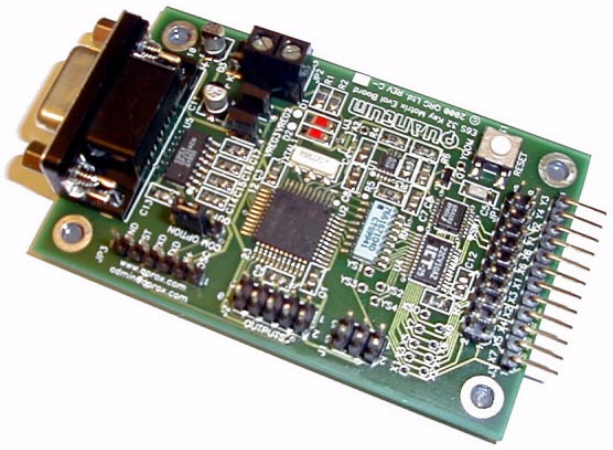
QMatrix™

E6S3 32-KEY MATRIX EVAL BOARD

SETUP AND OPERATION

16 November 2001



Quantum Research Group Ltd.

Rev 3.01/1101

GETTING STARTED - PLEASE READ THIS!

1. Adhere the 'Schurter' flex-matrix to a dielectric surface (e.g. the clear plastic panel included) by peeling off the backing paper and 'rolling' the flex circuit onto the plastic panel, rubbing the flex circuit down as you go to prevent air bubbles from forming; be very careful not to stress the matrix tail ribbon cable connector to avoid damaging the silver ink tracks!

   The matrix will not operate properly well without an overlying panel. The panel is an integral part of the charge-transfer mechanism that allows it to work; the plastic lets the sense fields 'mix' properly before they reach the surface. You can use panels of up to 2" thick glass with the 'Schurter' circuit.

2. Plug in the Schurter circuit tail into the board. Be careful to observe the pin 1 polarities of the cable and the boards.

3. Elevate the assembled panel from your work surface, plastic-side up, by 0.5cm or more, using rubber feet or objects at the corners or along the edges. *Do not place anything directly under a key pad area that could touch it*.

4. Connect a 9V battery clip (or other 8 to 12V DC power source) to the 2-terminal screw block.

5. Connect a RS232 serial extension cable (included) from the E6S3 board to an open serial com port on a PC. Com1 or Com2 can be used.

6. Start QmBtn on the PC, and power-up the board with a fresh 9V battery or a 8-12V DC supply. One of the LED's will glow brightly for a moment while the unit calibrates. This normally takes only a fraction of a second; do not touch the keys while the calibration is occurring or it will get a bad calibration point. (If this does happen, the QT60320 IC will automatically correct the error after 10 seconds).

7. The unit will begin to run normally after the LED goes out. If the E6S3 has been set up properly (see next page) an LED will flicker when a key is touched through the panel, and the QmBtn software will show the key currently being touched. The software also marks the last key touched with an 'X'; the 'X' is persistent and does not indicate detection *per se*.

ABOUT E6S3 TECHNOLOGY

E6S3 technology represents a significant advance in the science of capacitive key sensing. E6S3 builds on 7 years of continuous development in charge-transfer ("QT") sensing.

E6S3 makes use of a 2-electrode charge transfer mechanism, unlike Quantum's single-electrode sensing systems; just like a conventional scanned keyboard, a set of drive lines generates signals used to address the keys, while another set of lines are used to 'read out' the key signals. At the intersection of each drive and read line exists a 'key' defined as an interleaved or adjacent set of electrodes across which is coupled a tiny electric field.

Matrix sensing of this type is not new. What makes E6S3 technology revolutionary is the manner in which the signals are sensed at the 'front end', and how the signals are digitally processed thereafter. Instead of looking at the received voltage amplitudes or decay constants, E6S3 actually measures the amount of charge that has been transferred across each key electrode, using a 'virtual ground' charge detector. Charge cancellation methods are also applied to increase available dynamic range, allowing keys to have variable sizes and shapes within a matrix.

E6S3 uses wide dynamic range digital technology. It is inherently stable, since it does not rely on RF or high-gain circuits. The drive signals are digital pulses, while the receive circuitry uses simple analog switches and a low-cost CMOS opamp.

Algorithms built into E6S3 provide "high survivability" adaptive signal processing. These include Drift Compensation, Max On-Duration, Median Filtering, Spread Spectrum acquisition, variable Burst Lengths, and digitally controlled variable thresholds. Together these functions solve almost all the problems commonly associated with capacitive sensing.

E6S3 Hardware and Algorithm Features

Gated Summing-Junction Charge Sampling. Each key is sensed with a summing-junction charge detector that is narrowly sampled around the rising edge of the matrix drive signal. This produces a time-sampling that mirrors the use of high frequencies in the frequency domain. The net effect is a significant suppression of signals due to water films, since such films have a strong low-pass characteristic that does not pass high frequencies easily.

Low Impedance Sensing. The charge receiver is a low impedance 'summing junction', which helps enormously in reducing the effects of external RFI and surface contamination.

Individual Key Calibration. Each key is calibrated automatically on power up, independent of all other keys. This means that you do not have to worry about matching key sizes, shapes, or the overlying material thickness or type of material. You can physically place keys in scattered locations and under differing materials of different thicknesses. You also do not have to worry about wiring or trace lengths to the keys; all these aspects are automatically taken into account.

N-Key rollover. Any combination of keys can be simultaneously sensed. This is an inherent function of the charge-transfer sensing technique itself. Thus, making 'Shift' or 'Interlock' functions are easy to implement.

Auto Drift Compensation. The E6S3 is able to compensate for changes in signals caused by moisture, thermal drift, etc., yet remain fully capable of detecting touch. This feature operates silently and seamlessly together with all other processing features.

Stuck Key Recovery. The E6S3 will recalibrate automatically after 10 seconds of continuous detection at a particular key, and then continue to sense as though nothing had happened. This allows a foreign object to be placed on top of a

key without having the sensor permanently 'lock up'. After the timeout, the key recalibrates automatically and continues to function normally.

Variable Burst Length. Each key has its own burst length setting. Burst Length relates directly to the sensitivity of each key. It may be desirable to make some keys more sensitive than others; for example larger keys may need to be reduced in gain and smaller ones increased for best touch performance. Once the Burst Length is set for a particular key, it is stored in internal e^2prom.

Variable Thresholds. Each key has its own threshold level which can be altered individually and programmed into internal e^2prom. Together with the Burst Length parameter, this allows for tremendous fine-tuning.

Consensus Noise Filtering. Each key result is sampled 5 times and a consensus result of the signal is determined. This makes for a highly efficient impulse noise filter, dramatically reducing false detections caused by electrical interference.

QmBtn Software Notes

QmBtn Installation

Install the software by transferring the files to a subdirectory of your choice on your PC's hard drive. You may want to create a shortcut to the installed QmBtn.exe file on your desktop or in a folder.

Make sure the com port you intend to use is totally free of conflicts from mouse drivers, fax software, other modem programs and TSR's from programs like Laplink and Palm Pilot Sync Manager. A conflict will interfere with the program. If in doubt, delete the offending programs, or use another PC. QmBtn works under Win 98, NT, and Windows 2000.

QmBtn Operation

1. Power up the board. Start QmBtn and wait for it to locate and communicate with the Switch Matrix: the text "Capacitive Matrix Switch" is displayed when the Matrix has been identified. It will search for the E6S3 board on Com1 and Com2 automatically.

2. Select **Settings | Button Settings**... to view Gain and Sensitivity settings for a particular key. A second dialog box appears. These numbers can be altered on a key-by-key basis. This will also display the reference point and signal level for the key selected using a horizontal signal indicator.

3. Click on a particular matrix button or on the Slider (rightmost strip) in the viewer window to see the settings for that key in the **Gain & Sensitivity (G&S)** dialog - The Key Coordinate text at the top of G&S changes as you click on different buttons. At the same time, the G&S numbers are updated too, to reflect the current G&S for the key you have clicked on.

4. To change the G&S settings for a particular key, either type the new value directly into the G&S edit boxes or use the small scroller buttons. New G&S settings are uploaded into the E6S3 board after a brief delay (no other action is necessary). The QT60320 will completely recalibrate all of its keys after even one button has been reprogrammed for Gain. The new G&S settings are stored in the E6S2's internal e^2prom.

5. When a key's Sensitivity (threshold level) is altered, the new setting becomes immediately applicable and is stored to the E6S2's eeprom. No recalibration takes place, and the Cal LED does not glow in response.

6. **"Apply to All"** button sets the G&S settings (displayed in the two edit boxes) to all buttons on E6S3. You can then individually tailor G&S on other keys separately from the bulk of them.

7. **File | Save** and **File | Save As** options in the main menu allow you to save the G&S settings of all key combinations to disk. **File | Open** will allow you to retrieve these settings; they will be automatically downloaded into the current E6S3 board plugged into your PC.

The Button Settings dialog can be left open until all setting changes have been completed. It has no impact on the normal operation of the key viewer.

Gain and Threshold

- The Gain number relates to the amount of signal level available for processing. It actually controls the number of charge-transfer cycles in each burst, from 4 to 255; this in turn is directly relates to the amount of signal recovered from each switch: larger Gain numbers make a particular key more sensitive. The normal usable range of Gain setting is from 10 to 40. Above 40, with large keys it is possible to saturate the signals for that key. The QT60320 multiplies the Gain figure by 2 to set the burst length: so, setting Gain to 10 will create burst lengths of 20 QT cycles / burst.
- The ability to alter gain on a key-by-key basis is an extremely powerful feature of the E6S3. It permits you to design some keys larger than others, or to alter the size and amount of interleaving between electrodes at the key surface. So, you can make very small buttons and very large ones, and even though the E-field coupled to the finger is normally larger with the larger button, the gain setting for that button can be lowered to reduce it back to a normalized level.
- Threshold relates to the number of counts of signal strength required to create a detection. The higher this number is, the lower the sensitivity.

Gain, Threshold, and Drift Compensation – How They Interact

- **Gain** relates to the amount of signal level available for processing. It actually controls the number of charge-transfer cycles in each burst, from 4 to 256; this in turn is directly proportional to the amount of signal recovered from each switch. The Gain figure is multiplied by 2: A setting of 10 means a burst length of 20.
- **Threshold** (Sensitivity) interacts 100% with gain, and produces an almost indistinguishable effect from a behavioral viewpoint. If you double Gain, you make a key twice as sensitive from a signal perspective (although for 1/r reasons it may not *seem* twice as sensitive). If you cut Threshold in half, you also appear to make the key twice as sensitive.
- It is not a good idea to set Threshold to very low numbers, for example under 8. Low settings of Threshold will interfere with the setting of an appropriate level of signal hysteresis, and the sensor will also become more susceptible to electrical noise.
- The "**Drift Compensation**" algorithm is based on a slew-rate limited change in the reference level for each key. The slew rate is defined as a fixed number of seconds per change in the reference level count; the change in the reference level count is always just +/- 1 count. If you turn a key's Gain higher, the Drift Compensation algorithm ends up having less of an effect, since while the recovered signal will drift around faster with a higher Gain setting (as in any sensor system), the Drift Compensation algorithm is fixed at a certain number of seconds between compensating counts. Higher Gain settings can cause the sensor to momentarily false-detect or become less sensitive to touch, depending

on the underlying rate of signal drift (due to moisture, thermal swings, etc.), if the Drift Compensation mechanism can't keep up.

For this reason in 'drifty' environments it is better to turn Gain to a lower number if you can, and set the Threshold number lower to compensate key sensitivity for the lower Gain.

On the other hand, if Threshold is set *too* low to make a key more sensitive, signal drift can more easily cause a false trigger as the Drift Compensation algorithm may not keep up. So there are tradeoffs at certain points between Threshold and Gain, with an optimum setting 'somewhere'. There are no hard and fast rules, and experimentation and testing are advised.

Serial Protocol

The E6S3 uses an on-board MAX232 type RS232 driver, with software handshaking. There are no other outputs from the E6S3 board. The E6S3 responds to 'Evoke Codes' from the host on a polled basis.

Baud Rate: 9600
Parity: None
Length: 8 bits
Stop bits: 1

Evoke Byte Summary

ASCII Code (from host)	Hex Code	Purpose
s	**0x73**	Button State. Returns 4 bytes of on/off status for all buttons
S	**0x53**	Identification command. QT60320 responds with signature "3200002"
e	**0x65**	Error reporting. Returns 4 bytes corresponding to error bits for each key
/dXY	**0x2F 0x64 X Y**	Data Reporting. Returns signal and reference for a chosen key at location **X Y**
/eA	**0x2F 0x65 A**	Settings Reporting. Returns Gain and Threshold levels for chosen key at A
/EAv	**0x2F 0x45 A v**	Settings Write. Writes the Gain or Threshold value v for the chosen key at A
Ov	**0x4F v**	Port Write. Writes byte value **v** to the user output pins O1..O8
I	**0x49**	Port Read. Reads back the 4 bits of user port pins I1..I4

Important Note: None of the commands are to be followed by a CR or any other terminating character. Likewise, none of the responses includes a CR or other terminating character.

s Command – Button State

(0x73) After an 's' character is received, the E6S3 reports back the touch-state of the buttons; it responds by transmitting back 4 bytes of data containing a total of 4 x 8 = 32 data bits, where each bit represents the state of one button.

Each bytes holds the state of 8 buttons in a row along one Y line. The first byte corresponds to the top row (Y1) of the matrix and the 4th byte corresponds to the bottom row (Y4). Bit 0 of each byte holds the state of the matrix button corresponding to X1 and bit 7 holds the state of the matrix button corresponding to X8.

Response to 's' from E6S3 is an 8 byte stream as follows:

BIT #	7	6	5	4	3	2	1	0
BYTE #								
1	X8/Y1	X7/Y1	X6/Y1	X5/Y1	X4/Y1	X3/Y1	X2/Y1	X1/Y1
2	X8/Y2	X7/Y2	X6/Y2	X5/Y2	X4/Y2	X3/Y2	X2/Y2	X1/Y2
3	X8/Y3	X7/Y3	X6/Y3	X5/Y3	X4/Y3	X3/Y3	X2/Y3	X1/Y3
4	X8/Y4	X7/Y4	X6/Y4	X5/Y4	X4/Y4	X3/Y4	X2/Y4	X1/Y4

Multiple key presses will appear as multiple bits in the data stream.

Example: The key at intersection X5 / Y2 appears at bit 4 in the second byte. The bit is active low, so if the key is touched the bit is cleared (as viewed at QT60320 pin 15). All untouched bits will register as a '1' (high). RS232 data is inverted, so the transmitted data stream on the serial cable is 'active high'.

S Command - Identification Reporting

(0x53) Identification command: The E6S3 board responds to this with the 7-byte ASCII signature "3200002".

e Command - Error Reporting

(0x65) This command allows the host to retrieve the error status for all the buttons, in 4 bytes.

Error reporting bit mapping -

BIT #	7	6	5	4	3	2	1	0
BYTE #								
1	X8/Y1	X7/Y1	X6/Y1	X5/Y1	X4/Y1	X3/Y1	X2/Y1	X1/Y1
2	X8/Y2	X7/Y2	X6/Y2	X5/Y2	X4/Y2	X3/Y2	X2/Y2	X1/Y2
3	X8/Y3	X7/Y3	X6/Y3	X5/Y3	X4/Y3	X3/Y3	X2/Y3	X1/Y3
4	X8/Y4	X7/Y4	X6/Y4	X5/Y4	X4/Y4	X3/Y4	X2/Y4	X1/Y4

The appropriate bit (or bits) is set (high) when an error has been detected at the corresponding button(s). The error may be caused either by a short across the button or an open circuit at the button.

/dXY Command - Data Reporting

(0x2F 0x64 X Y) Reports back the signal level and reference value for the key chosen. X and Y are zero- referenced binary (not ASCII) values in the range 0..7 to indicate the button for which data is requested. If X=0 and Y=0, the device reports back with the data for key X1 Y1. If X=3 and Y=1, the key selected is X4 Y2. The last key is X8 Y4, so the highest /dXY command is /d73 where '7' and '3' are in binary.

The device reports back with two bytes: the first is the reference level, the second is the signal level. Both are binary bytes of range 0..255.

/eA Command - Settings Reporting

(0x2F 0x65 A) Allows the gain and threshold settings for a specific button to be read back to the host. The full command is **/eA** where **A** is a single binary byte specifying the address of the data to be read. Separate commands must be issued to read the gain and threshold for each button as these values are held at separate addresses, with the data for the threshold being held in the address immediately following that for the gain.

This function operates very much like a 'PEEK' command in Basic with respect to the data storage in the QT60320.

The gain and threshold settings for all buttons are held in a byte array, starting at address 100 (decimal). The first two sequential bytes in this array hold the data for button X1Y1, the next two bytes hold the data for button X2Y1 and so on. The gain is held in the first byte (of lower address) and the threshold in the second byte.

The button memory addresses are calculated as follows for a given key:

1. Multiply (Y-1) times 16 to get M. So, if the key is on Y3, M = (3-1)*16 = 32 (decimal)
2. Multiply (X-1) times 2 to get N. So, if the key is on X6, N = (6-1)*2 = 10 (decimal).
3. Add N + M +100 to get the first of the two memory addresses for the key.

Example: A key located at X4, Y3 would have its base address at:

(3-1)*16 + (4-1)*2 + 100 = 32 + 6 + 100 = 138 (decimal).

The first byte, at address 138 decimal is the gain, while address 139 has the threshold for X4 Y3. The address must be sent as a binary number (packed into 8 bits, i.e. 0x8a and 0x8b in this example), NOT as the ASCII string '1 3 8'.

The QT60320 reports back with a single binary byte containing the data requested.

EEPROM Read Access via /eA

It is possible to use part of the QT60320's internal eeprom as 'user storage'. This feature allows the elimination of a separate eeprom associated with a host MCU, reducing cost. The QT60320 has 86 bytes of spare eeprom available to the user for any function whatsoever. The eeprom can be read and written using the same /eA and /EAv commands used to examine and write key settings.

The first byte of eeprom is located at address 170 (decimal) and can be read via the command string /eA where A is the binary byte 170 (0xaa). The highest location is at 255 (0xff).

/EAv Command - Settings Write

(0x2F 0x45 A v) Allows the gain and threshold settings for a button to be written. **A** is a single binary byte specifying the address and **v** is a single binary byte value of the data to be written.

The addresses are calculated in an identical manner to those for the /eA command above. Note that a change in the gain value of even one key will cause the E6S3 to recalibrate all keys.

EEPROM Write Access via /EA

See notes under /eA for a more detailed description of this function.

The /EAv command can be used to write a byte to any of the 86 available internal eeprom locations. Like the /eA command, byte 'A' is the address of the byte from 0xaa to 0xff. The byte 'v' should contain the 8-bit binary data to be written to the address'A'.

Ov Command - Port Output Write

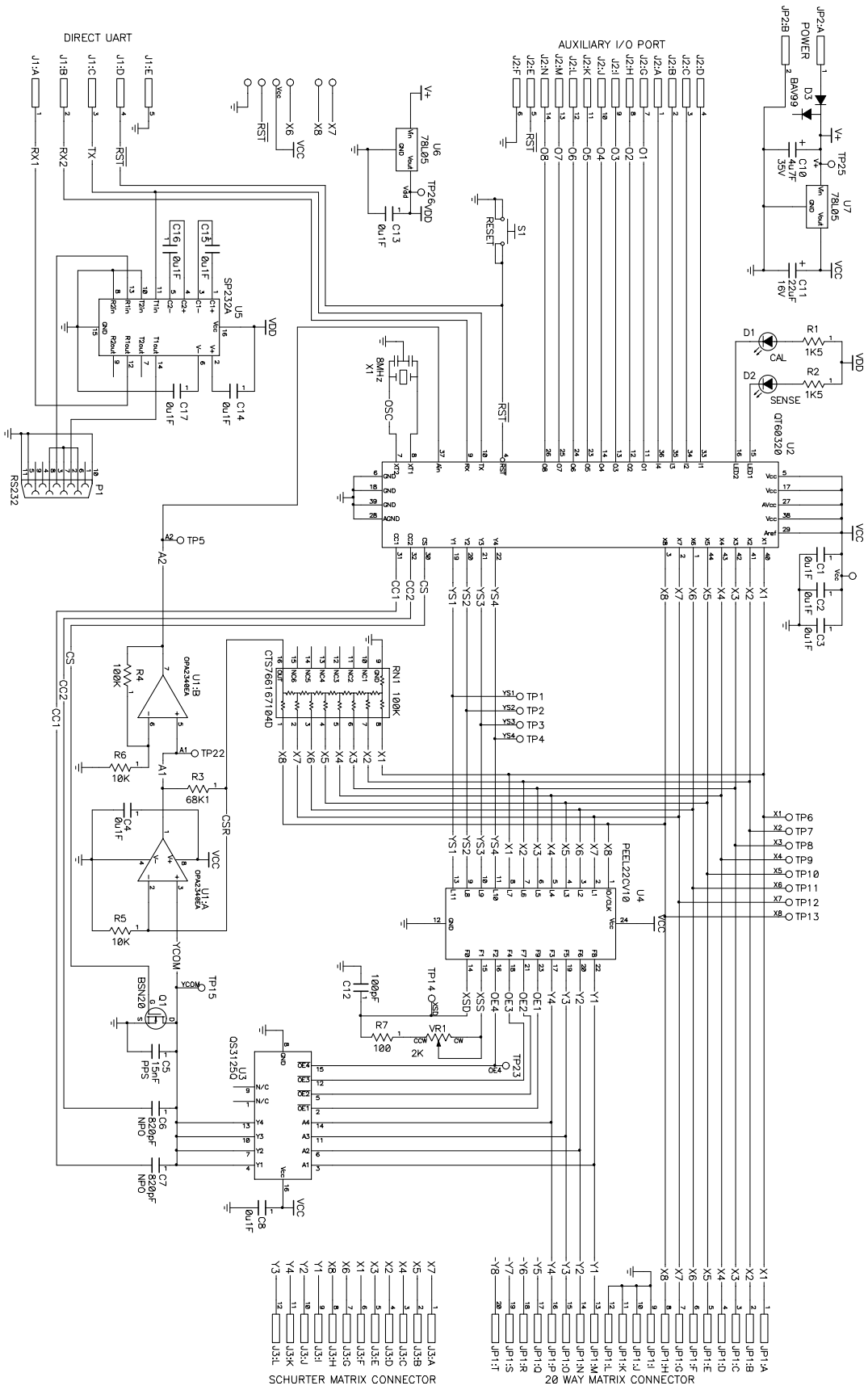
(0x4F v) Writes the value 'v', an 8-bit binary byte, to the QT60320's 8 output port pins O1…O8.

This feature can be used to drive LED's, a self-oscillating acoustic sounder, or other peripheral device. It can even be used in conjunction with the 'I' command (below) to scan a set of external electro-mechanical keys (up to 32 switches, e.g. in an 8x4 matrix) near the panel.

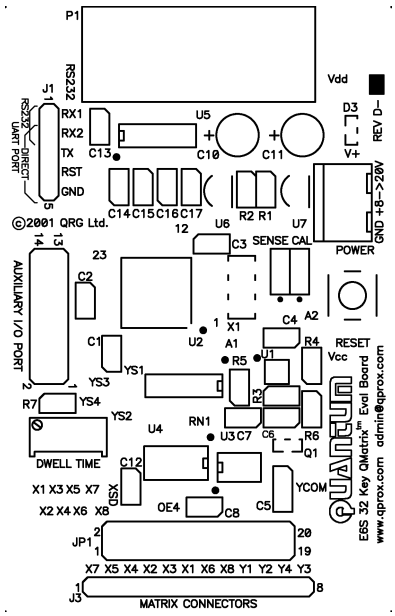
I Command - Port Input Read

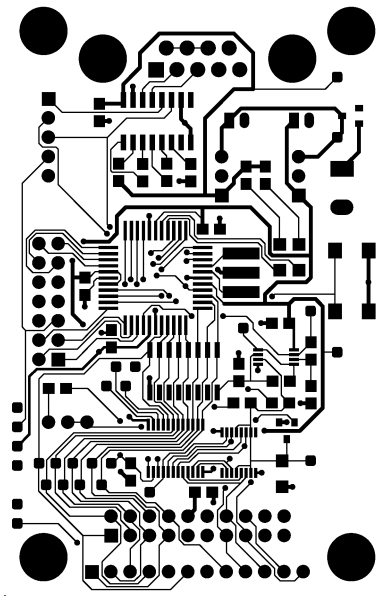
(0x49) Causes the QT60320 to return a binary byte from the port pins I1…I4. The value is returned in the lower nibble (bits 0,1,2,3) of the return byte. The high 4 bits are held at zero. Note that these 4 bits are available on the QT60320 itself but are not wired to any connector on the E6S3 in this release.

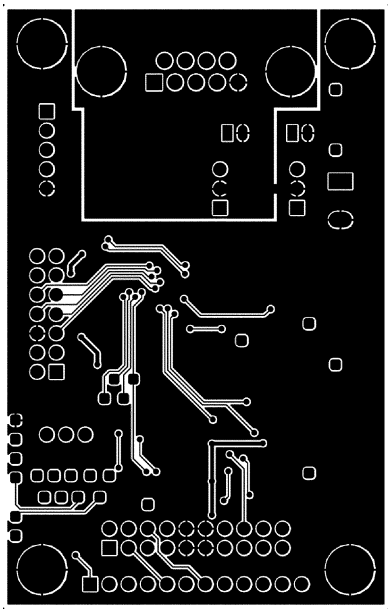
E6S3 Schematic Diagram



E6S3 PCB Layers







SILK COMPONENT BOTTOM

ICT EPLD SOURCE LISTING

Source listing for ICT PEEL22CV10AZ EPLD. The EPLD functions to generate control signals for the 3125 quad n-channel switch, to clamp unused Y lines during scanning, and to control the Y gate dwell time after the rise of an X line. This latter feature allows for moisture suppression since the charge transfer duration decreases (and thus is less responsive to the slow-moving charges from resistive water films) with decreasing dwell time.

```
TITLE 'E6S3'
PEEL22CV10A

"I/O CONFIGURATION DECLARATION
"IOC (PIN_NO        'PIN_NAME'          POLARITY          OUTPUT_TYPE       FEEDBACK_TYPE   )
"Inputs
XS1 PIN 8           "externally OR'd X inputs (8)
XS2 Pin 7
XS3 Pin 6
XS4 Pin 5
XS5 Pin 4
XS6 Pin 3
XS7 Pin 2
XS8 Pin 1
YS2 PIN 9
YS3 Pin 10
YS4 PIN 11
YS1 Pin 13

"Outputs
"Clamping outputs
 IOC (  22  'Y1'                Pos                 Com               Feed_Pin   )
 IOC (  20  'Y2'                Pos                 Com               Feed_Pin   )
 IOC (  19  'Y3'                Pos                 Com               Feed_Pin   )
 IOC (  17  'Y4'                Pos                 Com               Feed_Pin   )
"QS3125 drives
 IOC (  23  'F1'                Pos                 OutCom            Feed_Pin   )
 IOC (  21  'F2'                Pos                 OutCom            Feed_Pin   )
 IOC (  18  'F3'                Pos                 OutCom            Feed_Pin   )
 IOC (  16  'F4'                Pos                 OutCom            Feed_Pin   )

"Dly ckt in (& reset of delay cap drive out)
IOC (  14   'XSD'               Pos                 Com                          Feed_Pin   )
"Dly ckt drive out
IOC (  15   'XSS'               Pos                 OutCom            Feed_Pin   )

AR NODE 25  "Global Asynchronous Reset
SP NODE 26  "Global Synchronous Preset

EQUATIONS
AR = 0;
SP = 0;

y4.com = 0;                             " Clamp outputs
y3.com = 0;
y2.com = 0;
y1.com = 0;
y4.oe = !ys4;                   " drive outs to gnd when not used.
y3.oe = !ys3;
y2.oe = !ys2;
y1.oe = !ys1;
xss.com = xs1 # xs2 # xs3 # xs4 # xs5 # xs6 # xs7 # xs8; "sum all of x to drive out delay ckt
xsd.com = 0;
xsd.oe = !(xs1 # xs2 # xs3 # xs4 # xs5 # xs6 # xs7 # xs8); "clamp the delay cap when all 'xs' lines are 0
F4.com = !(ys4 & !xsd);         " enables to 3125 are active low!!
F3.com = !(ys3 & !xsd);
F2.com = !(ys2 & !xsd);
F1.com = !(ys1 & !xsd);
```

Quantum Research Group Ltd

651 Holiday Drive Bldg. 5 / 300
Pittsburgh, PA 15220 USA
Tel: 412-391-7367 Fax: 412-291-1015
admin@qprox.com
http://www.qprox.com

In the United Kingdom
Capstan House, High Street, Hamble Hants SO31 4HA
Tel: +44 (0)23 8045 3934 Fax: +44 (0)23 8045 3939