WESTSIDE PORT

An I/O Channel

For the Westside Chapter

EDITORIAL

Here we are with our second issue of the Westside Port, due to the positive response from the members present, at the last meeting. Input to the Port can be written/disk or any means that would be suitable to you, and give/send them to JOHN EASTON. Today's issue will contain some of the informative sections that we hope to continue to bring to you in future issues. This can only continue to function with your support; so try your best to be part of our next issue by submitting some "input".

We hope that you will enjoy this issue as much as we enjoy bringing it to you.

THE EDITORS (Vic Cassar & Alex Hovell)

ASSEMBLER and YOU

Over the next few issues, we will attempt to uncover the mysteries of ASSEMBLY programming. With the aid of a PUBLIC DOMAIN program called "SUPERMON" found in various forms, shapes and sizes on most TPUG library disks, we will guide you through programs with small content and easy to understand "BASIC sidelines."

An example of such a format would be:

```
.a C000 LDA #$41            : A = 65
.a C002 STA $0400           : POKE 1024 , A
.a C005 BRK                 : STOP
```

Note: The colon's are just to cut the page in half to separate the two versions of the program, under no circumstances should you type in both the colons and the BASIC sideline.

Now, how to start your journey into the amazing world of the ASSEMBLER....

1.) Freshen up your computer by turning it OFF and then ON to make sure that the machine is ready for use.

2.) Find "SUPERMON" on any of a TPUG disk and enter the following commands:

```
LOAD"SUPERMON",8    <RETURN>
```

Once SUPERMON has loaded, type:

```
RUN   <RETURN>
```

After typing RUN and pressing the RETURN key, you should see:

```
B*
   PC  SR AC XR YR SP
.;03FF F5 DC 20 33 F6 <=Numbers may vary
.
```

If you do not encounter the above example, then we suggest start at step 1 again. If this occurs more than once, try a different copy on a different disk. If all else fails, get in contact with TPUG as to where to get another copy.

Try typing in the following:

```
.A C000 LDA #$07
.A C002 STA $D020
.A C005 BRK
.A C006
.G C000
```

After you have typed in the above text, you will notice the border colour has changed to YELLOW, and the following lines appeared:

```
B*
   PC  SR AC XR YR SP
.;C005 75 07 20 33 F6
.
```

Now that we have completely baffled you and amazed you, we will explain the small details that got us here.

SUPERMON has a few commands to aid you in your quest to the better program. Listed below in alphabetical order:

```
A C000 LDA #$07
^
```

Assemble code (LDA #$07) at $C000 (49152 DECIMAL). Which is just a fancy way of saying that you have entered that command into memory as you would a basic program.

```
G C000
^
```

Goto the code at $C000 (49152 DECIMAL) is just another fancy way of saying GOTO (or RUN) starting at $C000.

After the program finished, you recieved some information from the machine about what you did. Here is a small explaination of what went on and what to expect.

```
B*
   PC    SR    AC    XR    YR    SP
.;C005   75    07    20    33    F6
. (1)   (2)   (3)   (4)   (5)   (6)
```

(1) The (P)rogram (C)ounter contains the last BYTE that was read to execute the LAST command (BRK). Therefore, you can tell where your program stopped by looking at the PC.

(2) The (S)tatus (R)egister contains 8 FLAGS that determine

special features of the computer which we will go into further once you are ready.

(3) The (AC)cumulator is the major "VARIABLE" of the machine, almost every command uses it. It is the gateway to every program. Without it, you couldn't do too much. LD(A) and ST(A) use the (AC) to keep track of the #$07. If you look at what AC says it has, it WILL be 07.

(4) The (X) Index (R)egister. This is used for many purposes. For example:

```
LDA $0400,X                    :  A=PEEK(1024+X)
          ^
```

X is being added to the $D000 location. The BASIC Sideline should give you enough explaination of it's most primary function.

(5) The (Y) Index (R)egister. This is used for almost the same purposes as the XR though, it is a bit more flexible. For example:

```
LDY $0400,X                    :  Y=PEEK(1024+X)
          ^
```

X is being added to the $D000 location and entered into Y. The BASIC Sideline should give you an insight to the fact that the YR can take information almost like the AC can, but still, it is limited as well.

(6) The (S)tack (P)ointer keeps track of where to return to after an RTS (ReTurn from Subroutine.) Like in BASIC, GOSUBs and RETURNs must also keep track of where the must return to after completing there Subroutine. Thus, a pointer must keep track of how many GOSUBs (JSR's) there are. And the SP does that very function. $F6 is a normal area for the pointer to be at. Every JSR (GOSUB) causes the SP to lower. Should the SP get too low (Under $20), then you should start to worry, as if it gets a chance to move around to $FF, the machine will not know what hit it!

We hope we haven't lost all of you into a maze of letters and numbers. As we hope to continue to teach you ASSEMBLY LANGUAGE. The easy way!

BROKEN BITS

In this section, we will make a note to errors in programs, current or previous, BASIC or ML. Any questions are welcome and so are new Broken Bits.

In this issue, we have attacked "A Two Button Mouse", PG.36 of the May 87 issue of Transactor. The routine to read the second (RIGHT) button on page 37 works SOME OF THE TIME and mostly causes great distortion. The Following Corrections, along side of the origional will prove to be foolproof and working. Also, since this is in ML and we have yet to break that surface, we have also included a BASIC version of the same reader (the working one of course!)

The Old Reader

```
RDPORT SEI
       LDA #$C0
       STA $DC02
       LDA #$80
       STA $DC00
       LDX #$00
       INX:BNE *-1
       LDX $D419
       LDY $D41A
       LDA #$FF
       STA $DC02
       RTS
```

The New Reader

```
RDPORT SEI
       LDA #$C0
       STA $DC02
       LDA #$80
       STA $DC00
       LDY #$00
       LDX #$00
CHECK  LDA $D419
       CMP #$FF
       BNE NOFIRE
       INX
       BNE CHECK
```

The difference between the NEW and the OLD is the NEW checks to see if the button is down 255 times before YR will go to 1. As his waits for a short sec and reads the paddles. Now that is jitter for you!

As promised the BASIC NEW READER:

```
10 POKE56322,192:POKE56320,128:Y=0:FORX=0TO2:
   IFPEEK(54297)=255THENNEXTX:Y=1
20 POKE56322,255:PRINT Y:END
```

We hope that you will benefit from this. And we would like to know if so.

SHORT CIRCUIT

In this section of the PORT, we look at small programs, one liners and many other facinating BASIC/ML programs.

This issue we have a small BASIC routine that can capitalize the first letter in every word in a string, leaving all others in lowercase. For example:

```
A$="jOhn easTon aNd TPUG."
```

After the routine is finished with O$ it will look like this:

```
O$="John Easton And Tpug."
```

And now, the routine!

```
      ↓space.
10 L$="":O$="":FORX=1TOLEN(A$):
   L=ASC(MID$(A$,X))AND127:
   L=LOR-128*(L>64ANDL<91ANDL$=" ")
20 L$=CHR$(L):O$=O$+L$:NEXTX
```

THAT'S IT! ..... you will need to shorten some commands to get line 10 properly entered.