ZTEST POOL and ZTEST PO64

The POOL driver is intended to allow the user of any system to quickly cycle through the available pools. With just one command it is possible to discard any number of pools, either by losing them or returning them. The POOL driver functions are needed to:

- Cycle through pool directories quickly.
- Immediately start fallback.
- Conveniently lose pools to recoup.

Note: The PO64 driver is the 64-bit version of the POOL driver. All functions are the same between the POOL driver and the PO64 driver.

Requirements and restrictions
None.

Format

```
>>>---ZTEST--++-----+-- --+-POOL-+-- ---------------------------------->  
  +-- '-PO64-'

>- +type-- --count>-- -Maxint-10-.
  |  + -Return>>---'+--+-++++++------++
  |   |   '+'-Toss-x'- | '+'-Maxint-y-'
  |   |   '-Double----------------'
  +-'-STATus--+-------+--+---------+-------------------------------+
    '- -All-'  '- -CLEAR-'  

+-ABORT--+-Help--------------------------------------------------------
```

*i* indicates the specific I-stream in which the driver will be run. If *i* is not specified, the test case(s) will be executed on the I-stream on which the command is entered.

* specifies the driver will be invoked on all currently defined and available I-streams.

**POOL**
required keyword that specifies the POOL driver is to be run.

**PO64**
required keyword that specifies the 64-bit version of the POOL driver is to be run.
type
indicates the pool record type, where type can be SST, SLT, SDP, LST, LLT, LDP, 4ST, 4LT, 4DP, or 4D6

count
discards the number of pools of the size specified.

Return
returns pool normally depending upon the type needs and RTA mounted for DUPT.

Double
specifies that pool is double released.

Maxint-y
specifies the number of pools to dispense during each interval of the driver. If not specified, y defaults to 10.

Toss-x
specifies how many pools will not be returned for each interval when combined with Return and Maxint. When Toss is greater than or equal to Maxint, all of the pools will be tossed.

STATus
displays the status of the active instances.

All
displays the status of all the logged instances.

CLEAR
clears the status table.

ABORT x
aborts the x instance of the driver.

Help
displays the correct syntax of the command.

Source code information
The POOL and PO64 drivers consist of the following program segments:

Header Files
None.

BSOs

<table>
<thead>
<tr>
<th>Module</th>
<th>Makefile</th>
<th>Segment</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>QDMA</td>
<td>N/A</td>
<td>qdma.asm</td>
<td>Real Time (64-bit) segment that parses the input.</td>
</tr>
<tr>
<td>QDMB</td>
<td>N/A</td>
<td>qdmb.asm</td>
<td>Real Time (64-bit) segment that can run the system out of pools by cycling through each directory of each valid pool type.</td>
</tr>
<tr>
<td>Module</td>
<td>Makefile</td>
<td>Segment</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>----------</td>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>QDMC</td>
<td>N/A</td>
<td>qdmc.asm</td>
<td>Real Time (31-bit) segment that can run the system out of pools by cycling through each directory of each valid pool type.</td>
</tr>
<tr>
<td>QDME</td>
<td>N/A</td>
<td>qdme.asm</td>
<td>Real Time (31-bit) segment that parses the input.</td>
</tr>
</tbody>
</table>

**CSOs**

None.

**Additional information**

- The following is a table of record IDs used by the POOL driver in order to obtain various pools:

<table>
<thead>
<tr>
<th>Record</th>
<th>Type</th>
<th>Dup</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1, Q2, Q3</td>
<td>short term</td>
<td>non-dup</td>
<td>Small, Large, 4K</td>
</tr>
<tr>
<td>PC, PE, QC</td>
<td>long term</td>
<td>dup</td>
<td>Small, Large, 4K</td>
</tr>
<tr>
<td>QD, QE, QF</td>
<td>long term</td>
<td>dup</td>
<td>Small, Large, 4K (to be used on WP and modified to be non-dup)</td>
</tr>
<tr>
<td>QG</td>
<td>long term</td>
<td>dup</td>
<td>4K-FARF6</td>
</tr>
</tbody>
</table>

Note that because only the WP subsystem has non-duplicate long-term pools, in order to test long term non-duplicate pools, you must issue:

WP/ZRTDM MODIFY REC-QD,RTP0-SLPA  
WP/ZRTDM MODIFY REC-QE,RTP0-LLPA  
WP/ZRTDM MODIFY REC-QD,RTP0-4LPA

Then run the pool driver on WP to dispense those non-duplicate long-term pool records.

- **PDU and double release of pool records by POOL driver:**

**Question:** When I use the pool driver with the DOUBLE option to release a pool record twice, and if I run PDU right the way, PDU does NOT detect the double released pool. If the PDU is run later, it then detects that particular pool record was released twice. WHY?

**Answer:** When a pool is released, the released address is recorded in a core block. This core block is also known as the CA block because its record ID is CA. The core block can hold between 65-130 released addresses, depending on what pool options are turned on. The core block is not filed to DASD until it is full or when pools are cycled down.

When you run a PDU, it only looks at the records on DASD ignoring whatever is in core. For your test, the trick is to fill up the core block so it gets filed to DASD or cycle to 1052 state, for example:

1. Run a PDU in 1052 state. This will return anything that has been hanging around and will give you clean PDUs later.
2. Cycle to NORM.
3. Generate the double return.
4. Cycle to 1052 state or fill up the CA block by issuing the following:

   ZTEST POOL SDP 200 RETURN

   **Note:** If you want to fill up the CA block, use SDP or LDP. By using different pool types, you can differentiate the addresses from steps 3 and 4. PDU uses LDP pools internally, so you will always have some LDP pools returned during a PDU.

5. Run the PDU.

**Examples**

The following example will discard 5 pools of type 4ST.

```
ZTEST POOL 4ST 5
```

The following example will display the help menu.

```
ZTEST POOL HELP
```

The following example will display the status of all logged instances.

```
ZTEST POOL STATUS ALL
```

The following example will abort the second instance of the driver.

```
ZTEST POOL ABORT 2
```

**Messages**

Below is a list of the POOL and PO64 driver messages.

**Note:** The prefix for 64-bit driver messages will be PO64. Everything else remains the same between the two versions of the driver.

**ZTEST POOL HELP**

- POOL0001I ZTEST POOL TYPE COUNT (Return) (Toss-x) (Maxint-y)
- POOL0001I (Double) (Maxint-y)
- POOL0001I ZTEST POOL STATUS (ALL) (CLEAR)
- POOL0001I ZTEST POOL ABORT x

- POOL0002E Pool type invalid. Use: SST, SLT, SDP, LST, LLT, LDP, 4ST, 4LT, 4DT, 4D6
- POOL0003E Your subsystem is invalid
- POOL0004E No pools of that type exist in this SS
- POOL0005E You must specify a non-zero number
- POOL0006I Pool driver starting in 1 second
- POOL0007I Pool driver now ending.
- POOL0008E Serious Error encountered. Record size mismatch.
- POOL0009E Error creating DECB
- POOL0010E GRRTC Failed
- POOL0011E NODOUBLE and NORETURN are not valid parms.
- POOL0012F SYSHEAP failure
POOL0013E Unable to process command
POOL0014I POOL Driver Status Display
   Instance ..
   Parms: ....................................
   Dispensed: ....
   First Addr: .............
   Last Addr: .............
   Started: ....-..........  
   Stopped: .....-.........

POOL0015E No instances logged.
POOL0016E Status cleared.
POOL0017E Unable to clear. Instances still alive.
POOL0018E Abort instance requested.
POOL0019E Invalid instance entered.
POOL0020I Instance aborted.
POOL0021E Dispensed pool type different than requested.
   Enter the following message and try again
   ZRTDM MODIFY RECID-.... RTP0-....A

POOL0022I Unable to abort. Instance already stopped.
POOL0025I Unable to start. Status table full.

References
For more information about reading syntax diagrams, also referred to as railroad diagrams, see Accessibility information in the TPF Product Information Center.