- TI "time slice out" interrupt introduced
- Paging retained without even-odd restrictions
- I/O Processing through 3 channels introduced
- Spooling and buffering for I/O through channels introduced
- Drum (secondary storage) introduced
- I/O interrupt introduced

NOTATIONS (Added):

TS: Time Slice
TSC: Time Slice Counter
CHi: Channel i i = 1, 2, 3
RD: Read
WT: Write
IS: Input Spool
OS: Ouput Spool
LD: Load
SWP: Swap
eb(q): Empty buffer (queue)
ifb(q): Inputful buffer (queue)
ofb(q): Outputful buffer (queue)
LQ: Load queue
RQ: Ready queue
SQ: Swap queue
IOQ: Input-Output (read/write) queue
TQ: Terminate (output spool) queue
IRi: Interrupt Routine for channel i i = 1, 2, or 3

SPOOLING AND BUFFERING INFO

- *Buffer Pool: 3 Types: Empty, Inputful, Outputful*
- *Channels: 3*
  - *Channel 1: Cardreader to Supervisor Memory*
  - *Channel 3: Supervisor Memory and Drum (either way)*
  - *Channel 2: Supervisor Memory to Printer*
- *Spooling: Input and Output*

(a) Input (Before Execution): Program and data cards transferred from Card Reader to Drum

Performed by Channels 1 and 3

Channel 1:

Started with an Empty buffer
Fills it with the next card from card reader
Returns Inputful buffer

Channel 3:

Started with the next Inputful buffer, and an available drum track
Writes the buffer to the drum track
Returns an Empty buffer

OUTPUT (After the program has terminated)

Output lines stored on drum tracks during execution sent to printer

Performed by channels 3 and 2

Channel 3:

Started with an Empty buffer, and the next output drum track
Fills the buffer with the next output line from the drum truck
Returns an Outputful buffer

Channel 2:

Started with the next Outputful buffer
Sends it to the printer
Returns an Empty buffer

Note that a channel cannot be started if appropriate type of buffer is not available.

INTERRUPT VALUES (Added):

TI = 1 on Time Slice Out
IOI: 1 channel 1 done
2 channel 2 done
4 channel 3 done

Error Message Coding: (No Change)

BEGIN
INITIALIZATION
IOI = 1

MOS (MASTER MODE)

Case TI and SI of

TI SI Action

0 or 1 1 Move PCB, RQ → IOQ (Read)

0 or 1 2 Move PCB, RQ → IOQ (Write)

0 or 1 3 Move PCB, RQ → TQ (Terminate [0])

2 1 Move PCB, RQ → TQ (Terminate [3])

2 2 Move PCB, RQ → IOQ (Write) then TQ (Terminate [3]) 2 3 Move PCB, RQ → TQ (Terminate [0])

Case TI and PI of

TI PI Action

0 or 1 1 Move PCB, RQ → TQ (Terminate [4])

0 or 1 2 Move PCB, RQ → TQ (Terminate [5])

0 or 1 3 Page Fault

If Valid

If Frame Available

Allocate

Update Page Table

Adjust IC, if necessary

Else

Move PCB, RQ → SQ

Else

Move PCB, RQ → TQ (Terminate [6])

2 1 Move PCB, RQ → TQ (TERMINATE [3,4]) 2 2 Move PCB, RQ → TQ (Terminate [3,5])

2 3 Move PCB, RQ → TQ (Terminate [3])

Case IOI of

0 No Action

1 IR1
2 IR2
3 IR2, IR1
4 IR3
5 IR1, IR3
6 IR3, IR2
7 IR2, IR1, IR3

IR1

Read next card in given eb, change status to ifb, place on if b (q)
If not e-o-f and eb(q) not empty
Get next eb
Start Channel 1

Examine ifb

$AMJ: Create and initialize PCB
Allocate frame for Page Table
Initialize Page Table and PTR
Set F ← P (Program cards to follow)
Change Status from ifb to eb
Return buffer to eb(q)
$DTA: Set F ← D (data cards to follow)
Change status from ifb to eb
Return buffer to eb(q)
$END: Place PCB on LQ, change status from ifb to eb, return buffer to eb(q)

Otherwise place ifb on ifb(q), save F information (program or data card for channel 3)

IR2

Print given ofb, change status from ofb to eb
Return buffer to eb(q)
If ofb(q) not empty,
Get next ofb
Start Channel 2

IR3 (First, complete the assigned task and the follow up action for channel 3 for each possible task, and then assign new task to it in priority order.)
Case Task of
IS: Write given ifb on given track
Place track number in P or D part of PCB
Change status from ifb to eb
Return buffer to eb(q)

OS: Read information (Output line) from given track into given eb
Change status from eb to ofb

Return buffer to ofb(q)
Release track
Decrement line count in PCB
If last line, fill two other ebs (if available) with blanks, change status from eb to ofb and place the buffers on ofb(q)
Release PCB, all remaining drum tracks and all memory blocks.
Prepare 2 lines of messages from next PCB (if available) on TQ, move them into ebs (if available), change status from eb to ofb, and place these buffers also on ofb(q) LD: Load program card from given track into indicated memory block
Decrement count in PCB
If zero, place PCB on RQ after all the initializations

RD: Read data card from given track into indicated memory block
Decrement count in PCB
Move PCB to RQ after setting TSC ← 0

WT: Write information from the indicated memory block to the given track
Increment line count (TLC) in PCB
If TI = 2 or 3, move PCB to TQ
Else move PCB to RQ after setting TSC ← 0
SQ(W): Write the information from the victim frame to the given track.
Locate drum track with faulted page
Task ← SQ(R)
Start Channel 3

SQ(R): Read drum track with faulted page in newly allocated frame
Move PCB, SQ → RQ after setting TSC ← 0

End-Case

(Now Assign New Task in Priority Order)

If a PCB on TQ (output spool first)
If eb(q) not empty
Get next buffer from eb(q)
Find track number of next output line
Task ← OS
Start Channel 3

Else (input spool next)
If ifb(q) not empty and a drum track available
Get next buffer from ifb(q)
Get a drum track

Task ← IS
Start Channel 3

Else (load next)
If a PCB on LQ (load next) and a memory frame available
Find track number of next program card
Allocate a frame
Update Page Table
Task ← LD
Start Channel 3

Else (now i/o)

If a PCB on IOQ
If Read (GD)
If no more data card
Move PCB, IOQ → TQ (Terminate [3])
Else
Find track number of next data card
Get memory RA
Task ← GD
Start Channel 3

Else If Write (PD)
If TLC > TLL, Move PCB IOQ → TQ (Terminate [2])
Else
Get a drum track, if available
Update PCB
Find memory RA
Task ← PD
Start Channel 3

Else (allocate memory)
If a PCB on SQ
If a memory frame now available
Allocate
Update page Table
Adjust IC, if necessary

Move PCB SQ → RQ with TSC ← 0

Else

Run page replacement algorithm
Find a victim frame
Allocate and Deallocate this frame
by updating both page tables
If victim frame not written into,
locate drum track for faulted page
Task ← SQ (R)
Start Channel 3

Else

Task ← SQ(W)
Start Channel 3

(END OF IR3)
START CHi
Adjust IOI (Subtract 1, 2, or 4)
Reset Ch timer to zero
Set Ch flag to busy.

STARTEXECUTION

IC ← 00
EXECUTEUSERPROGRAM
END (MOS)

EXECUTEUSERPROGRAM (SLAVE MODE)
ADDRESS MAP (VA, RA)
Accepts VA, either computes & returns RA or sets PI ← 2 (Operand Error) or PI ← 3 (Page Fault)
LOOP
ADDRESSMAP (IC, RA)
If PI ≠ 0, End-LOOP (F)
IR ← M[RA]
IC ← IC+1
ADDRESSMAP (IR[3,4], RA)
If PI ≠ 0, End-LOOP (E)
Examine IR[1,2]
LR: R ← M [RA]
SR: R → M [RA]
CR: Compare R and M [RA]
If equal C ← T else C ← F
BT: If C = T then IC ← IR [3,4]
GD: SI = 1 (Input Request)
PD: SI = 2 (Output Request)
H: SI = 3 (Terminate Request)

```
                Otherwise PI ← 1 (Operation Error)
            End-Examine

End-LOOP (X) X = F (Fetch) or E (Execute)

SIMULATION
        Increment TTC
        If TTC = TTL then TI ← 2
        Increment TSC
        If TSC = TS, then TI ← 1

        For all CHi, i = 1,2,3
                If CHi flag busy,
                        Increment Chi timer
                        If CHi timer = CHi total time
                                Increment IOI accordingly
                                (Set channel completion interrupt)
        End - For


If SI or PI or TI or IOI ≠ 0 then Master Mode, Else Slave Mode
```