# APPLICATION NOTE

## Interfacing ES601 to Other Devices using RS-232

Rev.: 1.0
Date: June 22, 2009
Doc.: ES601 Remote Control Application Note R1.wpd

# 1. Overview

Although ES601 display firmware Versions 2.0 and 3.0 are generally compatible with Datrend *AMPS-1*, Clinical Dynamics *AccuPulse* and Pronk Technologies *SimCube* in terms of RS-232 remote control operation, some improvements can be made to streamline operation with these specific test instruments in a User-Defined Tester AutoSequence or a Test Procedure.

In it's basic form of operation, the command structure for the RS-232 control was expected to be one or more ASCII characters followed by a carriage return, where the ASCII characters are sent sequentially with no delay between characters. While many test instruments follow this command structure, even Datrend products may deviate from this norm.

For instance, the AMPS-1 Patient Simulator follows an RS-232 command structure where there is a response from AMPS-1 to each character sent (with no carriage return), and the 'E' character terminates the command instead of a carriage return. If characters are sent with no delay between characters, errors can occur.

To deal with this peculiarity in the AMPS-1 command structure, a special command interpreter was created. To invoke the command interpreter, a special prefix character is placed in front of the regular command. When ES601 finds that character at the start of the command, it branches to the command interpreter which will deal with the special requirements of the AMPS-1.

Two other test instruments have been identified which have a command structure which is not fully compatible with the ES601. These devices are the Pronk SimCube and the Clinical Dynamics AccuPulse. For these devices an approach similar to that used to make ES601 compatible with AMPS-1 is used. During execution of an AutoSequence or Procedure, the presence of a specially-designated prefix character at the head of the instruction command string causes ES601 to branch to a special data transmission or data reception function which handles the unique RS232 communication requirements of the tester being controlled.

# 2. AMPS-1 Patient Simulator

For the case of AMPS-1, the pertinent instruction op codes are 'C' and 'R', and the designated character prefix is '&'. When ES601 detects this prefix in the instruction command string, ES601 branches to a special data transmission function which causes each ASCII character of the remote command to be sent individually, with a 200 millisecond delay between the characters. For AMPS-1, no carriage return is sent after transmitting the command, and response messages returned by AMPS-1 to ES601 are discarded. Note the '&' prefix is not transmitted as part of the remote command; the prefix simply acts as a trigger for ES601 when executing a 'C' or 'R' instruction, which causes ES601 perform an alternative form of RS232 data transmission that is suited to AMPS-1.

In order to receive commands AMPS-1 must be set to numeric mode (using the M command), after which a three digit numeric code changes AMPS-1 to a new function/parameter setting. Leading 0's in the three digit code are optional. For further information on the AMPS-1 RS232 command structure refer to the Operator's Manual, section 14.3.

For example, AMPS-1 commands converted to corresponding ES601 'C' and 'R' Instructions as follows:

| Op Code | Description | Command Field |
|---|---|---|
| C | Switch from ALPHA to NUMERIC mode | &M |
| R | Set ECG to NSR, 80BPM | &4E |
| R | Set ECG to Coarse AFIB | &104E |
| R | Set ECG to NSR, 120BPM | &15E |
| R | Set BP channel 1 to 160/110 | &181E |

The Prompt Field which forms part of a 'C' or 'R' Instruction will depend on the user's test application and can be edited arbitrarily as desired, using the Autosequence and/or Test Procedure Editor of *ES601pc* software.

'C', 'R', and additional ES601 Instructions can be combined in lists via the associated Editor to create complex automated tests for multiparameter patient monitors and similar devices. Refer to section 6.2 of the ES601Plus Operating Manual for additional information.

# 3. Pronk Technologies SimCube

## *3.1 Hardware Requirements*

Pronk SimCube (ver 4.x) can be connected to a Tester Port of ES601 using a RJ12 cable (P/N 3140-429) and a DB9M adaptor (P/N 3140-427).

The RS232 serial protocol for SimCube is 9600 baud, no parity, 8 data bits and 1 stop bit (9600,N,8,1). ES601 currently provides this setting for tester communication.

## *3.2 Control Command Requirements*

The SimCube "simplified protocol", based on the "SAE*xxxyyy*" (pressure) and "SAD*zzz*" (rate) commands, is compatible with ES601 Version 3.0 and earlier. These commands are for NIBP simulation only and cannot be used to control the overpressure or leak test.

The SimCube "native protocol" allows remote control of all SimCube test modes. The native protocol has the following unique communication requirements which are not adequately dealt with by ES601 firmware Version 3.0 and earlier:

3.2.1 Native commands begin with one whitespace character. This is problematic for the 'M' instruction of ES601, where a leading whitespace designates an 'M' instruction intended for Phase3 or DNI Impulse, in which case a command string is not to be sent to the tester.

3.2.2 A carriage return termination character is not used, either at the end of a command to be sent to SimCube or at the end of a response from SimCube. This is more problematic in detection of acknowledgments or measurement data received by ES601. It is also problematic for detection of a SimCube response to the ES601 "IDENT" command. (SimCube V4.5 and higher)

3.2.3 Native Commands intended to control the operating mode or the NIBP simulation must be sent ***twice*** to the SimCube, with a short delay between the repeated transmissions. These requirements are pertinent to 'C' and 'R' instructions of ES601.

3.2.4 Commands intended to retrieve data from the SimCube must be *sent one time only* (if, as is necessary for simulator control commands, the same data interrogation command is sent twice, the requested data is altered and often results in a return message corresponding to a zero measured value, even if the requested test result is not zero). This requirement is pertinent to the ES601 'M' instruction.

3.2.5 Measurement data sent from SimCube to ES601 is not human-readable,

at least not easily (The numeric data of interest are interspersed with lower-case delimiter characters and the '!' character, which terminates the response. Additionally, peak pressure data from the overpressure test incorporates a decimal point at the correct position, but the required decimal point is missing in the numeric data which comprises a leak test result). However, it is possible for ES601 to extract the relevant characters from the SimCube response message and thereby regenerate a readable result on the ES601 display. This requirement is also pertinent to the ES601 'M' instruction.

### *3.3 Firmware Improvements to Support SimCube*

The following changes are made in ES601 display firmware Version 3.1:

3.3.1 The ***[*** character is used as a prefix for all SimCube native commands. Per the native protocol, this character is followed by a whitespace, then one or more ASCII letters and digits. The last character in all native commands is ASCII ***!***. The ***[*** prefix is not sent to SimCube, but triggers ES601 to perform alternate forms of communication with SimCube depending on the instruction type.

3.3.2 For ***C*** and ***R*** instructions, a ***[*** prefix character causes ES601 to send the command string to the right of the ***[*** character twice, with a delay of 200 msec between transmissions, and without carriage return terminator. SimCube response is ignored and is discarded if received.

3.3.3 For the ***M*** instruction, a ***[*** prefix character causes ES601 to send the command string to the right of the ***[*** character <u>*once*</u>, with no carriage return, followed by a 200 msec delay to allow the command string to be sent over the connection and to be recognized by the tester at the far end.

Unlike conventional remote command strings which *do not* begin with the ***[*** prefix, the ES601 does not monitor for a condition indicating receipt of a carriage return, since SimCube will never send one. Instead, ES601 monitors its serial receive buffer for receipt of a first character on the serial connection, which for SimCube will normally be a whitespace character. Receipt of a first character then causes ES601 to delay 250 msec. This delay is sufficient for any tester to send approximately 250 characters at 9600 baud, the first 40 of which would be stored by ES601 in its serial receive buffer, and the 210 remaining characters of which would then be discarded.

After the delay, ES601 checks the *second* character of the response message which is expected to be now waiting in the serial receive buffer. If the second character in the return data is ASCII ***b***, then the tester sending the data is assumed to be a SimCube. In this case, a function is called which further processes the SimCube return data for display. If the

second character in the response is not ***b***, the tester is likely not a SimCube and the return data are then displayed verbatim on the ES601 LCD. This feature allows the enhanced ***M*** instruction of Version 3.1 to be used with any tester that does not terminate transmitted measurement data with a carriage return.

If the second character of the response message is ***b***, the SimCube data processing function is invoked and checks the *ninth* character of the response message waiting in the serial receive buffer. If the ninth character is ASCII ***r***, the return data then relates to a SimCube leak test result. In this case, the *tenth*, *eleventh* and *twelfth* characters are copied from the return data to a human-readable string of the form:

"***xx.x*** **mmHg/min**"

where the "*xx.x*" digits correspond to the response characters identified above. If necessary, a leading zero will be blanked. This human-readable string is then shown on the ES601 display as the SimCube test result, which can then be retained in the Clip Record by pressing the "PASS" or "FAIL" key on the keypad.

If the ninth character of the response is not ***r***, the return data then relates to the peak pressure captured by SimCube during an overpressure test. In this case, characters *two through six* are copied from the return data to a human-readable string of the form:

"***xxx.x*** **mmHg**"

where the "*xxx.x*" digits correspond to the response characters identified above.

3.3.4 The ES601 sends the "IDENT\r" command to a tester in response to depression of the "CHECK CONNECTION" button on the ES601 tester port setup menu. In ES601 Version 3.0 and earlier, ES601 checks for receipt of a carriage return character as a message terminator, which then indicates that a response to "IDENT\r" has been received and can then be displayed on the LCD to the right of the "CHECK CONNECTION" button. This approach will not work with SimCube since this tester does not send carriage return characters. For Version 3.1, the "CHECK CONNECTION" function is revised to look for receipt of *any* characters from a tester in response to "IDENT\r", rather than a carriage return terminator specifically. This modification makes the "CHECK CONNECTION"

function compatible with testers like SimCube ***and*** with testers like Infutest, which send a carriage return at the end of the "IDENT" response.

### *3.4 Using ES601 Version 3.1 With SimCube*

In practice, a User-Defined Tester AutoSequence or Test Procedure intended for use with SimCube will incorporate ***C***, ***R*** and ***M*** instructions for remote control of the tester, with the designated ***[*** prefix character being included at the head of each command string as specified in section 2.3. Additionally, it is necessary to include in the sequence additional ***K*** instructions to direct the user how to interact with the SimCube as this is not entirely obvious from the SimCube display alone. For those sequences incorporating the SimCube leak test, a ***D*** (delay) instruction is mandatory and must be included at the appropriate step in the sequence.

Below is an example Tester AutoSequence demonstrating the key testing functions provided by SimCube, implemented entirely with the native protocol.

| Step | OpCode | Command | Prompt |
|---|---|---|---|
| 1. | K | | CONNECT MONITOR AND CUFF TO SIMCUBE |
| 2. | R | [ A! | RUN MONITOR & CHECK BP=120/80 |
| 3. | R | [ H! | RUN MONITOR & CHECK BP=190/120 |
| 4. | R | [ L! | RUN MONITOR & CHECK BP=80/40 |
| 5. | C | [ P! | SELECT SIMCUBE OVERPRESSURE TEST |
| 6. | K | | SET UP MONITOR FOR OVERPRESSURE TEST |
| 7. | K | | INFLATE UNTIL MONITOR RELEASES PRESSURE |
| 8. | M | [ P! | CHECK PEAK PRESSURE < 350 MMHG |
| 9. | C | [ E! | SELECT SIMCUBE LEAK TEST |
| 10. | K | | LEAK TEST: MANUALLY INFLATE TO 300MMHG |
| 11. | D | 60 | HOLDING PRESSURE. CHECK FOR LEAKS. |
| 12. | M | [ E! | CHECK LEAK RATE < 5 MMHG/MIN |
| 13. | K | | END OF SEQUENCE |

Per the SimCube native protocol, note the use of leading whitespace and terminating exclamation point in the command strings to the right of the ES601 ***[*** prefix character. In the above example, steps (2), (3) and (4) invoke preset NIBP simulations on the SimCube. Alternatively, the SimCube custom control message (<whitespace>C, ...) could be used to configure the NIBP simulation at these steps or at additional steps in the sequence (custom control messages are not shown in the above example). According to the design specification for ES601 V3.1, control commands to the right of ***[*** are sent *twice* to the SimCube, with no carriage return terminator.

Steps (5)-(8) define the recommended method for performing a SimCube overpressure test; the maximum limit appearing at step (8) is arbitrary and will depend on the specifications of the monitor under test. According to the design specification for ES601

V3.1, the command to the right of ***[*** at step (5) is sent by ES601 to SimCube *twice* to initiate the test, using the C instruction (Op) code. Later at step (8), using the M instruction code, the same command is then used to retrieve the final test result obtained. Per the design specification, at step (8) the command to the right of ***[*** is sent by ES601 *only once* to SimCube, with no carriage return terminator, and the received response is processed to produce a test result of the form ***xxx.x*** **mmHg** which is then displayed on the ES601 LCD.

Steps (9)-(12) define the recommended method for performing a SimCube leak test; the inflation target appearing at step (10) and the acceptance limit at step (12) are arbitrary and will depend on the specifications of the monitor under test. Step (11) introduces the necessary delay for executing the test, the delay being arbitrarily set to 60 seconds in this example. According to the design specification, the C Op code command to the right of ***[*** at step (9) is sent by ES601 to SimCube *twice* to initiate the test, and later at step (12) the same command is then used to retrieve the final test result obtained. Per the design specification, at step (12) the M Op code command to the right of ***[*** is sent by E601 *only once* to SimCube, with no carriage return terminator, and the received response is processed to produce a test result of the form ***xx.x*** **mmHg/min** which is then displayed on the ES601 LCD.

# 4. Clinical Dynamics AccuPulse

## *4.1 Hardware Requirements*

The "COM1" port of Clinical Dynamics AccuPulse can be connected to a Tester Port of ES601 using a RJ12 cable (P/N 3140-429) and a DB9M adaptor (P/N 3140-427).

The RS232 serial protocol for AccuPulse is 9600 baud, no parity, 8 data bits and 1 stop bit (9600,N,8,1). ES601 currently provides this setting for tester communication.

## *4.2 Control Command Requirements*

The AccuPulse remote control protocol is somewhat more complex than the SimCube native protocol.

ES601 Version 3.1 has been designed to operate with AccuPulse firmware version *AP/2.01-16-Mar-07*.

The AccuPulse has these unique communication characteristics which must be accommodated by ES601:

4.2.1 Many AccuPulse commands require termination with a carriage return character, however, some commands will return a response to ES601 before ES601 has finished sending the carriage return. This is usually not

a problem for the ES601 firmware, and where this anomaly is problematic, it can be overcome through appropriate ordering of remote control instructions and non-remote control instructions in an AutoSequence or Test Procedure.

4.2.2 The AccuPulse "mode" command actually requires *five* characters plus carriage return to be sent by ES601, not four characters as specified in the Clinical Dynamics documentation. It turns out the fifth character of the command is arbitrary, but it *must be sent* before the carriage return. This is not a problem for ES601 firmware in particular, as the content of command strings sent by RS-232 is arbitrary. However, users will find this anomaly particularly frustrating if they are not aware of it, as the ES601-AccuPulse interface cannot be made to work unless "mode" commands comprising five characters are correctly input via the *ES601pc* instruction editor.

4.2.3 Several AccuPulse commands have a significant execution time of one to two seconds, for example, commands which switch the general operating mode of the tester from simulation to some other test function or visa versa. Selection of a NIBP "cal table" also requires significant execution time. In response to such commands, AccuPulse will eventually return a '>' response character once the mode switch has been effected, or the cal table has been loaded into memory. If a 'C' instruction is used to send such commands to AccuPulse, ES601 Version 3 and earlier firmware will timeout after 250 milliseconds and display a "NO RESPONSE FROM TESTER" message on the LCD. For Version 3.1, the 'C' instruction needs to be made more flexible in dealing with testers that return an immediate response to a command (for example, Oxitest), in addition to testers that require a few seconds to execute a command (for example, AccuPulse).

4.2.4 Several control functions provided by AccuPulse, in particular those functions associated with the NIBP simulation, require the ES601 to send *two* commands to configure *one* function, for example, to select a BP preset or set a simulated heart rate. This is possible in ES601 Version 3.0 firmware and earlier, but it requires two sequence instructions to implement control of a single setting on the simulator. To more efficiently support remote control of AccuPulse, ES601 Version 3.1 therefore requires some means of defining a double-command within a single 'C', 'R', or 'M' instruction.

4.2.5 AccuPulse is advantageous in that it provides completely automated tests for measurement of overpressure and air leakage. However, upon completion of such tests AccuPulse outputs a multiple-line test result via RS-232, where the *last* line of the response actually contains the desired information. This is a problem for ES601 Version 3.0 and earlier firmware, which will capture and display only the *first* line of a multi-line response, ignoring any subsequent lines that may be sent. To capture the most

relevant line of data in an AccuPulse test result, the 'M' instruction has been revised.

4.2.6 AccuPulse provides a mode of operation designated "streaming manometer". Given the alternative methods provided by AccuPulse for measuring overpressure and leakage, the streaming manometer feature is not particularly useful except for perhaps checking pressure calibration of a NIBP monitor at a number of preset levels. It turns out that revisions to the 'M' instruction required by (3.2.5) also make ES601 Version 3.1 inherently compatible with the streaming manometer function. As a streaming manometer, AccuPulse periodically transmits pressure measurement data in real-time via RS-232; such data can be interpreted as a multi-line test result where the last line sent (i.e. the current pressure measurement value) is the most relevant and is the one which is displayed by ES601 on the LCD.

4.2.7 Shortly after powering up AccuPulse, the tester will automatically output its firmware version via the RS-232 port along with a message describing self-test results. Following this initial output after power-on, the tester is then *nearly* ready for remote control operation. With the tester in this state, it is necessary to first send a carriage return to AccuPulse to initiate bi-directional communication via the RS-232 connection. Once the initial carriage return has been received, AccuPulse will return a '>' character as an acknowledgment and will then respond to remote control commands per the Clinical Dynamics "draft" specification.

It is not possible for ES601 to send only a carriage return as a command, however, sending a single ASCII character to AccuPulse followed by carriage return will produce the same effect of initiating RS-232 communications, provided the character sent is <u>*not*</u> ***A***. This requirement does not relate to ES601 firmware, but otherwise mandates a ***C*** instruction which must be included as the first step in any AccuPulse AutoSequence or Procedure. Provided this "wake up" command is <u>*not*</u> "A\r", it can be sent to AccuPulse at any time; if sent after power-on the command will enable RS-232 communications as described; if sent after a number of other commands have been sent, it will have no effect on the tester other than to cause AccuPulse to return a '>' character.

This is another anomaly which is easily overcome without a firmware change, but which users might find frustrating if they are not aware of it.

### *4.3 Firmware Improvements to Support AccuPulse*

The following changes are made in ES601 display firmware Version 3.1:

4.3.1 For ***C, R*** and ***M*** instructions, a ***{*** character is used as a prefix for all

AccuPulse operations that require transmission of two commands to the tester, including NIBP simulations and special automated test functions such as overpressure and leakage measurements. In instructions having a command string prefixed by ***{***, the ***}*** character then is used within the string to delimit the first and second commands. The result then is a command string having the apparent format:

**{**<*first_command*>**}**<*second_command*>

Note the ***{ }*** characters are not sent to AccuPulse, they only serve to delimit the first and second commands to be transmitted. ES601 extracts <*first_command*> from the instruction, appends a carriage return and sends the result to AccuPulse. After waiting 250 msec, ES601 discards any response which may have been received from AccuPulse, and then sends <*second_command*> after terminating this latter command with a carriage return.

The behavior of ES601 following transmission of <*second_command*> depends on the instruction type. For ***C*** instructions, ES601 monitors for a response as described in 4.3.2 below. For ***R*** instructions, no special action is taken and any data returned in response to <*second_command*> is ignored. For ***M*** instructions, ES601 delays 250 milliseconds and erases whatever may have been returned from AccuPulse over that short preliminary interval. This action is necessary since AccuPulse returns an initial response acknowledging <*second_command*> as received, but this initial response is an acknowledgment only and is not the desired test result being requested by the ***M*** instruction.

4.3.2 Monitoring of tester response to a command sent by a ***C*** instruction is enhanced in Version 3.1 to accommodate fast-responding testers like Oxitest, as well as testers having longer command execution time, such as AccuPulse. After transmitting a command (or two commands for a ***C*** instruction with ***{ }*** delimiters per 4.3.1), ES601 monitors for a tester response over a 250 msec interval. During this initial waiting interval, the ES601 display will not appear to have changed since completion of the immediately preceding instruction in the sequence. If a tester response is detected within the 250 msec timeout, ES601 automatically advances to the next instruction of the sequence without update of the LCD or user intervention. It is not necessary for the tester response to include a terminating carriage return for ES601 to proceed to the next instruction.

If no response is received within the initial 250 msec timeout, ES601 then enters a longer waiting period of up to two seconds. At this point, the user will now see the instruction TASK on the ES601 LCD so that the reason

for such waiting becomes evident. However, no keypad is provided and the user will not be able to advance the sequence until this subsequent timeout expires after two seconds of non-response. If a response is received within the two-second interval, ES601 automatically advances to the next instruction of the sequence without user intervention. It is not necessary for the tester response to include a terminating carriage return for ES601 to proceed to the next instruction.

If no tester response is received after a total waiting time of 2.25 seconds, ES601 then displays "NO RESPONSE FROM TESTER" along with the BACK / NEXT keypad on the LCD, allowing the user to advance the sequence manually if desired.

4.3.3 The ***M*** instruction is enhanced to accept multiple lines of measurement data returned in response to a transmitted command. In order to be individually recognized within a multi-line result, each line of text must be terminated by carriage return.

When a complete line of text is received, ES601 copies the line to the "DATA" field on the LCD display and prepares itself to receive a subsequent line. If a new line is received, it is copied to the "DATA" field on the LCD, overwriting the previous line posted there. Thereby, for testers such as AccuPulse which send multiple lines of data in response to a command, only the final line sent will be apparent to the user in the "DATA" field. As in earlier firmware versions, the ES601 does not advance the sequence in response to tester data received during execution of a ***M*** instruction, but waits until the user acknowledges information appearing in the "DATA" field by pressing the "PASS" or "FAIL" key on the keypad.

This approach is backward-compatible to previous testers such as Infutest and Phase 3, which return only a single line of text in response to a data-request command. The enhanced ***M*** instruction can also be used with testers that periodically send measurement data in real-time, such as when AccuPulse is commanded to operate in streaming manometer mode. In this mode, AccuPulse measures the inflation pressure applied to its input every 330 milliseconds, and sends this measurement data as a string of ASCII digits with decimal point, terminated by carriage return. This real-time measurement data will appear in the "DATA" field provided by the ***M*** instruction display; pressing one of the keys on the display causes the last value sent by the tester to be retained in the Clip Record and the sequence to advance to the next step.

### *4.4 Using ES601 Version 3.1 With AccuPulse*

In practice, a User-Defined Tester AutoSequence or Test Procedure intended for use

with AccuPulse will incorporate ***C***, ***R*** and ***M*** instructions for remote control of the tester, with the designated ***{ }*** delimiters being included in certain instructions requiring transmission of double-commands. Additionally, it is necessary to include in the sequence additional ***K*** instructions to direct the user how to interact with the AccuPulse as this is not entirely obvious from the AccuPulse display alone. Additionally, a "wake up" ***C*** instruction is necessary near the beginning of the sequence, to ensure the sequence will run correctly with an AccuPulse which has just been powered up (see 4.2.7).

On the following page is an example Tester AutoSequence demonstrating the key testing functions provided by AccuPulse.

| Step | OpCode | Command | Prompt |
|---|---|---|---|
| 1. | C | T | WAKE UP ACCUPULSE REMOTE CONTROL |
| 2. | K | | CONNECT MONITOR AND CUFF TO ACCUPULSE |
| 3. | C | APB20 | SELECT ACCUPULSE NIBP SIM MODE |
| 4. | C | {APB10}APY03 | SELECT DINAMAP CAL TABLE |
| 5. | R | {APB10}APCA4 | RUN NIBP & CHECK BP=120/80 |
| 6. | R | {APB10}APCA6 | RUN NIBP & CHECK BP=200/150 |
| 7. | R | {APB10}APCA2 | RUN NIBP & CHECK BP=80/50 |
| 8. | C | {APB10}APCA4 | RESET SIMULATOR BP TO 120/80 |
| 9. | R | {APB10}APD150 | RUN NIBP & CHECK HR=150BPM |
| 10. | R | {APB10}APD040 | RUN NIBP & CHECK HR=40BPM |
| 11. | C | {APB10}APD080 | RESET SIMULATOR HR TO 80BPM |
| 12. | C | APB40 | SELECT ACCUPULSE LEAK TEST MODE |
| 13. | K | | SET UP MONITOR FOR PRESSURE LEAK TEST |
| 14. | M | {API1}APJ1 | VERIFY PRESSURE DROP < 5MMHG |
| 15. | C | APB50 | SELECT ACCUPULSE OVERPRESSURE TEST |
| 16. | K | | SET UP MONITOR FOR OVERPRESSURE TEST |
| 17. | M | {API1}APJ1 | VERIFY PRES. RELEASE < 375MMHG |
| 18. | C | APB40 | SELECT ACCUPULSE LEAK TEST MODE |
| 19. | K | | SET UP MONITOR FOR PRESSURE CAL CHECK |
| 20. | M | {API0}APU2 | APPLY PRESSURE & CHECK CAL |
| 21. | C | {APU0}T | DISABLE REAL-TIME PRESSURE DATA |
| 22. | K | | END OF SEQUENCE |

In the above example, step (1) transmits "T\r" to AccuPulse. This causes the tester to enable RS-232 communications immediately following a power up. This step will have no effect on AccuPulse if remote control operation has previously been initiated with the tester (i.e., by running this ES601 AutoSequence at an earlier time).

Steps (3)-(8) show some commands relevant to blood pressure simulation. Note the mode command "APB*x*" as specified in the Clinical Dynamics draft documentation actually requires *five* characters "APB*x?*" plus carriage return to be correctly recognized

by AccuPulse, where the *?* "filler byte" can be any ASCII character. In the above example, a '0' has been used for the fifth character. Note that several other preset values of blood pressure may be optionally selected via the "APCA*y*\r" command. Simulated blood pressure can also be varied from the preset value via the BP Shift command "APE*zzzz**r*" (not shown in the above example). Other than the ***C*** instruction at step (3) which sets the tester to NIBP simulation mode, instructions at steps (4)-(8) all employ double-commands, where the first command sent ("APB10\r") prepares AccuPulse to receive "download data", i.e., a simulation parameter setting.

Steps (9)-(11) show commands for changing the simulation heart rate. Additional instructions could be added if observation of monitor performance at other heart rates is desired. As shown, double-commands are required for setting the rate.

Steps (12)-(14) show instructions required for performing an automated leak test with AccuPulse. Step (12) changes the operating mode of AccuPulse from NIBP simulation to leak test. Step (13) prompts the user to configure the monitor under test for a leak test; normally, this is done through a service mode provided by the operator interface of the monitor, which will provide a menu item for closing the deflate valve internal to the monitor. At step (14), AccuPulse responds to the double-command by automatically inflating the attached monitor and cuff by means of an internal air pump. AccuPulse will then monitor the applied pressure over a 60-second interval. At the end of this interval, a multi-line test result is transmitted from AccuPulse and the test then is completed. The third and final line of the transmitted information contains the measured leak rate in mmHg/min, which is displayed by ES601 in the "DATA" field on the LCD.

Steps (15)-(17) show instructions required for performing an automated overpresure test with AccuPulse. Step (12) changes the operating mode of AccuPulse from leak test to overpressure test. Step (13) prompts the user to configure the monitor under test for an overpressure test; normally, this would be the same configuration as used for a leak test. At step (17), AccuPulse responds to the double-command by automatically inflating the attached monitor and cuff by means of the internal air pump. AccuPulse monitors the increasing applied pressure until the relief valve internal to the monitor operates, at which point the pressure rapidly falls to zero. Once the relief valve is actuated (or if AccuPulse reaches its maximum limit of 400mmHg), a multi-line test result is transmitted from AccuPulse and the test then is completed. The second line of the transmitted information contains the measured peak pressure in mmHg, which is displayed by ES601 in the "DATA" field on the LCD.

Steps (18)-(21) show how AccuPulse can be configured to operate as a streaming manometer. Step (20) initiates real-time data transmission of measured pressure from the AccuPulse, at a rate of one reading every 330 milliseconds. This real-time data appears on the ES601 LCD in the "DATA" field, until the user chooses to advance the sequence via the "PASS", "FAIL", "BACK" or "SKIP" keys on the keypad. Step (21) inhibits the output of real-time pressure data from AccuPulse.

The ES601 display firmware, version 3.1, has been improved to deal with the special

requirements of RS-232 control of specific test instruments: the AMPS-1, the SimCube and the AccuPulse. The use of special characters to invoke an internal ES601 command processor allows the ES601 to accommodate the unique command structures of these devices, and potentially other devices with similar command structures.