Basic 1.2 (MVP) Common Partition Control

Computer Concepts Corp.
8001 W. 63rd Street
Shawnee Mission, KS. 66202

1.0 Breakdown of 0900 data and AR usage for Basic 2.x

This document outlines the functions of the 0900 section of data memory while running the Basic ~~1.2~~ operating system. It is believed that the contents of this memo are correct. Several areas of the 0900 memory segment are left blank due to insufficient evidence as to the function. Where data is questionable, supporting data will be presented to ascertain what the true function is.

All data memory is assigned by the Basic 2.x on partition initialization time. Partition 1 data memory segment starts at location 0C00 hex in low memory. The area from 0000 to 08FF is reserved by the system for Keyword translation, vector data for sub-keys, and system paramaters. Partitions are controlled by the 0900 to 0BFF region of data memory.

The maximum number of partitions allowed by the Wang is 16. This just happens to be a nice number in hex, permitting the even assignation of memory segments to connote various statuses of each partition. Our first discussions will indicate how the system has allocated memory segments to the partitions.

1.1 Partition Location in data memory

A general formula for computing the offset to find the status of a partition is:

(Address of function) + (Partition # - 1)

The addresses in data memory that tell us what addresses that are assigned to a Partition are 0990 for the memory location within a bank of memory, and 09E0 that tells us which memory bank that partition resides in. Assuming that we wish to find the status of partition 3, the formula gives us the following numbers:

0990 + (3-1) = 0990 + 2 = 0992
09E0 + (3-1) = 09E0 + 2 = 09E2

So the locations of interest to us are 0992 and 09E2. Figure 1 has a map of a sample configuration. From that Figure, we read the values 20 for 0992 and 40 for 09E2. The 0990 segment must be multiplied out by 256 base 10, to get a value of 2000. The starting address of Partition 3 is therefore 40:2000. 40 is the memory bank assignment, meaning the second bank of 64k. It is this area that contains all the individual stored paramaters for that partition, as well as the program. (Refer to the Partition decode section of this manual for further clarification as to the contents of this area.)

Any partitions' starting address can be derived in this manner. A valid partition cannot have a starting address of 00. Thus, if trying to find a partitions starting address, and you come up with the value 0, that partition has not been Sysgened.

1.2 IO determination (Other than CRT)

2.X

Basic 1.2 allows IO to occur among the partitions. Though this statement is obvious, this is not the norm in most operating systems. The usual method for doing this is to make a call to the Supervisor program, which in turn, does all the IO. Wang got around this drawback by the structure of the IO bus.

Simplifying the protocols required to communicate with peripheals allowed the individual partitions to control and talk to the peripheals directly. There are two series of addresses that correspond to the waiting for the completion of IO on a channel. These locations are 0960 and 09B0. Location 0960, (Plus the partition #-1), is set to the device number that the partition is in communication with at the current break.

Therefore, if the partition was accessing device /B50 at the time of partition break, location 0960 + Offset would be set to 50. No communication by any other channel can take place with this device until a full cycle is completed. If any other partitions wish to communicate with device /B50, the request is placed in the 09B0 section of memory, and the partition is effectively put to sleep until that device becomes available for the partition.

So any device number in the 0960 region means that the partition is active on IO, waiting for READY to be asserted by the Device. Any device number in the 09B0 section of memory indicates a REQUEST for IO, and the Partition is effectively sleeping till this occurs.

1.3 BREAK control

Partitions may be made inactive, either temporarily or permamently, in three ways:

1: BREAK (Val)	Gives up Val time slices
2: BREAK !	Permamently deactivates partition
3: IO Waiting	IO device selected is not available at this time, or Not Ready.

Item number 3 was explained in the previous section. The BREAK command subset actually does no more than set a location, 09C0 + partition # -1, to the value indicated by (Val). When that partitions' timeslice occurs, the system sees that a non zero value is here, and decrements that location by 1. It then immediately switches to the next partition in the chain. Eventually, the partition time slices decrement the counter to 0, and the partition becomes alive again.

It should be noted that no guarantee can be made as far as the amount of time that a partition will be kept sleeping. The status of the other partitions will influence the overall 'dead' time.

1.4 Terminal Configuration

Address 0970 + Offset contain the terminal status of the partition. This status word determines which terminal is assigned to that partition, whether it is attached or not, and to which MUX-D module the terminal belongs. Address 0970 is decoded as follows:

```
X X X X   X X X X

! ! ! !   ! ! !  !-----!
! ! ! !   ! ! !---------! These bits indicate Terminal #
! ! ! !   ! !------------) Not used
! ! ! !   !---------------) Terminal Not Assigned
! ! ! !                     (PARTITION NOT ATTACHED)
! ! ! !-------------------) 1 = Waiting for Terminal
! ! !---------------------- 0 = Terminal Attached. 1 = Not
! !-----------------------) MUXD assignemnent (LSB)
!-------------------------) MUXD assignments (MSB)
```

Example: 41 (0100 0001)

Indicates 2nd MUXD , Terminal 1. (5 plus terminal # = 6) Therefore, this is terminal 6.

1.5 Overall Partition Status

The overall partition status is defined by address 0940 + Offset. System paramaters may be gleaned from this status byte. The break-down of this byte is as follows:

```
X X X X   X X X X
! ! ! !   ! ! !  !------  Background Print
! ! ! !   ! ! !---------  Waiting For Output
! ! ! !   ! !------------ Waiting for Input
! ! ! !   !-------------- Undefined
! ! ! !                   TERMINAL JUST GOT ATTACHED
! ! ! !------------------ [redacted]
! ! !-------------------- IO Wait (Wait for Ready)
! !---------------------- Running Program (No IO)
!------------------------ End of Partitions
```

It can be seen that if the MVP Basic sees the 80 bit set, no further partitions are present, and the system reverts back to Partition 1.

1.6 Partition Programming Status

The actual control status of the Partition is derived from address 0980 + Offset. This byte contains information as to whether programming is enabled or disabled, Global accesses and Random number seed location status. The breakdown for the byte is as follows:

```
X  X  X  X   X  X  X  X
!  !  !  !   !  !  !  !----) 0 = Programming Enabled
!  !  !  !   !  !  !         1 = Programming Disabled
!  !  !  !   !  !  !-------) 0 = Normal. 1 = Load and Run
!  !  !  !   !  !            Program whose name is in Partition
!  !  !  !   !  !            Map + 0080.
!  !  !  !   !  !----------) 0 = Normal. 1 = Global Partition.
!  !  !  !   !-------------) 0 = Random Numbers Initialized
!  !  !  !                   1 = System Random Number seed
!__!__!__!                                    10 - LOAD UP PART+80
    !----------------------) Not used         20 - LOAD UP @ DISCONNECT
```

A special note about Global partitions. Though this bit, (04), determines that a partition is global, a Universal partition, residing in the first 8k of memory, is also defined by this bit, and by the fact that the LSB of the MAP address (09E0) is set to a 1. This is the only exception.

1.7 $PSTAT Message Area.

Basic allows the form $PSTAT=(Variable,Name, Ascii) to set the $PSTAT message area. This message area is commonly viewed by means of the @PSTAT program. The storage of the 8 characters that make up the message is located in Address 0B00 + (8 * (Partition# - 1)). Figure 2 indicates this area of memory with the contents of a typical program run.

1.8 $MSG Area

The system message, or banner, is stored at location 0B80 of memory. All users have access to this message.

PARTITION CONTROL AREA

```
00
0900 00 02 20 00 00 00 00 00 00 00 00 03 02 00 30 00 .. ...........0.
0910 02 01 30 00 00 00 00 00 00 00 00 00 00 00 00 00 ..0.............
0920 03 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0930 05 00 00 00 00 00 00 00 00 00 00 01 00 00 00 00 ................
0940 00 04 02 40 02 02 00 00 80 00 00 00 00 00 00 00 ...@............
0950 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0960 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0970 00 02 01 03 40 41 38 38 28 28 28 28 28 28 28 28 ....@A88((((((((
0980 08 08 08 08 08 08 08 08 00 00 00 00 00 00 00 00 ................
0990 0C 86 20 90 20 90 20 90 00 00 00 00 00 00 00 00 .. . . .........
09A0 C0 FF 0C 00 00 00 03 01 0D 12 2B 10 00 42 FF FF ..........+..B..
09B0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
09C0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
09D0 00 00 01 04 00 00 02 8E 00 00 00 00 00 00 00 00 ................
09E0 00 00 40 40 80 80 C0 C0 00 00 00 00 00 00 00 00 ..@@............
09F0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
```

```
00
0A00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0A10 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0A20 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0A30 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0A40 15 0F 10 83 20 8F 30 83 1D 01 19 0F 12 0F 00 00 .... .0.........
0A50 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0A60 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0A70 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0A80 00 00 04 00 04 00 04 00 04 03 04 00 04 00 04 00 ................
0A90 04 00 04 00 04 00 04 00 04 00 04 00 00 00 00 00 ................
0AA0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0AB0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0AC0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0AD0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0AE0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0AF0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
```

```
00
0B00 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
0B10 41 49 44 53 20 20 20 20 41 49 44 53 20 20 20 20 AIDS    AIDS
0B20 41 49 44 53 20 20 20 20 41 49 44 53 20 20 20 20 AIDS    AIDS
0B30 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
0B40 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
0B50 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
0B60 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
0B70 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
0B80 57 4F 52 54 48 20 42 52 4F 41 44 43 41 53 54 49 WORTH BROADCASTI
0B90 4E 47 20 49 4E 43 20 4D 4F 44 45 4C 20 42 2D 31 NG INC MODEL B-1
0BA0 32 32 31 20 42 59 20 43 4F 4D 50 55 54 45 52 20 221 BY COMPUTER
0BB0 43 4F 4E 43 45 50 54 53 20 20 20 20 20 20 20 20 CONCEPTS
0BC0 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
0BD0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0BE0 00 36 00 32 00 36 00 34 00 00 00 00 00 00 00 00 .6.2.6.4........
0BF0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
```

PARTITION CONTROL

```
       00
  1 - 0900 00 02 20 00 00 00 00 00 00 00 00 03 02 00 30 00 .. ...........0.
 17 - 0910 02 01 30 00 00 00 00 00 00 00 00 00 00 00 00 00 ..0.............
 33 - 0920 03 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
 49 - 0930 05 00 00 00 00 00 00 00 00 00 00 01 00 00 00 00 ................
 65 - 0940 00 04 02 40 02 02 00 00 80 00 00 00 00 00 00 00 ...@............
 81 - 0950 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
 97 - 0960 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
113 - 0970 00 02 01 03 40 41 38 38 28 28 28 28 28 28 28 28 ....@A88((((((((
129 - 0980 08 08 08 08 08 08 08 08 00 00 00 00 00 00 00 00 ................
145 - 0990 0C 86 20 90 20 90 20 90 00 00 00 00 00 00 00 00 .. . . .........
161 - 09A0 C0 FF 0C 00 00 00 03 01 0B 12 2B 10 00 42 FF FF ..........+..B..
177 - 09B0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
193 - 09C0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
209 - 09D0 00 00 01 04 00 00 02 8E 00 00 00 00 00 00 00 00 ................
225 - 09E0 00 00 40 40 80 80 C0 C0 00 00 00 00 00 00 00 00 ..@@............
241 - 09F0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
```

173

STATUS of PARTITION

WAITING FOR RDY FROM THIS DEVICE

TERMINAL STATUS

PROGRAMMING STATUS

ADDRESS of PARTITIONS

SYS CONFIGURATION AND CONTROL

SELECTED OPERATIONAL DEVICE

BREAK DATA

MEMORY BANK SELECTION

FIGURE 7

```
00
0B00 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
0B10 41 49 44 53 20 20 20 20 41 49 44 53 20 20 20 20 AIDS    AIDS
0B20 41 49 44 53 20 20 20 20 41 49 44 53 20 20 20 20 AIDS    AIDS
0B30 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
0B40 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
0B50 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
0B60 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
0B70 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
0B80 57 4F 52 54 48 20 42 52 4F 41 44 43 41 53 54 49 WORTH BROADCASTI
0B90 4E 47 20 49 4E 43 20 4D 4F 44 45 4C 20 42 2D 31 NG INC MODEL B-1
0BA0 32 32 31 20 42 59 20 43 4F 4D 50 55 54 45 52 20 221 BY COMPUTER
0BB0 43 4F 4E 43 45 50 54 53 20 20 20 20 20 20 20 20 CONCEPTS
0BC0 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
0BD0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0BE0 00 36 00 32 00 36 00 34 00 00 00 00 00 00 00 00 .6.2.6.4........
0BF0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
```

ϕPSTAT MESSAGE AREA - 8 CHARACTERS PER PARTITION

ϕMSG STORAGE AREA

FIGURE 2

1.9 Special System Areas 09A0

A special systems paramater block, located at address 09A0 is heavily utilized by Basic 2.x.

09A0 defines the highest memory address in the highest bank of Data memory divided by 256. Set only during the initial loading of Basic, but is accessed by the $PSTAT message.

09A1 contains the highest bank selection bits available to the system.

At location 09A2, the location of the program segment of memory currently being executed is stored.

At location 09A3, the bank selection of the currently executing partition is saved. Thus, the user can find which bank is being accessed, of the MVP by examining these locations.

Location 09A4 contains the terminal number of the partition currently being serviced.

Location 09A7 bit 1 indicates that the MVP has been initialized. If it is 0, then the $INIT command is allowed. A one bit indicates that the system has already performed a $INIT and cannot be reconfigured.

09AA and 09AB locations will contain the defaulted disk device. This device is passed initially to MVP by the prom bootstrap routine via AR 1A. After initialization time, the defaulted disk is set here by the $INIT routine.

Location 09AC through 09AF contain the active MUXD/E addresses that are on-line.

1.10 System statuses 09D0

Locations 09D0 - 09DF contains other interseting system information.

09D1 is normally set to a one (1), whenever a suspended IO Wait operation is in progress. That is, some partition wanted access to a device, but that device was busy, so that partition was put to sleep.

09D2 and 09D3 seem to be some sort of priority encoding scheme, though I don't have full comprehension of these features as yet.

Locations 09D6 and 09D7 contain a pointer to the error routine. If an recoverable error was encountered, these locations are read and used as a vector to the routine to goto. ERROR X=ERR!

— 09D8 reference by Magic terminal Status

Location 09DA is a magic location by Wang. If the high nibble is non-zero, a CBS with data set to 02 is sent to device FD, a timer test module made by engineering at Wang. This test is done at swap out time of a partition, and I reckon this to be a stop-timing code.

If location 09DA bit 1 is false, 0, then a CBS code of 01 is sent to the device FD at partition time entry, which I assume to be start timer code. However, if the 09DA bit 1 is true, 1, then the parition number is first compared with the low order nibble of this address, and only if they compare will the CBS 01 be executed.

This gives the option of monitoring just a single partition.

1.11 *Summary of 0900 Data Memory Functions*

Below is a total list, ordered, of the 0900 memory control region:

Address	Function
0900	Math TS
0910	Math TS
0920	Math TS
0930	Math TS
0940	Overall Partition status
0950	Unknown
0960	IO device being used by Partition
0970	Terminal Configuration
0980	Programming Status of Partition
0990	Addresses of Data memory for Partitions
09A0	System Configuration and Current Partition control
09B0	Devices selected and waiting for IO
09C0	Partition BREAK data
09D0	Various System statuses
09E0	Memory bank assignment for partitions
09F0	@24 area for control of MXE module
0A00	System common peripheals share area
0B00 to 0B7F	$PSTAT message area
0B80	$MSG storage area

2.0 AR usage in Basic 2.x

Though not necessarily linked with partition control, the AR usage section does indicate what the particular partition is doing at any on etime, and various other statuses about the partition.

We will first start with what I call the general AR's.

2.1 AR 00

AR 00 is used as both an input and output from subroutines that utilize paramater passing. In general, AR 00 will always point to the location of data being passed to or from a subroutine.

As an example, if a subroutine was evaluating the string:

STR(A$,10,200)

The contents of AR 00 would reflect the position of the variable A$ offset by 9 locations.

During your subroutine usage, AR 00 is most likely to be used as TS area.

2.2 AR 01 through AR 0F

No particular functions are associated with these AR's, as they are used by various routines for various functions. They may be used by you as TS AR's, as they are inclusive within ATOM decoding only. However, if you call various routines during the course of your programming, please be aware that they may not contain what you put into them.

AR 0D through AR 0F are used by the disk routines, so keep your hands off these.

2.3 AR 10

AR 10 always points to the current position within a line that is currently being executed. That is, as an atom is being processed, the AR 10 is being incremented through memory. This must be remembered by us, in that we must increment AR 10, or use routines that do this, during our own parsing of data.

Note that if an error occurs, AR 10 is the one referenced to display that funny up arrow!

2.4 AR 11

AR 11 always points to the last program location in data memory for that partition. Free memory is present from this point to the bottom of the variables.

2.5 AR 12

In conjuction with AR 10, AR 12 points to the beginning of a line of code in memory. Again, when an error is noted, AR 12 will allow the Basic program to display the full line. AR 12 will get reset to the next line by the normal GOTO, GOSUB, ON ALERT or normal fall out of a line.

2.6 AR 13

SYMBOLS

Temporary storage pointer for the bottom of the ~~value stack~~.

2.7 AR 14 ?? not sure ??

2.8 AR 15

AR 15 points to the current position in the Operator stack. The operator stack holds the RPN data, GOSUB data and FOR-NEXT information.

2.9 AR 16

Points to the bottom of the variable table that is available for the user. Free space may be detected by subtracting AR 11 from AR 16. However, data is stacked from this location downward during run time, so free space may elude us if large amounts of data is pumped here - discuss in class.

2.10 AR 17

The most mysterious of AR registers is AR 17. This register is the basic programming status of the system. Data here tells us whether the program is running or not, protected or not, scrambled or not, resolved or not.

2.11 AR 18   upper byte is colon 1E

This is a dual function register. The upper byte of the register is ill-defined, and the author has not played with it enough to confirm what it does. The lower byte tells us at what semi-colon, that is sub-element, of a line we are in. 00 in this byte tells us we are parsing the first piece of a line, while 02 would inform us that we are working on the third statement in a line.

2.12 AR 19   *?? no information ??*

2.13 AR 1A

Device address currently selected. This tells us what exact device is being accessed at this time.

2.14 AR 1B

AR 1B is the mathametical status register. Bits here tell us whether ROUND is in effect, and whether Radians, GRADS or Degrees is the default for the trigonometric functions. Furthermore, the upper byte tells us what error can be ignored. (C errors) SELECT ERROR) 62

```
AR 1B    ee  yy
             !--     bits 00 = Radian
                          10 = Degrees
                          20 = Grads
                          CO = No Round
                          40 = No round (Internal to Trig)
```

Where ee = error number to bypass

2.15 AR 1C *?? no comment ??*

2.16 AR 1D

The upper byte of AR 1D tells us what the desired output line width for the print device is. The lower byte tells us at what character position within a line we are at.

2.17 ***AR 1E***

Last location after program available for keyboard entry during either Immediate mode or new program.(??)

2.18 ***AR 1F***

Device that is to be selected currently. The upper byte indicates the ~~width~~ COLUMN of the device, AT TIME OF BREAK! while the lower byte dictates what device is to be selected.

2.19

In general, it is a good idea to keep your hands off any AR register above 10. Modification of these may result in disastrous results.