MULTIPROGRAMMING OPERATING SYSTEM (MOS) PROJECT

First Version

ASSUMPTIONS:

* Jobs entered without error in input file
* No physical separation between jobs
* Job outputs separated in output file by 2 blank lines
* Program loaded in memory starting at location 00
* No multiprogramming, load and run one program at a time
* SI interrupt for service request

NOTATION

M: memory; IR: Instruction Register (4 bytes)

IR [1, 2]: Bytes 1, 2 of IR/Operation Code

IR [3, 4]: Bytes 3, 4 of IR/Operand Address

M[&]: Content of memory location &

IC: Instruction Counter Register (2 bytes)

R: General Purpose Register (4 bytes)

C: Toggle (1 byte)

: Loaded/stored/placed into

MOS (MASTER MODE)

SI = 3 (Initialization)

Case SI of

1: Read

2: Write

3: Terminate

Endcase

READ

IR [4] ← 0

Read next (data) card from input file in memory locations IR [3,4] through IR [3,4] +9

If M [IR [3,4]] = $END, abort (out-of-data)

EXECUTEUSERPROGRAM

WRITE

IR [4] ← 0

Write one block (10 words of memory) from memory locations IR [3,4] through IR [3,4] + 9 to output file

EXECUTEUSERPROGRAM

TERMINATE

Write 2 blank lines in output file

MOS/LOAD

LOAD

m ← 0

While not e-o-f

Read next (program or control) card from input file in a buffer

Control card: $AMJ, end-while

$DTA, MOS/STARTEXECUTION

$END, end-while

Program Card: If m = 100, abort (memory exceeded)

Store buffer in memory locations m through m + 9

m ← m + 10

End-While

STOP

MOS/STARTEXECUTION

IC ← 00

EXECUTEUSERPROGRAM

EXECUTEUSERPROGRAM (SLAVE MODE)

Loop

IR ← M [IC]

IC ← IC+1

Examine IR[1,2]

LR: R ← M [IR[3,4]]

SR: R → M [IR[3,4]]

CR: Compare R and M [IR[3,4]]

If equal C ← T else C ← F

BT: If C = T then IC ← IR [3,4]

GD: SI = 1

PD: SI = 2

H: SI = 3

End-Examine

End-Loop