October 2007

IBM z/OS HCD & HCM Newsletter No 27

Multiple Subchannel Sets: An implementation view

Contact: ibmhcd@de.ibm.com
ibmhcm4z@cn.ibm.com

Multiple Subchannel Sets: An implementation view

October 2007

Iain Neville
Consulting IT Specialist, IBM System z
IBM Systems and Technology Group
IBM United Kingdom

Acknowledgements

Thanks to the following people for their contributions to this paper:

Friedrich Beichter
IBM Systems and Technology Group
HCD/HCM development

Harry Yudenfriend
Distinguished Engineer, IBM Systems and Technology Group
System z9 Software Design, System z9 I/O Strategy and Architecture

Background

The purpose of this document is to demonstrate a pragmatic implementation approach for the adoption of Multiple Subchannel Sets (MSS). Whilst the technology and functional requirements may be clearly understood, knowledge of the following must be available to facilitate both a practical and strategic implementation:

- Availability of device ranges within the existing 64K channel subsystem limit.
- DASD capacity planning detail for a period of 5 years+.
- Logical Subsystem alignment of existing and future Disk subsystems.
- Disk subsystem support of new technology (e.g. IBM HyperPAV).
- Performance profile of disk subsystems.
- Strategic plans for further peripheral deployment.
- DASD replication requirements.

Multiple Subchannel Sets – information items

These rules and guidelines should be understood prior to any implementation.

Hardware requirements

Two subchannel sets are available on System z9 servers. Within this, subchannel set-0 (SS0) may have up to 63.75k subchannels and subchannel set-1 (SS1) may have up to 64k subchannels. zSeries servers (pre z9) only have one subchannel set of 63k subchannels.

Software requirements

Multiple Subchannel Sets require z/OS 1.7 and above. Any LPAR < z/OS 1.7 cannot see devices in SS1.

Device number alignment

There is no required correspondence between addresses in the two sets. For example, we might have device number 8000 in subchannel set 0 and device number 8000 in subchannel set 1 and they might refer to completely separate devices. (We know that the device number in subchannel set 1 must be an alias for z/OS, but that is all we can know from the device number.) Likewise, device number 1234 (subchannel set 0) and device number 4321 (subchannel set 1) might be the base and an alias for the same device. There is no required correspondence between the device numbers used in the two subchannel sets.
Typically, base devices (3390B) are defined from the start of the range (00 forwards). Alias devices are defined from the last device in the range backwards. If the range is filled, this will be FF backwards. Unit addresses of base and alias devices on a single control unit must be unique. These can not be duplicated across subchannel sets.

Architecture

Only z9 processors support Multiple Subchannel Sets. An additional high order digit (either a 0 or a 1) is logically added to existing device numbers (e.g. 08000 for dev 8000 in SS0 & 18000 for dev 8000 in SS1). This enhancement is not delivered by system code as there is still an architectural requirement of four-digit addresses (device numbers, subchannels). However, some messages will contain the subchannel set number and one can mentally use it as a high-order digit for device numbers.

Preparation

Planning your device ranges must be done with a long term strategic view and devices should be predefined in Subchannel Set 1. This is established via the MAXDEV setting on the channel subsystem definition which correlates to the RESOURCE statement in the IOCP. This preallocates the HSA accordingly. A power on reset is required to change this setting.

Migration tools

For relocation of device ranges (UCBs) between SS0 and SS1, dynamic reconfiguration can be used.

For consolidation of device ranges involving migration of data, various tools are available; some of those are mentioned below.

- DFSMSDSS – for migration of data at either a volume level (DEVICE) or logical level (DATASET).
- Transparent Data Migration Facility (TDMF). This product moves data at the block and volume level without disruption.
- Logical Data Migration Facility (LDMF). This product migrates datasets at the volume level and is typically used to consolidate smaller volumes (e.g. 3390 model 2) to larger volumes (e.g. 3390 model 54).

Numbering Schemes

There is not a correct approach for all circumstances. The chosen approach will vary depending on the customer requirements. However, we can demonstrate the different options. Also, different implementations of PAV may have restrictions on the alignment of alias addresses within the range. This should be checked with your disk vendor prior to selecting a numbering scheme.

SCHEME1 - Compacting or rolling reuse of addresses

The numbering scheme is duplicated across SS0 and SS1 by using the UA field for those 3390A in SS1.

E.g.

5000,72	UNITADD=00 in SS0
5000,184	UNITADD=48 in SS1

Compaction is achieved in both SS0 and SS1 by using the follow on numbers in each SS (5048 is next in SS0, 50B7 is next in SS1). This achieves maximum reuse of device numbers. However, it should be noted that this does look confusing as the device addresses in SS1 are allocated unit addresses that do not match the numbering scheme:

IODEVICE ADDRESS=(50B8,184),UNITADD=48, CUNUMBR=(5000),STADET=Y*
,DESC='IBM 2107 DASD',UNIT=3390A,SCHSET=1

HCD examples with SCHEME1 :

The device definition below shows range 1000-1047 defined as 3390B devices in CNTLUNIT 1000.

FIGURE1: CNTLUNIT 1000, 3390B definitions for UA xx00-xx47

```
                        Device / Processor Definition
                                                                  Row 1 of 1
Command ===> ____________________________________________ Scroll ===> CSR

Select processors to change device/processor definitions, then press
Enter.

Device number  . . : 1000       Number of devices  . : 72
Device type  . . . : 3390B

                                        Preferred  Device Candidate List
/ Proc.CSSID  SS+  UA+  Time-Out  STADET  CHPID +    Explicit        Null
_ MINE.0      _    __   No        Yes     __         No              ___
****************************** Bottom of data ******************************




 F1=Help        F2=Split       F3=Exit        F4=Prompt      F5=Reset
 F6=Previous    F7=Backward    F8=Forward     F9=Swap       F12=Cancel
F22=Command
```

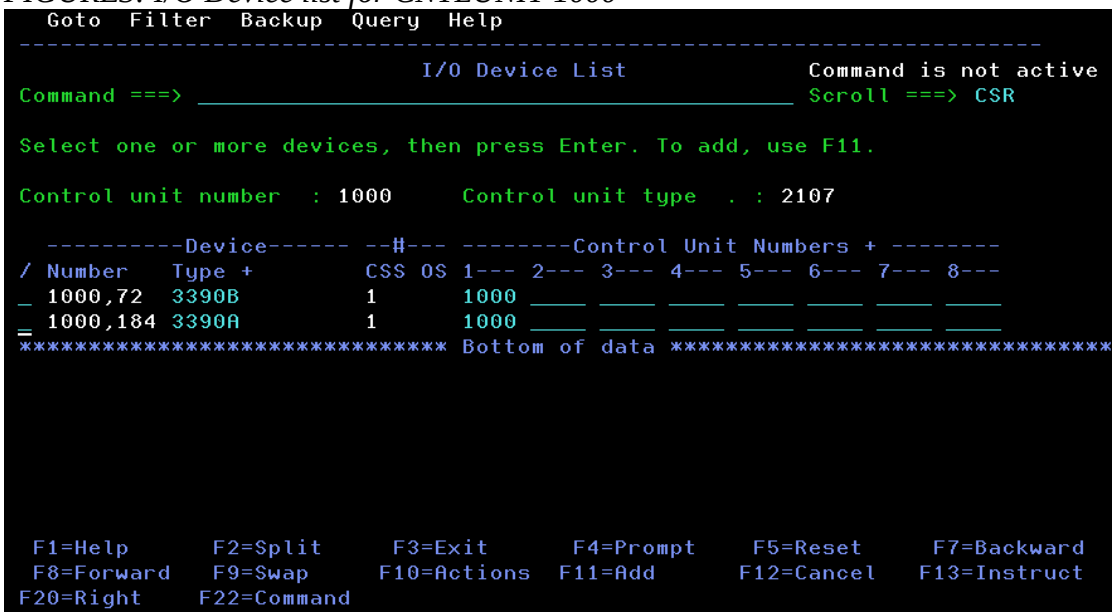
The remainder of this range (xx48-xxFF) will be defined as 3390A devices but with the same device address (1000). This is achieved using the UA field of the device definition as shown in Figure2 below. Note the alignment to SS1 (which is the default for alias devices attached to a z9 processor within HCD).

FIGURE2: CNTLUNIT 1000, 3390A definitions for UA xx48-xxFF

```
                        Device / Processor Definition
                                                                  Row 1 of 1
Command ===> ____________________________________________ Scroll ===> CSR

Select processors to change device/processor definitions, then press
Enter.

Device number  . . : 1000       Number of devices  . : 184
Device type  . . . : 3390A

                                        Preferred  Device Candidate List
/ Proc.CSSID  SS+  UA+  Time-Out  STADET  CHPID +    Explicit        Null
_ MINE.0      1    48   No        Yes     __         No              ___
****************************** Bottom of data ******************************




 F1=Help        F2=Split       F3=Exit        F4=Prompt      F5=Reset
 F6=Previous    F7=Backward    F8=Forward     F9=Swap       F12=Cancel
F22=Command
```

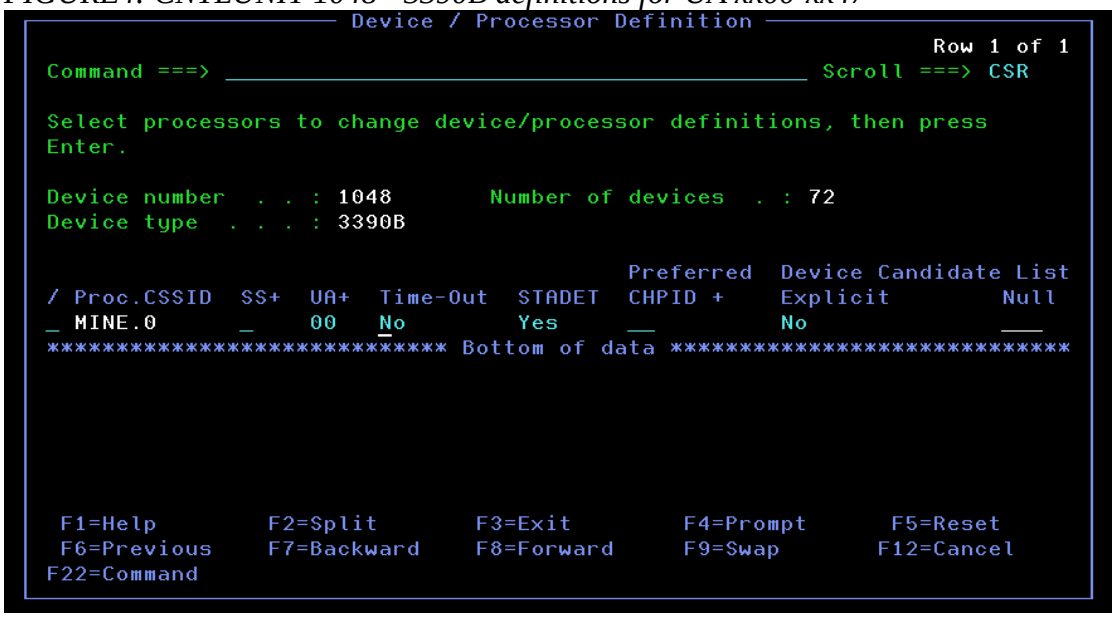
The resulting devices in CNTLUNIT 1000 are shown in Figure3 below.

FIGURE3: I/O Device list for CNTLUNIT 1000



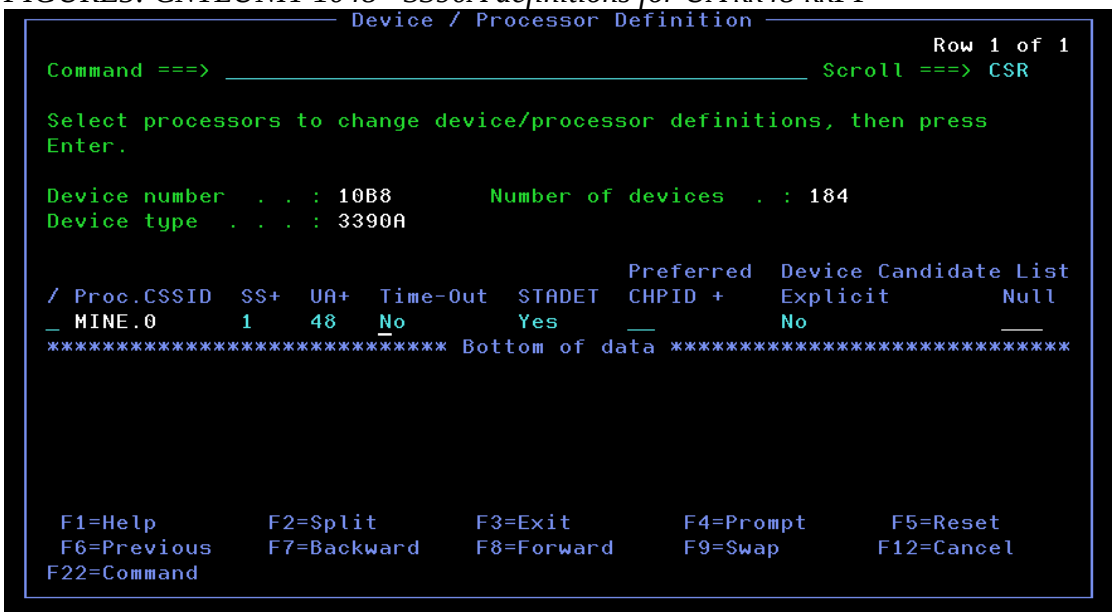
The next CNTLUNIT is numbered to align with the next available device address in SS0 (1048). The next available Devices are defined as bases (3390B) to SS0 as reflected in Figure4 below. Note that we specify UA of 00. This looks strange as we're defining devices 1048-8F but forcing a UA of 00 for 1048, 01 for 1049 etc.

FIGURE4: CNTLUNIT 1048 - 3390B definitions for UA xx00-xx47



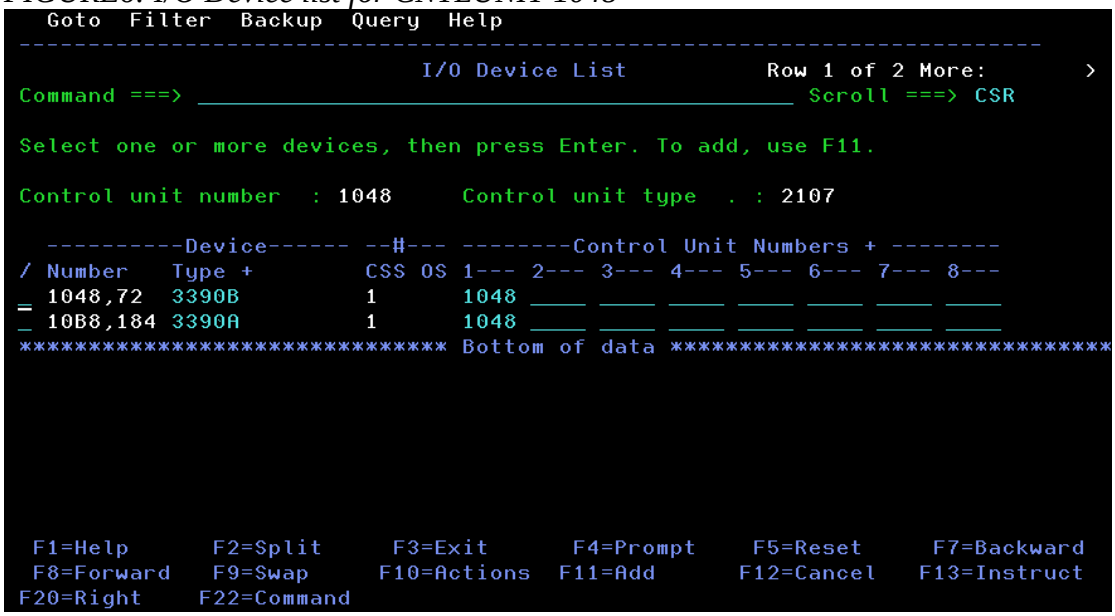
The remainder of the range in CNTLUNIT 1048 is then filled with the alias devices in SS1 (UA xx48-xxFF). Again, we generate a mismatch between the device number and the associated unit address (10B8 with UA of 48) but this is perfectly valid. This is shown in Figure5 below.

FIGURE5: CNTLUNIT 1048 - 3390A definitions for UA xx48-xxFF



The resulting device list for CNTLUNIT 1048 is shown in Figure6 below.

FIGURE6: I/O Device list for CNTLUNIT 1048



Example of IOCP statements with SCHEME1:

```
CNTLUNIT CUNUMBR=1000,PATH=((CSS(0),04,08,2C,31,33,37,6F,75)),*
        UNITADD=((00,256)), UNIT=2107

    IODEVICE ADDRESS=(1000,72),UNITADD=00,CUNUMBR=(1000),STADET=Y,*
        DESC='IBM 2107 DASD',UNIT=3390B
```

```
      IODEVICE ADDRESS=(1000,184),UNITADD=48,CUNUMBR=(1000),STADET=Y*
           ,DESC='IBM 2107 DASD',UNIT=3390A,SCHSET=1
```

Running totals

SS0	SS1
1000-1047	1000-10B7

```
CNTLUNIT CUNUMBR=1048,PATH=((CSS(0),04,08,2C,31,33,37,6F,75)),*
           UNITADD=((00,256)),UNIT=2107

      IODEVICE ADDRESS=(1048,72),UNITADD=00,CUNUMBR=(1048),STADET=Y,*
           DESC='IBM 2107 DASD',UNIT=3390B

      IODEVICE ADDRESS=(10B8,184),UNITADD=48,CUNUMBR=(1048),STADET=Y*
           ,DESC='IBM 2107 DASD',UNIT=3390A,SCHSET=1
```

Running totals

SS0	SS1
1000-1047	1000-10B7
1048-108F	10B8-116F

SCHEME2 - DASD versus "the rest"

This scheme reuses the freed device numbers in SS0 for all devices other than DASD:

E.g. 1000,72 for the base devices (UA=00..47), 1048,184 for the alias devices in SS1 (UA=48..FF), and use the freed device numbers in SS0 (1048..10FF) for device types other than DASD.

Example of IOCP statements for SCHEME2 :

```
CNTLUNIT CUNUMBR=1000,PATH=((CSS(0),04,08,2C,31,33,37,6F,75)),*
           UNITADD=((00,256)),UNIT=2107

      IODEVICE ADDRESS=(1000,72),UNITADD=00,CUNUMBR=(1000),STADET=Y,*
           DESC='IBM 2107 DASD',UNIT=3390B

      IODEVICE ADDRESS=(1048,184),UNITADD=48,CUNUMBR=(1000),STADET=Y*
           ,DESC='IBM 2107 DASD',UNIT=3390A,SCHSET=1
```

Running totals

SS0	SS1
1000-1047	1048-10FF

Devices 1048-10FF are now available in SS0 for use by non-DASD devices ("the rest").

SCHEME3 - Pairing

With pairing, consider the split of base and alias devices. The first half of an LSS range (128 devices) is allocated to the initial base range. We then reuse this device range for the next consecutive assignment of devices within SS1.

This scheme only works when no more than 128 alias devices are defined for a range of base devices. Devices are aligned to unit addresses with this scheme.

In essence, this approach facilitates double the addressable devices. This is reflected in the table below.

RANGE	CNTLUNIT	USED in SS0		USED in SS1
8000-80FF	**8000**	8000-807F		8080-80FF
	8080	8080-80FF		8000-807F
8100-81FF	**8100**	8100-817F		8180-81FF
	8180	8180-81FF		8100-817F

****Note:** The unitaddress (UA) field for each device will always start at 00 for base addresses and 80 for alias addresses. This reflects a difference between device address number and Unit Address when the range in SS0 starts on the xx80 boundary. For example devices 8080-80FF in SS0 above have UNITADD 00-7F whilst 8000-807F IN SS1 have UNITADD 80-FF.

This device range split (base versus alias) is very conservative in a FICON implementation, particularly if we consider HyperPAV which is discussed in more detail later on in this document.

HCD examples with SCHEME3 :

This section reflects the definition of control units to establish paired device ranges.

CNTLUNIT 8000

Figure7, below reflects that CNTLUNIT 8000 is defined with the following to fill the device range:

8000 - 807F as 3390B devices in SS0.
8080 - 80FF as 3390A devices in SS1

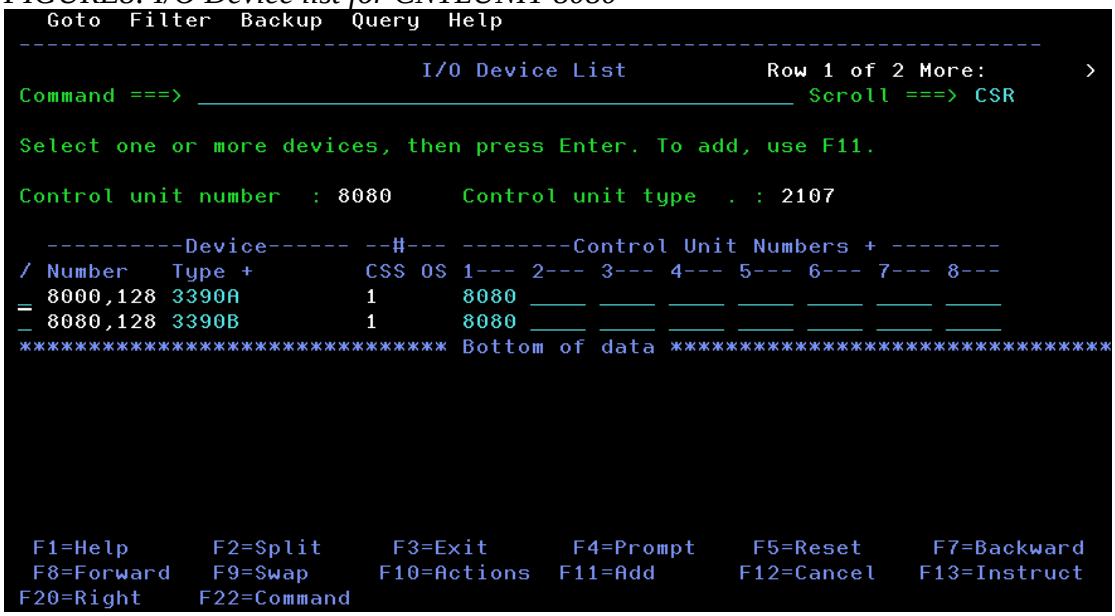
FIGURE7: I/O Device list for CNTLUNIT 8000

```
  Goto  Filter  Backup  Query  Help
--------------------------------------------------------------------------------
                            I/O Device List          Row 1 of 2 More:       >
Command ===> ____________________________________________ Scroll ===> CSR

Select one or more devices, then press Enter. To add, use F11.

Control unit number  : 8000     Control unit type  . : 2107

  ----------Device------ --#--- --------Control Unit Numbers + --------
/ Number    Type +        CSS OS 1--- 2--- 3--- 4--- 5--- 6--- 7--- 8---
_ 8000,128  3390B         1      8000 ____ ____ ____ ____ ____ ____ ____
_ 8080,128  3390A         1      8000 ____ ____ ____ ____ ____ ____ ____
******************************* Bottom of data ********************************



 F1=Help      F2=Split     F3=Exit      F4=Prompt    F5=Reset     F7=Backward
 F8=Forward   F9=Swap      F10=Actions  F11=Add      F12=Cancel   F13=Instruct
F20=Right     F22=Command
```

CNTLUNIT 8080
Figure8, below reflects the reuse of this device range in CNTLUNIT 8080.

8080 - 80FF as 3390B devices in SS0.
8000 - 807F as 3390A devices in SS1.

FIGURE8: I/O Device list for CNTLUNIT 8080



Example of IOCP statements – SCHEME3:

CNTLUNIT CUNUMBR=8000,PATH=((CSS(0),04,08,2C,31,33,37,6F,75)),*
UNITADD=((00,256)),UNIT=2107

IODEVICE ADDRESS=(8000,128),UNITADD=00,CUNUMBR=(8000),STADET=Y,*
DESC='IBM 2107 DASD',UNIT=3390B

IODEVICE ADDRESS=(8080,128),UNITADD=80,CUNUMBR=(8000),STADET=Y*
,DESC='IBM 2107 DASD',UNIT=3390A,SCHSET=1

Running totals
SS0 SS1
8000-807F 8080-80FF

CNTLUNIT CUNUMBR=8080,PATH=((CSS(0),04,08,2C,31,33,37,6F,75)),*
UNITADD=((00,256)),UNIT=2107

IODEVICE ADDRESS=(8080,128),UNITADD=00,CUNUMBR=(8080),STADET=Y,*
DESC='IBM 2107 DASD',UNIT=3390B

IODEVICE ADDRESS=(8000,128),UNITADD=80,CUNUMBR=(8080),STADET=Y*
,DESC='IBM 2107 DASD',UNIT=3390A,SCHSET=1

Running totals
SS0 SS1
8000-807F 8080-80FF
8080-80FF 8000-807F

SCHEME4 – Filling the ranges

This scheme preserves the ranges in SS0 and aligns devices to unit addresses. This scheme is driven by the desired BASE to Alias split that will have been calculated in advance (potentially via the PAV analysis tool). Basic principles used here are as follows:

1. Establish BASE to ALIAS ratio first. Align on to boundaries of 16 devices.
2. Stick to this alignment for all Device Ranges.
3. All Ranges are filled with 256 devices (00-FF).
4. Use up all Device Ranges. (A Range being an LSS of 256 devices 00-FF).
5. Define alias devices to SS1.
6. Reuse devices used in SS1 as bases in SS0.
7. Reuse devices allocated in SS0 in SS1.

Example: 192 Bases to 64 Aliases.

- There are 256 available ranges (256 x 256 = 65536 or FFFF).
- The 64 aliases in SS1 free up 256 x 64 = 16384 devices in SS0 for new ranges.
- 16384 / 192 = 85 additional ranges

Using the example above, a worksheet can be used to reflect the available ranges.

RANGE	CNTLUNIT	USED in SS0	Free in Range SS0		USED in SS1	Free in Range SS1
1000-10FF	**1000**	1000-10BF	10C0-10FF		10C0-10FF	1000-10BF
1100-11FF	**1100**	1100-11BF	11C0-11FF		11C0-11FF	1100-11BF
1200-12FF	**1200**	1200-12BF	12C0-12FF		12C0-12FF	1200-12BF

As this scheme is followed, we can see the available numbers in SS0 for reuse. Remember, there is no allegiance between the device number and the Unit Address. So, in an LSS, we can reuse the "Spare" device numbers accordingly. The example below reflects reuse of these ranges (i.e. those in columns titled "Free in Range SS0" for bases and "Free in Range SS1" for aliases).

Example: CNTLUNIT 1001:

- DEVICES in SS0 are reused from the worksheet above for BASE devices:

  10C0-10FF (UA 00-3F)
  11C0-11FF (UA 40-7F)
  12C0-12CF (UA 80-B0)

- DEVICES in SS1 are reused from the worksheet above for ALIAS devices:

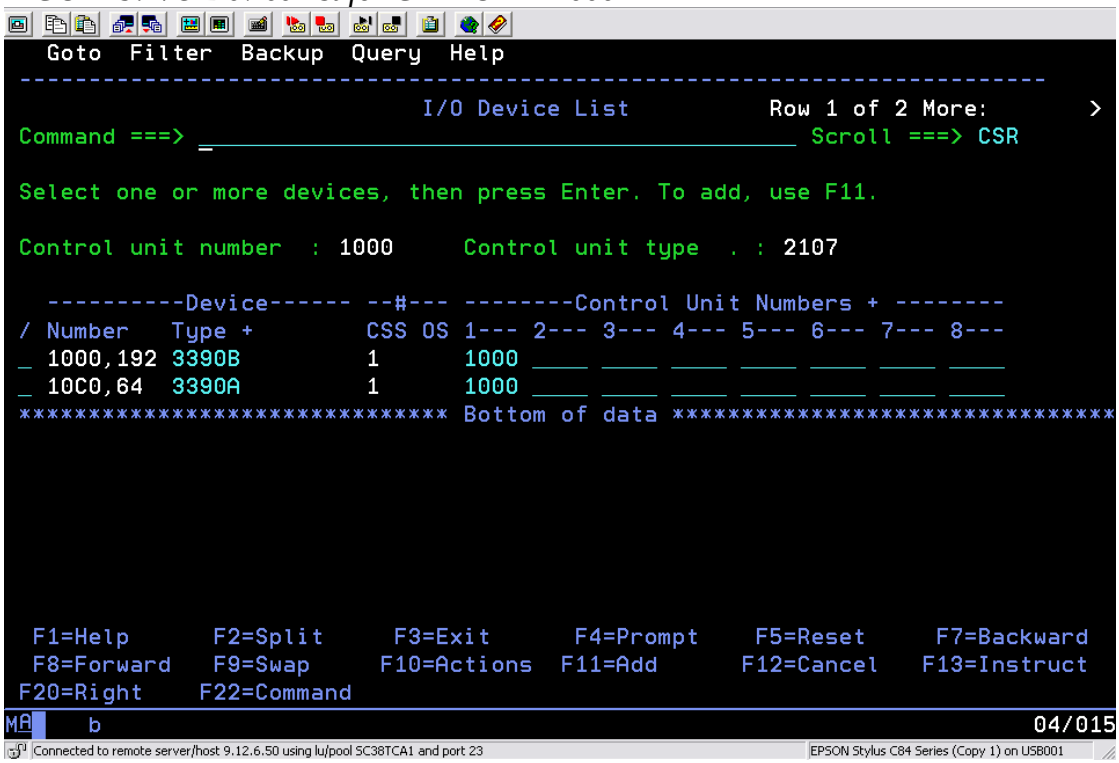
  1000-103F (UA C0-FF)

This makes up a range of 256 devices from the unused devices in each Subchannel set as shown in the worksheet above.

HCD examples for SCHEME4 :

This section reflects the HCD definitions in support of SCHEME4. The base to alias split is established as reflected in the worksheet above.

FIGURE9: I/O Device list for CNTLUNIT 1000



With the 256 ranges used, the devices unused in SS0 can be defined to additional CNTLUNITs as reflected in FIGURE10 below.

FIGURE10: I/O Device list for CNTLUNIT 1001

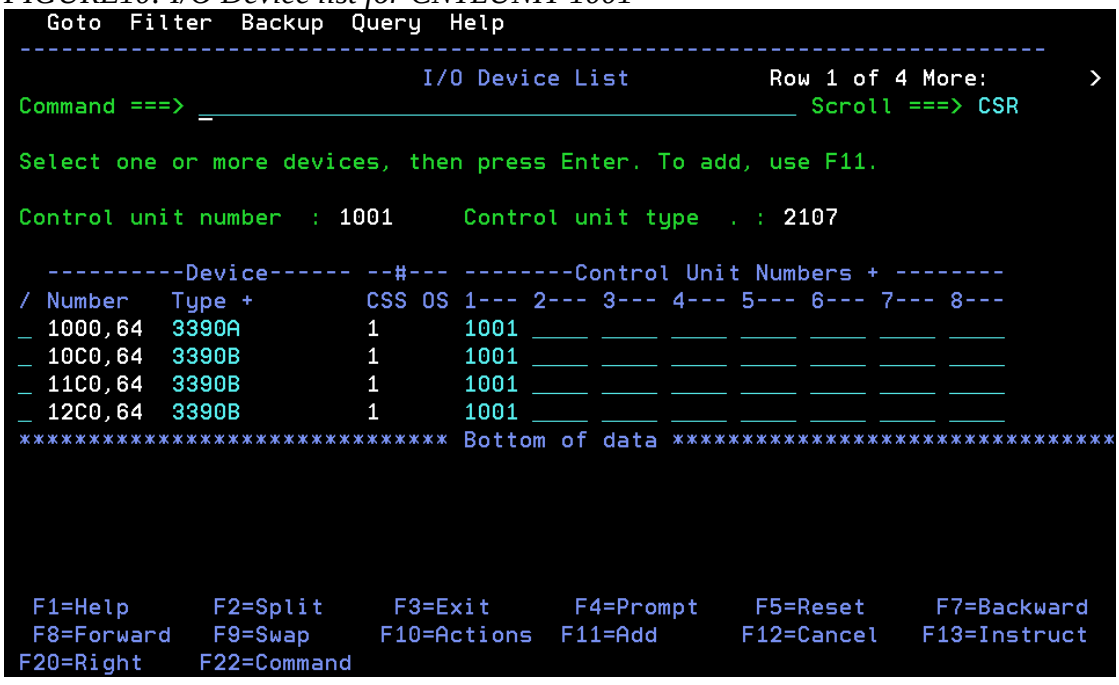


FIGURE10, above shows the reused device ranges in CNTLUNIT 1001. Ranges 10C0-10FF, 11C0-11FF and 12C0-12FF are defined as base addresses in SS0. Range 1000-103F is defined as alias addresses in SS1.

Example of IOCP statements with SCHEME4 :

The CNTLUNITs 1000, 1100 and 1200 would be defined as normal. An example is shown below for CNTLUNIT 1000. Similar definitions would be made for CNTLUNITs 1100 and 1200.

```
CNTLUNIT CUNUMBR=1000,PATH=((CSS(0),04,08,2C,31,33,37,6F,75)),*
        UNITADD=((00,256)),UNIT=2107

     IODEVICE ADDRESS=(1000,192),UNITADD=00,CUNUMBR=(1000),STADET=Y,*
        DESC='IBM 2107 DASD',UNIT=3390B

      IODEVICE ADDRESS=(10C0,64),UNITADD=C0,CUNUMBR=(1000),STADET=Y*
        ,DESC='IBM 2107 DASD',UNIT=3390A,SCHSET=1
```

Those devices left free in SS0 and SS1 respectively can then be reused to make complete ranges of 256 devices. An example is shown below for CNTLUNIT 1001:

```
CNTLUNIT CUNUMBR=1001,PATH=((CSS(0),05,09,2D,32,35,38,70,76)),*
        UNITADD=((00,256)),UNIT=2107

     IODEVICE ADDRESS=(10C0,64),UNITADD=00,CUNUMBR=(1001),STADET=Y,*
        DESC='IBM 2107 DASD',UNIT=3390B

     IODEVICE ADDRESS=(11C0,64),UNITADD=40,CUNUMBR=(1001),STADET=Y,*
        DESC='IBM 2107 DASD',UNIT=3390B

     IODEVICE ADDRESS=(12C0,64),UNITADD=80,CUNUMBR=(1001),STADET=Y,*
        DESC='IBM 2107 DASD',UNIT=3390B

      IODEVICE ADDRESS=(1000,64),UNITADD=C0,CUNUMBR=(1001),STADET=Y*
        ,DESC='IBM 2107 DASD',UNIT=3390A,SCHSET=1
```

Parallel Access Volumes (PAVs)

Technology has moved on significantly. This section discusses the different implementations of PAV and how this impacts the number of aliases we align to our base volumes. We'll not go into specific functional detail of PAV functionality as that's covered in other publications such as the following techdoc:

http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/TD100311

Dynamic versus Static PAVs

The initial implementation of Parallel Access Volumes was static PAV. Here, specific alias devices are assigned to bases within the logical subsystem (LSS). These aliases may not be re-assigned to any other base device.

With Dynamic PAV, alias devices may be unassigned and reassigned to different base devices. Aliases are assigned on an "as needed" basis with Workload Manager reacting to performance goals.

Predicting the split of bases and aliases

It is not trivial to predict the optimum ratio of Alias to Base addresses. If the ratio is too large, the amount of physical volumes is limited. If too small, this impacts IOSQ and response time.

Basic rules of thumb to achieve base to alias alignment are as follows:

1. Use as many aliases as you can afford.
   - Use the following table for a conservative recommendation. This is typically used for ESCON implementations.

Number of cylinders	Number of Aliases for Dynamic PAV	Number of Aliases for Static PAV
1-3339	1/3	1
3340-6678	2/3	2
6679-10,017	1	3
10,018-16,695	1 1/3	4
16,696-23,373	1 2/3	5
23,374-30,051	2	6
30,052-40,068	2 1/3	7
40,069-50,085	2 2/3	8
50,086-60,102	3	9
60,193<	3 1/3	10

2. Base on Channels per LSS
   - 6 x number of FICON channels to the LSS = recommended aliases
3. Allocate I/O rate * response time * .002
   - Example: 5 ms service time with 4000 I/O per second needs 40 aliases

HyperPAV

With HyperPAV technology, z/OS uses pools of aliases (by LSS). As each application I/O is requested, if the base volume is busy with another I/O, z/OS selects (and removes) a free alias from the pool, and starts the I/O to the base address through the selected alias. When the I/O completes, the alias device is used for another I/O on the LSS or is returned to the free alias pool. If too many I/Os are started simultaneously, z/OS will queue the I/Os at the LSS level. Queued I/O is done FIFO within assigned I/O priority.

The table directly below summarizes the functional differences:

Attribute	PAV	HyperPAV
Alias to base ratio	**Complex**: One alias per 9GBs, depending on workload and I/O rates and response time requirements. 192 PAV-aliases/64 PAV bases with Mod 27 volume sizes	**Simple:** (Peak I/O rate * average response time * 2) (10x reduction in the number of PAV-aliases). 20 PAV-aliases/236 PAV-bases (10,000/sec * 1 ms * 2)
Workload Management	**Sluggish**: WLM adjustments every 10 seconds when work is not meeting goals, every minute when goals met. I/O Priority by device **Multi-system overheads:** Aggregate multi-system measurements across SYSPLEX to make correct decisions	**On Demand**: Instantaneous response to changing work loads and I/O skews across devices. I/O priority by LSS. **No multi-system overhead**: No multi-system aggregation of measurements needed
RMF	Number of PAV-aliases bound to a base per interval.	Number of times I/O could not start because a PAV-alias was not available, high water mark for PAV-aliases in use for LSS
Improved Efficiency	**Alias used for specific base:** PAV-aliases bound to the same base across all operating system images **Interlocked**: Device-state-change interrupt required in order to insure multi-system coordination of dynamic PAV-aliases Limited parallelism	**Alias used for any base, any time on any system**: Each operating system image uses PAV-alias for a different base at the same time, multiplier for the amount of effective number of aliases. **No interlock required:** Improved parallelism for overcoming the speed of light penalty for replication at distance.
VSCR	PAV-aliases reside in 31 bit storage only.	10 x reduction in PAV-aliases UCBs and device related data structures.

Notice that in each z/OS image within the sysplex, aliases are used independently. WLM is not involved in alias movement so it does not need to collect information to manage HyperPAV aliases. If each LPAR needs 20 aliases then with Parallel access volumes, 60 aliases are required for 3 LPARS. With HyperPAV, the same requirement can be serviced with 20 aliases.

A simple HyperPav analogy:

- 4 x LPARs share a DASD LSS
- 20,000 I/Os per second go to the LSS
- Average response time is 1 millisecond

This means that on average, there are 20 I/Os outstanding at any given instance, 5 from each of the 4 LPARs. If all requests were for the same BASE device, 4 Aliases would be needed.

A total of 10 – 20 could be defined to cover spikes, depending on individual comfort zones.

HyperPAV prerequisites

The HyperPAV capability is available with version 2 of IBM's DS8000 storage subsystems. Software levels of z/OS 1.6 and above is necessary with the relevant maintenance from the HyperPAV subset of the PSP bucket:

http://www14.software.ibm.com/webapp/set2/psp/srchBroker

PAV analysis tool

To gauge a better understanding of your workload profile and achieve a more precise split, the PAV analysis tool may be used. This displays PAV alias usage over time in 3-D graphical representation. This is available from the following URL:

http://www-03.ibm.com/servers/eserver/zseries/zos/unix/bpxa1ty2.html