Advanced Transportation Controller (ATC) Standard

Comment Disposition

July 14, 2011

This is a Working Group Draft which is distributed for review and comment purposes only. You may reproduce and distribute this document within your organization, but only for the purposes of and only to the extent necessary to facilitate review and comment to AASHTO, ITE, or NEMA. Please ensure that all copies reproduced or distributed bear this legend. This document contains recommended information which is subject to approval.

Published by

American Association of State Highway and Transportation Officials (AASHTO)
444 North Capitol St. NW, Suite 249
Washington, DC 20001

Institute of Transportation Engineers (ITE)
1627 I(Eye) Street, NW, Suite 600
Washington, DC 20006

National Electrical Manufacturers Association (NEMA)
1300 N. 17th St., Suite 1752
Rosslyn, VA 22209-3806

© Copyright 2006-2011 AASHTO/ITE/NEMA. All rights reserved.
NOTICE
Joint AASHTO, ITE and NEMA Copyright
and
Advanced Transportation Controller (ATC) Working Group

These materials are delivered "AS IS" without any warranties as to their use or performance.

AASHTO/ITE/NEMA AND THEIR SUPPLIERS DO NOT WARRANT THE PERFORMANCE OR RESULTS YOU MAY OBTAIN BY USING THESE MATERIALS. AASHTO/ITE/NEMA AND THEIR SUPPLIERS MAKE NO WARRANTIES, EXPRESSED OR IMPLIED, AS TO NON-INFRINGEMENT OF THIRD PARTY RIGHTS, MERCHANTABILITY, OR FITNESS FOR ANY PARTICULAR PURPOSE. IN NO EVENT WILL AASHTO, ITE, OR NEMA OR THEIR SUPPLIERS BE LIABLE TO YOU OR ANY THIRD PART FOR ANY CLAIM OR FOR ANY CONSEQUENTIAL, INCIDENTAL, OR SPECIAL DAMAGES, INCLUDING ANY LOST PROFITS OR LOST SAVINGS, ARISING FROM YOUR REPRODUCTION OR USE OF THESE MATERIALS, EVEN IF AN AASHTO, ITE, OR NEMA REPRESENTATIVE HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Some states or jurisdictions do not allow the exclusion or limitation of incidental, consequential, or special damages, or exclusion of implied warranties, so the above limitations may not apply to you.

Use of these materials do not constitute an endorsement or affiliation by or between AASHTO, ITE, or NEMA and you, your company, or your products and services.

If you are not willing to accept the foregoing restrictions, you should immediately return these materials.

ATC is a trademark of AASHTO/ITE/NEMA.
The purpose of this document is to summarize comments (and their disposition) received from all interested parties regarding the ATC Controller Standard Version 5.2b.

Inquiries, additional comments and proposed or recommended revisions should be submitted to:

Standards Development Engineer
Institute of Transportation Engineers
1099 14th Street, NW, Suite 300 West
Washington, DC  20005-3438

voice: 202-289-0222
fax: 202-289-7722
ID#: 1-1

Commenter: Peter Ragsdale

Comment Type: Technical

Location: Section 2.2.5, Page 2-4
Section 4.3.3, Page 4-8

Clause: (n/a)

Issues and Proposed Revision

For compatibility purposes, we need to define a minimum level of GCC also … suggest 4.1.1 or 4.1.2 as a minimum (i.e. add requirement for 4.1.x or higher).

Disposition

Retitle Section 2.2.5, “Operating System Support and Development Requirements”:

Add to Section 2.2.5, following the list of BSP components:

Each engine board manufacturer shall provide all development tools (cross-compiler, linker, libraries, and header files) necessary to completely rebuild application programs for execution on their engine board. The development environment must operate on an x86-based platform and shall be available in both source and object format for both download from the manufacturer’s website in an unrestricted area and by physical media if requested. These tools shall be available for a period of not less than five (5) years after the last date of delivery of the associated engine board.
Advanced Transportation Controller (ATC) Standard

COMMENT DISPOSITION

ID#: 1-2
Commenter: Peter Ragsdale
Comment Type: Technical
Location: Section 2.2.5, Page 2-5
Clause: "A Linux console shall be provided on /dev/tty4 (/dev/console) at boot-up. All terminal outputs during boot time shall be made to that interface. All inputs are made via a terminal program to that interface. This interface is the default stdin / stdout of Linux. The communication parameters are initially 38400 baud, n, 8, 1. The same communication parameters shall be used by the boot-loader in order to ensure a continuous output to the serial terminal / console. After booting and all applications are loaded, /dev/tty4 shall be available to applications as /dev/sp4 unless the single-user mode Linux feature has been invoked via Ctrl-C during the boot process before control is passed to any application programs."

Issues and Proposed Revision

Console port baud rate during boot-up ... suggest increasing it to 115,200 baud which is typical for Linux and boot-loaders.

• This has boot time implications (e.g. removed 700 ms to the boot time on one engine board design)
• Please note that the current baud rate is not 2070 compatible anyways
• Do we also want to define the minimum amount of information that should come out of the port during boot?
• Please note that experience has shown that any information out of the port has potential negative impacts due to connected devices

Additionally, this baud rate is only applicable at boot-up ... the default is not applicable once control is passed to the application(s).

Disposition

Modify Section 2.2.5, Page 2-5, (in bold):

Notes:
1. A Linux console shall be available on /dev/tty4 (SP4) (/dev/console) at boot-up. All terminal outputs during boot time shall be made to that interface. All inputs shall be made via a terminal program to that interface. This interface is the default stdin / stdout of Linux. The communication parameters shall be initially set to 115.2 kbaud, n, 8, 1. These same communication parameters shall be used by the boot-loader in order to ensure a continuous output to the serial terminal / console. After booting and all applications are loaded, /dev/tty4 shall be available to applications as /dev/sp4 unless the single-user mode Linux feature has been invoked via Ctrl-C during the boot process before control is passed to any application program(s).
As external devices may be connected to SP4 at boot-up, a means of disabling and enabling the Linux console at boot-up shall be provided by one of the following two options:

**Option 1**
By default the Linux console and boot loader on `/dev/tty4` shall be disabled. Enabling of the console shall be invoked by pressing and holding the ESC key for at least 500ms following power up.

**Option 2**
By default the Linux console and boot loader on `/dev/tty4` shall be enabled, however SP4 shall be disabled by the host board during boot-up. This may be software controlled by the host board micro (independent of the Engine Board) or by a hardware jumper located on the host board. It is not intended that the hardware jumper be available to the user. If SP4 is under host board control, a means of enabling / disabling SP4 shall be provided that can be read by the host board micro on boot-up to determine whether to enable or disable SP4.
ID#: 1-3
Commenter: Dave Miller
Comment Type: Technical
Location: Section 2.2.5
Clause: (uCLibC timers - librt)

**Issues and Proposed Revision**

Wording should show “uCLibC or equivalent”.

In an effort to encourage standardization every attempt should be made to use existing standards as defined in Unix/Linux and not invent one-off versions of nearly the same thing.

Require uClibc 0.9.29, which includes librt.

**Disposition**

*Modify Section 2.2.5, Item 3:*

Increase uClibc minimum version to 0.9.29. Allow uClibc 'or equivalent'.

Add librt.so to list of required uClibc programs.

Add hrtimers package to table (new entry), per POSIX 1003.1b, with better than 100us resolution.
ID#: 1-4
Commenter: Dave Miller
Comment Type: Technical
Location: Section 2.2.5, Page 2-6
Clause: (Busybox programs)

Issues and Proposed Revision
Vendor shall verify that all utilities specified are provided and work correctly.

Disposition
Modify Section 2.2.5, Item 3:
Increase Busybox minimum version to 1.18.4.
Add ftp and ftpd to list of required Busybox programs.
ID#: 1-5

Commenter: Dave Miller

Comment Type: Technical

Location: Section 2.5.5

Clause: (reset)

**Issues and Proposed Revision**

There is no requirement in the ATC 5.2b for application code needing to call atc_cpu_reset() after a reboot. These devices do not reset instantaneously so some time must elapse from the call of this function and subsequent operations. The optimum place for this call is during the reboot period. Reset is required in Paragraph 2.2.5 of the ATC 5.2b standard as part of busybox. Processor peripherals should come up from reset in a known state otherwise boot time will be affected while a cpureset is being exercised. Boot time is not the right place for the application software to be timing out on a non-responsive front panel or fio.

During bootup, the CPURESET pin should be asserted and unasserted, in order to set the front panel and field I/O to a known state. Assertion time requirements shall match that of the 2070 ATC.

**Disposition**

*Modify Section 2.2.5, Item 3. Add “(Note 2)” to Function table at CPU Reset.*

Notes:

2. The CPURESET signal shall be asserted during the boot process with a minimum pulse width of 125ms and shall be deasserted no later than 250ms from the beginning of the boot process. The CPURESET signal shall not be activated outside of application program control after this 250ms period.
ID#: 1-6
Commenter: Peter Ragsdale
Comment Type: Technical
Location: Section 3.4, Page 3-7
Clause: “As a minimum, the ATC must provide the necessary interfaces to support the ITS Cabinet standard.”

Issues and Proposed Revision

An ATC should not be required to support the ITS Cabinet if used in a non-compatible cabinet such as the 332 or NEMA or hybrid (proprietary) … this is a contradiction and the port connectors & usage would be in conflict.

Disposition

Remove Section 3.4, last paragraph, Page 3-7.

As a minimum, the ATC must provide the necessary interfaces to support the ITS Cabinet standard. Additionally, the ATC should provide optional interface support for common legacy cabinets including Model 332, NEMA TS1 and NEMA TS2 types.
ID#: 1-7

Commenter: Peter Ragsdale

Comment Type: Technical

Location: Figure 3-4, Page 3-9
          Section 3.5.1.1, Page 3-9

Clause: “This feature allows the local operator or a remote computer to install or
        update the application software resident on the ATC.”

Issues and Proposed Revision

For safety reasons, I suggest that only a local operator is allowed to install or update the
application software resident on the ATC.

Disposition

Disposition deferred. Future revision of the standard may provide guidance (AEI) regarding
implications of remote access and control.
ID#: 1-8

Commenter: Peter Ragsdale

Comment Type: Technical

Location: Figure 3-4, Page 3-9
          Section 3.5.1.2, Page 3-10

Clause: “This feature allows the local operator to install or update the O/S resident on the ATC. Local upgrade capability is required while remote upgrade capability is considered an optional feature.”

Issues and Proposed Revision

For safety reasons, I suggest that only a local operator is allowed to install or update the O/S resident on the ATC.

Disposition

Disposition deferred. Future revision of the standard may provide guidance (AEI) regarding implications of remote access and control.
ID#: 1-9
Commenter: Peter Ragsdale
Comment Type: Technical
Location: Figure 3-4, Page 3-9
Section 3.5.1.7, Page 3-10
Clause: “This feature allows the operator to manage the starting, stopping and scheduling of one or more applications on the ATC.”

Issues and Proposed Revision

For safety reasons, I suggest that only a local operator is allowed to manage the starting, stopping and scheduling of applications on the ATC.

Disposition

Disposition deferred. Future revision of the standard may provide guidance (AEI) regarding implications of remote access and control.
ID#: 1-10

Commenter: Peter Ragsdale

Comment Type: Editorial

Location: Section 4.1.3, Page 4-2, Line 23

Clause: "degreed"

Issues and Proposed Revision

(spelling error – change to “degrees”)

Disposition

Disposed as proposed.
ID#: 1-11  
Commenter: Peter Ragsdale  
Comment Type: Technical  
Location: Section 4.1.3, Page 4-2  
Clause: "Once each hour, the BSP RTC driver shall automatically copy the current OST time to the RTC with an accuracy of 0.1 seconds."

Issues and Proposed Revision

Suggest that the RTC synchronization with the OST should be more frequent and/or that testing of repetitive power on/off should be conducted (e.g. 5 minutes on, 3 minutes off over a period of 8 hours with no drift). Can also specify no drift over a period of approximately 2.5 minutes of no power.

Disposition

*Add following text to end of Section 2.2.5:*

The C library shall use the /etc/localtime file to indicate the local system timezone. The /etc/localtime file shall be in the TZif2 format. The /etc/localtime file may be changed at any time; the C library shall always use the current file.

Resolution of the POSIX timers shall be 100µs or better.

*Add following text to Section 4.1.3:*

When the tod_set_timesrc is set to TOD_TIMESRC_LINESYNC, all timing functions and time keeping functions shall be referenced to the AC power line and shall drift with the AC power line and shall not attempt to track to a crystal or WWV (or equivalent) reference while AC power is available.

The controller time keeping functions, including but not limited to the time-of-day clock, shall not be adversely affected by transients, brown-out conditions, dropouts, and noise on the AC line; the controller shall include internal clock management circuitry such that short power outages (< 2.7 minutes @ 25C, <1.3 minutes over full temperature range) do not introduce any errors or drift into the controller’s clocks. The controller’s clock shall not deviate by more than +/- 8.2 ms from the AC line reference; this deviation shall not accumulate*. Outages longer than 2.7 minutes shall not cause the clock to drift by more than +/-0.005% of the duration of the outage over the full temperature range.

The controller clock may be set from an external source using any of the available communications ports or through the front panel keypad and display. When the clock is set, all fractional portions of the real time clock shall be cleared.
Informative: It is known that tracking the AC power line as a reference will cause the clock to "wander" by several seconds depending on the loading of the power grid. However, the time reference does not drift – i.e. there is no cumulative time gain or loss; the time reference simply "wanders" as the power grid control compensates by very small adjustments to the line frequency such that there is no continuing time drift. The difference between a NIST referenced time and a line frequency referenced time has been observed to wander as much as + and – six (6) to eight (8) seconds or more. The exact amount of the offset can be determined by contacting the regional power grid operator.

Authorized Engineering Information: For purposes of testing, the agency/testing lab needs to take into consideration possible deviations in the line frequency (power grid) and the accuracy of the test equipment used to measure time deviations.

* Authorized Engineering Information: As used here, “not accumulate” means that the controller time shall be within the required deviation of +/- 8.2 ms when referenced to the AC powerline regardless of the number of short power outages that occur. During power outages, the AC line frequency is assumed to be 60.000Hz.

Create new subsection, 9.10.4 Extended AC Line Sync Verification, as follows:

Verification to AC Line Sync for power outages of less than 2.7 minutes

1. The purpose of this test is to verify that time drift between ATC’s does not exceed requirements while running on the same AC source and for power outages of less than 2.7 minutes.
2. This test requires two or more ATC’s connected to the same AC power source.
3. Each ATC shall be configured with the same signal and timing plans and shall be running in a fully function cabinet.
4. This test requires that all units under test are set to the same time.
5. Time will be verified at different stages of testing. These stages, in no particular order are:
   a. During / After power interruption testing
   b. On start up
   c. During / After transient testing
   d. During / After 8 hour continuous operation
   e. After frequency sweep test
   f. During / After varying line voltage test
6. The tests defined above are considered as one continuous test, once testing has started the ATC’s times shall not be changed until completion of tests 5a through 5f.
7. To verify the time during each test, a chart recorder shall be used to monitor each ATC’s main street green output (green output 1). the chart recorder print out will be checked to ensure that the main street green of the ATC’s under test are synchronized. For the purpose of this test
   a. Main street green shall be monitored at the AC output of the load switch
   b. Synchronization shall mean that each ATC main street green shall go from the OFF to the ON state and the ON to the OFF state at the same time.
   c. To verify that each ATC’s main street green output is synchronized with each other, the waveforms recorded by the chart recorder shall be checked to ensure that the green on and off times occur within +/- one AC line cycle (non-accumulating over multiple power interruptions of less than 2.7 minutes) at the same zero line crossing for each ATC under test.
Power outages of more than 2.7 minutes

1. This test verifies that the controllers under test do not time drift by more than 0.005%. It shall also verify the operation of the time back up circuitry.
2. This test requires that all units under test are set to the same time as a WWV reference clock.
3. With time synchronized, record the current time of each ATC and reference clock. The power to each ATC is then removed for 7 days. At the end of the 7 day period switch the units back on and record the time of each ATC
4. Verify that the time drift between the ATC and reference is less then 0.005%.

Create new subsection, B.7 Time of Day Driver, as follows:

B.7 Time of Day Driver

The following standard Linux functions shall be supported by this driver:

- open()
- close()
- read()
- write()
- ioctl()

The special device file node for the time of day device shall be "/dev/tod".

open()

Allows an application to request shared access to the time of day device. The device supports the O_RDONLY, O_WRONLY, and O_RDWR open flags.

close()

Closes the time of day device. Closing the device cancels any requested signals.

read()

Read operations are ignored. All reads shall return 0.

write()

Write operations are ignored. All writes shall return 0.

ioctl()

Allows an application to configure and control the time of day device. This function shall only be called after a successful open of the respective device. The operation performed by the ioctl() function depends the command argument. The command argument determines the interpretation of any additional arguments. The supported IOCTL services are defined below.

Prototype

    int ioctl(int fd, int command, parameters);
### Advanced Transportation Controller (ATC) Standard

#### COMMENT DISPOSITION

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fd</td>
<td>This is the file descriptor of the device to access.</td>
</tr>
<tr>
<td>command</td>
<td>This specifies the desired operation to be performed.</td>
</tr>
<tr>
<td>parameter</td>
<td>This argument is an integer or a pointer to a source structure containing port configuration data or an integer or a destination structure where status information is placed by the ioctl function call.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Return Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>An error occurred. Consult errno.</td>
</tr>
<tr>
<td>else</td>
<td>The operation succeeded.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Errors</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EBADF</td>
<td>Invalid file descriptor specified.</td>
</tr>
<tr>
<td>EFAULT</td>
<td>There was a problem accessing data from the user specified buffer.</td>
</tr>
<tr>
<td>EINVAL</td>
<td>An invalid command parameter was specified.</td>
</tr>
</tbody>
</table>

### ATC_TOD_SET_TIMESRC

This command sets the time source. The time source may use AC line sync pulses or the RTC square wave output.

**Time sources:**

- ATC_TIMESRC_LINESYNC
- ATC_TIMESRC_RTCSQWR
- ATC_TIMESRCCRYSTAL
- ATC_TIMESRCEXTERNAL1
- ATC_TIMESRCEXTERNAL2

<table>
<thead>
<tr>
<th>ioctl() Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fd</td>
<td>This is the file descriptor of the device to access.</td>
</tr>
<tr>
<td>command</td>
<td>ATC_TOD_SET_TIMESRC</td>
</tr>
<tr>
<td>parameter</td>
<td>Time source integer.</td>
</tr>
</tbody>
</table>

**Errors:**

EINVAL Hardware does not support setting the time source requested.

### ATC_TOD_GET_TIMESRC

This command returns the current time source. The time sources are defined above.

<table>
<thead>
<tr>
<th>ioctl() Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fd</td>
<td>This is the file descriptor of the device to access.</td>
</tr>
</tbody>
</table>
command | ATC_TOD_GET_TIMESRC  
parameter | None.

**ATC_TOD_GET_INPUT_FREQ**

This command returns the current frequency that is driving the time of day clock. 0 is returned if the frequency is unknown or unusable as a tick signal.

<table>
<thead>
<tr>
<th>ioctl() Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fd</td>
<td>This is the file descriptor of the device to access.</td>
</tr>
<tr>
<td>command</td>
<td>ATC_TOD_GET_INPUT_FREQ</td>
</tr>
<tr>
<td>parameter</td>
<td>None.</td>
</tr>
</tbody>
</table>

**ATC_TOD_REQUEST_TICK_SIG**

This command requests a signal to be sent at each tick of the time of day clock as long as the file device remains opened. The param value passed to ioctl is the signal number that should be sent to the calling process at each time of day clock tick.

<table>
<thead>
<tr>
<th>ioctl() Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fd</td>
<td>This is the file descriptor of the device to access.</td>
</tr>
<tr>
<td>command</td>
<td>ATC_TOD_REQUEST_TICK_SIG</td>
</tr>
<tr>
<td>parameter</td>
<td>Signal number integer.</td>
</tr>
</tbody>
</table>

Errors:
EINVAL        The signal number is not a valid signal or the input frequency is unknown or unusable as a tick signal.

**ATC_TOD_CANCEL_TICK_SIG**

This releases the signal from being sent when the time of day clock ticks. If the file device is closed, the signal is automatically released.

<table>
<thead>
<tr>
<th>ioctl() Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fd</td>
<td>This is the file descriptor of the device to access.</td>
</tr>
<tr>
<td>command</td>
<td>ATC_TOD_CANCEL_TICK_SIG</td>
</tr>
<tr>
<td>parameter</td>
<td>None.</td>
</tr>
</tbody>
</table>

**ATC_TOD_REQUEST_ONCHANGE_SIG**

This command requests a signal to be sent each time the system time is changed. The param value passed to ioctl is the signal number that should be sent to the calling process.

| ioctl() Argument | Description |
This is the file descriptor of the device to access.

<table>
<thead>
<tr>
<th>command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATC_TOD_REQUEST_ONCHANGE_SIG</td>
<td></td>
</tr>
</tbody>
</table>

Signal number integer.

Errors:
EINVAL  The signal number is not a valid signal.

**ATC_TOD_CANCEL_ONCHANGE_SIG**

This releases the signal from being sent when the system time is changed. If the file device is closed, the signal is automatically released.

<table>
<thead>
<tr>
<th>ioctl() Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fd</td>
<td>This is the file descriptor of the device to access.</td>
</tr>
<tr>
<td>command</td>
<td>ATC_TOD_CANCEL_ONCHANGE_SIG</td>
</tr>
<tr>
<td>parameter</td>
<td>None.</td>
</tr>
</tbody>
</table>
ID#: 1-12
Commenter: Peter Ragsdale
Comment Type: Technical
Location: Section 5.2.1, Page 5-2
Figure 5-2, Page 5-4
Clause: “Component height, with the exception of the interface connectors, shall not exceed the overall Engine Board envelope dimensions, shall not exceed 0.742” on the top including the engine board PCB material thickness, nor exceed 0.100” on the bottom excluding the engine board PCB material thickness.”

Issues and Proposed Revision

The standard should not define the height of components above or below the Engine Board PCB, but rather just specify the envelope of the Engine Board as was the intent in lines 23-34 (“provided that the resulting engine board envelope remains within the overall dimensions specified in this standard”). Lines 34-37 give the necessary guidance (“The assembled distance between the engine board and the host module shall provide a minimum of 0.100” of clearance between the engine board envelope and any components on the host module below the engine board (including the actual host module PCB)” but lines 40-43 remove the flexibility without any user need.

I suggest that the overall envelope of the engine board be held the same, but that the manufacturer can change where the pcb is relatively provided that the clearance between the host board and engine board is maintained using the defined standard host board connectors. This allows one to use “taller” components on the bottom of the engine board without conflict.

I submit that this is true intention of the section ... defined Engine Board envelope and a minimum of 0.100” clearance between host module and engine board. Furthermore, we need to specify the maximum height of components under the Engine Board so that Engine Board interchangeability can be achieved (a user need).

Disposition

Modify Section 5.2.1, Page 5-2, last paragraph (in bold):

Component height, with the exception of the interface connectors, shall not exceed the overall Engine Board envelope dimensions as shown in Figure 5-2. The recommended component height shall not exceed 0.742” on the top including the engine board PCB material thickness, nor exceed 0.100” on the bottom excluding the engine board PCB material thickness.

Manufacturers may choose alternative height engine board connectors to provide additional bottom-side clearance. Under no condition shall any components extend beyond the overall Engine Board envelope dimensions as shown in Figure 5-2. Manufacturers who choose to use alternate height engine board connectors shall provide the proper height F-F standoffs for host board mating with any engine boards sold separately from an ATC.
Modify title to Figure 5-2 (in bold):

“Recommended Engine Board / Host Module Stackup (not to scale)”

Replace Figure 5-2 with the following:
ID#: 1-13
Commenter: Dave Miller
Comment Type: Technical
Location: Section 5.2.2, Table 5-1
Clause: (Ethernet LED indicators)

**Issues and Proposed Revision**

LEDs are not operational when internal Ethernet switch is not desired.

Some contracts require the use of an external Ethernet switch, meaning that the Engine Board Ethernet is not connected to the indicator LEDs.

Engine board design anticipated the use of Ethernet switch to generate the 10/100 and ACTIVITY indicator lamps. The 2070-1B board has one Ethernet port and four indicators. The 2070-1C board has two Ethernet ports and four indicators. Therefore, converting four of the ATC 5.2b Engine Board RESERVED pins to indicators would provide proper indications.

Recommend the following changes:

- Change Pin #34 from Reserved to ENET1 TxD
- Change Pin #35 from Reserved to ENET1 RxD
- Change Pin #37 from Reserved to ENET1 LINK
- Change Pin #43 from Reserved to ENET2 TxD
- Change Pin #44 from Reserved to ENET2 RxD
- Change Pin #46 from Reserved to ENET2 LINK

**Disposition**

Comment rejected. Pins to remain as Reserved. Agencies which require Ethernet indications driven by the engine board (for host modules without switches) must specify same by Special Provision.
ID#: 1-14

Commenter: Ron Johnson

Comment Type: Technical

Location: Section 5.2.2, Table 5-1

Clause: (Ethernet LED indicators)

Issues and Proposed Revision

Harris County uses external Ethernet switches in its traffic cabinets, and wishes to avoid the added expense of internal Ethernet switches in ATC controllers. This causes a problem with the Ethernet status LEDs, which are expected to be driven from the internal switch and not from the Engine Board.

Recommend the following change to the Engine Board pinout to accommodate ATC controllers without internal switches. Those with internal switches would ignore these interface pins (they would be no connects).

- Change Pin #34 from RESERVED to ENET1_LED_LNK/ACT (open-drain signal, active-LOW (LED ON))
  (LNK turns LED ON, ACT flashes OFF)
- Change Pin #35 from RESERVED to ENET1_LED_10/100 (open-drain signal, active-LOW (LED ON))
  (10Mb turns LED OFF, 100Mb turns LED ON)
- Change Pin #36 from RESERVED to ENET2_LED_LNK/ACT (open-drain signal, active-LOW (LED ON))
  (LNK turns LED ON, ACT flashes OFF)
- Change Pin #37 from RESERVED to ENET2_LED_10/100 (open-drain signal, active-LOW (LED ON))
  (10Mb turns LED OFF, 100Mb turns LED ON)

Disposition

Comment rejected. Pins to remain as Reserved. Agencies which require Ethernet indications driven by the engine board (for host modules without switches) must specify same by Special Provision.
ID#: 1-15

Commenter: Ron Johnson

Comment Type: Technical

Location: Section 5.2.2, Page 5-6, Table 5-1

Clause: “ENGINE_PRESENT”

Issues and Proposed Revision

Engine Board connector P1, pin 38, is specified to be ENGINE_PRESENT. There is no guidance in the standard regarding how this pin is to be used, and since there is no configuration in which an Engine Board would not be installed in an ATC, this pin is unnecessary.

Recommend that this pin be removed and renamed to RESERVED.

Disposition

Modify Section 5.4.5 under ENGINE_PRESENT (in bold):

ENGINE_PRESENT is an active-low, logic-level output signal from the engine board. This signal indicates the physical presence of an engine board to the host board. This signal shall not be used to carry power supply current and may be used for manufacturer test purposes only. It shall be a no-connect on standard ATC host module PCBs.
ID#: 1-16

Commenter: Dave Miller

Comment Type: Technical

Location: Section 5.3.3, Page 5-7

Clause: (FLASH memory size)

**Issues and Proposed Revision**

The minimum of 6 Mb available to the user is not enough memory space to load application and libraries. Adding more memory does not allow time at bootup to check each byte in memory.

There is no requirement in the ATC 5.2b standard to check the FLASH memory 100% at bootup time. Suggest that the memory size be increased and only a small number of FLASH memory bytes per sector be checked at boot-up time to keep the boot-up time within 4.5 seconds.

Standard already requires a minimum of 6 MB available to the application in addition to the kernel, bootcode and libraries. If more memory is required by the application, the end-user must specify the memory amount by contract. Since the 5.2b standard does not require a check of each memory byte, no changes to the standard are required.

**Disposition**

*Modify Section 5.3.3 under FLASH Memory* (in bold):

The engine board shall contain a minimum of 16MB of FLASH memory for application and O/S program storage. FLASH devices shall use a segmented architecture allowing erasing, writing and reading of individual segments. Access to this memory shall be accomplished with wait states totaling no more than 100 ns and a data bus width of no less than 16 bits.
ID#: 1-17
Commenter: Dave Miller
Comment Type: Technical
Location: Section 5.3.3, Page 5-7
Clause: (FLASH memory)

Issues and Proposed Revision

Require that the 6MB of FLASH memory provided exclusively for application program storage be formatted and mounted by the manufacturer (immediately available for use by application programs).

Disposition

Add to Section 5.3.3 under FLASH Memory:

All FLASH memory available for application program storage shall be formatted and mounted by the manufacturer and immediately available for use.
ID#: 1-18

Commenter: Ron Johnson

Comment Type: Technical

Location: Section 5.3.3, Page 5-8

Clause: “The engine board shall contain a minimum of 16MB of DRAM or equivalent volatile memory for application and O/S program execution.”

**Issues and Proposed Revision**

Recent deployments have shown that 16MB of DRAM is insufficient to permit loading of application programs and necessary libraries, particularly those which have been developed using C++.

Recommend that the minimum DRAM quantity be increased to 64MB.

**Disposition**

Comment approved. Minimum DRAM quantity increased to 64MB.
ID#: 1-19
Commenter: Peter Ragsdale
Comment Type: Technical
Location: Section 5.4.1, Page 5-10
Clause: “The engine board does not provide standby power in support of the SRAM or RTC. In the absence of VPRIMARY, these components shall be supported and maintained by VSTANDBY_5 provided from the host module (+5 VDC Standby Power from the Power Supply). That is, standby power shall be provided only from the host and not from any source located on the engine board itself.”

Issues and Proposed Revision

Remove the specification prohibiting the use of standby power source on the EB ... a standard should not prohibit enhanced functionality that has no negative impacts to the device or interchangeability.

With today’s current technology using voltages that are predominantly not 5V, manufacturers need to be able to design products which meet User needs without hindrances such as this.

Proposed revised text is as follows:

“The engine board does not provide standby power in support of the SRAM or RTC. In the absence of VPRIMARY, the SRAM and RTC components shall be supported and maintained by either VSTANDBY_5 provided from the host module (+5 VDC Standby Power from the Power Supply) or an on-board standby power source.”

Disposition

Modify Section 5.4.1 Power under Operating Voltages and Currents (in bold):

The engine board does not provide standby power in support of the SRAM or RTC. In the absence of VPRIMARY, SRAM and the RTC shall be supported and maintained by VSTANDBY_5 provided from the host module (+5 VDC Standby Power from the Power Supply) as well as any (optional) standby sources located on the engine board. That is, standby power shall be provided only from the host and not from any source located on the engine board itself. VSTANDBY_5 shall provide standby power to the engine board over the voltage range of VPRIMARY down to 2.0 VDC. VSTANDBY_5 is allowed to fall below 2.0 VDC, but in that case it will not be considered to be providing standby power. The maximum combined average current draw from VSTANDBY_5 plus any standby sources on the engine board shall be 8.0 µA over the standby voltage range of 4.5 VDC to 2.0 VDC. Alternatively, a maximum instantaneous current draw of 6.0 µA (sum of SRAM and RTC current) measured at the SRAM and RTC devices, at a voltage of 2.5 VDC and at 25˚ C, shall be considered equivalent to the maximum average current draw requirement stated above.
ID#: 1-20
Commenter: Dave Miller
Comment Type: Technical
Location: Section 5.4.3
Clause: (serial port default bit rates)

Issues and Proposed Revision

Paragraph 5.4.3 does not define defaults, and Paragraph 2.2.5 requires a default baud rate of 38.4K bps on SP4 only. No other default baud rates are required.

Default baud rates for each serial port should be added to the standard.

Suggest that the default baud rates match 2070 ATC default rates except for SP4, which is already specified by 5.2b to default to 38.4 Kbps.

Disposition

Modify Section 5.4.3 Serial Interface Ports:

Add “Default Rate (bps):” as table entry for each port.

Set default rate for SP4 to be 115.2kbps (ASYNC).
Set default rate for SP6 to be 38.4kbps.
Set default rate for all other ports to (none).
ID#: 1-21
Commenter: Peter Ragsdale
Comment Type: Technical
Location: Section 5.4.3, Pages 5-12, 5-13
Clause: “SPx_TXC_EXT: Transmit Clock External (I)”

Issues and Proposed Revision

I suggest changing SPx_TXC_EXT to SPx_DTR as a software-configurable option (possibly based on the Host EEPROM data … which would then require changes). This would provide better support of modems, etc and I do not know of any module that really uses this signal (yes, some 2070-7B modules can support it via a DIP switch, etc but I do not know of the signal being used anywhere).

Disposition

Comment rejected.
ID#: 1-22
Commenter: Peter Ragsdale
Comment Type: Technical
Location: Section 6.1.1, Page 6-2
Clause: “Communications circuitry may be embedded inside the ATC, providing the field connectors and pin assignments conform to this standard.”

Issues and Proposed Revision
Currently, the “standard” embedded ATC connectors (such as C50S, C60P, Port 1) conflict with section 6.1.1 … this section needs to be fixed.

Disposition
Replace Section 6.2.3 Mechanical Field Connections, through Section 6.2.3.3, as follows:

6.2.3 Standardized Legacy Field Connections
Where standardized legacy field connections are provided by an ATC, either as integral to the controller assembly or via a plug-in module in one of the communications slots, these connections shall be labeled appropriately and shall meet all applicable requirements of their respective standards as defined below.

An ATC may contain non-standardized field connections. These connections shall be thoroughly documented by the ATC manufacturer as to their pinout and electrical requirements.

6.2.3.1 NEMA Field Connections
The following field connections shall meet all applicable requirements as defined herein and in NEMA TS 2-2003, Traffic Controller Assemblies with NTCIP Requirements, v02.06.

Port 1
Port 1 provides an EIA-485 field connection for synchronous operation.

The Port 1 connector shall be a 15-pin metal shell D sub-miniature type connector. The connector shall utilize female contacts and shall be equipped with latching blocks. Pin connections shall be as follows:

<table>
<thead>
<tr>
<th>Pin</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Tx Data +</td>
</tr>
<tr>
<td>2</td>
<td>Logic Ground</td>
</tr>
<tr>
<td>3</td>
<td>Tx Clock +</td>
</tr>
<tr>
<td>4</td>
<td>Logic Ground</td>
</tr>
<tr>
<td>5</td>
<td>Rx Data +</td>
</tr>
<tr>
<td>6</td>
<td>Logic Ground</td>
</tr>
</tbody>
</table>
### Port 1

<table>
<thead>
<tr>
<th>Pin</th>
<th>Function</th>
<th>I/O</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>Rx Clock +</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Logic Ground</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Tx Data -</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Port 1 Disable (0VDC=disable) (*)</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Tx Clock -</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Earth Ground</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Rx Data -</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Rx Clock –</td>
<td></td>
</tr>
</tbody>
</table>

* Pin 10 shall be connected to Pin 8 when it is desirable to disable all Port 1 communications activity.

### Port 2

Port 2 provides an EIA-232 asynchronous interface for connection to a printer or personal computer.

The Port 2 connector shall be a 25-pin metal shell D sub-miniature type connector. The connector shall utilize female contacts and shall be equipped with latching blocks. Pin connections shall be as follows:

<table>
<thead>
<tr>
<th>Pin</th>
<th>Function</th>
<th>I/O</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Earth Ground</td>
<td>[-]</td>
</tr>
<tr>
<td>2</td>
<td>Transmitted Data</td>
<td>[O]</td>
</tr>
<tr>
<td>3</td>
<td>Received Data</td>
<td>[I]</td>
</tr>
<tr>
<td>4</td>
<td>Request To Send</td>
<td>[O]</td>
</tr>
<tr>
<td>5</td>
<td>Clear To Send</td>
<td>[I]</td>
</tr>
<tr>
<td>6</td>
<td>Not Used</td>
<td>[-]</td>
</tr>
<tr>
<td>7</td>
<td>Logic Ground</td>
<td>[-]</td>
</tr>
<tr>
<td>8</td>
<td>Received Line Signal Detector</td>
<td>[I]</td>
</tr>
<tr>
<td>9-19</td>
<td>Not Used</td>
<td>[-]</td>
</tr>
<tr>
<td>20</td>
<td>DTE Ready</td>
<td>[O]</td>
</tr>
<tr>
<td>21-25</td>
<td>Not Used</td>
<td>[-]</td>
</tr>
</tbody>
</table>

### Port 3

Port 3 provides a four-wire full-duplex communications interface over an unconditioned Type 3002 voice grade channel utilizing FSK modulation.

The Port 3 connector shall be a nine-pin metal shell D sub-miniature type connector. The connector shall utilize male contacts and shall be equipped with latching blocks. Pin connections shall be as follows:

<table>
<thead>
<tr>
<th>Pin</th>
<th>Function</th>
<th>I/O</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Transmit 1</td>
<td>[O]</td>
</tr>
<tr>
<td>2</td>
<td>Transmit 2</td>
<td>[O]</td>
</tr>
<tr>
<td>3</td>
<td>Reserved</td>
<td>[-]</td>
</tr>
<tr>
<td>4</td>
<td>Receive 1</td>
<td>[I]</td>
</tr>
<tr>
<td>5</td>
<td>Receive 2</td>
<td>[I]</td>
</tr>
<tr>
<td>6</td>
<td>Earth Ground</td>
<td>[-]</td>
</tr>
<tr>
<td>7</td>
<td>Reserved</td>
<td>[-]</td>
</tr>
</tbody>
</table>
6.2.3.2 2070 Field Connections

The following field connections shall meet all applicable requirements as defined herein and in Caltrans Transportation Electrical Equipment Specifications (TEES), dated March 12, 2009, plus all applicable errata.

**C50S**

C50S provides an EIA-232 asynchronous interface for connection to a printer or personal computer.

The C50S connector shall be a nine-pin metal shell D sub-miniature type connector. The connector shall utilize female contacts and shall be equipped with jack screws. Pin connections shall be as follows:

<table>
<thead>
<tr>
<th>Pin</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>C50 Enable</td>
</tr>
<tr>
<td>2</td>
<td>SP4 Receive Data</td>
</tr>
<tr>
<td>3</td>
<td>SP4 Transmit Data</td>
</tr>
<tr>
<td>4</td>
<td>Reserved</td>
</tr>
<tr>
<td>5</td>
<td>DC Ground #1</td>
</tr>
<tr>
<td>6</td>
<td>Reserved</td>
</tr>
<tr>
<td>7</td>
<td>Reserved</td>
</tr>
<tr>
<td>8</td>
<td>Reserved</td>
</tr>
<tr>
<td>9</td>
<td>Reserved</td>
</tr>
</tbody>
</table>

**C50J**

C50J provides an EIA-232 asynchronous interface for connection to a printer or personal computer.

The C50J connector shall be an eight-pin modular RJ45-type connector. Pin connections shall be as follows:

<table>
<thead>
<tr>
<th>Pin</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Reserved</td>
</tr>
<tr>
<td>2</td>
<td>SP4 Receive Data</td>
</tr>
<tr>
<td>3</td>
<td>C50 Enable</td>
</tr>
<tr>
<td>4</td>
<td>Reserved</td>
</tr>
<tr>
<td>5</td>
<td>SP4 Transmit Data</td>
</tr>
<tr>
<td>6</td>
<td>DC Ground #1</td>
</tr>
<tr>
<td>7</td>
<td>Reserved</td>
</tr>
<tr>
<td>8</td>
<td>Reserved</td>
</tr>
</tbody>
</table>

**C60P**

C60P provides an EIA-232 asynchronous interface for connection to a remote front panel interface device.
The C60P connector shall be a nine-pin metal shell D sub-miniature type connector. The connector shall utilize male contacts and shall be equipped with jack screws. Pin connections shall be as follows:

<table>
<thead>
<tr>
<th>Pin</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>* Power [I]</td>
</tr>
<tr>
<td>2</td>
<td>SP6 Receive Data  [I]</td>
</tr>
<tr>
<td>3</td>
<td>SP6 Transmit Data [O]</td>
</tr>
<tr>
<td>4</td>
<td>Reserved [-]</td>
</tr>
<tr>
<td>5</td>
<td>DC Ground #3 [-]</td>
</tr>
<tr>
<td>6</td>
<td>Reserved [-]</td>
</tr>
<tr>
<td>7</td>
<td>CPURESET [O]</td>
</tr>
<tr>
<td>8</td>
<td>Reserved [-]</td>
</tr>
<tr>
<td>9</td>
<td>CPULED [O]</td>
</tr>
</tbody>
</table>

* Pin 1 shall provide +5VDC, 350mA maximum draw, referenced to DC Ground #3 (isolated). All signals on C60P shall be referenced to isolated DC Ground #3.

C12S

C12S provides an EIA-485 field connection for synchronous operation.

The C12S connector shall be a 25-pin metal shell D sub-miniature type connector. The connector shall utilize female contacts and shall be equipped with jack screws. Pin connections shall be as follows:

<table>
<thead>
<tr>
<th>Pin</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SP5 Tx Data +</td>
</tr>
<tr>
<td>2</td>
<td>SP5 Rx Data +</td>
</tr>
<tr>
<td>3</td>
<td>SP5 Tx Clock +</td>
</tr>
<tr>
<td>4</td>
<td>SP5 Rx Clock +</td>
</tr>
<tr>
<td>5</td>
<td>SP3 Tx Data +</td>
</tr>
<tr>
<td>6</td>
<td>SP3 Rx Data +</td>
</tr>
<tr>
<td>7</td>
<td>SP3 Tx Clock +</td>
</tr>
<tr>
<td>8</td>
<td>SP3 Rx Clock +</td>
</tr>
<tr>
<td>9</td>
<td>LINESYNC +</td>
</tr>
<tr>
<td>10</td>
<td>NRESET +</td>
</tr>
<tr>
<td>11</td>
<td>POWERDOWN +</td>
</tr>
<tr>
<td>12</td>
<td>* Power</td>
</tr>
<tr>
<td>13</td>
<td>DC Ground #2</td>
</tr>
<tr>
<td>14</td>
<td>SP5 Tx Data -</td>
</tr>
<tr>
<td>15</td>
<td>SP5 Rx Data -</td>
</tr>
<tr>
<td>16</td>
<td>SP5 Tx Clock -</td>
</tr>
<tr>
<td>17</td>
<td>SP5 Rx Clock -</td>
</tr>
<tr>
<td>18</td>
<td>SP3 Tx Data -</td>
</tr>
<tr>
<td>19</td>
<td>SP3 Rx Data -</td>
</tr>
<tr>
<td>20</td>
<td>SP3 Tx Clock -</td>
</tr>
<tr>
<td>21</td>
<td>SP3 Rx Clock -</td>
</tr>
<tr>
<td>22</td>
<td>LINESYNC -</td>
</tr>
<tr>
<td>23</td>
<td>NRESET -</td>
</tr>
<tr>
<td>24</td>
<td>POWERDOWN -</td>
</tr>
</tbody>
</table>
25 Equipment Ground

* Pin 12 shall provide +5VDC referenced to DC Ground #2 (isolated). All signals on C12S shall be referenced to isolated DC Ground #2.

**C13S**

C13S provides an EIA-485 field connection for asynchronous/synchronous operation.

The C13S connector shall be a 25-pin metal shell D sub-miniature type connector. The connector shall utilize female contacts and shall be equipped with jack screws. Pin connections shall be as follows:

<table>
<thead>
<tr>
<th>Pin</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SP8 Tx Data +</td>
</tr>
<tr>
<td>2</td>
<td>SP8 Rx Data +</td>
</tr>
<tr>
<td>3</td>
<td>SP8 Tx Clock +</td>
</tr>
<tr>
<td>4</td>
<td>SP8 Rx Clock +</td>
</tr>
<tr>
<td>5</td>
<td>SP8 Request To Send +</td>
</tr>
<tr>
<td>6</td>
<td>SP8 Clear To Send +</td>
</tr>
<tr>
<td>7</td>
<td>SP8 Carrier Detect +</td>
</tr>
<tr>
<td>8</td>
<td>Reserved</td>
</tr>
<tr>
<td>9</td>
<td>LINESYNC +</td>
</tr>
<tr>
<td>10</td>
<td>NRESET +</td>
</tr>
<tr>
<td>11</td>
<td>POWERDOWN +</td>
</tr>
<tr>
<td>12</td>
<td>* Power</td>
</tr>
<tr>
<td>13</td>
<td>DC Ground #2</td>
</tr>
<tr>
<td>14</td>
<td>SP8 Tx Data -</td>
</tr>
<tr>
<td>15</td>
<td>SP8 Rx Data -</td>
</tr>
<tr>
<td>16</td>
<td>SP8 Tx Clock -</td>
</tr>
<tr>
<td>17</td>
<td>SP8 Rx Clock -</td>
</tr>
<tr>
<td>18</td>
<td>SP8 Request To Send -</td>
</tr>
<tr>
<td>19</td>
<td>SP8 Clear To Send -</td>
</tr>
<tr>
<td>20</td>
<td>SP8 Carrier Detect -</td>
</tr>
<tr>
<td>21</td>
<td>Reserved</td>
</tr>
<tr>
<td>22</td>
<td>LINESYNC -</td>
</tr>
<tr>
<td>23</td>
<td>NRESET -</td>
</tr>
<tr>
<td>24</td>
<td>POWERDOWN -</td>
</tr>
<tr>
<td>25</td>
<td>Equipment Ground</td>
</tr>
</tbody>
</table>

* Pin 12 shall provide +5VDC referenced to DC Ground #2 (isolated). All signals on C13S shall be referenced to isolated DC Ground #2.

**C21S and C22S (DE-9S versions)**

C21S/22S (DE-9S versions) provide an EIA-232 asynchronous interface for connection to an external communications device or personal computer.

The C21S/22S (DE-9S versions) connector shall be a nine-pin metal shell D sub-miniature type connector. The connector shall utilize female contacts and shall be equipped with jack screws. Pin connections shall be as follows:
Pin Function
1 Carrier Detect (DCD) [I]
2 Receive Data (RXD) [I]
3 Transmit Data (TXD) [O]
4 Reserved [-]
5 * Interface Ground [-]
6 Reserved [-]
7 Request To Send (RTS) [O]
8 Clear To Send (CTS) [I]
9 Reserved [-]

* Pin 5 shall provide an interface ground which is isolated from all internal ground references. All signals on C21S/22S shall be referenced to this isolated ground.

C21S and C22S (DA-15S versions)

C21S/22S (DA-15S versions) provides an EIA-485 field connection for synchronous operation.

The C21S/22S (DA-15S versions) connector shall be a 15-pin metal shell D sub-miniature type connector. The connector shall utilize female contacts and shall be equipped with jack screws. Pin connections shall be as follows:

Pin Function
1 Tx Data +
2 * Interface Ground
3 Tx Clock +
4 * Interface Ground
5 Rx Data +
6 * Interface Ground
7 Rx Clock +
8 Reserved
9 Tx Data -
10 * Interface Ground
11 Tx Clock -
12 * Interface Ground
13 Rx Data -
14 * Interface Ground
15 Rx Clock –

* Pins 2, 4, 6, 10, 12 and 14 shall provide an interface ground which is isolated from all internal ground references. All signals on C21S/22S shall be referenced to this isolated ground.

C2S and C20S

C2S/20S provide EIA-232 and FSK field connections for asynchronous operation.

The C2S/20S connector shall be a 14-pin AMP M14 type connector. The connector shall utilize female contacts and shall be equipped with spring latch supports. Pin connections shall be as follows:
<table>
<thead>
<tr>
<th>Pin</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Audio In 1</td>
</tr>
<tr>
<td>B</td>
<td>Audio In 2</td>
</tr>
<tr>
<td>C</td>
<td>Audio Out 1</td>
</tr>
<tr>
<td>D</td>
<td>* Power</td>
</tr>
<tr>
<td>E</td>
<td>Audio Out 2</td>
</tr>
<tr>
<td>F</td>
<td>Reserved</td>
</tr>
<tr>
<td>H</td>
<td>Carrier Detect (DCD) (EIA-232)</td>
</tr>
<tr>
<td>J</td>
<td>Request To Send (RTS) (EIA-232)</td>
</tr>
<tr>
<td>K</td>
<td>Receive Data (RXD) (EIA-232)</td>
</tr>
<tr>
<td>L</td>
<td>Transmit Data (TXD) (EIA-232)</td>
</tr>
<tr>
<td>M</td>
<td>Clear To Send (CTS) (EIA-232)</td>
</tr>
<tr>
<td>N</td>
<td>** Interface Ground</td>
</tr>
<tr>
<td>P</td>
<td>Reserved</td>
</tr>
<tr>
<td>R</td>
<td>Reserved</td>
</tr>
</tbody>
</table>

* Pin D shall provide +5VDC referenced to interface ground (Pin N).

** Pin N shall provide an interface ground which is isolated from all internal ground references. All EIA-232 signals on C21S/22S shall be referenced to this isolated ground.
ID#: 1-23
Commenter: Peter Ragsdale
Comment Type: Technical
Location: Section 6.2.2, Pages 6-5, 6-6
Clause: (+12V ISO VDC and DCGND2 pins)

Issues and Proposed Revision

+12VDC ISO and DCGND2 should not be required on the modem slot pinout. According to TEES, 2070-6x and 2070-7x modules are not allowed to use the +12V ISO source. Additionally, a NEMA version of an ATC v5.2b controller does not generate +12V ISO (it has +24VDC instead).

Disposition

Comment accepted. See Comment 1-25 disposition.
ID#: 1-24
Commenter: Peter Ragsdale
Comment Type: Technical
Location: Section 6.2.2, Pages 6-5 and 6-6
Clause: (Both communications slots define a signal called “INSTALLED”)

**Issues and Proposed Revision**

Both of the communication slots specify an “Installed” signal but no where does the standard define what it does or how one uses it.

We suggest that if the INSTALLED signal is not “active” (i.e. module not present) then the ATC can do the following items:

- Turn-off +12V and -12V to the communication slots
- If the communication slot(s) are the only place in the ATC where these voltages are used, then the ATC should be allowed to turn off these voltages completely if none of the INSTALLED signals is active and even extinguish Power Supply LEDs for +12V and -12V
- Disable the serial port drivers to the communication slot(s) if desired

We suggest that if the INSTALLED signal is active (i.e. communication modem is installed), then the ATC should be allowed to disable its integrated serial ports that could be in conflict. For example, if a 2070-6A module is installed in the Mandatory slot, then the ATC could disable any integrated SP1 or SP2 serial ports it may have.

**Disposition**

*Add Section 6.3.1.6, as follows:*

**6.3.1.6 INSTALLED Signals**

INSTALLED signals (at connector position B23 in each communications slot) are active-low signals, driven by the installed module, indicating the installation of a module into the corresponding slot. The ATC may utilize these signals to disable any integrated serial ports which may be in conflict with active ports on the installed module.

Should neither of the INSTALLED signals be active, the ATC may (optionally):

- disable all serial port drivers to the communications slots
- disable ±12VDC (referenced to DCGND1) to the communications slots
- if ±12VDC (referenced to DCGND1) is ONLY used by the communications slots and not by any integrated serial ports within the ATC, those supplies may be completely extinguished
ID#: 1-25

Commenter: Peter Ragsdale

Comment Type: Technical

Location: Section 6.2.2, Pages 6-5 and 6-6

Clause: {pin-outs for both the Mandatory and Optional slot(s)}

Issues and Proposed Revision

Communication slots: Suggest partial signals only be included on the slot(s) such as SP1, SP2, and possibly the Ethernet.

(see attached material – immediately following – three pages)

What other signals are required in this case?

Disposition

Replace Section 6.2.2 as follows:

(continued next page)
Mandatory Communications Slot:

<table>
<thead>
<tr>
<th>PIN</th>
<th>ROW A</th>
<th>ROW B</th>
<th>ROW C</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SP1TXD+</td>
<td>SP6TXD+(**)</td>
<td>SP5TXD+(**)</td>
</tr>
<tr>
<td>2</td>
<td>SP1TXD-</td>
<td>SP6TXD-(**)</td>
<td>SP5TXD-(**)</td>
</tr>
<tr>
<td>3</td>
<td>SP1RXD+</td>
<td>SP6RXD+(**)</td>
<td>SP5TXC+(**)</td>
</tr>
<tr>
<td>4</td>
<td>SP1RXD-</td>
<td>SP6RXD-(**)</td>
<td>SP5TXC-(**)</td>
</tr>
<tr>
<td>5</td>
<td>SP1RTS+</td>
<td>SP1TXCO+</td>
<td>SP5RXD+(**)</td>
</tr>
<tr>
<td>6</td>
<td>SP1RTS-</td>
<td>SP1TXCO-</td>
<td>SP5RXD-(**)</td>
</tr>
<tr>
<td>7</td>
<td>SP1CTS+</td>
<td>SP1TXCI+</td>
<td>SP5RXC+(**)</td>
</tr>
<tr>
<td>8</td>
<td>SP1CTS-</td>
<td>SP1TXCI-</td>
<td>SP5RXC-(**)</td>
</tr>
<tr>
<td>9</td>
<td>SP1DCD+</td>
<td>SP1RXC+</td>
<td>SP3TXD+(**)</td>
</tr>
<tr>
<td>10</td>
<td>SP1DCD-</td>
<td>SP1RXC-</td>
<td>SP3TXD-(**)</td>
</tr>
<tr>
<td>11</td>
<td>SP2TXD+</td>
<td>SP4TXD+(**)</td>
<td>SP3RXD+(**)</td>
</tr>
<tr>
<td>12</td>
<td>SP2TXD-</td>
<td>SP4TXD-(**)</td>
<td>SP3RXD-(**)</td>
</tr>
<tr>
<td>13</td>
<td>SP2RXD+</td>
<td>SP4RXD+(**)</td>
<td>SP3RTS+(**)</td>
</tr>
<tr>
<td>14</td>
<td>SP2RXD-</td>
<td>SP4RXD-(**)</td>
<td>SP3RTS-(**)</td>
</tr>
<tr>
<td>15</td>
<td>SP2RTS+</td>
<td>SP2TXCO+</td>
<td>SP3CTS+(**)</td>
</tr>
<tr>
<td>16</td>
<td>SP2RTS-</td>
<td>SP2TXCO-</td>
<td>SP3CTS-(**)</td>
</tr>
<tr>
<td>17</td>
<td>SP2CTS+</td>
<td>SP2TXCI+</td>
<td>SP3DCD+(**)</td>
</tr>
<tr>
<td>18</td>
<td>SP2CTS-</td>
<td>SP2TXCI-</td>
<td>SP3DCD-(**)</td>
</tr>
<tr>
<td>19</td>
<td>SP2DCD+</td>
<td>SP2RXC+</td>
<td>SP3TXCO+(**)</td>
</tr>
<tr>
<td>20</td>
<td>SP2DCD-</td>
<td>SP2RXC-</td>
<td>SP3TXCO-(**)</td>
</tr>
<tr>
<td>21</td>
<td>DCGND1</td>
<td>(reserved)</td>
<td>SP3TXCI+(**)</td>
</tr>
<tr>
<td>22</td>
<td>NETWK1 Tx+</td>
<td>(reserved)</td>
<td>SP3TXCI-(**)</td>
</tr>
<tr>
<td>23</td>
<td>NETWK2 Tx-</td>
<td>INSTALLED</td>
<td>SP3RXC+(**)</td>
</tr>
<tr>
<td>24</td>
<td>(reserved)</td>
<td>LINESYNC</td>
<td>SP3RXC-(**)</td>
</tr>
<tr>
<td>25</td>
<td>NETWK3 Rx+</td>
<td>POWERUP</td>
<td>CPU_RESET</td>
</tr>
<tr>
<td>26</td>
<td>NETWK4 Rx-</td>
<td>POWERDOWN</td>
<td>CPU_ACTIVE(**)</td>
</tr>
<tr>
<td>27</td>
<td>DCGND1</td>
<td>DCGND1</td>
<td>DCGND1</td>
</tr>
<tr>
<td>28</td>
<td>+12 VDC</td>
<td>-12 VDC</td>
<td>+5 VDC STANDBY</td>
</tr>
<tr>
<td>29</td>
<td>+5 VDC</td>
<td>+5 VDC</td>
<td>+5 VDC</td>
</tr>
<tr>
<td>30</td>
<td>DCGND1</td>
<td>DCGND1</td>
<td>DCGND1</td>
</tr>
<tr>
<td>31</td>
<td>+12 ISO VDC(**)</td>
<td>+12 ISO VDC(**)</td>
<td>+12 ISO VDC(**)</td>
</tr>
<tr>
<td>32</td>
<td>DCGND2(**)</td>
<td>DCGND2(**)</td>
<td>DCGND2(**)</td>
</tr>
</tbody>
</table>
## Optional Second Communications Slot:

<table>
<thead>
<tr>
<th>PIN</th>
<th>ROW A</th>
<th>ROW B</th>
<th>ROW C</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SP3TXD+</td>
<td>SP6TXD+(**)</td>
<td>SP5TXD+(**)</td>
</tr>
<tr>
<td>2</td>
<td>SP3TXD-</td>
<td>SP6TXD-(**)</td>
<td>SP5TXD-(**)</td>
</tr>
<tr>
<td>3</td>
<td>SP3RXD+</td>
<td>SP6RXD+(**)</td>
<td>SP5TXC+(**)</td>
</tr>
<tr>
<td>4</td>
<td>SP3RXD-</td>
<td>SP6RXD-(**)</td>
<td>SP5TXC-(**)</td>
</tr>
<tr>
<td>5</td>
<td>SP3RTS+</td>
<td>SP3TXC0+</td>
<td>SP5RXD+(**)</td>
</tr>
<tr>
<td>6</td>
<td>SP3RTS-</td>
<td>SP3TXC0-</td>
<td>SP5RXD-(**)</td>
</tr>
<tr>
<td>7</td>
<td>SP3CTS+</td>
<td>SP3TXC1+</td>
<td>SP5RXC+(**)</td>
</tr>
<tr>
<td>8</td>
<td>SP3CTS-</td>
<td>SP3TXC1-</td>
<td>SP5RXC-(**)</td>
</tr>
<tr>
<td>9</td>
<td>SP3DCD+</td>
<td>SP3RXC+</td>
<td>SP3TXD+(**)</td>
</tr>
<tr>
<td>10</td>
<td>SP3DCD-</td>
<td>SP3RXC-</td>
<td>SP3TXD-(**)</td>
</tr>
<tr>
<td>11</td>
<td>SP4TXD+</td>
<td>SP4TXD+(**)</td>
<td>SP3RXD+(**)</td>
</tr>
<tr>
<td>12</td>
<td>SP4TXD-</td>
<td>SP4TXD-(**)</td>
<td>SP3RXD-(**)</td>
</tr>
<tr>
<td>13</td>
<td>SP4RXD+</td>
<td>SP4RXD+(**)</td>
<td>SP3RTS+(**)</td>
</tr>
<tr>
<td>14</td>
<td>SP4RXD-</td>
<td>SP4RXD-(**)</td>
<td>SP3RTS-(**)</td>
</tr>
<tr>
<td>15</td>
<td>(reserved)</td>
<td>(reserved)</td>
<td>SP3CTS+(**)</td>
</tr>
<tr>
<td>16</td>
<td>(reserved)</td>
<td>(reserved)</td>
<td>SP3CTS-(**)</td>
</tr>
<tr>
<td>17</td>
<td>(reserved)</td>
<td>(reserved)</td>
<td>SP3DCD+(**)</td>
</tr>
<tr>
<td>18</td>
<td>(reserved)</td>
<td>(reserved)</td>
<td>SP3DCD-(**)</td>
</tr>
<tr>
<td>19</td>
<td>(reserved)</td>
<td>(reserved)</td>
<td>SP3TXCO+(**)</td>
</tr>
<tr>
<td>20</td>
<td>(reserved)</td>
<td>(reserved)</td>
<td>SP3TXCO-(**)</td>
</tr>
<tr>
<td>21</td>
<td>DCGND1</td>
<td>C50 ENABLE</td>
<td>SP3TXC1+(**)</td>
</tr>
<tr>
<td>22</td>
<td>NETWK1 Tx+</td>
<td>(reserved)</td>
<td>SP3TXC1-(**)</td>
</tr>
<tr>
<td>23</td>
<td>NETWK2 Tx-</td>
<td>INSTALLED</td>
<td>SP3RXC+(**)</td>
</tr>
<tr>
<td>24</td>
<td>(reserved)</td>
<td>LINESYNC</td>
<td>SP3RXC-(**)</td>
</tr>
<tr>
<td>25</td>
<td>NETWK3 Rx+</td>
<td>POWERUP</td>
<td>CPU_RESET</td>
</tr>
<tr>
<td>26</td>
<td>NETWK4 Rx-</td>
<td>POWERDOWN</td>
<td>CPU_ACTIVE(**)</td>
</tr>
<tr>
<td>27</td>
<td>DCGND1</td>
<td>DCGND1</td>
<td>DCGND1</td>
</tr>
<tr>
<td>28</td>
<td>+12 VDC</td>
<td>-12 VDC</td>
<td>+5 VDC STANDBY</td>
</tr>
<tr>
<td>29</td>
<td>+5 VDC</td>
<td>+5 VDC</td>
<td>+5 VDC</td>
</tr>
<tr>
<td>30</td>
<td>DCGND1</td>
<td>DCGND1</td>
<td>DCGND1</td>
</tr>
<tr>
<td>31</td>
<td>+12 ISO VDC(**)</td>
<td>+12 ISO VDC(**)</td>
<td>+12 ISO VDC(**)</td>
</tr>
<tr>
<td>32</td>
<td>DCGND2(**)</td>
<td>DCGND2(**</td>
<td>DCGND2(**)</td>
</tr>
</tbody>
</table>

**Notes:**

1. Signal directions are referenced to the engine board, not the communications interface. For example, SP1_TXD (SP1TXD+ and SP1TXD-) is Serial Port 1 data transmitted from the engine board to the communications interface. SP1_RXD (SP1RXD+ and SP1RXD-) is Serial Port 1 data received by the engine board from the communications interface.

2. Signals noted with (**) are optional. If included they must represent the specified standard signal. If not included they are reserved and shall not be used for any purpose.
Discussion of Communication Slot Support

What is the User Need/Purpose of the Communication Slot?

To reuse/be compatible with 2070-type communications modules by adhering to form fit and function.

2070 Compatibility Guidelines:

1. Although not explicitly stated anywhere in TEES, SP6 cannot be used by any 2070 module in any slot (A1-A4) since the 2070-3x modules have exclusive use of this port and a front panel is always present in a 2070.
2. A 2070 module installed in slots A2-A4 shall not use SP4 since they do not have C50_Enable signal brought to them, and therefore would conflict with SP4 (C50S) on the front panel (2070-3x modules).
3. A 2070 module installed in 2070 slots A1, A2, or A4 cannot use SP5 unless a module is installed in the A3 slot (typically I/O) and any such module installed in A3 assumes that it has exclusive control of SP5.
4. All 2070 modules assume that they “own” the serial ports that they are connected to. For example, one cannot put a 2070-6A module in both the A2 & A4 slots as they will conflict on SP1 and SP2.
5. The SP1 and SP2 ports are disabled unless a module is installed in A2 (in which case it assumes 100% ownership of both SP1 and SP2).

So where does this bring us in an ATC v5.2b controller?

First, let’s review the pinouts for the ATC mandatory and optional communication slots:

Mandatory Slot:

- has the same pinout as slots A2, A3, & A4 in a 2070
- there is no specification for what the “installed” pin does
- includes support for SP1, SP2, SP3, SP4, SP5, SP6, & ENET

2nd (optional) Slot:

- has the same pinout as slot A1 in a 2070
- it supports the “C50_enabled” pin
- there is no specification for what the “installed” pin does
- includes support for SP3, SP4, SP5, SP6, & ENET
Now, let’s specify what the User Needs for each communication slot really are:

2nd (optional) Slot:

- form, fit, and pinout as required to support desired ports
- support the “C50_enabled” pin (if SP4 is included in the pinout)
- suggest that it is only required to support “channel 1 and channel 2” being SP3 and SP4 respectively, plus ENET as an option (how do we prevent both slots from using the ENET signals?)
- suggest that the “installed” pin should be used to disable any ATC integrated/embedded SP3 connector

Mandatory Slot:

- form, fit, and pinout as required to support desired ports
- cannot add “C50_Enabled” (otherwise channel 2, SP2, would be disabled). Therefore, remove SP4 from the pinout
- suggest that only SP1, SP2, and ENET be supported (i.e. remove SP3, SP4, SP5, and SP6)
- suggest that the “Installed” pin should be used to disable any ATC integrated/embedded SP1 & SP2 connectors

Also, if neither “Installed” pin is active, then I suggest that one could turn off the +/- 12V supplies is one desired.

Problems identified so far that can cause conflicts, errors, and inconsistencies:

- If ENET is one both communication slots, there will be a conflict on the Hub
  - Two options:
    - Remove ENET from the optional slot, or
    - Add another port to the hub (e.g. use the “internal expansion” port)
- If two modules are installed that use the same ports, there will be a conflict as 2070 modules are not designed to handle this (this problem also exists in the 2070)

The next issue is whether or not we used support a 2070-2B module in the communication slot, or in a new to-be-defined I/O slot.
Proposed Modem Slot Pin-out (minimum requirements):

<table>
<thead>
<tr>
<th>PIN</th>
<th>ROW A</th>
<th>ROW B</th>
<th>ROW C</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SP1TXD+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>2</td>
<td>SP1TXD-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>3</td>
<td>SP1RXD+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>4</td>
<td>SP1RXD-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>5</td>
<td>SP1RTS+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>6</td>
<td>SP1RTS-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>7</td>
<td>SP1CTS+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>8</td>
<td>SP1CTS-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>9</td>
<td>SP1DCD+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>10</td>
<td>SP1DCD-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>11</td>
<td>SP2TXD+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>12</td>
<td>SP2TXD-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>13</td>
<td>SP2RXD+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>14</td>
<td>SP2RXD-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>15</td>
<td>SP2RTS+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>16</td>
<td>SP2RTS-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>17</td>
<td>SP2CTS+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>18</td>
<td>SP2CTS-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>19</td>
<td>SP2DCD+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>20</td>
<td>SP2DCD-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>21</td>
<td>DCGND1</td>
<td>NA</td>
<td>-</td>
</tr>
<tr>
<td>22</td>
<td>NETWK1 Tx+</td>
<td>NA</td>
<td>-</td>
</tr>
<tr>
<td>23</td>
<td>NETWK2 Tx-</td>
<td>INSTALLED</td>
<td>-</td>
</tr>
<tr>
<td>24</td>
<td>NA</td>
<td>LINESYNC</td>
<td>-</td>
</tr>
<tr>
<td>25</td>
<td>NETWK3 Rx+</td>
<td>POWERUP</td>
<td>CPU_RESET</td>
</tr>
<tr>
<td>26</td>
<td>NETWK4 Rx-</td>
<td>POWERDOWN</td>
<td>-</td>
</tr>
<tr>
<td>27</td>
<td>DCGND1</td>
<td>DCGND1</td>
<td>DCGND1</td>
</tr>
<tr>
<td>28</td>
<td>+12 VDC</td>
<td>-12 VDC</td>
<td>+5 VDC Standby</td>
</tr>
<tr>
<td>29</td>
<td>+5 VDC</td>
<td>+5 VDC</td>
<td>+5 VDC</td>
</tr>
<tr>
<td>30</td>
<td>DCGND1</td>
<td>DCGND1</td>
<td>DCGND1</td>
</tr>
<tr>
<td>31</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>32</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Do we want to add the following signals: SP1TXCO+, SP1TXCO-, SP1RXC+, SP1RXC-, SP2TXCO+, SP2TXCO-, SP2RXC+, and SP2RXC-. 
ID#: 1-26
Commenter: Peter Ragsdale
Comment Type: Technical
Location: Section 6.2.3.2, Page 6-8
Clause: (No support for NEMA 15 pin Port 1, etc)

Issues and Proposed Revision
Currently the TIA-485 version of the 15-pin D-sub conflicts with the TS2 Port 1 pinout. This needs to be corrected … but then you may run into a conflict with 2070 pinouts.

Disposition
Comment accepted. See Comment 1-22 disposition.
ID#: 1-27
Commenter: Peter Ragsdale
Comment Type: Technical
Location: Section 6.3.2.1, Page 6-12
Clause: “…without ability to disable the transmitter, unless C50_Enabled is active.”

Issues and Proposed Revision

Based on the text in section 6.3.2.1 any EIA-574 serial port shall be disabled if C50_Enable is active … this was not the intent and only SP4 connections should be disabled including those that are not EIA-574.

If SP4 is not available elsewhere, then suggest that C50_Enable is not required to be implemented on C50. but that a cable that uses it should be compatible.

Disposition

*Replace Section 6.3.2.1 EIA-574, under Description, as follows:*

The EIA-574 versions of the communications interface shall convert the ATC EIA-485 signals to EIA-574 bipolar simplex, meaning each signal is unidirectional and point-to-point.

If the ATC front panel contains a C50S interface with an active (LOW) C50_ENABLED signal such that one or more of the EIA-574 interfaces could be in contention, the C50_ENABLED signal may be used to disable the associated transmitter on the communications interface.
ID#: 1-28

Commenter: Peter Ragsdale

Comment Type: Technical

Location: Section 7.1.1, Page 7-1

Clause: "In this standard, alternative user interfaces may be included provided that the minimal interface is provided also."

Issues and Proposed Revision

C60P should be optional if an integrated keypad/LCD is present.

Submit that the intent was not that both an optional and minimal interfaces be provided, but that an optional interface (such as the 2070-3B) could be provided instead of the minimum. Changes are needed to sections 7.1.1.1 and 7.1.1.2 also.

Reword as "In this standard, one or more optional user interfaces may be included provided that at least one of the minimal user interfaces is provided also."

Disposition

Comment accepted with alternate resolution. See Comment 1-29 disposition.
ID#: 1-29

Commenter: Peter Ragsdale

Comment Type: Technical

Location: Section 7.1.1.1, Page 7-1

Clause: “The minimum user interface to the Application shall consist of the following:”

Issues and Proposed Revision

One should be able to select between one or more alternative User interfaces with conflicts (such as the C60P conflicting with 2070-3B).

Reword as “The minimum user interface to the Application shall consist of either of the following two alternatives:” Next add, “Alternative #1:” ahead of the existing C60P requirement. Lastly, add “Alternative #2:” and the 2070-3x requirements from section 7.1.2.1 (currently option #1).

Disposition

Delete Sections 7.1.1.1, 7.1.1.2, 7.1.2.1 and 7.1.2.2 relating to UI/Application and UI/OS.
Delete Section 7.1.4.3 Infrared Port.

Replace content of Section 7.1.1 Minimum User Interface, with the following:

The minimum user interface shall consist of the following:

- EIA-574 SP4 connector, nine-pin “D” (Guidance: C50S of ATC 2070)
- CPU ACTIVE LED Indicator
- Ethernet Port (Guidance: Internal ATC 10/100 hub Port 2)
- USB Port, for removable memory device only.

- EIA-574 SP6 connector, nine-pin “D” (Guidance: C60P of ATC 2070)
- OR-  
- Keyboard, LCD, Bell (Guidance: traditional ATC 2070)

Replace content of Section 7.1.2 Optional User Interfaces, with the following:

In addition to the minimum user interface, the ATC may include one or more optional user interfaces:

- Data Key (see Section 7.1.4.6)
- AUX Switch (only available for Keyboard, LCD, Bell configuration)
ID#: 1-30
Commenter: Peter Ragsdale
Comment Type: Technical
Location: Section 7.1.1.1, Page 7-2
Clause: “Data Key (Consistent with the requirements of ATC 2070)”

Issues and Proposed Revision
The Datakey should be optional. The (other) suggested changes to section 7.1.1.1 of stating “Alternative” minimum user interfaces partially addresses this idea.

Disposition
Comment accepted. See Comment 1-29 disposition.
ID#: 1-31
Commenter: Peter Ragsdale
Comment Type: Editorial
Location: Section 7.1.1.2, Page 7-2
Clause: "Internal ATC 10/100 hub Port 2"

Issues and Proposed Revision

Remove the word “internal” as this port is on the outside of the ATC (external).

Disposition

Disposed as proposed.
ID#: 1-32
Commenter: Peter Ragsdale
Comment Type: Technical
Location: Section 7.1.2.1, Page 7-2
Clause: “Option 1: Keyboard, LCD, Bell, and AUX Switch (Guidance: traditional ATC 2070)”
“Option 2 & Option 3”

Issues and Proposed Revision

Move the current option #1 to the revised section 7.1.1.1.
Renumber options #2 & #3 as #1 & #2 respectively.

Disposition

Comment accepted with alternate resolution. See Comment 1-29 disposition.
ID#: 1-33
Commenter: Peter Ragsdale
Comment Type: Technical
Location: Section 7.1.2.1, Page 7-2
Clause: “Keyboard, LCD, Bell, and AUX Switch”

**Issues and Proposed Revision**

If User interface Option #1 is selected, then the AUX switch should be optional. (If it is optional to even have a keyboard, etc then the individual components should be optional also).

**Disposition**

*Comment accepted with alternate resolution. See Comment 1-29 disposition.*
ID#: 1-34

Commenter: Peter Ragsdale

Comment Type: Technical

Location: Section 7.1.4.1, Page 7-3

Clause: “The keyboard, at a minimum, shall be capable of the complete single keystroke functionality (without key translations) of the standard ATC 2070 front panel.”

Issues and Proposed Revision

Standard implies that one can supply any keyboard that has a minimum of 28 keys … how do we handle the codes for the extra keys? Please note that several manufacturers have created products that have more than 28 keys (functions).

Disposition

Disposition deferred to a future revision of the standard.
ID#: 1-35

Commenter: Peter Ragsdale

Comment Type: Technical

Location: Section 7.1.4.1, Page 7-3

Clause: “The keyboard, at a minimum, shall be capable of the complete single keystroke functionality (without key translations) of the standard ATC 2070 front panel.”

Issues and Proposed Revision

What does “without key translations” mean? Add text to clarify since a manufacturer can label the keys as they want to.

Disposition

Modify Section 7.1.4.1, Item Keyboard, as follows (in bold):

The keyboard, at a minimum, shall be capable of the complete single keystroke functionality (without key translations) of the standard ATC 2070 front panel. It shall not be necessary to translate or otherwise effect alternate non-standard keypresses in order to provide the keystrokes available on a standard ATC 2070 front panel.
ID#: 1-36

Commenter: Peter Ragsdale

Comment Type: Technical

Location: Section 7.1.4.1, Page 7-3

Clause: “Each key shall be engraved or embossed with its function character.”

Issues and Proposed Revision

Requirement should be that the function of each key should be clearly indicated … one should be able to place the legend for the key function on the chassis either above/below the actual key instead of mandating that it be on the key.

Disposition

Modify Section 7.1.4.1, Item Keyboard, as follows (in bold):

The keyboard, at a minimum, shall be capable of the complete single keystroke functionality (without key translations) of the standard ATC 2070 front panel. Each key shall have a legend which is permanently attached to the ATC, either on the front panel immediately adjacent to the key; or engraved, embossed or otherwise permanently affixed to the key cap. Minimum key surface area shall be 0.09 square inches. Minimum key spacing shall be 0.5” on centers. The actual keypad arrangement is not specified here.
ID#: 1-37

Commenter: Peter Ragsdale

Comment Type: Technical

Location: Section 7.1.4.1, Page 7-3

Clause: "Minimum key size shall be 0.3" x 0.3". Minimum key spacing shall be 0.5" on centers."

Issues and Proposed Revision

Key size should be defined as minimum surface area and minimum separation spacing.

For example, minimum key surface area should be 0.09 square inches, and key separation should be a minimum of 0.05 inches (this value should be checked against the 2070-3x keypads … they do not set the minimum but we cannot exclude them).

Disposition

*Modify Section 7.1.4.1, Item Keyboard, as follows (in bold):*

The keyboard, at a minimum, shall be capable of the complete single keystroke functionality (without key translations) of the standard ATC 2070 front panel. Each key shall be engraved or embossed with its function character. **Minimum key surface area shall be 0.09 square inches.** Minimum key spacing shall be 0.5" on centers. The actual keypad arrangement is not specified here.
ID#: 1-38
Commenter: Peter Ragsdale
Comment Type: Technical
Location: Section 7.1.4.1, Page 7-3
Clause: (n/a)

Issues and Proposed Revision

For consistency purposes, we should add a statement that the CPU_ACTIVE LED should be turned ON following a Power-on Reset.

Disposition

*Replace Section 7.1.4.1, under CPU_ACTIVE LED Indicator, as follows:*

The cathode of the CPU_ACTIVE LED Indicator shall be electrically connected to the CPU_ACTIVE signal. The current-limiting resistor providing power to the CPU_ACTIVE LED Indicator anode shall be located on the front panel.

The CPU_ACTIVE LED Indicator shall be turned ON following a power-on reset.
ID#: 1-39

Commenter: Peter Ragsdale

Comment Type: Technical

Location: Section 7.1.4.1, Page 7-4

Clause: “The User Interface shall be reset once power is applied or have a momentary control reset switch on the PCB that is logic ORed with the CPU_RESET Line, producing a USER Interface RESET.”

Issues and Proposed Revision

Suggest that having a momentary push button on the PCB does not relieve the front panel User Interface from the requirement of resetting itself upon application of power.

Reword as “A USER Interface RESET shall be generated whenever power is applied, upon activation of a momentary control reset switch on the PCB (if it exists), and upon activation of the CPU_RESET Line.”

Disposition

Replace Section 7.1.4.1, under Reset, opening paragraph, as follows:

A user interface RESET shall be generated whenever power is applied, upon activation of a momentary reset switch on the PCB (if it exists), and upon activation of the CPU_RESET line. Following the user interface RESET being active for a minimum of 25 ms, or receipt of a valid Soft Reset display command, the following shall occur:
ID#: 1-40

Commenter: Peter Ragsdale

Comment Type: Technical

Location: Section 7.1.4.1, Page 7-4

Clause: (n/a)

**Issues and Proposed Revision**

For consistency purposes, we should add bullet #8 to the reset list that specifies that the Cursor shall be turned "on" as part of the reset process.

**Disposition**

*Add to Section 7.1.4.1, under Reset, as follows:*

8. The cursor shall be turned ON.
ID#: 1-41

Commenter: Peter Ragsdale

Comment Type: Technical

Location: Section 7.1.4.1, Page 7-6

Clause: “Displayable characters shall be refreshed at least 20 times per second.”

Issues and Proposed Revision

Reduce the LCD refresh rate to 10 Hz, and preferably just for pixels/characters that changed. There is no User need that can demand refresh rate. This still allows for the specified flash (on/off times), but enables one to support larger displays.

Disposition

Disposition deferred to a future revision of the standard.
ID#: 1-42
Commenter: Peter Ragsdale
Comment Type: Technical
Location: Section 7.1.4.1, Page 7-4
Clause: "If two or more keys are depressed simultaneously, no code shall be sent. If a key is depressed while another key is depressed, no additional code shall be sent."

Issues and Proposed Revision

I would like to suggest that multiple key presses be allowed, but that their operation is implementation dependent and potentially not portable. Suggest removing these two sentences from the standard.

Further, we suggest that the double key press of “ESC” + “ENTER” returns 08h (“backspace” key) … this was chosen to “emulate” the concept of the backspace key being pressed when a 2070 is booting up.

Disposition

Modify Section 7.1.4.1, under Key Press, as follows (in bold):

When a key press is detected, the appropriate key code shall be transmitted to SP6 RXD.

If two or more keys are depressed simultaneously, no code shall be sent. If a key is depressed while another key is depressed, no additional code shall be sent. The following sequence shall be transmitted:

| ESC | K P1 P2 | 1B 5B 4B P1 P2 | P1: key one pressed; P2: key two pressed (NOTE: key codes for P1 and/or P2 may be multi-byte per Key Codes table) |

No key code shall be transmitted upon release of one or both keys following a two keypress sequence.

If more than two keys are pressed simultaneously, no code shall be sent.

Add to end of table in Section 7.1.4.1, under Key Codes, as follows:

| Double Key Press (P1; P2) | ESC | K P1 P2 | 1B 5B 4B P1 P2 |
ID#: 1-43

Commenter: Peter Ragsdale

Comment Type: Technical

Location: Section 7.1.4.1, Page 7-8, Table 7-1

Clause: Section 10.1.13.2 on page 10-5 requires “When such heaters are used, they shall only be energized at low temperature to support operator interaction and shall be controllable through the application software.” and yet no method to control them is possible/defined.

Issues and Proposed Revision

Extend “Configuration Command Codes” to include support for TEES 2007 proposed additions to control the display heater:

<table>
<thead>
<tr>
<th>Command</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESC [ ? 12 h</td>
<td>1B 5B 3F 31 32 68</td>
<td>Heater on</td>
</tr>
<tr>
<td>ESC [ ? 12 l</td>
<td>1B 5B 3F 31 32 6C</td>
<td>Heater off</td>
</tr>
</tbody>
</table>

These were included in a previous comment regarding front panel protocol corrections/extensions.

Disposition

Comment accepted. Add records to end of Table 7-1, Configuration Command Codes, as follows:

<table>
<thead>
<tr>
<th>Command</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESC [ ? 12 h</td>
<td>1B 5B 3F 31 32 68</td>
<td>Heater on (subject to temperature constraints)</td>
</tr>
<tr>
<td>ESC [ ? 12 l</td>
<td>1B 5B 3F 31 32 6C</td>
<td>Heater off</td>
</tr>
</tbody>
</table>

Add paragraph to Section 7.1.4.1, under Display, as follows:

The display may optionally contain a heating element for low-temperature operation. This heating element shall be application-controllable using configuration commands for that purpose found in Table 7-1. The front panel shall ignore commands to turn on the heating element should the ambient temperature be high enough for proper operation.
ID#: 1-44

Commenter: Peter Ragsdale

Comment Type: Technical

Location: Section 7.1.4.1, Page 7-8, Table 7-1

Clause: (n/a)

Issues and Proposed Revision

Extend “Inquiry – Command Response Codes” to include support for TEES 2007 proposed additions to inquire about the display heater status and the display type. Also, extend display type to include “H” meaning “Other - Host Board EEPROM contains display type and/or parameters”:

```
ESC [ h n 1B 5B 68 6E
ESC [ P1 R 1B 5B P1 52
P1: Heater (h, l)
```

```
ESC [ c 1B 5B 63
ESC [ P1 R 1B 5B P1 52
P1: Type (A,B,D,H)
```

These were included in a previous comment regarding front panel protocol corrections/extensions.

Disposition

Comment accepted. Add records to end of Table 7-1, Inquiry Command-Response Codes, as follows:

```
ESC [ h n 1B 5B 68 6E
ESC [ P1 R 1B 5B P1 52
P1: Heater (h, l)
```

```
ESC [ c 1B 5B 63
ESC [ P1 R 1B 5B P1 52
P1: Type (A,B,D,H)
```
ID#: 1-45
Commenter: Peter Ragsdale
Comment Type: Technical
Location: Section 7.1.4.1, Page 7-8, Table 7-1
Clause: (n/a)

**Issues and Proposed Revision**

Query the keypad ... we propose the following:

<table>
<thead>
<tr>
<th>ESC [ K n</th>
<th>1B 5B 4B 6E</th>
<th>(normal key code)</th>
<th>(normal key code)</th>
<th>No response if no key is currently pressed</th>
</tr>
</thead>
</table>

On reset, what happens if a key is pressed ... suggest that it must be reported shortly after the power-up string is sent.

**Disposition**

*Comment accepted. Add record to end of Table 7-1, Inquiry Command-Response Codes, as follows:*

<table>
<thead>
<tr>
<th>ESC [ K n</th>
<th>1B 5B 4B 6E</th>
<th>(normal key code)</th>
<th>(normal key code)</th>
<th>No response if no key is currently pressed</th>
</tr>
</thead>
</table>
ID#: 1-46
Commenter: Peter Ragsdale
Comment Type: Technical
Location: Section 7.1.4.1, Pages 7-7 to 7-9
Table 7-1
Clause: (There are numerous errors in the front panel protocol tables)

Issues and Proposed Revision

Please correct the front panel protocol errors.
(see attached material – immediately following – four pages)

Disposition

Correct protocol errors in Table 7-1, as follows (in bold):

<table>
<thead>
<tr>
<th>ASCII REPRESENTATION</th>
<th>HEX VALUE</th>
<th>FUNCTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>HT</td>
<td>09</td>
<td>Move cursor to next tab stop</td>
</tr>
<tr>
<td>CR</td>
<td>0D</td>
<td>Position cursor at first position on current line</td>
</tr>
<tr>
<td>LF</td>
<td>0A</td>
<td>(Line Fee) Move cursor down one line</td>
</tr>
<tr>
<td>BS</td>
<td>08</td>
<td>(Backspace) Move cursor one position to the left and write space</td>
</tr>
<tr>
<td>ESC [ Py j Px f</td>
<td>1B 5B Py 3B Px 66</td>
<td>Position cursor at (Px, Py)</td>
</tr>
<tr>
<td>ESC [ Pn C</td>
<td>1B 5B Pn 43</td>
<td>Position cursor Pn positions to right</td>
</tr>
<tr>
<td>ESC [ Pn D</td>
<td>1B 5B Pn 44</td>
<td>Position cursor Pn positions to left</td>
</tr>
<tr>
<td>ESC [ Pn A</td>
<td>1B 5B Pn 41</td>
<td>Position cursor Pn positions up</td>
</tr>
<tr>
<td>ESC [ Pn B</td>
<td>1B 5B Pn 42</td>
<td>Position cursor Pn positions down</td>
</tr>
<tr>
<td>ESC [ H</td>
<td>1B 5B 48</td>
<td>Home cursor (move to 1,1)</td>
</tr>
<tr>
<td>ESC [ 2 j</td>
<td>1B 5B 32 4A</td>
<td>Clear screen with spaces without moving cursor</td>
</tr>
<tr>
<td>ESC c</td>
<td>1B 63</td>
<td>Soft reset</td>
</tr>
<tr>
<td>ESC P P1 [ Pn j Pn...f</td>
<td>1B 50 P1 5B Pn 3B...Pn 66</td>
<td>Compose special character number Pn (1–8) at current cursor position</td>
</tr>
<tr>
<td>ESC [ &lt; Pn V</td>
<td>1B 5B 3C Pn 56</td>
<td>Display special character number Pn (1–8) at current cursor position</td>
</tr>
<tr>
<td>ESC [ 25 h</td>
<td>1B 5B 32 35 68</td>
<td>Turn Character blink on</td>
</tr>
<tr>
<td>ESC [ 25 j</td>
<td>1B 5B 32 35 6C</td>
<td>Turn Character blink off</td>
</tr>
<tr>
<td>ESC [ &lt; 5 h</td>
<td>1B 5B 3C 35 68</td>
<td>Illuminate Backlight</td>
</tr>
<tr>
<td>ESC [ &lt; 5 l</td>
<td>1B 5B 3C 35 6C</td>
<td>Extinguish Backlight</td>
</tr>
<tr>
<td>ESC [ 33 h</td>
<td>1B 5B 33 33 68</td>
<td>Cursor blink on</td>
</tr>
<tr>
<td>ESC [ 33 l</td>
<td>1B 5B 33 33 6C</td>
<td>Cursor blink off</td>
</tr>
<tr>
<td>ESC [ 27 h</td>
<td>1B 5B 32 37 68</td>
<td>Reverse video on – Note 2</td>
</tr>
<tr>
<td>ESC [ 27 l</td>
<td>1B 5B 32 37 6C</td>
<td>Reverse video off – Note 2</td>
</tr>
<tr>
<td>ASCII REPRESENTATION</td>
<td>HEX VALUE</td>
<td>FUNCTION</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------</td>
<td>----------</td>
</tr>
<tr>
<td>ESC [ 24 h</td>
<td>1B 5B 32 34 68</td>
<td>Underline on – Note 2</td>
</tr>
<tr>
<td>ESC [ 24 l</td>
<td>1B 5B 32 34 6C</td>
<td>Underline off – Note 2</td>
</tr>
<tr>
<td>ESC [ 0 m</td>
<td>1B 5B 30 6D</td>
<td>All attributes off</td>
</tr>
<tr>
<td>ESC H</td>
<td>1B 48</td>
<td>Set tab stop at current cursor position</td>
</tr>
<tr>
<td>ESC [ Pn g</td>
<td>1B 5B Pn 67</td>
<td>Clear tab stop Pn = 0,1,2 at cursor = 3 all tab stops</td>
</tr>
<tr>
<td>ESC [ ? 7 h</td>
<td>1B 5B 3F 37 68</td>
<td>Auto-wrap on</td>
</tr>
<tr>
<td>ESC [ ? 7 l</td>
<td>1B 5B 3F 37 6C</td>
<td>Auto-wrap off</td>
</tr>
<tr>
<td>ESC [ ? 8 h</td>
<td>1B 5B 3F 38 68</td>
<td>Auto-repeat on</td>
</tr>
<tr>
<td>ESC [ ? 8 l</td>
<td>1B 5B 3F 38 6C</td>
<td>Auto-repeat off</td>
</tr>
<tr>
<td>ESC [ ? 25 h</td>
<td>1B 5B 3F 32 35 68</td>
<td>Cursor on</td>
</tr>
<tr>
<td>ESC [ ? 25 l</td>
<td>1B 5B 3F 32 35 6C</td>
<td>Cursor off</td>
</tr>
<tr>
<td>ESC [ &lt; 47 h</td>
<td>1B 5B 3C 34 37 68</td>
<td>Auto-scroll on</td>
</tr>
<tr>
<td>ESC [ &lt; 47 l</td>
<td>1B 5B 3C 34 37 6C</td>
<td>Auto-scroll off</td>
</tr>
<tr>
<td>ESC [ &lt; Pn S</td>
<td>1B 5B 3C Pn 53</td>
<td>Set Backlight timeout value to Pn (0–63)</td>
</tr>
<tr>
<td>ESC [ Pu</td>
<td>1B 5B 50 55</td>
<td>String sent to ENGINE BOARD when EIPA power up</td>
</tr>
</tbody>
</table>

**NOTE:**
1. Numerical values have one ASCII character per digit without leading zero.
2. Reverse video and underline NOT required for integrated front panel displays.

<table>
<thead>
<tr>
<th>COMMAND</th>
<th>RESPONSE</th>
<th>FUNCTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Engine Board to Front Panel Module</td>
<td>Front Panel Module to Engine Board</td>
<td></td>
</tr>
<tr>
<td>ASCII Representation</td>
<td>HEX Value</td>
<td>ASCII Representation</td>
</tr>
<tr>
<td>ESC [ 6 n</td>
<td>1B 5B 36 6E</td>
<td>ESC [ Py; Px R</td>
</tr>
<tr>
<td>ESC [ B n</td>
<td>1B 5B 42 6E</td>
<td>ESC [ P1; P2;....P6 R</td>
</tr>
<tr>
<td>ESC [ A n</td>
<td>1B 5B 41 6E</td>
<td>ESC [ P1 R</td>
</tr>
</tbody>
</table>

All protocol enhancements covered by other comment dispositions.

(Disposition End)
Front Panel Protocol Corrections & Enhancements

Section 7.1.4.1 of the ATC v5.2b standard has numerous errors in the Front Panel protocol tables. The following tables have been corrected and also include required/suggested enhancements.

A summary of the enhancements required of the Front Panel Protocol is as follows:

- Command Heater On: \texttt{ESC} [ ? 12 h
- Command Heater Off: \texttt{ESC} [ ? 12 l
- Query Heater Status: \texttt{ESC} [ h n
  - Response – Heater On: \texttt{ESC} [ h R
  - Response – Heater Off: \texttt{ESC} [ l R
- Double key press of “ESC” + “ENTER” returns 08h (“backspace” key)
- Query Display Type: 
  - Added response code type “H” to means “details in Host Board EEPROM”
- Query Keypad: \texttt{ESC} [ K n
  - Response is normal key code if a key is pressed; otherwise no response

<table>
<thead>
<tr>
<th>ASCII REPRESENTATION</th>
<th>HEX VALUE</th>
<th>FUNCTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>HT</td>
<td>09</td>
<td>Move cursor to next tab stop</td>
</tr>
<tr>
<td>CR</td>
<td>0D</td>
<td>Position cursor at first position on current line</td>
</tr>
<tr>
<td>LF</td>
<td>0A</td>
<td>(Line Feed) Move cursor down one line</td>
</tr>
<tr>
<td>BS</td>
<td>08</td>
<td>(Backspace) Move cursor one position to the left and write space</td>
</tr>
<tr>
<td>\texttt{ESC} [ Py ; Px f</td>
<td>1B 5B Py 3B Px 66</td>
<td>Position cursor at (Px, Py)</td>
</tr>
<tr>
<td>\texttt{ESC} [ Pn C</td>
<td>1B 5B Pn 43</td>
<td>Position cursor Pn positions to right</td>
</tr>
<tr>
<td>\texttt{ESC} [ Pn D</td>
<td>1B 5B Pn 44</td>
<td>Position cursor Pn positions to left</td>
</tr>
<tr>
<td>\texttt{ESC} [ Pn A</td>
<td>1B 5B Pn 41</td>
<td>Position cursor Pn positions up</td>
</tr>
<tr>
<td>\texttt{ESC} [ Pn B</td>
<td>1B 5B Pn 42</td>
<td>Position cursor Pn positions down</td>
</tr>
<tr>
<td>\texttt{ESC} [ H</td>
<td>1B 5B 48</td>
<td>Home cursor (move to 1,1)</td>
</tr>
<tr>
<td>\texttt{ESC} [ 2 J</td>
<td>1B 5B 32 4A</td>
<td>Clear screen with spaces without moving cursor</td>
</tr>
<tr>
<td>\texttt{ESC} c</td>
<td>1B 63</td>
<td>Soft reset</td>
</tr>
<tr>
<td>\texttt{ESC} P P1 [ Pn ; Pn…f</td>
<td>1B 50 P1 5B Pn 3B…Pn 66</td>
<td>Compose special character number Pn (1-8) at current cursor position</td>
</tr>
<tr>
<td>\texttt{ESC} [ &lt; Pn V</td>
<td>1B 5B 3C Pn 56</td>
<td>Display special character number Pn (1-8) at current cursor</td>
</tr>
</tbody>
</table>
### CONFIGURATION COMMAND CODES

<table>
<thead>
<tr>
<th>ASCII REPRESENTATION</th>
<th>HEX VALUE</th>
<th>FUNCTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>position</td>
<td>position</td>
<td></td>
</tr>
<tr>
<td>ESC [ 25 h</td>
<td>1B 5B 32 35 68</td>
<td>Turn Character blink on</td>
</tr>
<tr>
<td>ESC [ 25 l</td>
<td>1B 5B 32 35 6C</td>
<td>Turn Character blink off</td>
</tr>
<tr>
<td>ESC [ &lt; 5 h</td>
<td>1B 5B 3C 35 68</td>
<td>Illuminate Backlight</td>
</tr>
<tr>
<td>ESC [ &lt; 5 l</td>
<td>1B 5B 3C 35 6C</td>
<td>Extinguish Backlight</td>
</tr>
<tr>
<td>ESC [ 33 h</td>
<td>1B 5B 33 33 68</td>
<td>Cursor blink on</td>
</tr>
<tr>
<td>ESC [ 33 l</td>
<td>1B 5B 33 33 6C</td>
<td>Cursor blink off</td>
</tr>
<tr>
<td>ESC [ 27 h</td>
<td>1B 5B 32 37 68</td>
<td>Reverse video on – Note 2</td>
</tr>
<tr>
<td>ESC [ 27 l</td>
<td>1B 5B 32 37 6C</td>
<td>Reverse video off – Note 2</td>
</tr>
<tr>
<td>ESC [ 24 h</td>
<td>1B 5B 32 34 68</td>
<td>Underline on – Note 2</td>
</tr>
<tr>
<td>ESC [ 24 l</td>
<td>1B 5B 32 34 6C</td>
<td>Underline off – Note 2</td>
</tr>
<tr>
<td>ESC [ 0 m</td>
<td>1B 5B 30 6D</td>
<td>All attributes off</td>
</tr>
<tr>
<td>ESC H</td>
<td>1B 48</td>
<td>Set tab stop at current cursor position</td>
</tr>
<tr>
<td>ESC [ Pn g</td>
<td>1B 5B Pn 67</td>
<td>Pn = 0: Clear tab stop at Current Position,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Pn = 3: Clear All Tabs</td>
</tr>
<tr>
<td>ESC [ ? 7 h</td>
<td>1B 5B 3F 37 68</td>
<td>Auto-wrap on</td>
</tr>
<tr>
<td>ESC [ ? 7 l</td>
<td>1B 5B 3F 37 6C</td>
<td>Auto-wrap off</td>
</tr>
<tr>
<td>ESC [ ? 8 h</td>
<td>1B 5B 3F 38 68</td>
<td>Auto-repeat on</td>
</tr>
<tr>
<td>ESC [ ? 8 l</td>
<td>1B 5B 3F 38 6C</td>
<td>Auto-repeat off</td>
</tr>
<tr>
<td>ESC [ ? 12 h</td>
<td>1B 5B 3F 31 32 68</td>
<td>Heater on</td>
</tr>
<tr>
<td>ESC [ ? 12 l</td>
<td>1B 5B 3F 31 32 6C</td>
<td>Heater off</td>
</tr>
<tr>
<td>ESC [ ? 25 h</td>
<td>1B 5B 3F 32 35 68</td>
<td>Cursor on</td>
</tr>
<tr>
<td>ESC [ ? 25 l</td>
<td>1B 5B 3F 32 35 6C</td>
<td>Cursor off</td>
</tr>
<tr>
<td>ESC [ &lt; 47 h</td>
<td>1B 5B 3C 34 37 68</td>
<td>Auto-scroll on</td>
</tr>
<tr>
<td>ESC [ &lt; 47 l</td>
<td>1B 5B 3C 34 37 6C</td>
<td>Auto-scroll off</td>
</tr>
<tr>
<td>ESC [ &lt; Pn S</td>
<td>1B 5B 3C Pn 53</td>
<td>Set Backlight timeout value to Pn (0-63)</td>
</tr>
<tr>
<td>ESC [ PU</td>
<td>1B 5B 50 55</td>
<td>String sent to ENGINE BOARD when EIPA power up</td>
</tr>
</tbody>
</table>

**NOTE:**
1. Numerical values have one ASCII character per digit without leading zero.
2. Reverse Video & Underline are not required for integrated Front Panel displays.
## INQUIRY COMMAND – RESPONSE CODES

<table>
<thead>
<tr>
<th>COMMAND</th>
<th>RESPONSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Engine Board to Front Panel Module</td>
<td>Front Panel Module to Engine Board</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ASCII Representation</th>
<th>HEX Value</th>
<th>ASCII Representation</th>
<th>HEX Value</th>
<th>FUNCTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESC [ 6 n</td>
<td>1B 5B 36 6E</td>
<td>ESC [ Py; Px R</td>
<td>1B 5B Py 3B Px 52</td>
<td>Inquire Cursor Position</td>
</tr>
<tr>
<td>ESC [ B n</td>
<td>1B 5B 42 6E</td>
<td>ESC [ P1; P2;…P6 R</td>
<td>1B 5B P1 3B P2 3B…P6 52</td>
<td>Status Cursor Position</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ESC [ A n</td>
<td>1B 5B 41 6E</td>
<td>ESC [ P1 R</td>
<td>1B 5B P1 52</td>
<td>P1: AUX Switch (h, l)</td>
</tr>
<tr>
<td>ESC [ h n</td>
<td>1B 5B 68 6E</td>
<td>ESC [ P1 R</td>
<td>1B 5B P1 52</td>
<td>P1: Heater (h, l)</td>
</tr>
<tr>
<td>ESC [ e</td>
<td>1B 5B 63</td>
<td>ESC [ P1 R</td>
<td>1B 5B P1 52</td>
<td>P1: Type (A,B,D,H)</td>
</tr>
<tr>
<td>ESC [ K n</td>
<td>1B 5B 4B 6E</td>
<td>(normal key code)</td>
<td>(normal key code)</td>
<td>No response if no key is currently pressed</td>
</tr>
</tbody>
</table>
### Key Codes

<table>
<thead>
<tr>
<th>Key</th>
<th>ASCII DATA (TEXT)</th>
<th>ASCII DATA (HEX)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>30</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>31</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>32</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>33</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>34</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>35</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>36</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>37</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>38</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>39</td>
</tr>
<tr>
<td>A</td>
<td>A</td>
<td>41</td>
</tr>
<tr>
<td>B</td>
<td>B</td>
<td>42</td>
</tr>
<tr>
<td>C</td>
<td>C</td>
<td>43</td>
</tr>
<tr>
<td>D</td>
<td>D</td>
<td>44</td>
</tr>
<tr>
<td>E</td>
<td>E</td>
<td>45</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>46</td>
</tr>
<tr>
<td>(UP ARROW)</td>
<td>ESC [ A</td>
<td>1B 5B 41</td>
</tr>
<tr>
<td>(DOWN ARROW)</td>
<td>ESC [ B</td>
<td>1B 5B 42</td>
</tr>
<tr>
<td>(RIGHT ARROW)</td>
<td>ESC [ C</td>
<td>1B 5B 43</td>
</tr>
<tr>
<td>(LEFT ARROW)</td>
<td>ESC [ D</td>
<td>1B 5B 44</td>
</tr>
<tr>
<td>ESC</td>
<td>ESC O S</td>
<td>1B 4F 53</td>
</tr>
<tr>
<td>NEXT</td>
<td>ESC O P</td>
<td>1B 4F 50</td>
</tr>
<tr>
<td>YES</td>
<td>ESC O Q</td>
<td>1B 4F 51</td>
</tr>
<tr>
<td>NO</td>
<td>ESC O R</td>
<td>1B 4F 52</td>
</tr>
<tr>
<td>*</td>
<td>*</td>
<td>2A</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>2B</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>2D</td>
</tr>
<tr>
<td>ENTER</td>
<td>CR</td>
<td>0D</td>
</tr>
<tr>
<td>ESC + ENTER</td>
<td>BS</td>
<td>08</td>
</tr>
</tbody>
</table>

### AUX Switch Codes

<table>
<thead>
<tr>
<th>SWITCH POSITION</th>
<th>ASCII DATA (TEXT)</th>
<th>ASCII DATA (HEX)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ON</td>
<td>ESC O T</td>
<td>1B 4F 54</td>
</tr>
<tr>
<td>OFF</td>
<td>ESC O U</td>
<td>1B 4F 55</td>
</tr>
</tbody>
</table>
ID#: 1-47

Commenter: Peter Ragsdale

Comment Type: Technical

Location: Section 7.1.4.2, Page 7-10

Clause: "Intent is that heater is controlled by front panel activity only."

Issues and Proposed Revision

This contradicts section 10.1.13.2 which states that application program will control Heater. As such the proposed extensions to the front panel protocol regarding the heater control and status are required.

Disposition

Comment accepted. See Comment 1-43 disposition.
ID#: 1-48
Commenter: Peter Ragsdale
Comment Type: Technical
Location: Section 7.2, Page 7-16
Clause: (implied)

Issues and Proposed Revision
Power Supply LEDs, On/Off Switch, and Fuse should not be required (implied) to be on the power supply but rather anywhere within the ATC.

Either add a new line to section 7.2 stating that they can be anywhere, or add an appropriate statement in each of sections 7.2.1, 7.2.2, and 7.2.3.

Disposition
Replace Sections 7.2, 7.2.1 and 7.2.2 with the following text:

7.2 Power Supply General Description

The power supply shall consist of one or more subassemblies that can be removed and reinstalled using standard hand tools. All electrical connections to the power supply subassembly(s) shall be through a connector(s) rated to handle required power supply voltage, current and signal levels defined in Section 7.2. It shall be cooled by convection only. The power supply shall be capable of supporting the internal ATC circuitry, plus provide power for each optional module. The power supply shall convert service voltage to the proper DC voltages at the power rating needed to support the unit and any external power as described in paragraph 7.2.6.

The power supply must produce all output voltages with the specified tolerances and capacities within 750 ms after the application of external power to the ATC. The power supply must also raise the POWERUP and POWERDOWN signals to a HIGH state, indicating that power is stable and available, within this same 750 ms time period.

7.2.1 "On/Off" Power Switch (Optional)

If provided, an "On/Off" Power switch shall disconnect the AC line of the service voltage from the power supply. The "ON" position of the Power Switch shall be clearly labeled on the switch or the panel to which the switch is mounted.

Guidance: The Power Switch is not required to be located on the Power Supply subassembly(s).
7.2.2 LED DC Power Indicators (Optional)

If provided, LED DC power indicators shall indicate that all DC voltages generated by the power supply meet the following conditions:

a) That the +5 VDC is within 4.8V to 5.25V and that the ± 12 VDC voltages are within ±8% of nominal
b) For 332 Parallel I/O versions, that the +12 VDC ISO is within ±8% of nominal
c) For NEMA versions, that the +24 VDC is within the NEMA TS-2 tolerance for +24VDC

Guidance: The LED DC Power Indicators are not required to be located on the Power Supply subassembly(s).
ID#: 1-49

Commenter: Peter Ragsdale

Comment Type: Technical

Location: Section 7.2.2, Page 7-16

Clause: “Four LED DC power indicators shall be provided …”

Issues and Proposed Revision

ATC v5.2b requires 4 LEDs for power supply status in all cases; however, a NEMA TS2 Type 1 only has 3 voltages (+24VDC does not exist).

Modify the text to state that a LED must be provided to indicate the status of each voltage created by the power supply.

Disposition

Comment accepted. See Comment 1-48 disposition.
ID#: 1-50
Commenter: Peter Ragsdale
Comment Type: Technical
Location: Section 7.2.6.5, Page 7-23
Clause: “The power supply shall supply +5 VDC current budget for 550 ms after power loss at 100 VAC. The supply shall be capable of holding up the ATC for two 500 ms power loss periods occurring in a 1.5-second period at 100 VAC. Since the engine board is powered completely by +5 VDC, no other power supply output voltages shall need to be maintained during power loss to prevent reboot.”

Issues and Proposed Revision

Clarify / correct the Power Supply hold-up requirements such that the ATC provides hold up on I/O voltages also (e.g. NEMA requires this). Need to discuss whether communication slot voltages are also required (e.g. +/- 12VDC).

Disposition

*Modify Section 7.2.6.5 as follows (in bold):*

The power supply shall supply sufficient current budget for all internal voltages, except for display heater and backlight, for 550 ms after power loss at the minimum line voltage specified in section 7.2.6.1 (i.e. 100 VAC or 89 VAC). The supply shall be capable of holding up the ATC for two 500ms power loss periods occurring in a 1.5-second period at the minimum line voltage specified in section 7.2.6.1. All USB port(s) and communication(s) slot shall be able to constantly draw their maximum current for each voltage as specified in section 7.2.6 during the outages. The ATC power supply shall recover sufficiently such that this power loss test can be repeated continuously once every 60 seconds over the full operating temperature range. Note that the I/O voltage used for internal purposes (e.g. pull-up resistors) must be maintained throughout the outages.
ID#: 1-51

Commenter: Peter Ragsdale

Comment Type: Technical

Location: Section 7.2.6.5, Page 7-23, Lines 21-22

Clause: “The supply shall be capable of holding up the ATC for two 500 ms power loss periods occurring in a 1.5-second period at 100 VAC.”

Issues and Proposed Revision

Suggest that we enhance the AC loss test to be repetitive pulses over a longer period of time.

Propose new wording of:

“The supply shall be capable of holding up the ATC when continuously subjected to 375 ms power loss periods followed by 550 ms supply of 100 VAC.”

Disposition

Disposition deferred to a future revision of the standard.
ID#: 1-52

Commenter: Peter Ragsdale

Comment Type: Technical

Location: Section 8.1.1, Page 8-1

Clause: “The parallel I/O connects the ATC to transportation cabinets including, but not limited to, …”

Issues and Proposed Revision

Enhance the text in section 8.1.1 that allows other parallel connections to be more clear in that they can be provided (but how?). For example, certain agencies have specific parallel connectors within their cabinets that they require to be used and they want to use ATC controllers. Section 8.1.1 states that such cabinets are authorized, but how the parallel connections are supported is missing (e.g. adding new I/O module response codes).

Disposition

*Disposition deferred to a future revision of the standard.*
ID#: 1-53

Commenter: Peter Ragsdale

Comment Type: Technical

Location: Section 8.2.1.13, Page 8-18

Clause: “The PI/O identification command frame shall be used to request the PI/O identification value response of "1" for the Model 332 PI/O, "2" for the NEMA TS-2 Type 2 PI/O, “3” for the NEMA TS-2 Type 1 PI/O. The identification value response for ITS cabinet SIUs and CMU shall be frame address.”

Issues and Proposed Revision

For additional / proprietary I/O modules, who will maintain and assign the response codes for section 8.2.1.13? I suggest that an Annex be created and that manufacturers submit requests for “codes” and they be assigned and included in the next version of the standard.

How do we handle extensions to a module? For example, if a manufacturer extends a module’s capabilities should they use the same code or be assigned a new one?

Looking forward to ITS Cabinet V2, how are we going to handle the situation of the SIU/AMU/CMU replacements (if any)? What about concepts from the NEMA Next Generation Cabinet?

Disposition

Disposition deferred to a future revision of the standard.
ID#: 1-54

Commenter: Peter Ragsdale

Comment Type: Technical

Location: Section 8.2.2.4, Page 8-20

Clause: “O78 shall normally change its state every 100 ms.”

Issues and Proposed Revision

The actual operation of deployed products (2070-8) do not appear to be behave as described in this section. Please review section 8.2.2.4 in its entirety and correct. Coordination with Caltrans re TEES and the ATC2070 are also required.

Disposition

No problem found.
ID#: 1-55

Commenter: Peter Ragsdale

Comment Type: Technical

Location: Section 8.2.3, Page 8-22

Clause: “…Fault Monitor Logic Output and Output Frame Byte 9 Bit 6 to …”

Issues and Proposed Revision

It was my understanding that the Fault Monitor Logic Output was the same as the Output Frame Byte 9 Bit 6 … please correct/clarify. Also, please change “Output Frame Byte 9 Bit 6” to be the proper output number (e.g. O78 or O79) similar to the representation used in section 8.2.2.4.

Disposition

*Modify Section 8.2.3 Connection to NEMA TS-2 Type 1 Cabinets, under Description, as follows:*  

The TS-2 Type 1 PI/O provides a TS2-1 compatible interface, AC Power to the ATC, and Fault Monitor Logic Output and Output Frame Byte 9 Bit 6 to the NEMA TS2 Cabinet Monitor Unit (CMU).

*Modify Section 8.2.3 Connection to NEMA TS-2 Type 1 Cabinets, under Fault Monitor, as follows (in bold):*

An FCU output **O78** shall drive an open collector transistor whose output shall be routed to Connector A Pin F for use as a FAULT MONITOR Output. The transistor shall be capable of sinking 200 mA at 30 VDC.
ID#: 1-56
Commenter: Peter Ragsdale
Comment Type: Editorial
Location: Section 8.2.3, Page 8-22
Clause: “...NEMA TS2 Cabinet Monitor Unit (CMU)”

Issues and Proposed Revision
NEMA TS2 does not have a CMU ... please change to “Malfunction Management Unit (MMU)”

Disposition
Disposed as proposed.
ID#: 1-57
Commenter: Peter Ragsdale
Comment Type: Editorial
Location: Section 8.2.3, Page 8-22, Line 29
Clause: “Portport”

Issues and Proposed Revision

Typo … change to “Port”. Also in the line 27 change “port 1” to “Port 1.”

Disposition

Disposed as proposed.
ID#: 1-58
Commenter: Peter Ragsdale
Comment Type: Technical
Location: Section 8.2.3, Page 8-22
Clause: “An FCU output shall drive an open collector transistor whose output shall be routed to Connector A Pin F for use as a FAULT MONITOR Output.”

Issues and Proposed Revision
The FCU output needs to be clarified (what it is, how it works, how one controls it, etc).

Disposition
See Comment 1-55 disposition.
ID#: 1-59

Commenter: Peter Ragsdale

Comment Type: Technical

Location: Section 8.2.3, Page 8-22

Clause: “An FCU output shall drive an open collector transistor whose output shall be routed to Connector A Pin F for use as a FAULT MONITOR Output.”

Issues and Proposed Revision

A previous comment regarding a “suggestion that we define the TS2 FM pin as being connected to the SPx_RTS pin” is officially retracted … although this comment makes sense technically and financially, there are ATC units that have implemented this and it will create incompatibilities if changed at this point (and also with the 2070-2N).

Disposition

Comment withdrawn by commenter.
ID#: 1-60
Commenter: Peter Ragsdale
Comment Type: Editorial
Location: Section 8.3, Page 8-25
Clause: “Port 7 provides a permanent connection for internal ATC expansion …”

Issues and Proposed Revision
Port 7 is defined elsewhere as an “external” port. Change text to “Port 7 provides a permanent connection for external ATC expansion …”

Disposition
Disposition deferred to a future revision of the standard.
ID#: 1-61
Commenter: Peter Ragsdale
Comment Type: Technical
Location: Section 7.1.3, Page 7-3
Section 8.4, Page 8-26
Clause: "DCGND3 is the ground reference for all signals on front panel SP4 and SP6 connectors and the LCD Heater Power Supply."

"Isolation is not required when the serial port is connected to another integrated ATC assembly. For example, an integrated front panel is considered to be part of the ATC, meaning the SP6 connection to the front panel device need not be electrically isolated. However, a removal front panel or SP6 external connector such as C60P is not considered to be part of the ATC and isolation is required."

"SP4 (C50S), SP6 (C60P), and the LCD heater supply must be isolated from the engine board; however, they do not have to isolated from each other."

Issues and Proposed Revision

There is potential confusion possible regarding whether or not the LCD heater needs to be isolated from the engine board in cases where the front panel is integrated into the ATC. In an integrated unit, the processor on SP6 is controlling the heater (on/off) and should have the same ground reference as there is no added protection offered by an isolated heater.

We suggest that if the front panel is integrated (and not removable), then both SP6 and any LCD Heater do not have to be isolated however any SP4 connector would require isolation. Furthermore, if the front panel is not integrated (i.e. removable or connected to a connector on the ATC) then isolation is required for SP6, the LCD Heater (if present), and SP4.

Minor text changes are required to the identified sections to provide clarity.

Disposition

Modify Section 7.1.3 User Interface Pin Connections, under Guidance, as follows (in bold):

2. DCGND3 is the ground reference for all externally-accessible signals on front panel SP4 and SP6 connectors and the LCD Heater Power Supply. If the front panel is integrated, SP6 signals and the LCD heater are not required to be isolated and may use the engine board ground reference.

(continued next page)
Modify Section 8.4 Isolation Requirements, as follows (in bold):

- Isolation is not required when the serial port is connected to another integrated ATC assembly. For example, an integrated front panel is considered to be part of the ATC, meaning the SP6 connection to the front panel device and any LCD heater installed need not be electrically isolated from the engine board. However, a removal front panel or SP6 external connector such as C60P is not considered to be part of the ATC and isolation from the engine board is required for all externally-accessible signals.

- SP4 (C50S), SP6 (C60P), and the LCD heater supply must be isolated from the engine board; however, they do not have to isolated from each other.

- SP4 (C50S) must always be isolated from the engine board; however, for non-integrated front panels, SP4 may share the same isolated ground reference as SP6 and any LCD heater.
ID#: 1-62
Commenter: Peter Ragsdale
Comment Type: Technical
Location: Section 10
Clause: (n/a)

Issues and Proposed Revision

Although practically a de facto User Requirement, do we want to specifically state that horizontal PCBs are not permitted?

Disposition

Modify Section 10.1.15 as follows (in bold):

No components, traces, brackets, or obstructions shall be within 0.125 inch of a PC board edge (guide edges). The manufacturer's name or logo, model number, serial number and circuit issue or revision number shall appear and be readily visible on all PCBs. Devices to prevent the PCB from backing out of its assembly connectors shall be provided. All screw type fasteners shall utilize locking devices or locking compounds except for finger screws, which shall be captive. Solder quality should conform to IPC 610 specification for Industrial ratings. Serial numbers on PCBs shall be permanent. All PCBs shall be mounted such that their component surface plane is vertically oriented when the ATC is mounted in its normal operating position.
ID#: 1-63
Commenter: Peter Ragsdale
Comment Type: Technical
Location: Section 10.1.10, Page 10-3
Clause: “Fuse size rating shall be labeled on the holder.”

Issues and Proposed Revision
Suggest that fuse rating only needs to be clearly displayed on the ATC unit ... it could be on the chassis near the fuseholder or on the fuseholder (probably custom).

Disposition
Replace Section 10.1.10 Fuses, as follows:

10.1.10 Single-Use Over-Current Protection Device
All single-use over-current protection devices (OCPD) shall be resident in a holder which is readily accessible. Single-use OCPDs shall not require any tools other than a screwdriver for their replacement. A label indicating the OCPD rating shall be located next to the OCPD holder.

Fuses shall not be dislodged during shipping and handling.

Replace Section 7.2.3 Service Voltage Fuse, as follows:

7.2.3 Service Voltage Over-Current Protection Device
An over-current protection device (OCPD) shall be provided to protect from over-current on the AC Line side of the service voltage connection to the ATC power supply. The Service Voltage OCPD shall be accessible from the front of the unit.

Guidance: The Service Voltage OCPD is not required to be located on the power supply subassembly(s).
ID#: 1-64
Commenter: Peter Ragsdale
Comment Type: Technical
Location: Section 10.1.10, Page 10-3
Clause: “All fuses shall be 3AG slow blow type and resident in a holder. ”

Issues and Proposed Revision
What about allowing other fuse types and sizes? As it is, there is no compatibility between units guaranteed at all (e.g. Model 1 could use 1A and Model B uses 3A … even if from the same manufacturer).

Disposition
Comment approved. See Comment 1-63 disposition.
ID#: 1-65
Commenter: Peter Ragsdale
Comment Type: Technical
Location: Annex A, Page A-1, Line 11
Section 2.2.5, Page 2-6, Line 2
Clause: Linux is version 2.6.9+
BusyBox version is 1.00p8

Issues and Proposed Revision

Increase the minimum versions of both BusyBox and Linux due to significant errors found and corrected. Suggest that BusyBox be increased to v1.10+ and Linux be changed to 2.6.14+.

Disposition

Comment accepted. See comments 1-4 and 1-67.
ID#: 1-66

Commenter: Peter Ragsdale

Comment Type: Technical

Location: Annex A, Page A-1

Clause: “This Annex specifies the minimum Linux kernel configuration features that shall be included in all kernel builds and the minimum required set of utilities.”

Issues and Proposed Revision

Doug Tarico has pointed out that the newer versions of Linux configurator are not compatible with this Annex. The text needs to be enhanced to state that the “equivalent functionality to that specified” shall be included.

Disposition

*Replace Annex A, opening paragraph, as follows:*

This standard requires a Linux-compatible 2.6.x kernel (where x >= 18).

Platform-specific or Linux version-specific configuration options shall be selected by the manufacturer based on satisfying the functional requirements implied below.

The following items or their equivalent shall be the minimum Linux kernel configuration features that are included in the kernel build. A manufacturer may include additional configuration options at their discretion.

*Replace Annex A, under Kernel Configuration, as follows:*

```bash
# # General setup #
CONFIG_SYSVIPC=y
CONFIG_IKCONFIG=y
CONFIG_IKCONFIG_PROC=y
CONFIG_EMBEDDED=y
CONFIG_SYSCTL=y
CONFIG_HOTPLUG=y
CONFIG_ELF_CORE=y
#
# Loadable module support #
CONFIG_MODULES=y
CONFIG_MODULE_UNLOAD=y
CONFIG_KMDD=y
#
# Processor type and features #
CONFIG_PREEMPT=y
```
# Executable file formats
# CONFIG_BINFORM_ELF=y
# CONFIG_BINFORM_AOUT=m
#
# Networking options
#
# CONFIG_PACKET=y
# CONFIG_PACKET_MMAP=y
# CONFIG_USER=y
# CONFIG_INET=y
# CONFIG_INET_MULTICAST=y
# CONFIG_INET_PNP=y
# CONFIG_INET_PNP_DHCP=y
# CONFIG_INET_PNP_BOOTP=y
# CONFIG_SYN_COOKIE=y
#
# 1P: Virtual Server Configuration
#
# CONFIG_NETFILTER=y
#
# Block devices
#
# CONFIG_BLK_DEV_LOOP=y
# CONFIG_BLK_DEV_RAM=y
# CONFIG_BLK_DEV_INITRD=y
#
# SCSI device support
#
# CONFIG_SCSI=y
# CONFIG_SCSI_PROC_FS=y
#
# SCSI support type (disk, tape, CD-ROM)
# CONFIG_BLK_DEV_SD=y
#
# Ethernet (10 or 100Mbit)
# CONFIG_NET_Ethernet=y
#
# Non-8250 serial port support
#
# CONFIG_UNIX98_PTYS=y
# CONFIG_LEGACY_PTYS=y
# CONFIG_LEGACYPTY_COUNT=256
#
# USB support
#
# CONFIG_USB=y
#
# Miscellaneous USB options
#
# CONFIG_USB_DEVI_CEFS=y
#
# NOTE: USB_STORAGE enables SCSI, and 'SCSI disk support'
# may also be needed; see USB_STORAGE Help for more information
# CONFIG_USB_STORAGE=y
# CONFIG_USB_STORAGE_FREECOM=y
CONFIG_USB_STORAGE_SD200=y
CONFIG_USB_STORAGE_DPCM=y
#
# File systems
#
CONFIG_EXT2_FS=y
#
# DOS/FAT/NT Filesystems
#
CONFIG_FAT_FS=y
CONFIG_MSDOS_FS=y
CONFIG_VFAT_FS=y
CONFIG_FAT_DEFAULT_CODEPAGE=437
CONFIG_FAT_DEFAULT_CODECCHARSET="iso8859-1"
#
# Pseudo filesystems
#
CONFIG_PROC_FS=y
# CONFIG_PROC_KCORE is not set
CONFIG_SYSFS=y
CONFIG_TMPFS=y
CONFIG_RAMFS=y
#
# Network File Systems
#
CONFIG_NFS_FS=y
CONFIG_NFS_V3=y
CONFIG_NFSD=y
CONFIG_NFSD_V3=y
CONFIG_NFSD_TCP=y
CONFIG_ROOT_NFS=y
CONFIG_LOCKD=y
CONFIG_LOCKD_V4=y
CONFIG_EXPORTFS=y
CONFIG_NFS_COMMON=y
CONFIG_SUNRPC=y
#
# Native Language Support
#
CONFIG_NLS_DEFAULT="iso8859-1"
#
# Kernel hacking
#
# CONFIG_PRENTK_TIME is not set
# CONFIG_MAGIC_SYSRQ is not set
# CONFIG_UNUSED_SYMBOLS is not set
# CONFIG_DEBUG_KERNEL is not set
# CONFIG_DEBUG_VERBOSE is not set
# CONFIG_DEBUG_FS is not set
# CONFIG_UNWIND_INFO is not set
# CONFIG_DOUBLEFAULT is not set
#
# Security options
#
# CONFIG_KEYS is not set
# CONFIG_SECURITY is not set
#
# Cryptographic options
#
CONFIG_CRYPTO=y
CONFIG_CRYPTO_HMAC=y
CONFIG_CRYPTO_MD5=y
#
# Library routines
#
CONFIG_CRC32=y
CONFIG_ZLIB_INFLATE=y
CONFIG_ZLIB_DEFLATE=y
ID#: 1-67
Commenter: Peter Ragsdale
Comment Type: Technical
Location: Annex A, Page A-1
Clause: “Currently, only a Linux compatible kernel in the range of 2.6.x, (where x >= 9)...”

Issues and Proposed Revision
Due to major changes in Linux, the minimum of level of kernel should be increased to 2.6.14 (i.e. x>=14).

Disposition
Modify Annex A, Paragraph 1, last line:

This standard requires a Linux-compatible 2.6.x kernel (where x >= 18).
ID#: 1-68

Commenter: Peter Ragsdale

Comment Type: Technical

Location: Annex B.1, Page B-1

Clause: “The BSP shall provide an external function, void atc_cpu_reset (void) …”

Issues and Proposed Revision

Where is “atc_cpu_reset”? Should it be a library or include (macro)? From my previous e-mail to the ATC list, I suggest that we create a standard library named, libatc.so, that will contain all BSP functions defined now or in the future. Although this may seem like overkill right now (there will be only one function in it), it is well suited to the future when other functions might be defined and included.

Disposition

Add table row to 2.2.5 Operating System & Board Support Package Requirements, as follows:

<table>
<thead>
<tr>
<th>manufacturer-specific</th>
<th>libatc.so</th>
<th>ATC-specific library functions (atc_cpu_reset)</th>
</tr>
</thead>
</table>

Add to Annex B, Section B.1 ATC CPU_RESET, under Overview, as follows:

The function atc_cpu_reset shall be provided with the BSP within a library named libatc.so.
ID#: 1-69
Commenter: Peter Ragsdale
Comment Type: Technical
Location: Annex B.2
Clause: (n/a)

**Issues and Proposed Revision**

Recommend updated GPIO driver model.

(see attached material – immediately following – one page)

**Disposition**

*Replace Annex B.2, subsection Overview, with the following:*

**Overview**

The PIO Driver Interface allows the user to control the CPU Active LED, determine whether or not the Datakey is inserted, reset peripheral devices via the CPU Reset signal, and read the Powerdown signal status.

**Supported Device File Operations:**

- `open()`;
- `close()`;
- `read()`;
- `write()`;

**open()**

The following dev entries shall exist:

/dev/datakeypresent
/dev/cpuactive
/dev/powerdown
/dev/cpureset

**close()**

Closes the file descriptor.

**read()**

```
int read(int filp, void *buf, int count);
```

This allows for reading the state of the Powerdown signal and for determining whether or not the Datakey is inserted. The value passed in the count parameter must be 1 or no bytes will be read.
write()
int write(int filp, void *buf, int count);

Allows changing the state of the CPU Active LED and the CPU Reset signal.

Writing a single nonzero character to the /dev/cpuactive device shall turn on the CPU active LED. Writing a zero to the device will turn off the LED.

(Disposition End)
GPIO Driver

The GPIO driver allows the user to control the CPU active LED, determine if the Datakey is present, reset peripheral devices, and power down peripheral devices.

Supported Device File Operations:
- `open();`
- `close();`
- `read();`
- `write();`

`open()`

The following dev entries shall exist:
- `/dev/datakeypresent`
- `/dev/cpuactive`
- `/dev/powerdown`
- `/dev/cpureset`

`close()`

Closes the file descriptor.

`read()`

```c
int read(int filp, void *buf, int count);
```

This allows for reading the state of the power down pin and for reading the state of whether the Datakey is inserted. The value passed in the count parameter must be 1 or no bytes will be read.

`write()`

```c
int write(int filp, void *buf, int count);
```

Allows changing the state of the CPU Active LED and the CPU reset signal.

Writing a single nonzero character to the `/dev/cpuactive` device shall turn on the CPU active LED and writing zero will turn off the LED.
ID#: 1-70
Commenter: Peter Ragsdale
Comment Type: Technical
Location: Annex B.3
Clause:

Issues and Proposed Revision

Correct SPI driver errors and add enhancements.
Correct Host EEPROM errors and add extensions.

(see attached material – immediately following – two pages)

Disposition

Replace Section B.3 in it’s entirety with the following:

B.3 ATC Serial Peripheral Interface (SPI) Devices

B.3.1 SPI EEPROM Driver

The following standard Linux functions shall be supported by this driver:

- `open()`
- `close()`
- `read()`
- `write()`
- `lseek()`

The special device file node for the host EEPROM shall be “/dev/eeprom”.

B.3.2 Host Board EEPROM Content and Organization

The following describes the content and organization of the Host EEPROM. If a Host EEPROM does not exist within the ATC, then application software should assume the default configuration which represents an ATC 2070L. The data format in the Host EEPROM for the Latitude and Longitude fields shall comply with IEEE/ANSI 754-1985 STD. All the other fields shall follow a Big Endian Format as implemented in Freescale 68000-family CPUs.

- Host EEPROM Version (uint8 0..255; Standard enumerated values are: v5.2b – 1; v6 – 2)
- Host EEPROM Size in bytes (uint16 0..65535; Use Defaults = 0)
- # Modules (uint8 0..255). For each module:
  - Module Location (uint8 0..255; Standard enumerated values are: Other – 1, Host – 2, Display – 3, I/O – 4, Power Supply – 5, A1 – 6, A2 – 7)
Module Make (uint8 0..255; manufacturer – NEMA assigned NTCIP code except Generic = 0)
Module Model (OCTET STRING)
Module Version (seven bytes: YYYYMMDDrrmmpp … where YYYYMMDD is date & version is rr.mm.pp … stored in BCD)
Module Type (uint8 0..255; Standard enumerated values are: Other – 1, Hardware – 2, Software/Firmware – 3)

Display properties:
- # Char Lines (uint8 0..255 - use 0 if no display attached)
- # Char Columns (uint8 0..255 - use 0 if no display attached)
- # Graphic Rows-1 (y_max) (uint8 0..255 - use 0 if no display attached)
- # Graphic Columns-1 (x_max) (uint16 0..1023 - use 0 if no display attached)

Ethernets (uint8 0..255). For each Ethernet:
- Type (uint8 0..255; Standard enumerated values are: Other - 1, Hub/Phy/Other Direct Port - 2, Unmanaged Switch - 3, Managed Switch - 4, Router - 5)
- IP Address (four bytes)
- Switch/Router MAC Address (six bytes); use 000000 000000 if Hub/Phy/Other Direct Port
- Subnet Mask (four bytes)
- Default Gateway (four bytes)
- Engine Board Interface (integer 0..255; Standard enumerated values are: Other - 1, Phy – 2, RMII – 3, MII – 4)

SPI3 Purpose (uint8 0..255; Standard enumerated values are: Unused – 0, reserved 1..255)

SPI4 Purpose (uint8 0..255; Standard enumerated values are: Unused – 0, otherwise manufacturer specific code 1..255)

Host Board Serial Ports Used – uint16 0..65535 bit encoded as follows:
- Bit 0: SP1
- Bit 1: SP1s
- Bit 2: SP2
- Bit 3: SP2s
- Bit 4: SP3
- Bit 5: SP3s
- Bit 6: SP4
- Bit 7: SP5s
- Bit 8: SP6
- Bit 9: SP8 (although always present in an ATC, it is optional in an ATC2070)
- Bit 10: SP8s (although always present in an ATC, it is optional in an ATC 2070)

Host Board Serial Ports Present – uint16 0..65535 bit encoded as follows:
- Bit 0: SP1
- Bit 1: SP1s
- Bit 2: SP2
- Bit 3: SP2s
- Bit 4: SP3
• Bit 5: SP3s
• Bit 6: SP4
• Bit 7: SP5s
• Bit 8: SP6
• Bit 9: SP8 (although always present in an ATC, it is optional in an ATC 2070)
• Bit 10: SP8s (although always present in an ATC, it is optional in an ATC 2070)

- Serial Bus #1 Port – uint16 0..65535 bit encoded as follows (0 = SB #1 non-existent/unused):
  - Bit 0: Reserved
  - Bit 1: SP1s
  - Bit 2: Reserved
  - Bit 3: SP2s
  - Bit 4: Reserved
  - Bit 5: SP3s
  - Bit 6: Reserved
  - Bit 7: SP5s
  - Bit 8: Reserved
  - Bit 9: Reserved
  - Bit 10: SP8s (although always present in an ATC, it is optional in an ATC 2070)

- Serial Bus #2 Port – uint16 0..65535 bit encoded as follows (0 = SB #2 non-existent/unused):
  - Bit 0: SP1
  - Bit 1: SP1s
  - Bit 2: SP2
  - Bit 3: SP2s
  - Bit 4: SP3
  - Bit 5: SP3s
  - Bit 6: SP4
  - Bit 7: SP5s
  - Bit 8: SP6
  - Bit 9: SP8 (although always present in an ATC, it is optional in an ATC 2070)
  - Bit 10: SP8s (although always present in an ATC, it is optional in an ATC 2070)

- TS 2 Port 1 Port – uint16 0..65535 bit encoded as follows (0 = NEMA Port 1 non-existent):
  - Bit 0: Reserved
  - Bit 1: SP1s
  - Bit 2: Reserved
  - Bit 3: SP2s
  - Bit 4: Reserved
  - Bit 5: SP3s
  - Bit 6: Reserved
  - Bit 7: SP5s
  - Bit 8: Reserved
  - Bit 9: Reserved
  - Bit 10: SP8s (although always present in an ATC, it is optional in an ATC 2070)

- Expansion Bus type (uint8 0..255; Standard enumerated values are: None – 1, Other – 2, VME – 3)
- Encoded SPI Addressing (uint8 0..255; Standard enumerated values are: No – 1, Yes – 2)
- CRC #1 (16-bit)
- Latitude (4 bytes)
- Longitude (4 bytes)
• Controller ID (uint16 0..65535)
• Communication Drop # (uint16 0..65535)
• Reserved for Agency (35 bytes … per TEES)
• CRC #2 (8-bit)
• User Data (to end)

The default configuration is as follows:
• Host EEPROM Version  = 2
• Host EEPROM Size in bytes  = 0
• # Modules = 4  (Note: 2070L is 2070-1B, 2070-2A, 2070-3B, & 2070-4B only)
  o Module 1 Location = 2
  o Module 1 Make = 0
  o Module 1 Model = “2070-1B Host”
  o Module 1 Version = 20040608020200
  o Module 1 Type = 2
  o Module 2 Location = 4
  o Module 2 Make = 0
  o Module 2 Model = “2070-2A”
  o Module 2 Version = 20040608020200
  o Module 2 Type = 2
  o Module 3 Location = 3
  o Module 3 Make = 0
  o Module 3 Model = “2070-3B”
  o Module 3 Version = 20040608020200
  o Module 3 Type = 2
  o Module 4 Location = 5
  o Module 4 Make = 0
  o Module 4 Model = “2070-4B”
  o Module 4 Version = 20040608020200
  o Module 4 Type = 2

• Display properties:
  o # Char Lines = 8
  o #Char Columns = 40
  o #Graphic Rows-1 (y_max) = 63
  o #Graphic Columns-1 (x_max) = 239

• # Ethernets = 1
  o Ethernet 1 Type = 2
  o Ethernet 1 IP Address = 10.20.70.51
  o Ethernet 1 Switch/Router MAC Address = 000000 000000
  o Ethernet 1 Subnet Mask = 255.255.255.0
  o Ethernet 1 Default Gateway = 10.20.70.254
  o Ethernet 1 Engine Board Interface = 2

• SPI3 Purpose = 0
• SPI4 Purpose = 0
• Host Board Serial Ports Used = 2047:
• # Ports used for I/O = 1:
  o Port 1 ID = 5
  o Port 1 Mode = 3
  o Port 1 Baud Rate = 614400

• Serial Bus #1 Port = 128
• Serial Bus #2 Port = 0
• TS 2 Port 1 Port = 0
• Expansion Bus type = 1
• Encoded SPI Addressing = 1 (No)
• CRC #1 (16-bit) ... n/a, simulate if required
• Latitude = 0.0 ... or read from Datakey if present
• Longitude = 0.0 ... or read from Datakey if present
• Controller ID = 65535 ... or read from Datakey if present
• Communication Drop # = 65535 ... or read from Datakey if present
• Reserved for Agency (35 bytes of 0xFF) ... or read from Datakey if present
• CRC #2 (8-bit) ... n/a, simulate if required ... or read from Datakey if present
• User Data (to end, each 0xFF) ... n/a, or read from Datakey if present

B.3.3 SPI Datakey Driver

The following standard Linux functions shall be supported by this driver:

• open() 
• close( ) 
• read ( ) 
• write( ) 
• lseek() 
• ioctl()

The special device file node for the Datakey shall be “/dev/datakey”.

The ioctl function prototype is as follows:

```c
int ioctl(int d, int request, ...);
```

The following ioctl requests shall be supported and are defined in the common atc.h header file. IOCTL requests are defined according to Linux conventions as described in ”Documentation/ioctl/ioctl-decoding.txt”.

**ATC_DATAKEY_ERASE_ALL**

Perform a Bulk Erase operation, which sets all bits to 1.

**ATC_DATAKEY_ERASE_SECTOR**

Perform a Sector Erase operation for the sector containing the address passed as a long integer parameter.

**ATC_DATAKEY_READ_STATUS_BITS**

Returns the contents of the device Status Register in the least significant byte of the value returned.

**ATC_DATAKEY_WRITE_PROTECT_BITS**
Sets the Block Protect Bits in the device Status Register to the value contained in the least significant bits of the value passed as a long integer parameter.

**ATC_DATAKEY_GET_DEVICE_SIZE**

Returns the total size of the device in bytes.

**ATC_DATAKEY_GET_SECTOR_SIZE**

Returns the size of one sector in bytes.

### B.3.4 Datakey Header Format

The format of the first 28 bytes of the datakey is defined in the following structure. Only datakeys with \texttt{version} = 1 or 2 are currently defined. A non-defined value for \texttt{version} or \texttt{type} is an error. Bytes 29 to 63 are reserved for extension of the header format and shall contain the value 0xFF if unused. All additional memory following the 64th byte is undefined and available for application use.

```c
typedef struct atc_datakey {
    uint16 fcs; // 16 bit Frame Check Sequence (FCS) calculated as defined in clause 4.6.2 of ISO/IEC 3309. This FCS is calculated across bytes 3-64
    uint8 type; // Key Type See table below
    uint8 version; // Header Version: Only 1 and 2 are valid values currently.
    uint32 latitude;
    uint32 longitude;
    unit16 id; // Controller ID
    unit16 drop; // Communications drop number
    uint32 ipaddress; // IP address
    uint32 subnet; // Subnet mask
    uint32 gateway; // Default gateway
} ;
```

Structure member “\texttt{type}” shall contain the Key Type value as defined in the following table:

<table>
<thead>
<tr>
<th>Key Type</th>
<th>Model No</th>
<th>Memory Size</th>
<th>Sector Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>SFK2M</td>
<td>2Mb</td>
<td>64 Kbytes</td>
</tr>
<tr>
<td>4</td>
<td>SFK4M</td>
<td>4Mb</td>
<td>64 Kbytes</td>
</tr>
<tr>
<td>5</td>
<td>SFK8M</td>
<td>8Mb</td>
<td>64 Kbytes</td>
</tr>
<tr>
<td>6</td>
<td>SFK32M</td>
<td>32Mb</td>
<td>64 Kbytes</td>
</tr>
</tbody>
</table>

The data format in the Datakey header for the Latitude and Longitude fields shall comply with IEEE/ANSI 754-1985 STD. All the other fields shall follow a Big Endian Format as implemented by Freescale 68000-family CPUs.

(Disposition End)
SPI / EEPROM Driver

The EEPROM driver provides full capability for reading and writing to EEPROM.

Supported Device File Operations:

- open();
- close();
- read();
- write();
- lseek();
- ioctl();

open()

The dev entry for the host EEPROM shall be /dev/eeprom. The EEPROM can be opened for Read, Write, or Read/Write.

Examples:

- fd = open("/dev/eeprom", O_RDONLY);
- fd = open("/dev/eeprom", O_WRONLY);
- fd = open("/dev/eeprom", O_RDWR);

close()

Closes the file descriptor.

read()

int read(int filp, void *buf, int count);

Reads up to count bytes into buf and returns the number of bytes read. The read occurs at the current position within the device.

Note: The current position can be determined using the ioctl. The current position can be changed using the lseek() function.

Possible Errors:

EIO if end of file condition has already been reached
write()
int write(int filp, void *buf, int count);

Writes count bytes to the device at the current file position within the device. If all of the bytes specified by count cannot be written before the end of the device no bytes shall be written and an error shall be returned. The number of bytes written shall be returned. If the value returned is less than count, then the returned value of bytes were written correctly, but the remaining bytes contain errors. In this case it is necessary to try the write again for the remaining bytes or repeat the same write again until the number of bytes returned matches the count.

Possible Errors:
   EIO if end of file condition would occur writing the number of bytes specified.

lseek()
lseek(int fd, int pos, int type);

Seeks to a specified position in the device. Both absolute and relative types of seeking are supported. If relative seeking is specified the pos value may be positive or negative. If absolute seeking is specified the file position is assigned to the pos value. If seeking outside the device size is attempted an error is returned and no change to the file position takes place.

Type Definitions:
   ATC_EEPROM_SEEK_ABS
   ATC_EEPROM_SEEK_REL

ioctl()
ioctl(int fd, unsigned int cmd, unsigned long param);

The ioctl function supports multiple different commands, each described separately.

Command Definitions:
   ATC_EEPROM_GET_FILE_POS
   ATC_EEPROM_GET_DEVICE_SIZE

ATC_EEPROM_GET_FILE_POS

Returns the current file position. The param value is ignored.

ATC_EEPROM_GET_DEVICE_SIZE

Returns the size of the EEPROM device in bytes. The param value is ignored.
ID#: 1-71
Commenter: Peter Ragsdale
Comment Type: Technical
Location: Annex B.3.2, Page B-5
Clause: (The Datakey driver is minimal and should be extended to be practical given the size of the Datakey.)

**Issues and Proposed Revision**

Add BSP extensions to support easier use of the Datakey.

(see attached material – immediately following – three pages)

**Disposition**

See Comment 1-70 disposition.

(Disposition End)
Datakey Driver

This driver provides full capability for manipulating Datakey devices. Datakeys of sizes 131072, 262144, 524288, and 1047576 bytes are supported.

Supported Device File Operations:
- `open()`;
- `close()`;
- `read()`;
- `write()`;
- `lseek()`;
- `ioctl()`;

`open()`

The dev entry for the host EEPROM shall be /dev/datakey. The EEPROM can be opened for Read, Write, or Read/Write.

Examples:
- `fd = open("/dev/datakey", O_RDONLY);`
- `fd = open("/dev/datakey", O_WRONLY);`
- `fd = open("/dev/datakey", O_RDWR);`

`close()`

Closes the file descriptor.

`read()`

`int read(int filp, void *buf, int count);`

Reads up to count bytes into buf and returns the number of bytes read. The read occurs at the current position within the device.

Note: The current position can be determined using the ioctl. The current position can be changed using the lseek() function.

Possible Errors:
- ENXIO if Datakey is not present
- EBUSY if the signature changes
- EIO if end of file condition has already been reached
write()
int write(int filp, void *buf, int count);

Writes count bytes to the device at the current file position within the device. If all of the bytes specified by count cannot be written before the end of the device no bytes shall be written and an error shall be returned. The number of bytes written shall be returned. If the value returned is less than count, then the returned value of bytes were written correctly, but the remaining bytes contain errors. In this case it is necessary to try the write again for the remaining bytes or repeat the same write again until the number of bytes returned matches the count.

Possible Errors:
- ENXIO if Datakey is not present
- EBUSY if the device signature changes (ie. Someone switched devices really fast)
- EIO if end of file condition would occur writing the number of bytes specified.

lseek()
lseek(int fd, int pos, int type);

Seeks to a specified position in the device. Both absolute and relative types of seeking are supported. If relative seeking is specified the pos value may be positive or negative. If absolute seeking is specified the file position is assigned to the pos value. If seeking outside the device size is attempted an error is returned and no change to the file position takes place.

Type Definitions:
- ATC_DATAKEY_SEEK_ABS
- ATC_DATAKEY_SEEK_REL

ioctl()
ioctl(int fd, unsigned int cmd, unsigned long param);

The ioctl function supports multiple different commands, each described separately. If the Datakey is not inserted ENXIO is returned as the error code.

Command Definitions:
- ATC_DATAKEY_GET_FILE_POS
- ATC_DATAKEY_ERASE_ALL
- ATC_DATAKEY_ERASE_SECTOR
- ATC_DATAKEY_READ_PROTECT_BITS
- ATC_DATAKEY_WRITE_PROTECT_BITS
- ATC_DATAKEY_GET_DEVICE_SIZE
- ATC_DATAKEY_GET_SECTOR_SIZE

ATC_DATAKEY_GET_FILE_POS

Returns the current file position. The param value is ignored.
ATC_DATAKEY_ERASE_ALL

Erases all data on the Datakey. The param value is ignored. The CPU active light blinks with high frequency during erasure. Always returns 0.

Note: When data is erased, all values are read as 0xFF.

ATC_DATAKEY_ERASE_SECTOR

Erases all data in the sector containing the address specified by param. The CPU active light blinks at high frequency during erasure. Returns 0 on success or EINVAL on invalid address. The sector size can be determined using the appropriate ioctl() in order to know what address ranges will be erased by this command.

ATC_DATAKEY_READ_PROTECT_BITS

Returns the value of the protect bits directly read from the Datakey. The data format will be in accordance with the datasheet for the Datakey being used (not the same for different device sizes). This function is provided so the user can ensure that the device is not protected. The param value is ignored.

ATC_DATAKEY_WRITE_PROTECT_BITS

Writes the value specified in param directly to the Datakey protection byte. The data format varies in accordance with the datasheet for the Datakey being used. This function is provided primarily so that the user can remove protection if writing is being prevented.

ATC_DATAKEY_GETDEVICE_SIZE

Returns the size of the Datakey device in bytes. The param value is ignored.

ATC_DATAKEY_GETSECTOR_SIZE

Returns the sector size of the Datakey in bytes. The param value is ignored.
ID#: 1-72

Commenter: Peter Ragsdale

Comment Type: Technical

Location: Annex B.3.3, Page B-6

Clause: "If a Host EEPROM does not exist within the ATC, then the BSP should assume the default configuration which represents an ATC 2070L."

Issues and Proposed Revision

Correct standard to state that if the Host EEPROM does not exist then the application assumes default 2070L configuration.

Disposition

Comment accepted. Annex B.3 modified accordingly.
ID#: 1-73
Commenter: Peter Ragsdale
Comment Type: Technical
Location: Annex B.4.4, Page B-10
Clause: “Standard Linux drivers required to implement a ramfs file system …”

Issues and Proposed Revision

In order to support the intent of battery-backed RAM, we need to change the type of SRAM driver that is to be provided (type is currently wrong … it needs to be a Linux compatible system on top of an MTD block).

Disposition

*Modify Annex B, Section B.4.4 Static RAM, as follows* *(in bold)*:

Standard Linux drivers required to implement an **MTD-type persistent filesystem mounted as /sram** on the +5 VDC Standby Power-backed static RAM shall be provided.
ID#: 1-74

Commenter: Peter Ragsdale

Comment Type: Technical

Location: Annex B.5.4, Page B-20, Line 21

Clause: “};”

**Issues and Proposed Revision**

To be "useful", a C typedef is usually of the form:

```c
typedef struct name {
    {list of variables};
} name_t;
```

We need to fix the typedef of "atc_spzx_config" as there is no "name" assigned to it for use by the application programs. Suggest changing line #21 to: "} struct atc_spzx_config_t;"

**Disposition**

Comment accepted. Replace struct in Annex B.5.4 as follows:

```c
typedef struct atc_spzx_config{
    uint8 protocol; // set the port protocol (ATC_SDLC, ATC_SYNC, ATC_HDLC)
    uint8 baud; // set the port baud rate (ATC_B1200, ATC_B4800, ATC_B9600, ATC_B19200, ATC_B38400, ATC_B57600, ATC_B76800, ATC_115200, ATC_B153600, ATC_B614400)
    uint8 transmit_clock_source; // (ATC_CLK_INTERNAL, ATC_CLK_EXTERNAL)
    uint8 transmit_clock_mode; // sets whether the sync transmit clock is on continuously or bursts with the data frame (ATC_CONTINUOUS, ATC_BURST)
} atc_spzx_config_t;
```
ID#: 1-75
Commenter: Peter Ragsdale
Comment Type: Technical
Location: Annex B.5.5, Pages B-20, B-21
Clause: // Available baud rates
#define ATC_B1200 1200
#define ATC_B2400 2400
#define ATC_B4800 4800
#define ATC_B9600 9600
#define ATC_B19200 19200
#define ATC_B38400 38400
#define ATC_B57600 57600
#define ATC_B76800 76800
#define ATC_B115200 115200
#define ATC_B153600 153600 // NEMA default
#define ATC_B614400 614400 // ITS default

Issues and Proposed Revision

We need to fix the baud rate defines so that all of the values can fit within the “uint8 baud” field defined in the atc_spxs_config struct (typedef) stated in section 5.5.4. We suggest an enumeration be used as follows:

#define ATC_B1200 0
#define ATC_B2400 1
#define ATC_B4800 2
#define ATC_B9600 3
#define ATC_B19200 4
#define ATC_B38400 5
#define ATC_B57600 6
#define ATC_B76800 7
#define ATC_B115200 8
#define ATC_B153600 9
#define ATC_B614400 10

Furthermore, so that an application can determine the baud when reading the serial port structure, we propose adding the following to atc_spxs.h also:

const int ATC_B[] = {1200, 2400, 4800, 9600, 19200, 38400, 57600, 76800, 115200, 153600, 614400};

Disposition

Comment accepted. Modify Annex B.5.5 as recommended:
ID#: 1-76

Commenter: Dave Miller

Comment Type: Technical

Location: Annex B.5.5

Clause: (header files and BSP components)

**Issues and Proposed Revision**

Vendor shall verify that all header files and BSP components provided match the ATC v5.2b standard.

Recommend changing the standard to use 32-bit int instead of 8-bit int.

**Disposition**

*Comment rejected. Size should be based on the need of the object. Acceptance would break existing deployments.*
ID#: 1-77

Commenter: Peter Ragsdale

Comment Type: Technical

Location: Annex B

Clause:

Issues and Proposed Revision

Suggest new header files be adapted for consistency in application development.

(see attached material – immediately following – three pages)

Disposition

Update content of Section B.5.5 as follows:

These constants shall reside in a file named ATC_SPXS.H.

```
#ifndef __ATC_SPXS_H
#define __ATC_SPXS_H

#define ATC_SPXS_WRITE_CONFIG 0
#define ATC_SPXS_READ_CONFIG 1

#define ATC_SDLC 0
#define ATC_SYNC 1
#define ATC_HDLC 2

#define ATC_B1200 0
#define ATC_B2400 1
#define ATC_B4800 2
#define ATC_B9600 3
#define ATC_B19200 4
#define ATC_B38400 5
#define ATC_B57600 6
#define ATC_B76800 7
#define ATC_B115200 8
#define ATC_B153600 9
#define ATC_B614400 10

const int ATC_B[] = { 1200, 2400, 4800, 9600, 19200, 38400,
                     57600, 76800, 115200, 153600, 614400 };

#define ATC_CLK_INTERNAL 0
#define ATC_CLK_EXTERNAL 1
#define ATC_GATED 0
```

```c
#define ATC_CONTINUOUS 1

typedef struct atc_spcx_config {
    unsigned char protocol;
    unsigned char baud;
    unsigned char transmit_clock_source;
    unsigned char transmit_clock_mode;
} atc_spxs_config_t;

#endif

Add new subsection B.8 to Annex B, defining ATC.H, as follows:

B.8 General ATC Include File Definitions

These constants shall reside in a file named ATC.H.

#ifndef __ATC_H
#define __ATC_H

// Device File Names
#define ATC_HOST_EEPROM_DEV "/dev/eeprom"
#define ATC_DATAKEY_DEV "/dev/datakey"
#define ATC_GPIO_POWERDOWN_DEV "/dev/powerdown"
#define ATC_GPIO_DATAKEY_DEV "/dev/datakeypresent"
#define ATC_GPIO_CPUACTIVE_DEV "/dev/cpuactive"
#define ATC_GPIO_CPURESET_DEV "/dev/cpureset"
#define ATC_TIMING_TOD_DEV "/dev/tod"

#define ATC_SP1 "/dev/sp1"
#define ATC_SP2 "/dev/sp2"
#define ATC_SP3 "/dev/sp3"
#define ATC_SP4 "/dev/sp4"
#define ATC_SP5 "/dev/sp5"
#define ATC_SP6 "/dev/sp6"
#define ATC_SP8 "/dev/sp8"

#define ATC_SP1S "/dev/sp1s"
#define ATC_SP2S "/dev/sp2s"
#define ATC_SP3S "/dev/sp3s"
#define ATC_SP5S "/dev/sp5s"
#define ATC_SP8S "/dev/sp8s"

// Datakey Driver Definitions
#define ATC_DATAKEY_ERASE_ALL _IO('D', 1)
#define ATC_DATAKEY_ERASE_SECTOR _IOW('D', 2, unsigned long)
#define ATC_DATAKEY_READ_PROTECT_BITS _IOR('D', 3, unsigned long)
#define ATC_DATAKEY_WRITE_PROTECT_BITS _IOW('D', 4, unsigned long)
#define ATC_DATAKEY_GET_DEVICE_SIZE _IOR('D', 5, unsigned long)
#define ATC_DATAKEY_GET_SECTOR_SIZE _IOR('D', 6, unsigned long)

// Time of Day Driver Definitions
```
#define ATC_TOD_SET_TIMESRC _IOW('T', 1, unsigned long)
#define ATC_TOD_GET_TIMESRC _IO('T', 2)
#define ATC_TOD_GET_INPUT_FREQ _IO('T', 3)
#define ATC_TOD_REQUEST_TICK_SIG _IOW('T', 4, unsigned long)
#define ATC_TOD_CANCEL_TICK_SIG _IO('T', 5)
#define ATC_TOD_REQUEST_ONCHANGE_SIG _IOW('T', 6, unsigned long)
#define ATC_TOD_CANCEL_ONCHANGE_SIG _IO('T', 7)

timesrc_enum
{
    ATC_TIMESRC_LINESYNC,
    ATC_TIMESRC_RTCSQWR,
    ATC_TIMESRC_CRYSTAL,
    ATC_TIMESRC_EXTERNAL1,
    ATC_TIMESRC_EXTERNAL2
};

#endif

(Disposition End) ----------------------------------------------------------------------------------------------------------
The content of atc_spxs.h is displayed on this page.

```c
#ifndef __ATC_SPXS_H
#define __ATC_SPXS_H

#define ATC_SPXS_WRITE_CONFIG   0
#define ATC_SPXS_READ_CONFIG    1
#define ATC_SDLC                0
#define ATC_SYNC                1
#define ATC_HDLC                2
#define ATC_B1200               0
#define ATC_B2400               1
#define ATC_B4800               2
#define ATC_B9600               3
#define ATC_B19200              4
#define ATC_B38400              5
#define ATC_B57600              6
#define ATC_B76800              7
#define ATC_B115200             8
#define ATC_B153600             9
#define ATC_B614400             10

const int ATC_B[] = { 1200, 2400, 4800, 9600, 19200, 38400,
                     57600, 76800, 115200, 153600, 614400 };

#define ATC_CLK_INTERNAL        0
#define ATC_CLK_EXTERNAL        1
#define ATC_GATED               0
#define ATC_CONTINUOUS          1

typedef struct atc_spcx_config {
    unsigned char protocol;
    unsigned char baud;
    unsigned char transmit_clock_source;
    unsigned char transmit_clock_mode;
} atc_spxs_config_t;

#endif
```
The content of atc.h is displayed on the following two pages.

```c
#ifndef __ATC_H
#define __ATC_H

// Device File Names
#define ATC_HOST_EEPROM_DEV    "/dev/eeprom"
#define ATC_ENGINE_EEPROM_DEV   "/dev/engine_eeprom"
#define ATC_DATAKEY_DEV        "/dev/datakey"
#define ATC_GPIO_POWERDOWN_DEV  "/dev/powerdown"
#define ATC_GPIO_DATAKEY_DEV    "/dev/datakeypresent"
#define ATC_GPIO_CPUACTIVE_DEV  "/dev/cpuactive"
#define ATC_GPIO_CPURESET_DEV   "/dev/cpureset"
#define ATC_TIMING_TOD_DEV      "/dev/tod"
#define ATC_TIMING_TIMERS_DEV   "/dev/timers"

#define ATC_SP1     "/dev/sp1"
#define ATC_SP2     "/dev/sp2"
#define ATC_SP3     "/dev/sp3"
#define ATC_SP4     "/dev/sp4"
#define ATC_SP5     "/dev/sp5"
#define ATC_SP6     "/dev/sp6"
#define ATC_SP8     "/dev/sp8"

#define ATC_SP1S    "/dev/sp1s"
#define ATC_SP2S    "/dev/sp2s"
#define ATC_SP3S    "/dev/sp3s"
#define ATC_SP4S    "/dev/sp4s"
#define ATC_SP5S    "/dev/sp5s"
#define ATC_SP6S    "/dev/sp6s"
#define ATC_SP8S    "/dev/sp8s"

// DATAKEY IOCTL CONSTANTS
#define ATC_DATAKEY_GET_FILE_POS 3
#define ATC_DATAKEY_ERASE_ALL 6
#define ATC_DATAKEY_ERASE_SECTOR 7
#define ATC_DATAKEY_READ_PROTECT_BITS 8
#define ATC_DATAKEY_WRITE_PROTECT_BITS 9
#define ATC_DATAKEY_GET_DEVICE_SIZE 10
#define ATC_DATAKEY_GET_SECTOR_SIZE 11

// DATAKEY LSEEK CONSTANTS
#define ATC_DATAKEY_SEEK_REL 0
#define ATC_DATAKEY_SEEK_ABS 1

// EEPROM IOCTL CONSTANTS
#define ATC_EEPROM_GET_FILE_POS 3
#define ATC_EEPROM_GETDEVICE_SIZE 10

// EEPROM LSEEK CONSTANTS
#define ATC_EEPROM_SEEK_REL 0
#define ATC_EEPROM_SEEK_ABS 1
```

// Time of Day driver Definitions
#define ATC_TOD_SET_TIMESRC 1
#define ATC_TOD_GET_TIMESRC 2
#define ATC_TOD_GET_INPUT_FREQ 3
#define ATC_TOD_REQUEST_TICK_SIGNAL 5
#define ATC_TOD_CANCEL_TICK_SIGNAL 6
#define ATC_TOD_DST_ENABLE 10
#define ATC_TOD_DST_DISABLE 11
#define ATC_TOD_DST_SETINFO 12
#define ATC_TOD_DST_GETINFO 13

// TIMING Driver Definitions
#define ATC_TIMER_GET_STATUS 0x1C
#define ATC_TIMER_STATE_NULL 0x0000 /* timer is first initialized */
#define ATC_TIMER_STATE_SIG 0x1000 /* one-shot signal is pending */
#define ATC_TIMER_STATE_CYC 0x1001 /* periodic signal is pending */
#define ATC_TIMER_STATE_START 0x1002 /* free running */
#define ATC_TIMER_STATE_STOP 0x1003 /* not active */
#define ATC_TIMER_STATE_RESET 0x1004 /* timer is reset */

typedef struct
{
    unsigned int value;
    unsigned int mode;
    unsigned int signal;
    unsigned int period;
} Timer_status;

typedef struct
{
    unsigned int code;
    unsigned int param1;

    union
    {
        unsigned int param;
        void __user *pointer;
    } param2;
} timing_params_t;

#define ATC_TIMER_SET_PARAMS 0
#define ATC_TIMER_GET_PARAMS 1

#define ATC_SET_TIMESRC 1
#define ATC_TIMESRC_LINESYNC 0
#define ATC_TIMESRC_RTCSQWR 1
#endif
ID#: 1-78

Commenter: Peter Ragsdale

Comment Type: Technical

Location: Annex B

Clause: (There are no Timers defined in the ATC standard minimal requirements.)

Issues and Proposed Revision

Add “Timer” extensions to the BSP similar to TEES.

(see attached material – immediately following – three pages)

Disposition

Comment rejected. Standard to require Linux hrtimers (with a maximum minimum resolution of 100 us) and POSIX 1003.1b timer API support. See Comment 1-4 disposition.
Timer Driver

This driver provides an abstraction for controlling up to 16 timers with 100us resolution simultaneously. A timer can be used to send a one-shot or periodic signal to a process. A timer can be used in a free running mode where the timer is either restarted (stopped and cleared), started (running), or stopped. When the timer device node is opened, a timer is assigned automatically to the caller if one is available, thus eliminating the need for user applications to know which timers the other applications are using to avoid collisions.

Supported Device File Operations:
- open();
- close();
- read();
- ioctl();

open()

The dev entry for the timer driver shall be /dev/timers. When the device is opened, a timer is automatically assigned to the caller if there is one available, otherwise an error is returned to the caller.

close()

Closes the file descriptor and reinitializes the timer, making it available to be reused.

read()

A call to read with a count of at least 4 bytes will read a binary 32-bit unsigned integer containing the current value of the open timer.

ioctl()

ioctl(int fd, unsigned int cmd, unsigned long param);

This ioctl passes a parameter structure for the parameters. The structure used is defined as follows:

typedef struct {
    u32 code;
    u32 param1;
    union {
        u32 param;
        void *pointer;
    } param2;
} timing_params_t;
The ioctl supports getting and setting a timer status structure defined as follows:

```c
typedef struct {
    u32 value;
    u32 mode;
    u32 signal;
    u32 period;
} Timer_status;
```

**Command Definitions:**
- **ATC_TIMER_GET_PARAMS**
- **ATC_TIMER_SET_PARAMS**

**ATC_TIMER_GET_PARAMS**

When this command is issued, a timing_params_t shall be passed as the parameter. The params.code value shall be set to ATC_TIMER_GET_STATUS, params.param2.pointer shall point to a Timer_status structure, and params.param1 shall be the number of bytes allocated for the Timer_status structure. The current timer status shall be copied into the location at params.param2.pointer or an error will be returned if an invalid length or invalid pointer was passed to the ioctl.

**ATC_TIMER_SET_PARAMS**

This function sets the mode of the timer based on the parameter code value in the structure of type timing_params_t that is passed as the parameter. As an example in the explanation of the parameter code definitions, the following variable will be used: timing_params_t params;

**Parameter Code Definitions:**
- **ATC_TIMER_STATE_SIG**
- **ATC_TIMER_STATE_CYC**
- **ATC_TIMER_STATE_START**
- **ATC_TIMER_STATE_STOP**
- **ATC_TIMER_STATE_RESET**

**ATC_TIMER_STATE_SIG**

This command sends a one-time signal to the caller process after a specified time.

Example to set up a one-time signal to be sent after 1/10 of a second:

```c
params.code = ATC_TIMER_STATE_SIG;
params.param1 = SIGALRM;  // signal code
params.param2 = 1000;  // 1/10 of a second period
ioctl(fd, ATC_TIMER_SET_PARAMS, &params);
```

**ATC_TIMER_STATE_CYC**

This command sets up a one-shot signal to be sent to the caller process after a specified time. While in this mode, the current timer value can be read at any time by calling the read() function.
Example to set up cyclical signal to occur every 1/10 of a second:

```c
params.code = ATC_TIMER_STATE_CYC;
params.param1 = SIGALRM; // signal code
params.param2 = 1000; // 1/10 of a second period
ioctl(fd, ATC_TIMER_SET_PARAMS, &params);
```

**ATC_TIMER_STATE_START**

This command starts the timer without clearing its value. The timer value will be incremented every 100us. The current value can be read by calling read() or by calling the ioctl with command ATC_TIMER_GET_PARAMS, and read the period member of the Timer_status structure.

Example:

```c
params.code = ATC_TIMER_STATE_START;
ioctl(fd, ATC_TIMER_SET_PARAMS, &params);
```

**ATC_TIMER_STATE_STOP**

This command stops the timer without clearing its value. The current value can still be read while the timer is stopped.

Example:

```c
params.code = ATC_TIMER_STATE_STOP;
ioctl(fd, ATC_TIMER_SET_PARAMS, &params);
```

**ATC_TIMER_STATE_RESET**

This command stops the timer and resets the timer value. The timer value will read as 0 when in reset state.

Example:

```c
params.code = ATC_TIMER_STATE_RESET;
ioctl(fd, ATC_TIMER_SET_PARAMS, &params);
```
ID#: 1-79
Commenter: Peter Ragsdale
Comment Type: Technical
Location: Annex B
Clause: (n/a)

Issues and Proposed Revision

Add the ability to load files, o/s, boot loader, auto run scripts, etc upon USB insertion or power up with USB inserted in a standardized manner.

QTC has defined a directory structure, etc to support this (and we will provide it if desired).

Disposition

Add the following paragraph to Annex B, at the end of Section B.4.1 (USB):

During boot-up, the ATC shall check for the presence of an executable binary or POSIX shell script named /media/usb/startup on an attached USB device. If found, the ATC shall automatically execute that file. This automatic startup from USB shall only be performed if the device is inserted at initial ATC boot-up, and NOT performed for devices newly inserted at any other time. The maximum boot-up time requirement does not apply when a USB device is attached. If the ATC has multiple USB ports, only the first port is required to support this automatic startup feature.
ID#: 1-80
Commenter: Peter Ragsdale
Comment Type: Technical
Location: Annex B
Clause: (n/a)

**Issues and Proposed Revision**
Suggest that we standardize the baseline directory structure.

**Disposition**

*Modify Section 2.2.5, adding Item 4:*

4) A root filesystem including the following directories, or symbolic links to directories, and compatible with version 2.3 (or later) of the Filesystem Hierarchy Standard:

<table>
<thead>
<tr>
<th>Directory</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>/bin</td>
<td></td>
</tr>
<tr>
<td>/dev</td>
<td></td>
</tr>
<tr>
<td>/etc</td>
<td></td>
</tr>
<tr>
<td>/etc/init.d</td>
<td>Linux Standard Base location for initialization scripts</td>
</tr>
<tr>
<td>/lib</td>
<td></td>
</tr>
<tr>
<td>/media</td>
<td></td>
</tr>
<tr>
<td>/media/usb</td>
<td>Mount point for usb storage devices</td>
</tr>
<tr>
<td>/mnt</td>
<td></td>
</tr>
<tr>
<td>/opt</td>
<td></td>
</tr>
<tr>
<td>/sram</td>
<td>Mount point for SRAM filesystem</td>
</tr>
<tr>
<td>/sbin</td>
<td></td>
</tr>
<tr>
<td>/tmp</td>
<td></td>
</tr>
<tr>
<td>/usr</td>
<td></td>
</tr>
<tr>
<td>/var</td>
<td></td>
</tr>
</tbody>
</table>
ID#: 1-81

Commenter: Dave Miller

Comment Type: Technical

Location: Annex B

Clause: (C++ libraries)

**Issues and Proposed Revision**

Applications may be written in C++, requiring C++ libraries, as described in Paragraph 3.2.3, while the 5.2b standard does not require C++ libraries.

End user must supply the proper C++ libraries, as well as to increase the memory size to allow room for the libraries along with the application.

Require manufacturers to supply gLibC 2.5 or later C++ libraries on any processor that includes an MMU.

**Disposition**

*Memory concerns addressed by Comments 1-16, 1-17. glibc not required.*

*Modify Section 2.2.5, Item 3:*

Add GCC/libstdc++.so package to table (new entry).
ID#: 1-82
Commenter: Ralph Boaz
Comment Type: Technical
Location: Annex B
Clause: (n/a)

**Issues and Proposed Revision**

During the development of the API standard (June/July 2006) it was agreed upon between the leadership of the API and Controller WGs (and other members of the WGs) that the subsequent version to ATC 5.2b would include a kernel interface for the SDLC driver. This allows implementations of the API to use a kernel level interface to the SDLC if they so choose.

(see attached material – immediately following – one page)

**Disposition**

*Add new subsection B.6 to Annex B, as follows:*

**B.6 ATC SPxs Synchronous Driver Kernel Level Interface**

**B.6.1 Overview**

This section defines a generalized synchronous driver Kernel Level interface for the serial ports of the ATC Controller Engine Board. The action and meaning of the functions and parameters selected by this standard shall follow those defined by the Open Group Base Specifications Issue 6 IEEE STD 1003.1, 2004 Edition. However, only those flags and commands defined here require implementation to ensure conformance.

An additional Kernel Level interface is required to allow Kernel Level programming usage of the SPxs Synchronous Driver. The symbols mentioned herein must be exported to the Kernel so that Loadable Kernel Modules (LKM) may link to these functions.

**B.6.2 Channel Numbers**

Each synchronous serial port shall have a special channel number associated with it. Channel numbers shall be defined as follows:

<table>
<thead>
<tr>
<th>Device</th>
<th>Channel Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>SP1</td>
<td>1</td>
</tr>
<tr>
<td>SP2</td>
<td>2</td>
</tr>
<tr>
<td>SP3</td>
<td>3</td>
</tr>
<tr>
<td>SP5</td>
<td>5</td>
</tr>
<tr>
<td>SP8</td>
<td>8</td>
</tr>
</tbody>
</table>
B.6.3 SPxs Functions

All function symbols listed herein must be exported to the Kernel environment to allow linking by Loadable Kernel Modules (LKM).

*sdlc_kernel_open(*)*

This function allows kernel code to request exclusive ownership of a serial port, for read and write access. Upon opening, the port is made ready to be configured via an *sdlc_kernel_ioctl(*)* function call and then accessed via the *sdlc_kernel_read( ), sdlc_kernel_write( ) and sdlc_kernel_close( )* functions.

**Prototype**

```c
void * sdlc_kernel_open( int channel );
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>channel</td>
<td>This is the channel number to open.</td>
</tr>
</tbody>
</table>

**Return Value**

<table>
<thead>
<tr>
<th>Return Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IS_ERR()</td>
<td>An error occurred, or the channel could not be opened. IS_ERR() is called with the returned void * to determine if an error occurred.</td>
</tr>
<tr>
<td>Pointer to driver block</td>
<td>A pointer to a device driver data block, specific to this driver and channel. This pointer will be passed in subsequent sdlc_kernel_calls.</td>
</tr>
</tbody>
</table>

**Errors**

<table>
<thead>
<tr>
<th>Errors</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENODEV</td>
<td>Incorrect device specified or device cannot be found.</td>
</tr>
<tr>
<td>EBUSY</td>
<td>The device is already open or the device is busy.</td>
</tr>
<tr>
<td>EINTR</td>
<td>The open system call was interrupted by a signal.</td>
</tr>
</tbody>
</table>

**Example**

```c
void *context;
context = sdlc_kernel_open( ATC_SP5 );              /* Get the file context for sp5s */
if ( IS_ERR( context ) )
    {
        printk( KERN_ALERT "sdlc_kernel_open() failure on sp5s, errno = %d\n",
                PTR_ERR( context ) );
    }
else

The returned context is the ownership “handle” passed by kernel code to the *sdlc_kernel_close( ), sdlc_kernel_read( ), sdlc_kernel_write( ) and sdlc_kernel_ioctl( )* functions.

*sdlc_kernel_close(*)*

This function allows kernel code to release ownership of the serial port pointed to by context. This function should only be called after a successful *sdlc_kernel_open( )* of the respective port and
after any previous write calls have had time to be transmitted.. Upon closing the port, all settings and configurations are put in a reset state and the context is no longer valid. This call shall immediately perform the close and return without delay.

Prototype

```
int sdlc_kernel_close( void *context );
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>context</td>
<td>This is the context of the port to be closed.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Return Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;0</td>
<td>An error occurred. The negative error code is returned.</td>
</tr>
<tr>
<td>0</td>
<td>The operation succeeded.</td>
</tr>
</tbody>
</table>

Errors

<table>
<thead>
<tr>
<th>Errors</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EBADF</td>
<td>Incorrect file descriptor specified.</td>
</tr>
</tbody>
</table>

EINTR The close system call was interrupted by a signal.

Example

```
int status;
status = sdlc_kernel_close( context);
if ( 0 != status )
    printk( KERN_ALERT "sdlc_kernel_close() failure, error = %d\n", status);
return(status);
```

```
sdlc_kernel_read( )
```

This function allows kernel code to read data from the open serial port referenced by the void *context. This function should only be called after a successful sdlc_kernel_open() of the respective port. The function reads up to count bytes from the next available valid packet received. The function places the requested data in the memory location pointed to by *buf. When count bytes is less than the number of bytes in the next available packet, the remainder of the packet shall be discarded by the driver. The driver shall not copy the packet's CRC bytes to buf. This is a non-blocking operation. The function will return 0 (zero) if no data is available.

Prototype

```
ssize_t sdlc_kernel_read( void *context, void *buf, ssize_t count);
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>context</td>
<td>This is the context of the port to access.</td>
</tr>
<tr>
<td>buf</td>
<td>The data sdlc_kernel_read will be put here.</td>
</tr>
</tbody>
</table>
count This is the desired number of bytes to read.

<table>
<thead>
<tr>
<th>Return Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt; 0</td>
<td>An error occurred. The negative error code is returned.</td>
</tr>
<tr>
<td>count</td>
<td>The operation succeeded. The number of bytes actual read is returned.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Errors</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EBADF</td>
<td>Incorrect file descriptor specified.</td>
</tr>
<tr>
<td>EFAULT</td>
<td>There was a problem copying data into the user specified buffer.</td>
</tr>
<tr>
<td>EINVAL</td>
<td>The call was interrupted by a signal before any data was read.</td>
</tr>
</tbody>
</table>

**Example**

```c
ssize_t status;

status = sdlc_kernel_read( context, &buf, count );
if ( 0 > status )
    printk( KERN_ALERT "sdlc_kernel_read() failure, error = %d\n", status );
else

sdlc_kernel_write( )
```

This function allows kernel code to write data to the open serial port referenced by the void *context. This function should only be called after a successful open of the respective device. The function writes count bytes to the driver to be sent out as a single packet. The function takes the data from the memory location pointed to by *buf. The driver shall add the crc bytes to the data copied from buf as part of sending the out the packet. If count bytes can not be sent as a single packet, the EINVAL error shall be returned and no packet shall be sent. This is a non-blocking operation; therefore if the write would block the call shall return immediately with an EAGAIN error.

**Prototype**

```c
ssize_t write( void *context, const void *buf, ssize_t count);```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>context</td>
<td>This is the context of the port to access.</td>
</tr>
<tr>
<td>buf</td>
<td>The data written comes from here.</td>
</tr>
<tr>
<td>count</td>
<td>This is the desired number of bytes to write.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Return Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt; 0</td>
<td>An error occurred. The negative error code is returned.</td>
</tr>
</tbody>
</table>
count | The operation succeeded.

<table>
<thead>
<tr>
<th>Errors</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EBADF</td>
<td>Incorrect file descriptor specified.</td>
</tr>
<tr>
<td>EFAULT</td>
<td>There was a problem copying data from the user specified buffer.</td>
</tr>
<tr>
<td>EINVAL</td>
<td>The maximum number of bytes to be written has been exceeded.</td>
</tr>
<tr>
<td>EAGAIN</td>
<td>Returned if the write operation would block.</td>
</tr>
</tbody>
</table>

**Example**

```c
ssize_t status;
status = sdlc_kernel_write( context, &buf, count );
if ( 0 > status )
    printk( KERN_ALERT "sdlc_kernel_write() failure, error = %d\n", status );
```

**sdlc_kernel_ioctl( )**

This function allows kernel code to configure, control and monitor status of the serial port referenced. by void *context. This function should only be called after a successful sdlc_kernel_open( ) of the respective port. The operation performed by the sdlc_kernel_ioctl( ) function depends on the command argument. The command argument determines the interpretation of any additional arguments. The supported IOCTL services are defined below.

**Prototype**

```c
int sdlc_kernel_ioctl( void *context, int command, parameters );
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>context</td>
<td>This is the context of the port to access.</td>
</tr>
<tr>
<td>command</td>
<td>This specifies the desired operation to be performed.</td>
</tr>
<tr>
<td>parameters</td>
<td>This argument is an integer or a pointer to a source structure containing port configuration data or an integer or a destination structure where information is placed by the sdlc_kernel_ioctl( ) function call.</td>
</tr>
</tbody>
</table>

**Return Value | Description**

<table>
<thead>
<tr>
<th>Return Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt; 0</td>
<td>An error occurred. The negative error code is returned.</td>
</tr>
<tr>
<td>0</td>
<td>The operation succeeded.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Errors</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EBADF</td>
<td>Invalid file descriptor specified.</td>
</tr>
<tr>
<td>EFAULT</td>
<td>There was a problem accessing data from the user specified buffer.</td>
</tr>
<tr>
<td>ENOTTY</td>
<td>An invalid command parameter was specified</td>
</tr>
</tbody>
</table>
EINVAL An invalid command parameter was specified.

Valid SPxs IOCTL Commands

**ATC_SPXS_WRITE_CONFIG**

This command passes the atc_spxs_config structure to the serial port pointed to by context. This command is used to set the baud rate, protocol and clocking options of a port. The atc_spxs_config structure is defined in the Advanced Transportation Controller (ATC) standard section B.5.4

<table>
<thead>
<tr>
<th>ioctl( ) Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>context</td>
<td>This is the context of the device to access.</td>
</tr>
<tr>
<td>command</td>
<td>ATC_SPXS_WRITE_CONFIG</td>
</tr>
<tr>
<td>parameter</td>
<td>struct atc_spxs_config * Pointer to a kernel space config structure to write.</td>
</tr>
</tbody>
</table>

**Example**

```c
int status;

status = sdlc_kernel_ioctl( context, ATC_SPXS_WRITE_CONFIG, &atc_sp1s_configure);
if( 0 > status )
    printk( KERN_ALERT "sdlc_kernel_ioctl() failure, error = %d\n", status);
```

**ATC_SPXS_READ_CONFIG**

This command copies the data from the port's atc_spxs_config structure to the buffer pointed to by parameter. This command is used to check the state of the baud rate, protocol and clocking options of a port. The atc_spxs_config structure is defined in the Advanced Transportation Controller (ATC) standard section B.5.4

<table>
<thead>
<tr>
<th>ioctl( ) Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>context</td>
<td>This is the context of the device to access.</td>
</tr>
<tr>
<td>command</td>
<td>ATC_SPXS_READ_CONFIG</td>
</tr>
<tr>
<td>parameter</td>
<td>struct atc_spxs_config * Pointer to a kernel space destination</td>
</tr>
</tbody>
</table>

**Example**

```c
int status;

status = sdlc_kernel_ioctl( context, ATC_SPXS_READ_CONFIG, &port_configuration );
if( 0 > status )
    printk( KERN_ALERT "sdlc_kernel_ioctl() failure, error = %d\n", status );
```
FIONREAD

This command returns the number of bytes to be read from the next available valid packet received. If no packet is available the call will return success and a count of zero.

<table>
<thead>
<tr>
<th>ioctl( ) Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>context</td>
<td>This is the context of the device to access.</td>
</tr>
<tr>
<td>command</td>
<td>FIONREAD</td>
</tr>
<tr>
<td>parameter</td>
<td>int * Pointer to the kernel space location to return the count of bytes available.</td>
</tr>
</tbody>
</table>

Example

```c
int status;
int rcount;

status = sdlc_kernel_ioctl(cd, FIONREAD, &rcount);
if (0 > status)
    printk(KERN_ALERT _"sdlc_kernel_ioctl() failure, error = %d\n",status);
```

(Disposition End)
SDLC Kernel-Level Interface Functions

**int sdlc_kernel_write( int channel, unsigned char * buf, size_t len )**

A functional interface must be provided that will allow the FIO Manager to pass a SDLC protocol formatted buffer, that will be sent on a previously opened SDLC port as indicated by channel. The interface will allow the passing of a channel, a pointer to a Kernel space buffer to be sent, and the number of bytes in this buffer. This buffer will be owned by the FIO Manager and must not be freed by the SDLC driver. The return code will indicate success or failure to pass the buffer to the SDLC driver, not the actual transmission of this frame. This is a non-blocking interface.

**int sdlc_kernel_read ( int channel, unsigned char * buf, size_t len )**

A functional interface must be provided that will allow the FIO Manager to read SDLC frames that MAY have been received by the SDLC Driver. This is a non-blocking interface; so a read will return 0 when no data is available. The interface will allow the passing of a channel, a pointer to a buffer that the SDLC driver should copy the response into and the number of bytes in this buffer. The FIO Manager owns this buffer and must not be freed by the SDLC Driver. The return code will indicate the number of bytes returned in the buffer, with 0 indicating no frame to return.

**int sdlc_kernel_ioctl ( int channel, int command, void * arg )**

A functional interface must be provided that will allow the FIO Manager to modify the behavior of the SDLC driver. At this time, the only known behavior that must be modifiable by the FIO Manager is the baud rate of the SDLC port. The baud rates that must be supported are 153.6 Kbps and 614.4 Kbps. The channel, an ioctl command and a structure defined by the SDLC driver for passing baud rate information, will be passed – standard ioctl() interface. The return code will indicate success or failure of the operation.

The following interfaces are optional but may be required to support the existing driver.

**void * context sdlc_kernel_open( int channel )**

A functional interface could be provided that will be utilized to open 1 of 3 serial ports (SP3, SP5 or SP8) for SDLC communications. This interface would be called once for each port that the FIO Manager may desire to open. A port name, device name or port number would be passed as input – your choice. Upon open, it will be expected that a context structure will be returned (SDLC handle); that will be passed back in upon subsequent sdlc_ calls. A NULL will be returned to indicate failure (port does not exist / cannot be opened) of the open. This open must be exclusive, no other device opens for the requested port can be allowed.

**sdlc_kernel_close( void * context )**

A functional interface must be provided that will allow the FIO Manager to close a previously opened SDLC communication port. The SDLC handle returned by the sdlc_kernel_open() call will be passed. No return codes are expected. The function will free the context structure acquired from the open.

The actual function names were not decided.
ID#: 1-83

Commenter: Dave Miller

Comment Type: Technical

Location: Annex B

Clause: (SDLC driver)

**Issues and Proposed Revision**

Cannot query the SDLC driver to find out how many bytes are available. Suggest adding ioctl call to SDLC driver for fionread functions to indicate the current number of bytes in the present packet for a read. Also require that the SDLC driver is in a single-packet read mode.

No maximum timeout required for an SDLC blocking read. Suggest adding an ioctl call to the SDLC driver to implement the OS-9 equivalent of pd_iotime. This requirement is suggested to minimize the number of bugs in code that is ported from OS-9 to Linux.

**Disposition**

*Comment accepted. See Comment 1-82 disposition.*
ID#: 3-1

Commenter: Mike Gallagher / Intelight

Comment Type: Technical

Location: Annex B.2

Clause: "This section defines a generalized driver interface for port pins defined on the ATC engine board."

Issues and Proposed Revision

Linux has a generic GPIO infrastructure (CONFIG_GPIOLIB) for user-space access to GPIO pins described at http://www.kernel.org/doc/Documentation/gpio.txt. Suggest replacing section with reference to standard Linux API.

Disposition

Comment rejected. See Comment 1-69 disposition.
ID#: 3-2
Commenter: Mike Gallagher / Intelight
Comment Type: Technical
Location: Annex B.3.1
Clause: SPI ioctl calls

Issues and Proposed Revision

SPI device API describes ioctl function codes for reading and writing with offset, but no read/write buffer is specified in either case. Suggest removal of SPI driver section in favor of individual drivers relating to SPI device type, i.e. Linux MTD for SPI memories, etc.

Disposition

See Comment 1-70 disposition.
ID#: 3-3  
Commenter: Mike Gallagher / Intelight  
Comment Type: Technical  
Location: Annex B.5  
Clause: "This section defines a generalized synchronous driver interface for the serial ports of the ATC engine board."

**Issues and Proposed Revision**

Linux has a Generic HDLC layer (CONFIG_HDLC) which uses the Linux 'net device' model, better suited to synchronous packet-oriented protocols than the 'tty' or character based serial driver model as specified. Suggest replacing the existing section, requiring the "spxs" ports driver be a network device compatible with the Linux Generic HDLC Layer described at: [http://www.kernel.org/pub/linux/utils/net/hdlc/](http://www.kernel.org/pub/linux/utils/net/hdlc/).

**Disposition**

Comment rejected. Driver as indicated not supported.
ID#: 3-4  
Commenter: Mike Gallagher / Intelight  
Comment Type: Technical  
Location: Annex B.4.4  
Clause: "Standard Linux drivers required to implement a ramfs file system on the +5 VDC Standby Power-backed static RAM shall be provided."

Issues and Proposed Revision

"ramfs" is a dynamically-sized filesystem type based upon the Linux disk-caching mechanism, and unsuitable for a memory region of fixed location and size. Suggest access to the SRAM memory be via Linux MTD device, possibly mounted as "/r0" with "ext2" or "minix" filesystem.

Disposition

Comment accepted. See Comment 1-73 disposition.
ID#: 3-5
Commenter: Mike Gallagher / Intelight
Comment Type: Technical
Location: General (new)
Clause: (new)

Issues and Proposed Revision

The ATC Standard 5.2b does not provide a definition or guidelines for the organization of the Linux root filesystem. We suggest this is necessary so that user-state applications have a standard way to install and access resources on each vendor platform. Suggest reference to the Filesystem Hierarchy Standard (http://www.pathname.com/fhs/) or Buildroot (http://buildroot.uclibc.org/) conventions or a new section with minimum filesystem requirements.

Disposition

Comment accepted. See Comment 1-80 disposition.
ID#: 3-6
Commenter: Mike Gallagher / Intelight
Comment Type: Technical
Location: General (new)
Clause: (new)

Issues and Proposed Revision

Toolchain requirements for applications portability: The ATC requirements for a vendor-supplied BSP should also require a copy of the gcc cross-compiler and libraries (toolchain) necessary to build applications for the particular cpu architecture of the vendors' platform.

Disposition

Comment accepted. See Comment 1-1 disposition.
ID#: 3-7

Commenter: Mike Gallagher / Intelight

Comment Type: Technical

Location: General (new)

Clause: (new) (Secure Shell (SSH) replacement for telnet/ftp)

Issues and Proposed Revision

Suggest a requirement to provide the busybox-related secure shell service “dropbear” (http://matt.ucc.asn.au/dropbear/dropbear.html) with support for secure shell (ssh) and secure copy (scp) as a replacement for the telnet and ftp services.

Disposition

Modify Section 2.2.5, Item 3:

Add Secure Shell package to table (new entry), Version SSH-2, programs sshd, ssh, scp.
ID#: 3-8

Commenter: Mike Gallagher / Intelight

Comment Type: Technical

Location: General (new)

Clause: (new) (Time synchronization with LineSync and/or GPS sources)

Issues and Proposed Revision

It is not clear that the Linux system clock tick should be directly tied to the LineSync pulse input, excluding other clock synchronization sources, and it is common in the Linux world to use the NTP system for time synchronization, allowing for user-space selection and configuration of the time synchronizing source clock. It is likely that the ATC will be required to operate with at least the following clock synchronization sources: LineSync pulse on GPIO pin; a GPS unit providing a Pulse Per Second (PPS) signal via serial port; network based NTP server. Suggest that the LineSync signal be converted to a PPS source by a required device driver compatible with the NTP PPS infrastructure (http://www.ietf.org/rfc/rfc2783.txt?number=2783). This way, the user may configure NTP to use whichever of the clock sources as available, to provide highly accurate timekeeping.

Disposition

Comment withdrawn by commenter.
ID#: 3-9

Commenter: Mike Gallagher / Intellight

Comment Type: Technical

Location: 2.2.5 (3) BSP - Utilities

Clause: "Utility applications, modules, libraries and supporting data which include, but are not limited to, the following:"

Issues and Proposed Revision

Suggest adding "rpm" and/or "dpkg" to list of busybox utility applets or a separate requirement for an embedded package management utility such as "ipkg" in order to facilitate application installation and updates with dependency checking, as used widely under Linux.

Disposition

Comment approved. Modify Section 2.2.5, Item 3:

Add rpm to list of required Busybox programs.
ID#: 3-10

Commenter: Doug Tarico / McCain

Comment Type: Technical

Location: Section 5.4.3, page 5-13

Clause: (SPI_SEL lines)

Issues and Proposed Revision

Comment

SPI_SEL_1 through SPI_SEL_4 on the P2 connector should be treated as a 4-bit binary encoded value to allow 16 SPI select lines instead of 4. This can and should be defined in a way that is backwards compatible so that interchangeability between new and old hardware is preserved.

Suggested resolution

Update text as follows (changes from ATC 5.2b are highlighted in yellow). Corresponding changes would also be needed in Annex B, section B.3.1, which describes the SPI driver.

Disposition

Replace Section 5.4.3, under Serial Peripheral Interface Port, in it’s entirety with the following:

The engine board shall provide a synchronous Serial Peripheral Interface Port. All SPI interface pins shall be at HCT logic-levels. Input pins are indicated by (I), output pins by (O).

The SPI select pins on the P2 connector shall be interpreted by the engine board and host module as a 4-bit binary coded address as shown in the following table, allowing up to 15 different SPI devices to be addressed.

<table>
<thead>
<tr>
<th>SPI_SEL_4</th>
<th>SPI_SEL_3</th>
<th>SPI_SEL_2</th>
<th>SPI_SEL_1</th>
<th>Encoded SPI Address</th>
<th>Device</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>None</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>DataKey</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>Host EEPROM</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>Reserved</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>X</td>
<td>X</td>
<td>4-7</td>
<td>Reserved</td>
</tr>
<tr>
<td>1</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>8-15</td>
<td>Manufacturer Specific</td>
</tr>
</tbody>
</table>

The encoded SPI address 1 is required to support DataKey operations.

The encoded SPI address 2 is required to support a host module serial EEPROM device containing controller configuration information. The content and organization of the information will be described in the BSP. This EEPROM device shall have the following characteristics:

- Function in a manner similar to a 25020-type (2K-bit) SPI EEPROM device
- Have a minimum size of 2 Kbit organized as 256 words of 8 bits each
- Provide 5V interface signals
- Operate properly with up to a 2.0 MHz SPI clock
- Utilize SPI Mode 0 (CPOL=0, CPHA=0)
- Be write-protected (using "WP pin) whenever POWERUP is LOW
- Be readable from application software during normal ATC operation
- Support the following instruction set:

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
<th>Instruction Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>WREN</td>
<td>Write Enable</td>
<td>0000 X110</td>
</tr>
<tr>
<td>WRDI</td>
<td>Write Disable</td>
<td>0000 X100</td>
</tr>
<tr>
<td>RDSR</td>
<td>Read Status Register</td>
<td>0000 X101</td>
</tr>
<tr>
<td>WRSR</td>
<td>Write Status Register</td>
<td>0000 X001</td>
</tr>
<tr>
<td>READ</td>
<td>Read from Memory Array</td>
<td>0000 A8011</td>
</tr>
<tr>
<td>WRITE</td>
<td>Write to Memory Array</td>
<td>0000 A8010</td>
</tr>
</tbody>
</table>

Note: A8 represents MSB address bit A8

Encoded SPI addresses 3-7 are currently unimplemented and are reserved for future SPI-related expansion in this standard.

Encoded SPI addresses 8-15 are available for manufacturer specific use. To support Engine Board interchangeability, if an ATC Controller includes any manufacturer-specific SPI devices, then the ATC Controller manufacturer shall provide documentation to allow third parties to write device drivers which utilize those devices. To support application software portability, if an Engine Board includes any device drivers for manufacturer specific SPI devices, then the Engine Board manufacturer shall provide documentation to allow third parties to write application programs which utilize those device drivers.

For backwards compatibility, drivers that use more than one concurrent SPI_SEL select line to select an encoded address shall query the "Expanded SPI Supported" field in the Host EEPROM to determine if encoded SPI address selection is supported by the host. This query shall be performed when the open() function is called. If the Host EEPROM version is less than 2 or if no Host EEPROM is present, then the driver shall assume that encoded SPI address selection is not supported, open() shall return -1 and errno shall be set to ENODEV.

Principal Usage: DataKey / serial EEPROM interface
Operating Modes: SYNC
Synchronous Rates (bps): (application-specific)
Interface Pins:
- SPI_MOSI: Master-Out-Slave-In (O)
- SPI_MISO: Master-In-Slave-Out (I)
- SPI_CLK: Clock (O)
- SPI_SEL_1: Encoded Select 1 (O)
- SPI_SEL_2: Encoded Select 2 (O)
- SPI_SEL_3: Encoded Select 3 (O)
- SPI_SEL_4: Encoded Select 4 (O)
Serial Peripheral Interface Port

The engine board shall provide a synchronous Serial Peripheral Interface Port. All SPI interface pins shall be at HCT logic-levels. Input pins are indicated by (I), output pins by (O).

The SPI select pins on the P2 connector shall be interpreted by the engine board and host module as a 4-bit binary coded value as shown in the following table, allowing up to 16 different SPI devices to be addressed.

<table>
<thead>
<tr>
<th>SPI_SEL_4</th>
<th>SPI_SEL_3</th>
<th>SPI_SEL_2</th>
<th>SPI_SEL_1</th>
<th>Encoded SPI Select</th>
<th>Device</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>None</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>DataKey</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>Host EEPROM device</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>Reserved</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>4</td>
<td>Reserved</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>5</td>
<td>Reserved</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>6</td>
<td>Reserved</td>
</tr>
<tr>
<td>1</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>7</td>
<td>Reserved</td>
</tr>
<tr>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>8-15</td>
<td>Manufacturer Specific</td>
</tr>
</tbody>
</table>

The encoded SPI Select value 1 is required to support DataKey operations.

The encoded SPI Select value 2 is required to support a host module serial EEPROM device containing controller configuration information. The content and organization of the information will be described in the BSP. This EEPROM device shall have the following characteristics:

- Function in a manner similar to a 25020-type (2K-bit) SPI EEPROM device
- Have a minimum size of 2 Kbit organized as 256 words of 8 bits each
- Provide 5V interface signals
- Operate properly with up to a 2.0 MHz SPI clock
- Utilize SPI Mode 0 (CPOL=0, CPHA=0)
- Be write-protected (using *WP pin) whenever POWERUP is LOW
- Be readable from application software during normal ATC operation
- Support the following instruction set:

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
<th>Instruction Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>WREN</td>
<td>Write Enable</td>
<td>0000 X110</td>
</tr>
<tr>
<td>WRDI</td>
<td>Write Disable</td>
<td>0000 X100</td>
</tr>
<tr>
<td>RDSR</td>
<td>Read Status Register</td>
<td>0000 X101</td>
</tr>
<tr>
<td>WRSR</td>
<td>Write Status Register</td>
<td>0000 X001</td>
</tr>
<tr>
<td>READ</td>
<td>Read from Memory Array</td>
<td>0000 A0011</td>
</tr>
<tr>
<td>WRITE</td>
<td>Write to Memory Array</td>
<td>0000 A0010</td>
</tr>
</tbody>
</table>

Note: A₀ represents MSB address bit A₀

Encoded SPI Select values 3-7 are currently unimplemented and are reserved for future SPI-related expansion. Encoded SPI Select values 8-15 are available for manufacturer specific use.
ID#: 3-11

Commenter: Doug Tarico / McCain

Comment Type: Technical

Location: Section 5.4.3, page 5-17, line 11

Clause: (Engine Board Ethernet speed)

**Issues and Proposed Revision**

**Comment**
The engine board should not be limited to 10 Mbit ethernet. The engine board should provide the necessary connections to support 100 Mbit ethernet. 100 Mbit support can remain optional on the engine board, but the host module should support it if available on the engine board.

**Suggested resolution**
Redefine "reserved" pins on P1 connector (Table 5-1):
34 = ENET1_TX_CT
35 = ENET1_RX_CT
43 = ENET2_TX_CT
44 = ENET2_RX_CT

Note: there are several other places in the document where the text will need to be updated if this change is made.

**Disposition**

*Comment withdrawn by commenter.*
ID#: 3-12
Commenter: Doug Tarico / McCain
Comment Type: Technical
Location: Section 5.4.4, page 5-17
Clause: (GPIO pins)

Issues and Proposed Revision

Comment
We would like to have some I/O pins available for manufacturer specific use on the engine board P1 or P2 connectors. The PROG_TEST pins on the P2 connector are an excellent candidate for this since they are already defined for manufacturer-specific usage. However, this section precludes usage of these pins on production host modules.

Suggested resolution
Delete the following sentence:

"Manufacturers are free to designate these pins for these purposes in any configuration on special engine board test adapter hosts, however all mating PROG_TEST pins on production ATC host modules shall be no-connects."

Disposition

Disposition deferred to a future revision of the standard.
ID#: 3-13
Commenter: Doug Tarico / McCain
Comment Type: Technical
Location: Section 7.1.2.1, page 7-2, line 17
Clause: (AUX switch)

Issues and Proposed Revision

Comment
The AUX switch should be optional since it is part of an optional user interface.

Suggested resolution
Change line 17 to:

“Option 1: Keyboard, LCD and Bell (Guidance: traditional ATC 2070)”

Disposition

Comment accepted with alternate resolution. See Comment 1-29 disposition.
ID#: 3-14
Commenter: Doug Tarico / McCain
Comment Type: Technical
Location: Section 7.1.4.1, page 7-3, line 32
Clause: (LCD contrast)

Issues and Proposed Revision

Comment
The requirement for use of the * key for software contrast control precludes the use of a custom "CONTRAST" key on keyboard with additional keys.

Suggested resolution
Delete the following sentences:

“If using a software controlled contrast, the contrast control shall be accomplished by pressing the (*) key to enable the adjustment, followed by the (+) key to darken and the (-) key to lighten the contrast. By pressing the (*) key again will disable the contrast adjustment.”

Disposition

Disposition deferred to a future revision of the standard.
ID#: 3-15

Commenter: Doug Tarico / McCain

Comment Type: Technical

Location: Section 7.1.4.1, page 7-7, line 17

Clause: (bell frequency)

Issues and Proposed Revision

Comment
The bell frequency is specified here as 4 KHz, but the 2070 specification calls for 2 KHz. These should be consistent.

Suggested resolution
Change requirement to 2 KHz to match 2070 specification.

Disposition

Disposition deferred to a future revision of the standard.
ID#: 3-16
Commenter: Doug Tarico / McCain
Comment Type: Technical
Location: Section 7.2.2, page 7-16
Clause: (power indicators)

Issues and Proposed Revision

Comment
The DC power indicators should be optional. NEMA controllers have traditionally not required power supply status indicators.

Suggested resolution
Remove section 7.2.2 entirely.

Disposition

Comment accepted (power indicators as optional). See Comment 1-49 disposition.
ID#: 3-17
Commenter: Doug Tarico / McCain
Comment Type: Editorial
Location: Section 8.4, page 8-26, line 4
Clause: ---

Issues and Proposed Revision

Comment
Typo: "removal" should be "removable"

Suggested resolution
Fix as indicated.

Disposition

Disposed as proposed.
ID#: 3-18

Commenter: Doug Tarico / McCain

Comment Type: Technical

Location: Annex B, driver interfaces

Clause: (driver)

**Issues and Proposed Revision**

Comment
Recommend creating a device /dev/linesync that can be used to provide a signal to an application program at a frequency that is synchronized to the A/C linesync signal.

Suggested resolution
Add a definition of this driver to Annex B. Specific language TBD.

**Disposition**

*Comment withdrawn by commenter.*
ID#: 4-1
Commenter: Peter Ragsdale
Comment Type: Technical
Location: (numerous)
Clause: (numerous) – interval timing

Issues and Proposed Revision

Comment

During NTCIP 1214 standards development, it was raised that some controllers are not accurately timing intervals. ATC v5.2b standard puts some requirements on clocks and timing (4.1.3) but does not go far enough.

Suggested resolution

1. In section 4.1.3, suggest a stronger wording that under no circumstances shall the OST miss a LINESYNC pulse/edge, and that applications shall ensure that all timing is relative to OST such that no cumulative or drift errors shall occur in timing intervals.
2. Somewhere we need to add a requirement that any interval timed shall not deviate by more than 100 milliseconds from its set value at a power source frequency of 60 hertz.
3. In section 9.1.1, there should be a test that communications loading does not affect timing accuracy per defined specifications (e.g. toggle a signal every 8.333 ms, etc)
4. In section 9.7.2 to 9.7.6 inclusively, remove the statement “skip tests 2. 2.2.11 Timing Accuracy Tests and 2. 2.2.11.2 Repeatability” ...we need to ensure that timing accuracy and repeatability is maintained.
5. Add requirements specifically to define timing accuracy and repeatability.
6. Add a requirement in section 9.6 for a Chart recorder sufficient to verify/validate timing intervals, etc.(e.g. 24 channel, 1ms resolution, etc)
7. In section 4.1.3. besides mandating that the OS update the RTC from the OST once per hour, should we also add a requirement that the RTC be updated if it differs from the OST by more than x ms? E.g. one cannot achieve the NYC requirements as currently stated as there can be too much drive between OST and RTC such that if a power outage occurs and OST is reloaded from RTC that offset in the time is introduced.

Disposition

Disposition deferred to a future revision of the standard.
ID#: 4-2

Commenter: Robin Harrison

Comment Type: Technical

Location: 8.2.2.5 Page 8-21 and 8-22

Clause: Mechanical Details – D connector

Issues and Proposed Revision

Comment

NEMA does not define a D connector, the D connector defined in the standard refers to the 2070 NEMA module D connector. Historically D connectors have been customer and/or manufacturer specific and many NEMA cabinets still require support of these custom D connectors, while other NEMA cabinets have no requirement for the D connector.

Suggested resolution

1. Change the standard so that the D connector is optional. For TS2 type 2 or TS1 applications the minimum connectors should be A, B and C.
2. Allow the use of custom D connectors to allow for continuing support of cabinets in the field.
3. Define the D connector currently in the standard as the ATC D and include the pin out in the ATC standard.

Disposition

Replace Section 8.2.1.13, Module Identification, as follows:

8.2.1.13 I/O Module Identification

The I/O Module Identification command frame shall be used to request the I/O identification value response of:

1. Model 332 PI/O
2. NEMA TS-2 Type 2 PI/O per TEES / ATC2070 (includes A, B, C, and D connectors)
3. NEMA TS-2 Type 1 PI/O with standard A connector only
4. NEMA TS-2 Type 2 PI/O with standard A, B, and C connectors only
5. NEMA TS-2 Type 2 PI/O with standard A, B, and C plus custom additional connector(s)
6. Custom PI/O
7-15. Reserved

The remaining I/O identification values (16..255) are set aside for manufacturer or agency specific configurations which are not specified herein. However, in order for the device to be considered conformant to this standard, it must support one of the options listed above (1-4) in addition to the custom I/O connector(s); this has been done to allow agencies to procure ATC controllers that provide the advanced capabilities and conform to the balance of this standard while still
supporting agency specific legacy parallel I/O or D connector(s). Please note that the SDLC address for these I/O modules is 20. The command and response frames are as follows:

<table>
<thead>
<tr>
<th>Description</th>
<th>msb</th>
<th>lsb</th>
<th>Byte Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Type Number = 60)</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

**I/O Module Identification Command**

<table>
<thead>
<tr>
<th>Description</th>
<th>msb</th>
<th>lsb</th>
<th>Byte Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Type Number = 188)</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>PI/O I D byte</td>
<td>x</td>
<td>x</td>
<td>2</td>
</tr>
</tbody>
</table>

Create new sections 8.2.1.15 and 8.2.1.16, as follows:

### 8.2.1.15 I/O Input/Output Sizing

The I/O Input/Output Sizing command frame shall be used to request the number of inputs and outputs supported by the I/O module. Please note that the SDLC address for these I/O modules is 20. The command and response frames are as follows:

<table>
<thead>
<tr>
<th>Description</th>
<th>msb</th>
<th>lsb</th>
<th>Byte Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Type Number = 68)</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

**I/O Module Identification Command**

<table>
<thead>
<tr>
<th>Description</th>
<th>msb</th>
<th>lsb</th>
<th>Byte Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Type Number = 196)</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Number of Inputs Supported</td>
<td>x</td>
<td>x</td>
<td>2</td>
</tr>
<tr>
<td>Number of Outputs Supported</td>
<td>x</td>
<td>x</td>
<td>3</td>
</tr>
</tbody>
</table>

(continued next page)
8.2.1.16 I/O Module Extended Description

The I/O Module Extended Description command frame shall be used to request details of the I/O module. These details include the Manufacturer NTCIP ID as assigned by the NTCIP Coordinator and a Manufacturer Specific Code (proprietary) which can provide additional information concerning the actual I/O module provided. Use of this command frame is not required by an application if the Module ID return code (frame = 60) is 1, 2, 3, or 4 (i.e. module defined completely by a standard). If this command frame is received by one of these standard modules, then it shall return the Manufacturer NTCIP ID and 0 (zero) for the Manufacturer Specific Code. Please note that the SDLC address for these I/O modules is 20. The command and response frames are as follows:

### I/O Module Extended Description Command

<table>
<thead>
<tr>
<th>Description</th>
<th>msb</th>
<th>lsb</th>
<th>Byte Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Type Number = 69)</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

### I/O Module Extended Description Response

<table>
<thead>
<tr>
<th>Description</th>
<th>msb</th>
<th>lsb</th>
<th>Byte Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Type Number = 197)</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Manufacturer NTCIP ID</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Manufacturer Specific Code</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
</tbody>
</table>
ID#: 4-3
Commenter: Robin Harrison
Comment Type: Technical
Location: 8.2.2.4 Page 8-20
Clause: Fault and Voltage Monitor - Monitor Logic

**Issues and Proposed Revision**

Defining what type of logic gate is too specific; rather the required functionality should be defined. It should be up to the manufacture to determine what type of circuit to use.

**Disposition**

*Disposition deferred to a future revision of the standard.*
ID#: 4-4

Commenter: Robin Harrison

Comment Type: Technical

Location: 8.4.3

Clause: Serial I/O Page 8-26

Issues and Proposed Revision

The requirement of isolating all serial ports, while good electrical practice has at least one practical limitation.

Per section 5.3.3 of NEMA TS2-2003 page 163 and 164 the cabinet’s port 1 cable does not have DC ground populated and relies on all equipment being referenced to the cabinet’s DC ground.

In this case a fully isolated port 1 will not function in a NEMA cabinet.

Suggest making all Serial I/O for NEMA applications to be optionally isolated.

Disposition

Comment rejected.

Editor’s Note: Section 8.4.3 states, “Each serial port shall be referenced to the attached equipment.” which implies that, even with the required isolation, the ‘cabinet’ side of the serial port used by NEMA PORT1 would be referenced to DCGND2.
ID#: 4-5
Commenter: Robin Harrison
Comment Type: Technical
Location: 7.2.6 Page 7-22
Clause: External Power Supply Requirements

Issues and Proposed Revision
Currently external power requirements are defined on one page and engine board power is defined elsewhere.
Suggest merging requirements so that power supply load requirements are in one place.

Disposition
Disposition deferred to a future revision of the standard.
ID#: 4-6
Commenter: Robin Harrison
Comment Type: Technical
Location: 7.2.6 Page 7-22
Clause: External Power Supply Requirements

Issues and Proposed Revision

Currently external power requirements define a minimum and maximum current for each voltage output. Defining a minimum, while not the intent, allows the designer to design a power supply to the minimum load requirements.

Disposition

Disposition deferred to a future revision of the standard.
ID#: 4-7
Commenter: Robin Harrison
Comment Type: Technical
Location: 7.2.6 Page 7-22
Clause: External Power Supply Requirements

Issues and Proposed Revision

The second paragraph looks incomplete.

Reads: For each Optional Communications Interface Module (note that power supply should be sized appropriately based on the number of communication interface slots the ATC provides): optional communication interface m……

Disposition

Delete superfluous text in Section 7.2.6, as follows:

For each Optional Communications Interface Module (note that power supply should be sized appropriately based on the number of communication interface slots the ATC provides): optional communication interface m
ID#: 4-8
Commenter: Robin Harrison
Comment Type: Technical
Location: 7.2.6 Page 7-22
Clause: External Power Supply Requirements

Issues and Proposed Revision

Instead of stating that the power supply should be sized appropriately based on the number of communication standards, suggest stating what the load requirements are per communication slot.

Disposition

*Existing standard meets requested intent. See Comment 4-5 disposition.*
ID#: 4-9  
Commenter: Robin Harrison  
Comment Type: Technical  
Location: 7.2.6 Page 7-22  
Clause: External Power Supply Requirements  

Issues and Proposed Revision
Currently external power requirements defines the +24VDC to be 500mA maximum. NEMA requires 500mA to be available for external applications. The 500mA maximum stated in the ATC standard does not take into account the parallel IO load.

Disposition
Existing standard meets requested intent.
ID#: 4-10
Commenter: Robin Harrison
Comment Type: Technical
Location: 7.2.6 Page 7-22
Clause: External Power Supply Requirements

Issues and Proposed Revision
What is 24V DC referenced? Is it DCGND1?

Disposition
DCGND2. See Figure 8-4, Page 8-27.
ID#: 4-11

Commenter: Robin Harrison

Comment Type: Technical

Location: 7.2.2 Page 7-16

Clause: LED DC Power Indicators

Issues and Proposed Revision

Which +5VDC is the standard referring to?

Disposition

Modify Section 7.2.2, Item a. (in bold):

a. **Primary** +5 VDC is within 4.8V to 5.25V and the ± 12 VDC voltages are within ±8% of nominal (all voltages referenced to DCGND1).
ID#: 4-12  
Commenter: Robin Harrison  
Comment Type: Technical  
Location: 9.8 and 9.9 Page 9-9  
Clause: Shock and vibration

Issues and Proposed Revision

The NEMA defined shock and vibration test method is out of date.

Suggest defining our own standard as follows:

1. Place the test unit on the shaker's test table. Securely fasten the unit to the test table.
2. Placement of sensors:
   a.) Place one sensor on the end of the shaker table – this will measure the force level with which the test unit is subjected.
   b.) Place a second sensor on the rear of the table – this will safeguard the test unit by shutting down test fixture should excessive force level occurs.
   c.) Place a third sensor on the test unit itself – this will measure the actual force level of the test unit.
3. Resonant Search –
   a.) Set the test table for a double amplitude displacement of 0.015 inch.
   b.) Cycle the test table over a search range from 5 to 30 Hertz for a period of 6 minutes.
   c.) Note and record the resonant frequency.
      i. In the event of more than one resonant frequency in a given axis, record the most severe resonance.
      ii. If resonant frequencies appear equally severe, record each resonant frequency.
      iii. If no resonant frequency occurs for a given axis within the prescribed range, 30 Hz shall be recorded.
4. Endurance test – Vibrate the test unit in each plane at its resonant frequency for a period of 1 hour at an amplitude resulting in 0.5 g acceleration.
   a) When more than one resonant frequency has been recorded, the test period of one hour shall be divided equally between the resonant frequencies.
   b) The total time of the endurance test shall be limited to 3 hours, 1 hour in each axis.
5. Program the Shaker table to subject the test unit to a 10G force having duration of 11 milliseconds using a waveform suitable for simulating a drop test, such as a saw tooth. Subject the test unit to three impacts of both Positive & Negative pulses.
6. Repeat 1-5 for all axis of test.

Disposition

Disposition deferred to a future revision of the standard.
ID#: 4-13

Commenter: Robin Harrison

Comment Type: Technical

Location: 7.2.1 Page 7-16

Clause: ON/OFF Power Switch

**Issues and Proposed Revision**

Not all users/applications require a power switch. For example NEMA cabinets have a cabinet equipment power switch.

Suggest updating the standard to define the power switch as optional or define what applications require a switch and which ones do not.

**Disposition**

*Comment accepted. See Comment 1-48 disposition.*
ID#: 4-14
Commenter: Robin Harrison
Comment Type: Technical
Location: Chapter 8
Clause: n/a

Issues and Proposed Revision

Much of the non cabinet specific information is placed under section 8.2.1 Parallel Connection to Model 332 Cabinets.

All non cabinet specific information should be under its own section, with separate sections for each application.

Currently the section reads as if its biased heavily to 332 cabinet applications.

Disposition

Disposition deferred to a future revision of the standard.