A primary goal for CICS event processing support was that the overhead of detecting, capturing and emitting events should be minimal.

The performance data presented in these slides show that this goal has been achieved and continues to be achieved across the CICS releases in which CICS event support is available.
Important Disclaimer

THE INFORMATION CONTAINED IN THIS PRESENTATION IS PROVIDED FOR INFORMATIONAL PURPOSES ONLY.

WHilst EFForts WERE MADE TO VERIFY THE COMPLETENESS AND ACCURACY OF THE INFORMATION CONTAINED IN THIS PRESENTATION, IT IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED.

IN ADDITION, THIS INFORMATION IS BASED ON IBM’S CURRENT PRODUCT PLANS AND STRATEGY, WHICH ARE SUBJECT TO CHANGE BY IBM WITHOUT NOTICE.

IBM SHALL NOT BE RESPONSIBLE FOR ANY DAMAGES ARISING OUT OF THE USE OF, OR OTHERWISE RELATED TO, THIS PRESENTATION OR ANY OTHER DOCUMENTATION.

NOTHING CONTAINED IN THIS PRESENTATION IS INTENDED TO, OR SHALL HAVE THE EFFECT OF:

• CREATING ANY WARRANTY OR REPRESENTATION FROM IBM (OR ITS AFFILIATES OR ITS OR THEIR SUPPLIERS AND/OR LICENSORS); OR

• ALTERING THE TERMS AND CONDITIONS OF THE APPLICABLE LICENSE AGREEMENT GOVERNING THE USE OF IBM SOFTWARE.

Important Disclaimer – Notes

• Please take note of this important disclaimer.

• It is always possible that unintended errors have crept into this material.

• As always with performance data, your mileage may vary.
CICS EP Performance Topics

- **CICS TS V4.1 Performance Results**
  - Application events and EP adapters in detail

- **CICS TS V4.2 Performance Results**
  - Application events summary
  - System events

- **CICS TS V5.1 Performance Results**
  - EP adapter sets

CICS EP Performance Topics – Notes

- This material contains a summary of performance measurements for event processing support for CICS Transaction Server for z/OS.

- Event support was introduced in CICS TS V4.1, and comprehensive performance data was obtained, to demonstrate that the goal of minimal overhead due to events had been achieved. This data remains applicable for CICS event support on later releases.

- CICS TS V4.2 introduced enhancements to event support, and in particular support for system events. Further performance measurements were made, which also remain applicable to later releases.

- CICS TS V5.1 introduced further new capabilities, including support for EP Adapter sets. The performance data for V5.1 focuses on the EP adapter set results.
CICS TS V4.1 Performance Environment

- **Hardware**
  - Z10 2097-763 model E64
    - LPAR with 3 dedicated CPs
    - 2097 separate LPAR for Network drivers
    - Best fit for scaling CP speeds – LSPR 2097-703

- **Software**
  - z/OS 1.9, CICS TS V4.1

CICS TS V4.1 Performance Environment – Notes

- This describes the hardware and software environment used on CICS TS V4.1 to obtain these performance guidelines.
- TPNS was used to drive the test transactions.
- CPU numbers quoted in this presentation should be scaled to other machines using the Large Systems Performance Reference or LSPR (see [http://www-03.ibm.com/systems/z/advantages/management/lspr/](http://www-03.ibm.com/systems/z/advantages/management/lspr/)).
- Although the machine used was a 2097-763, because the tests were run on a dedicated 3 CP LPAR, the best fit for scaling is the relative ITR from a 2097-703.
### Event Capture Processing

<table>
<thead>
<tr>
<th>Event Proc On/Off</th>
<th>API in SPEC</th>
<th>pri-ary pred?</th>
<th>pri pred. match?</th>
<th>Other pred. match</th>
<th>data filter match</th>
<th>capture process</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>OFF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>negligible</td>
</tr>
<tr>
<td>ON NO</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>negligible</td>
</tr>
<tr>
<td>ON YES NO</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.110 microsecs</td>
</tr>
<tr>
<td>ON YES NO</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.180 microsecs</td>
</tr>
<tr>
<td>ON YES YES</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.225 microsecs</td>
</tr>
<tr>
<td>ON YES YES YES</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>+0.01 microsecs per predicate</td>
</tr>
<tr>
<td>ON YES YES YES NO</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>+0.01 microsecs per data filter</td>
</tr>
<tr>
<td>ON YES YES YES YES NO</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>+11 microsecs</td>
</tr>
</tbody>
</table>

### Event Capture processing – Notes

The values on this slide indicate the costs involved in determining whether a request matches the conditions for capturing an event. The processing involves various checks for a match with the capture specification, and as soon as a match fails then no further checking is carried out and no data is captured. The figures in the data show the cumulative cost involved in getting as far as each stage in the checking, so do not directly represent the cost of each type of check.

The notes on the table indicate:

1. If Event Processing is not started (OFF), then CICS takes no action. The cost of determining this is negligible.
2. If Event Processing is started (ON), there is a quick check to see if there is an Event specification covering this API. The cost is again negligible.
3. If the API is one of interest, CICS checks whether a primary predicate is specified (e.g. whether there is a predicate specifying the FILE on a WRITE command). If not, then checking will continue against any other predicates and data filters.
4. If the API is one of interest and a primary predicate is specified, CICS checks whether the primary predicate matches. If a full name is defined in the specification for the primary predicate (i.e. when filterOperator="EQ"), a fast lookup occurs. Otherwise, if the name is generic then the check is not optimized. NOTE that the data in the table does not relate directly to the cost of carrying out a primary predicate check, but includes extra processing that is incurred if the checks get as far as evaluating the primary predicate.
5. If the primary predicate matches, CICS checks any other predicates from either the application context or the application command options.
6. Finally, CICS checks if any data filters match, e.g. offset 8 for len 4 = ‘xxxx’
7. If all specified conditions match, CICS then captures the event itself. In these measurements, the event does not have any captured data other than the standard application context.
Event Dispatching

- Captured events dispatched under dispatcher tasks
  - Execute on open L8 TCBs
- Number of dispatcher tasks is varied depending on event volumes
  - Initially two CEPD dispatcher tasks started
  - New dispatcher tasks created if current dispatcher tasks are busy
- EP dispatcher tasks allocated from MAXOPENTCBS
  - To ensure event processing does not monopolise open TCBs, maximum TCBs used for EP dispatchers limited to one third MAXOPENTCBS

Event Dispatching – Notes

The dispatching of captured events has considerations in terms of TCB usage:

- CICS runs multiple EP dispatcher tasks which execute on L8 TCBs, in order to service the emitting of events. These dispatcher tasks count towards the MAXOPENTCBS limit.
- The number of dispatcher tasks will vary during the running of CICS.
- Initially two CEPD dispatcher tasks will be started, and these tasks will be considered as static in nature in that they will always be active.
- New dispatcher tasks will be created if events arrive to be dispatched and the current dispatcher tasks are busy.
- The newly started dispatcher tasks are dynamic in nature, and will terminate after a dynamic timeout value, following the end of their event workload.
- So that EP does not monopolise the L8 TCBs, the maximum number of L8 TCBs used for EP dispatchers is limited to one third of the MAXOPENTCBS value.
## EP Adapter Processing

<table>
<thead>
<tr>
<th>Capture Fields</th>
<th>EP Adapter</th>
<th>Event Format</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-n</td>
<td>TS</td>
<td></td>
<td>+62+(5*n) microsecs</td>
</tr>
<tr>
<td>1-n</td>
<td>CUST</td>
<td></td>
<td>+112+(5*n) microsecs</td>
</tr>
<tr>
<td>1-n</td>
<td>TRAN</td>
<td></td>
<td>+114+(27*n) microsecs</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(NOTE includes cost of started tran)</td>
<td></td>
</tr>
<tr>
<td>1-n</td>
<td>WMQ WBE</td>
<td></td>
<td>+97+(6*n) microsecs</td>
</tr>
<tr>
<td>1-n</td>
<td>WMQ CBE</td>
<td></td>
<td>+104+(6*n) microsecs</td>
</tr>
<tr>
<td>1-n</td>
<td>WMQ CFE</td>
<td></td>
<td>+93+(5*n) microsecs</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

### EP Adapter Processing – Notes

- This shows the performance considerations associated with the choice of EP adapter (which would of course be chosen based on the use to be made of the event, rather than on performance costs).
- These measurements were obtained prior to the introduction of the HTTP EP adapter, which is therefore not included.
- Column 1 indicates the number of data items being captured. There can be performance considerations associated with capturing a large number of individual items of data.
- Column 2 indicates the EP adapter used to format and emit the event.
- Column 3 indicates the event format used (only shown for the WMQ Queue EP adapter, which is the only EP adapter which offers a choice of format).
- Important Notes:
  - The cost per data item includes the cost of capturing the data when the event is captured, as well as the cost of emitting the data.
  - The TS queue EP adapter figures are based on using Main Temporary Storage.
  - The Transaction start EP adapter figures include not only the cost of the EP adapter itself, but also of a simple consumer program which reads all the containers passed in the CCE format event.
  - The custom EP adapter does not include any processing to emit the event, just that to read the containers passed on the EP adapter interface.
Event Processing example

This shows the event processing cost for a workload.

- 1 millisecond of CPU per user task
- A transaction was used which issues 100 API commands
  - including 1 WRITEQ TS command which was used to capture events, with the capture specifications varied to control when the event was captured.
  - Many instances of the transaction were run in the system.
- Captured events were emitted using the WebSphere MQ EP adapter with WBE format.
- Test T1: from 1 in 100 tasks, WRITEQ TS command events captured
- Test T2: from 1 in 10 tasks, WRITEQ TS command events captured
- Test T3: from 1 in 4 tasks, WRITEQ TS command events captured
- Test T4: from 1 in 2 tasks, WRITEQ TS command events captured
- Test T5: from 1 in 1 tasks, WRITEQ TS command events captured
  - Note that it would be very unlikely that you would want to capture an event from every single WRITEQ TS command, so T5 is very much an extreme case. T1 and perhaps T2 are much more representative of the typical situation.
CICS Events Performance
Updates for CICS TS V4.2 including system events

CICS TS V4.2 Event performance areas covered

- This section covers performance measurements for CICS TS V4.2
  - The cost of application events
    - Filtering costs
    - Capture and emission costs
    - CICS TS V4.2 comparison with CICS TS V4.1
  - Additional cost of assured event emission
    - New in CICS TS V4.2
  - The cost of system events
    - New in CICS TS V4.2
Environment used for V4.2 Performance measurements

- **Hardware**
  - z196 2817-779 model M80
  - LPAR with 5 dedicated CPs
  - Separate LPAR for network drivers
  - Best fit for scaling CP speeds – LSPR 2817-705
  - DASD DS8800
  - Internal Coupling Facility with ICP links

- **Software**
  - z/OS V1.12
  - CICS TS V4.1 (for comparisons)
  - CICS TS V4.2

CICS TS V4.2 Performance Environment – Notes

- This slide describes the hardware and software environment used on CICS TS V4.2 to obtain these performance guidelines.
- Any of the CPU numbers quoted in this presentation can be scaled to other machines using the LSPR at
### Application Event CPU Costs

<table>
<thead>
<tr>
<th>EP component</th>
<th>EP Adapter</th>
<th>CPU microsecs</th>
<th>CPU extra</th>
<th>comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>EP started</td>
<td>0</td>
<td></td>
<td></td>
<td>overhead of having EP started</td>
</tr>
<tr>
<td>Filter</td>
<td>+0.01</td>
<td></td>
<td></td>
<td>cost of a single data filter operation</td>
</tr>
<tr>
<td>Capture + Emit</td>
<td>WebSphere MQ</td>
<td>+68</td>
<td>+4 QMGR</td>
<td>WebSphere MQ cost with non-persistent queue</td>
</tr>
<tr>
<td></td>
<td>HTTP</td>
<td>+102</td>
<td>+6 TCPI</td>
<td>cross LPAR TCPI costs</td>
</tr>
<tr>
<td></td>
<td>Transaction Start</td>
<td>+79</td>
<td></td>
<td>transaction just reads passed containers</td>
</tr>
<tr>
<td></td>
<td>TS Queue</td>
<td>+46</td>
<td></td>
<td>main TS queue</td>
</tr>
<tr>
<td></td>
<td>TSQ (recov)</td>
<td>+116</td>
<td></td>
<td>recoverable aux TS queue</td>
</tr>
<tr>
<td>Custom</td>
<td>+51</td>
<td></td>
<td></td>
<td>no emission, just reads containers</td>
</tr>
</tbody>
</table>

**Example:**

Emitting to WebSphere MQ non-persistent queue with 100 data filters in capture specification:
- CICS CPU overhead $(100 \times 0.01) + 68 = 69$ microsecs
- WebSphere MQ CPU overhead = 4 microsecs

### Application Event CPU Costs – Notes

- “CPU microseconds” is the number of microseconds of CPU to run each EP component
- “CPU extra” is the additional microseconds of CPU consumed in other address spaces
- Note that the CICS Statistics reports “Total Event Filter operations” within “EVENTBINDING GLOBAL STATISTICS”:
  - This statistic is incremented by 1 for each set of data filters in an Event Capture Specification.
  - The cost of a single data filter operation is specified here.
  - Thus an Event Capture Specification containing 100 data filters will cost $100 \times 0.01$ microseconds, but the “Total Event Filter operations” statistic will be incremented by 1 (not 100).
- Adapters:
  - In these performance tests, the Transaction Start adapter task simply reads the passed containers and does no further processing.
  - Also, the Custom adapter runs a transaction that simply reads the passed containers, and does not emit the event. If using a Custom EP adapter, there is likely to be an additional emission cost.
Application Events measurements

- **Non-transactional, non-assured events measured**
  - other combinations covered in assured events section
- **Running Event Processing**
  - note that there is no significant CPU cost in having EP started
    - assuming no filtering or capturing takes place
- **Filtering (or predicate) costs**
  - 3 types of predicates:
    - Context predicates eg userid
    - Event options predicates eg filename of an FC WRITE API command
    - Application data predicates eg customer account number
  - including a primary predicate
    - a predicate with a filter operator of EQ
- **Capture and Emission costs with various adapters**

Application Events measurements – Notes

This slide contains some details about the application event measurements which follow.

- Event filtering is performed by defining predicates which specify the conditions under which an event is to be captured.
- If no predicates are specified, events are emitted every time the capture point you choose runs in the target CICS system.
- Each predicate consists of information identifying the predicate, an operator, and a value.
- You can specify as many predicates as you want to identify the situation in which you want this capture point to emit an event.
- The three types of predicates that you can specify are Context, Event Options, and Application Data.
- Some capture points have a primary predicate.
  - A primary predicate is a predicate for which you select Equals as the Operator setting and provide a value to avoid a performance impact as you add more capture specifications at a capture point.
Application Events primary predicate measurement workload

- **User task issues 1,000 FC READs to same file**
  - File referenced is NOT specified in any capture specifications

- **Predicates specified in performance test run**
  - 100 capture specifications with:
    - different FILE filter values
    - test #1 with Event Processing STOPPED
    - test #2 with filterOperator="EQ" (exact match – directory lookup – primary predicate)
    - test #3 with filterOperator="SW" (generic – sequential scan)

- **Predicate effects of each user task:**
  - 1,000 file predicate checks and no event capture
  - high number of FC calls and capture specifications per task
    - to show CPU overhead differences

---

Application Events primary predicate measurement – Notes

These notes describe the workload used for the measurements to investigate the significance of using the primary predicate.

- The file being referenced by the application was not specified in any capture specifications so that:
  1. a full scan occurs, and
  2. the overhead of capturing and emitting the event is not included in the CPU overhead (i.e. there was never a successful match when the event processing capture specifications enabled in the system were checked)

- SW stands for “starts with”
Application Events predicate workload - CPU usage

This slide shows the results of the measurements with primary predicates:

- **test #1: no filtering** – the workload was run with CICS event processing stopped
- **test #2: 100 “EQ”** – the workload was run with 100 non-matching FILE READs defined with a filterOperator of “EQ”
- **test #3: 100 “SW”** – the workload was run with 100 non-matching FILE READs defined with the filterOperator=“SW”

These results show that running with an equal match on a primary predicate gives very little overhead compared with having no event processing active at all, but there is more overhead associated with less specific tests.
Application Events filtering measurement workload

- **User task issues 1,000 FC READs to same file**
  - matches the file primary predicate
- **Event filtering set with 100 application data filters for this file**
  - First 99 data filters match, 100th data filter does not match
    - this avoids the event capture and emission process
  - test #1 with Event Processing STOPPED
  - test #2 with 1000*100 filters per task
- **Filtering effects of each user task:**
  - 1,000*100 data filter checks and no event capture
  - heavy filtering used to show CPU overheads

Application Events filtering measurement workload – Notes

This slide describes the workload used to measure the cost of filtering.
Application Events Filtering - CPU usage (exaggerated)

This slide shows the results of the filtering measurements:

- **test #1**: no filtering – workload with EP stopped
- **test #2**: 1000*100 filters – workload with 100 non-matching FILE READs defined with filterOperator="EQ", where the first 99 filters match but the 100th filter does not match.

This shows that, while there is some overhead involved in processing such a large number of filters, this scales linearly and not very steeply with growing numbers of transactions to process.
Application Events adapter measurements

- **EP adapters:**
  - WebSphere MQ (CBE, CFE, WBE message formats)
  - HTTP (CBER, CBE, WBE message formats)
  - Temporary Storage (CBER, CBE, WBE message formats)
  - Transaction Start (CCE message format)
  - Custom

- **Event format**
  - The event message format used has no significant effect on performance

---

Application Events adapter measurements – Notes

Measurements were obtained to investigate the performance implications of different EP adapters and event formats. As of CICS TS V4.2, more event formats are supported by some of the EP adapters:

- **CBE** - Common Base Event format, for consumption by products such as IBM Business Monitor
- **CBER** – Common Base Event REST format, for the IBM Business Monitor REST HTTP server
- **CCE** - CICS Channel-based Event format
- **CFE** – CICS Flattened Event format
- **WBE** – WebSphere Business Events XML format, used by the event engine of IBM Operational Decision Manager
Application Event CPU costs

<table>
<thead>
<tr>
<th>EP component</th>
<th>Adapter</th>
<th>CPU micro</th>
<th>CPU extra</th>
<th>comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>EP started</td>
<td></td>
<td>0</td>
<td></td>
<td>overhead of having EP started</td>
</tr>
<tr>
<td>Filter</td>
<td></td>
<td>+0.01</td>
<td></td>
<td>cost of a single data filter operation</td>
</tr>
<tr>
<td>Capture + Emit</td>
<td>MQ</td>
<td>+68</td>
<td>+4 QMGR</td>
<td>MQ cost with non-persistent queue</td>
</tr>
<tr>
<td>HTTP</td>
<td></td>
<td>+102</td>
<td>+6 TCP/IP</td>
<td>cross LPAR TCP/IP costs</td>
</tr>
<tr>
<td>Trans Start</td>
<td></td>
<td>+79</td>
<td></td>
<td>trans just reads passed containers</td>
</tr>
<tr>
<td>TSQ</td>
<td></td>
<td>+46</td>
<td></td>
<td>main TSQ</td>
</tr>
<tr>
<td>TSQ recov</td>
<td></td>
<td>+116</td>
<td></td>
<td>recoverable aux TSQ</td>
</tr>
<tr>
<td>Custom</td>
<td></td>
<td>+51</td>
<td></td>
<td>no emission, just reads containers</td>
</tr>
</tbody>
</table>

Example:
Emitting to MQ non-persistent queue with 100 data filters:
CICS CPU overhead (100\*0.01)+68=69 microsecs
MQ CPU overhead=4 microsecs

Application Event CPU costs – Notes

- “CPU micro” is the number of microseconds of CPU consumed in each component within CICS EP support
- “CPU extra” is the additional microseconds of CPU consumed in other address spaces
- Note that the CICS Statistics reports “Total Event Filter operations” within “EVENTBINDING GLOBAL STATISTICS”.
- This statistic is incremented by 1 for each set of dataFilters in an Event Capture Specification.
- The cost of a single data filter operation is specified here.
- Thus an Event Capture Specification containing 100 dataFilters will cost 100\*0.01 microseconds, but the “Total Event Filter operations” statistic will be incremented by 1 (not 100).

Adapters:
- In these performance tests, the Transaction Start adapter task simply reads the passed containers and does no further processing.
- Also, the Custom adapter runs a transaction that simply reads the passed containers, and does not emit the event.
- So if using the Custom adapter, there will be an additional emission cost.
Application Events summary

- **Filtering costs**
  - Generally low CPU overhead
    - especially if the EQ filter operator is used

- **Event Capture and Emission costs**
  - 46 to 116 microseconds of CPU to capture and emit an event
    - depending on EP adapter used

- **CICS TS V4.2 comparison with CICS TS V4.1**
  - Identical Event Processing performance tests ran for 4.1 and 4.2
    - No CPU increase between 4.1 and 4.2

Application Events summary – Notes

This slide summarises the findings from the performance measurements of Application Events in CICS TS V4.2
Assured Event emission

- **Assured Events**
  - Used to assure that certain Application Events are emitted
  - Synchronous processing of event results in assured emission
    - Emission performed within the UOW of the capturing transaction
  - when used with WebSphere MQ:
    - event delivery can be assured as well
      - by defining MQ queue as PERSISTENT

Assured and Transactional event emission

- **Synchronous events**
  - captured and emitted within the user task associated with the event capture
  - thus the event emission process is assured

- **Asynchronous events**
  - captured and emitted within a thread of an EP dispatcher task, or a separately attached user task
  - the user task may complete normally, but the event emission process may fail
  - the event emission process is not assured

- **Transactional events**
  - capturing and emission is delayed until a syncpoint is issued (implicitly or explicitly)
  - thus the event is not emitted if the UOW rolls back

- **Non-transactional events**
  - capturing and emission occurs immediately
  - the event is emitted even if the user task subsequently fails
Assured Application Events workloads

- **WebSphere MQ EP adapter used for assured event workloads**
  - as MQ adapter supports all combinations
    - Synchronous, Transactional
    - Synchronous, Non-transactional
    - Asynchronous, Transactional
    - Asynchronous, Non-transactional
- **user task issues a single PUT CONTAINER**
- **all PUT CONTAINER events captured**
- **no filtering**

Assured Application Events workloads – Notes

This slide summarizes the workloads used to measure the performance for assured event emission. (Assured emission applies only to application events, and not to system events.)
Assured Application Events (MQ) – additional CPU costs

<table>
<thead>
<tr>
<th>Adapter</th>
<th>Sync/Async</th>
<th>Transactional</th>
<th>Persistent Q</th>
<th>CICS CPU</th>
<th>MQ CPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>MQ CBE</td>
<td>Async</td>
<td>No</td>
<td>No</td>
<td>68</td>
<td>4</td>
</tr>
<tr>
<td>MQ CBE</td>
<td>Async</td>
<td>Yes</td>
<td>No</td>
<td>+4</td>
<td>+0</td>
</tr>
<tr>
<td>MQ CBE</td>
<td>Sync</td>
<td>No</td>
<td>No</td>
<td>+24</td>
<td>+18</td>
</tr>
<tr>
<td>MQ CBE</td>
<td>Sync</td>
<td>Yes</td>
<td>No</td>
<td>+30</td>
<td>+19</td>
</tr>
<tr>
<td>MQ CBE</td>
<td>Async</td>
<td>No</td>
<td>Yes</td>
<td>+21</td>
<td>+24</td>
</tr>
<tr>
<td>MQ CBE</td>
<td>Async</td>
<td>Yes</td>
<td>Yes</td>
<td>+25</td>
<td>+24</td>
</tr>
<tr>
<td>MQ CBE</td>
<td>Sync</td>
<td>No</td>
<td>Yes</td>
<td>+41</td>
<td>+43</td>
</tr>
<tr>
<td>MQ CBE</td>
<td>Sync</td>
<td>Yes</td>
<td>Yes</td>
<td>+43</td>
<td>+38</td>
</tr>
</tbody>
</table>

Additional CPU:

Synchronous processing results in an MQ Commit call at user task syncpoint time

Additional CPU:

MQPUT1 call for persistent queues consumes more CPU than non-persistent queues

Notes:

- This slide shows the additional CPU costs which are incurred when using synchronous (assured) event emission. All times are in microseconds.
- These measurements were obtained using the WebSphere MQ EP adapter. When synchronous event emission is used, the user task syncpoint results in MQ commit processing, which does not occur with asynchronous emission.
- This results in an additional CPU overhead in both CICS and MQ address spaces.
- There is also an additional CPU overhead in both CICS and MQ address spaces for MQ persistent queues, due to increased CPU on an MQPUT1 call.
- This shows that there are additional overheads involved, over and above the cost of synchronous event emission, when using the WebSphere MQ EP adapter to emit events.
Assured Events summary

- **WebSphere MQ EP adapter**
  - most common adapter used with assured events
  - additional CPU costs for synchronous events
  - plus more CPU with persistent MQ queues
  - but overheads relatively low
  - if user task running on the QR TCB and event is synchronous:
    - TCB switches will cause an additional CPU overhead
    - switches to and from the L8 TCB to issue the MQPUT1 and MQCOMMIT calls

Assured Events Summary – Notes

This summarizes the findings for assured events, using WebSphere MQ to emit the events.
System Events

- **System Events supported**
  - MAXTASKS threshold
  - TRANCLASS TASK threshold
  - Relatively rare System Events:
    - DB2 connection status change
    - File enable/disable status change
    - File open/close status change
    - Unhandled transaction abend

System Events – Notes

- DB2 connection status changes, file status changes and transaction abends should be rare events.
- Monitoring and capturing these rare events will therefore not adversely affect overall system performance.
MAXTASKS System Events workload measurements

- **Max tasks**
  - No overhead in having System Events enabled
  - Test #1 MAXTASKS=10 and EP stopped
  - Test #2 MAXTASKS=10 with MAXTASKS System Events enabled
    - event captured when current # tasks goes higher than 60% of MAXTASKS
    - this test generated a 2.8% capture rate
    - Transaction Start EP adapter used
      - this overhead has been included in the CPU costs
  - CICS Statistics Event Filter operations
    - Statistic increased when any MXT threshold is crossed (up or down)
    - Threshold values can be set to:
      - 50%, 60%, 70%, 80%, 90% or 100%

MAXTASKS System Events workload measurements – Notes

- Performance measurements were obtained for the “Max Tasks” system event, which enables events to be emitted when the number of active tasks in a region goes above or below a certain percentage of the MXT (max tasks) limit.
- The performance measurements were obtained using an event specified for 60% of max tasks
- Measurements were made with:
  1. MAXTASKS set to 10 and EP stopped (so no events captured)
  2. MAXTASKS set to 10 and EP enabled (resulting in events captured for 2.8% of user tasks)
- These figures do not give the cost of a single emission, but show the general overheads
With MAXTASKS=10, approximately 2.8% of user tasks resulted in a System Event being emitted.

- The CPU cost includes emitting the event using a Transaction Start EP adapter.

- The CPU costs are compared with MAXTASKS=10 and Event Processing stopped.
TRANCLASS System Events workload measurements

- **Transaction Class**
  - No overhead in having System Events enabled
  - Test #1 TRANCLASS=10 and EP stopped
  - Test #2 TRANCLASS=10 with TRANCLASS System Events enabled
    - Event captured when current # tasks in class goes higher than 60% of TRANCLASS
    - this test generated a 2% capture rate
    - Transaction Start EP adapter used
      - this overhead has been included in the CPU costs
  - CICS Statistics Event Filter operations
    - Statistic increased when any Transaction Class threshold is crossed (up or down)
    - Threshold values can be set to:
      - 50%, 60%, 70%, 80%, 90% or 100%

TRANCLASS System Events workload measurements – Notes

Transaction Class
- Run with 60% threshold set in the capture specification, and with:
  1. TRANCLASS set to 10 and EP stopped
  2. TRANCLASS set to 10, so events captured for 2% of user tasks
- Does not show the cost of a single emission, but general overheads
System Events – Transaction Class CPU usage

- With TRANCLASS=10 for a transaction, approximately 2% of user tasks resulted in a System Event being emitted.
- The CPU cost includes emitting the event using a Transaction Start adapter.
- The CPU costs are compared with TRANCLASS=10 and Event Processing stopped.
System Events summary

- **MAXTASKS and TRANCLASS performance measurements**
  - MAXTASKS with a 2.8% capture rate:
    - results in an overall 3 microsecond increase in CPU per user task
      - including the emission cost of a Transaction Start adapter
  - TRANCLASS with a 2% capture rate:
    - results in an overall 2 microsecond increase in CPU per user task
      - including the emission cost of a Transaction Start adapter

- **System Events emission**
  - No detectable CPU increase when System events enabled and no emissions
  - During normal operation number of emissions should be low
  - Event Filtering and capturing CPU costs low
  - System overheads should thus be negligible

System Events summary – Notes

- Note that the workloads were designed to generate a relatively high percentage capture rate.
- If these capture rates were detected in a production environment, then the MAXTASKS/TRANCLASS limits should be reviewed.
CICS Events Performance
Updates for CICS TS V5.1
– EP adapter sets

CICS TS V5.1 Updates – Notes

The previous measurements remain valid for V5.1.
New in V5.1 are EP adapter sets, for which performance measurements were obtained.
Environment for V5.1 Measurements

- **Hardware**
  - z196 2817-779 model M80
  - LPAR with up to 16 dedicated CPs
    - See each benchmark description for number of CPs actually used
  - Separate LPAR with 4 dedicated CPs for Network driver
  - DASD DS8800
  - Internal Coupling Facility with ICP links

- **Software**
  - z/OS 1.13
  - CICS TS V4.2
  - CICS TS V5.1

Environment for V5.1 Measurements – Notes

- Any of the CPU numbers quoted in this section can be scaled to other machines using the LSPR at
Event Processing Adapter Sets

- **EP Adapter Sets**
  - Prior to CICS TS V5.1:
    - events specified in an Event Binding could only be emitted by one EP adapter
    - EP adapter definition could be:
      - embedded in Event Binding or
      - defined as a separate EP adapter resource and referenced in the Event Binding
    - Emitting events to multiple EP adapters required multiple identical Event Bindings
  - New to CICS TS V5.1:
    - events specified in an event binding can be emitted by multiple EP adapters
      - Events specified in an event binding reference an EP adapter set
      - EP adapter set contains a list of EP adapter resources
      - Each EP adapter emits the same event (in the format specified for that EP adapter)

Event Processing Adapter Sets – Notes

- Prior to CICS TS V5.1, to emit an event to multiple EP adapters (for example, write to an MQ queue and also start a CICS transaction), required multiple identical Event Bindings specifying different EP adapters.
- Thus multiple events are captured when, for example, a CICS API command is issued, with each event resulting in an event emission.
- Now in CICS TS V5.1, to emit an event to multiple EP adapters can be achieved with a single Event Binding referencing an EP adapter set.
- The EP adapter set contains a list of EP adapters, each of which emit the event.
- Thus only one event is captured when, for example, a CICS API command is issued.
CICS Event Processing V5.1 testing

- Event processing without adapter sets installed
  - there is no regression in CPU/tran or response times
    - comparing CICS TS V4.2 with CICS TS V5.1

- Event processing with adapter sets installed
  - CICS TS V5.1 performance test comparing:
    - EP adapter set containing one adapter with
    - EP adapter not contained within an EP adapter set
    - CPU/tran and response times identical

CICS Event Processing V5.1 Testing – Notes

Event processing testing for CICS TS V5.1
- This consisted of running a workload that emitted one non-transactional, asynchronous event per CICS transaction.
- The event was emitted to a WMQ queue.
- The CPU per transaction and response times were measured for the workload running on CICS TS 4.2 and CICS TS V5.1.

EP Adapter Set testing:
- This consisted of running a workload that emitted one non-transactional, asynchronous event per CICS transaction.
- The event was emitted to a WMQ queue.
- The CPU per transaction and response times were measured for the workload running on CICS TS V5.1 with:
  a) an EP adapter set containing one EP adapter
  b) an EP adapter installed that's NOT part of an adapter set
Event Processing Adapter Set performance comparison

- **Comparing:**
  - Test #1: One Event Binding referencing:
    - Adapter Set containing 3 WMQ adapters
    - Single event captured
  - Test #2: 3 Event Bindings each referencing:
    - a single WMQ adapter
    - 3 events captured

EP Adapter Set performance benefits - CPU usage
CICS TS V5.1 Summary

- **No performance regression with CICS TS V5.1**
- **Multiple emissions per captured event**
  - with adapter sets vs. without adapter sets
    - similar performance characteristics

Thank You!