Hardware details of the Aurora board

1. Graphics:

1.1 Screen memory locations:

There are 3 screen areas (instead of 2 like on the standard QL).

In addition to the standard SCR0 (at $20000) and SCR1 (at $28000) there is the new high-res screen area at $4C0000, the size of this screen memory is 240 kbytes. Please note that SCR0 and SCR1 are physically resident in the high-res area as well. The hardware automatically re-codes accesses to SCR0 into the top left-hand corner of the high-res area, and SCR1 into the top right-hand corner of the high-res area.

However, accesses to SCR1 are disabled when the resolution and mode selected is anything over 512x256 mode 4 or 256x256 mode 8. Accesses to SCR0 are disabled if any mode other than 4 or 8 is selected.

Access disabling means that the data written to the SCR0 or SCR1 area respectively will not appear in the high-res screen area or on the screen. Read-back will still be possible because the GC/SGC shadows SCR0 and SCR1 with it's own RAM, which will provide readback data. This disabling was done to prevent old programs which directly access SCR0/SCR1 to interfere with the new graphics when new modes are in use. If mode 4 or 8 is in use with a higher resolution, SCR0 will automatically get written over the contents of the top left-hand corner of the screen, the only way to disable this is not to write into SCR0!!!

Our next product will move the location of the high-res memory into a different (higher) address as there will be a higher maximum RAM limit.

1.2 Screen memory organization:

There are 4 possible screen memory layouts, which depend on the mode in use. In all modes the lowest address in the memory area represents the top left-hand pixel, as usual, with higher addresses progressing to the right and downwards.

Mode 4 and 8 have the usual QL layout with one difference: The screen memory size is ALLWAYS 1024x960 mode 4 pixels. If a lower resolution is selected, the top left-hand corner of the 1024x960 is used. Therefore, the line length is CONSTANT, 256 bytes, and the maximum X and Y coordinates change only (and the number of lines and screen size). Therefore, the line length must NOT be calculated from the max X and Y coordinates, but the proper value from the window definition block must be used.

In modes 16 and 256 the screen area is configured differently, as a 512 x 480 byte field. Therefore, the line length is again constant, 512 bytes. In mode 16, each byte holds two pixels, top 4 bits are the 'left-hand' pixel, and bottom 4 bits are the 'right-hand' pixel. In mode 256, each byte represents one pixel.

Again, resolutions lower than the maximum memory size use only the upper left-hand corner of the memory field.

In any mode, colors are generated from the bits in memory by converting the various layouts to 3-bit red, green and blue components, i.e. R,G,B values from 0 to 7. In modes 4 and 8, each of the components can only assume a value of 0 or 7. In mode 8, the flash bit is stored, but ignored by the hardware (i.e. pixels do not flash).

The bit layout in mode 16 and 256 is as follows:

Mode 16:

Bit:	7	6	5	4	3	2	1	0
Pixel info:	Gl	Rl	Bl	Il	Gr	Rr	Br	Ir

G = green, R = red, B = blue, I = intensity.

The color component values generated are:

GRBI	G	R	B	
0000	0	0	0	Black
0001	1	1	1	Dark gray
0010	0	0	4	Dark blue
0011	0	0	7	Blue
0100	0	4	0	Dark red
0101	0	7	0	Red
0110	0	4	4	Dark magenta
0111	0	7	7	Magenta
1000	4	0	0	Dark green
1001	7	0	0	Green
1010	4	0	4	Dark cyan
1011	7	0	7	Cyan
1100	4	4	0	Dark yellow
1101	7	7	0	Yellow

1110	4	4	4	Gray
1111	7	7	7	White

Mode 256:

Bit:	7	6	5	4	3	2	1	0
Pixel info:	G2	R2	B2	G1	R1	B1	G0	RB0

G2,1,0 = green, G2=MSB, G0=LSB
R2,1 = red, R2=MSB
B2,1 = blue, B2=MSB
RB0 = Red/Blue compound bit 0.

G2..1 translate directly into a 3-bit value for the green component
R2..1 and B2..1 translate directly into the top 2 bits of the 3-bit red and blue components.
RB0 generates, in conjunction with R2..1 the LSB of red, R0, and in conjunction with B2..1 the LSB or blue, B0, as follows:

```
R0 = R[2] * RB[0]
   + R[1] * RB[0]
   + /R[2] * /R[1] * /B[2] * /B[1] * RB[0]

B0 = B[2] * RB[0]
   + B[1] * RB[0]
```

Therefore:

R2	R1	B2	B1	RB0	R	B
0	0	0	0	0	0	0
0	0	0	0	1	1	0
0	0	0	1	0	0	2
0	0	0	1	1	0	3
0	0	1	0	0	0	4
0	0	1	0	1	0	5
0	0	1	1	0	0	6
0	0	1	1	1	0	7
0	1	0	0	0	2	0
0	1	0	0	1	3	0
0	1	0	1	0	2	2
0	1	0	1	1	3	3
0	1	1	0	0	2	4
0	1	1	0	1	3	5
0	1	1	1	0	2	6
0	1	1	1	1	3	7
1	0	0	0	0	4	0

1	0	0	0	1	5	0
1	0	0	1	0	4	2
1	0	0	1	1	5	3
1	0	1	0	0	4	4
1	0	1	0	1	5	5
1	0	1	1	0	4	6
1	0	1	1	1	5	7
1	1	0	0	0	6	0
1	1	0	0	1	7	0
1	1	0	1	0	6	2
1	1	0	1	1	7	3
1	1	1	0	0	6	4
1	1	1	0	1	7	5
1	1	1	1	0	6	6
1	1	1	1	1	7	7

This color model has been chosen over one with three bits of green and red and two bits of blue because the discrete colors reproduced cover a more uniform area out of the standard color triangle.

1.3 Detecting the Aurora board

The Aurora can be detected at reset as follows:
At reset the Aurora mimics the 8301 ULA and will automatically set itself into 512x256 mode 4, SCR0 active. Because of the hardware remapping of SCR0 into the high-res screen area, anything written into the first 128 bytes of SCR0 can be read in the first 128 bytes of the high-res area, BUT NOT THE OTHER WAY AROUND!!! because the GC/SGC shadows only SCR0 and SCR1 and not the high-res area. Write only to SCR0 and read only from the high-res area to test for Aurora, not the other way around. In the previous specifications this step was replaced by a test for RAM at $4C0000, I do not recommend that because the address will change with our next product, and the current address will most likely hold ordinary RAM and not Aurora screen RAM.
After the presence of a high-res area is detected, the amount of the high-res area RAM should be tested. This will be either 240 kbytes or 128 kbytes. If the amount is 240 kbytes, Aurora is detected. If the amount is 128 kbytes, a LCD board is detected (see 1.5 for details).

1.4 Control registers

The graphics portion of the Aurora is controlled by 3 registers. One is the standard, write only, mode control register (MCR) as found on the 8301 ULA. There is also a write only extended mode control register (EMCR) for controlling the additional resolutions and modes, and a read only monitor preset register (MPR):

ADDRESS	D7	D6	D5	D4	D3	D2	D1	D0	
W$18043	AR	////	////	M1	M0	////	HR1	HR0	EMCR
W$18063	SCR1	////	////	////	M0	////	BLK	////	MCR
R$18043	////	////	////	MT1	////	MT0	////	IE	MPR

The bits in the MCR and EMCR are automatically reset to 0 when the system is reset.

The MCR has the usual assignment, for compatibility:
BLK = blank screen when 1 is written, enable screen when 0 is written
M0 = select mode 4 (0) or mode 8 (1)
SCR1 = select SCR0 (0) or SCR1 (1) to be displayed on screen

The behavior is slightly different in conjunction with the EMCR. In particular, the bits behave in a QL compatible manner when the EMCR bits are all 0. M0 in the EMCR is the same bit as M0 in the MCR. SCR1 in the MCR should be 0 if anything but the QL compatible resolutions and modes are used, otherwise the screen will be garbled.

The EMCR controls the new modes and resolutions:

HR1..0 select the horizontal resolution (future implementations will use bit 2 as HR2, so currently it must be written as 0):

HR1	HR0	Horizontal resolution
0	0	512 pixels
0	1	640 pixels
1	0	768 pixels
1	1	1024 pixels

M1..0 select the color mode (future implementations will use bit 5 as M2, so currently it must be written as 0):

M1	M0	Mode
0	0	Mode 4
0	1	Mode 8 (implemented only for compatibility)

1	0	Mode 16
1	1	Mode 256

The AR bit controls the aspect ratio, and hence the vertical resolution in conjunction with HR1..0 (Future implementations will use bit 6 as AM, aspect modifier, currently bit 6 must be written as 0):

AR	Aspect ratio
0	2:1 (vertical res. = horizontal res. * 1/2), QL style pixels
1	4:3 (vertical res. = horizontal res. * 3/4), square pixels

IMPORTANT:
The actual resolution displayed will depend on the monitor preset, which can be read from the MPR AND the mode selected (for reasons of limited high-res screen area size). The resolution selected by HR1..0 and AR in principle does NOT depend on the mode, except in mode 8, where the resolution selected refers to mode 4, but the number of pixels in one line is halved, as usual in mode 8 (this was done to maintain compatibility), and by limit of the high-res area size. Because the high-res area size is fixed, 240 kbytes, the resolutions in modes with more colors will be limited. The limiting logic is simple - if the resolution chosen is higher than a limit, the limit is used instead. Limits apply independently for x and y directions. In particular:

Mode 4: No limits (high-res area size is larger than maximum resolution, that being 1024 x 768).
Mode 16: Maximum vertical resolution is limited to 480 lines.
Mode 256: Horizontal resolution is limited to 512 pixels, and maximum vertical resolution is limited to 480 lines.

Additional limits may apply depending on the monitor preset values. Where more limits apply, the lowest value is used as the actual limit. The maximum x and y coordinates have to be adjusted according to these limits for every given resolution and monitor preset setting.

The MPR has three bits:
MT1..0 = return the general type of monitor selected
IE = returns interlace enable bit

Maximum vertical resolutions obtainable for any MT1..0 and IE combination are as follows:

MT1	MT0	IE	Monitor type	Max. vert. resolution

0	0	0	QL standard, NI	288 lines
0	0	1	QL standard, I	576 lines
0	1	0	VGA NI	576 lines
0	1	1	VGA I	768 lines
1	0	0	SVGA NI	576 lines
1	0	1	SVGA I	768 lines
1	1	0	Multisynch NI	768 lines
1*	1*	1*	Multisynch diag.*	960 lines*

* This is a special diagnostic mode which displays a 1024x960 interlaced picture on a multisynch monitor when 1024x768 is selected, hence displaying the contents of the whole high-res screen area. Whether the software will support this is optional - this combination of MT and IE bits is not used in normal operation.

1.5 The LCD board

The LCD board is a fringe development of the Aurora. It can operate on it's own (standard QL) or with the Aurora. When it operates with a standard QL, it offers increased resolution (up to 640x480, the size of the LCD panel), and will be detected by the fact that the high-res area is 128 kbytes in size. When it operates with the Aurora, it is transparent (cannot be detected) and follows settings for the Aurora whenever possible (resolutions higher than 640x480 have the top left-hand corner displayed on the LCD panel, and the panel is blanked if mode 16 or 256 is selected).

The MCR and EMCR registers appear as on the Aurora. Writing a 1 into M1 in EMCR will blank the screen as the LCD does not support a 16 or 256 color mode. An additional screen size limit always applies, which is that the maximum screen resolution is 640x480. Mode 8 is not accurately reproduced (similar to the QVME). There is no MPR register, and no additional facilities except graphics. Automatic SCR0 and SCR1 area relocation operates exactly as on the Aurora.

2. ROM disc, extended ROM sizes

2.1 Extended ROM concept

The Aurora features an extended ROM socket which can hold ROM-like chips (EPROM, Flash...) with up to 512 kbytes capacity. Because the memory map does not allow direct access to all of the 512 kbytes, the ROM is paged into 32 kbyte pages. To maintain QL compatibility, the hardware has been arranged so that the first 48 kbytes of the ROM (any size) initially appear at address $00000, so the system can start up from that address. In order for this to happen, a valid QL ROM image has to be present in the first 48 kbytes of the ROM, or a bootstrap program which will be correctly recognized by the GC/SGC firmware. The GC/SGC shadow the ROM area, by copying it to faster RAM. Once the GC/SGC firmware is initialized, the contents of addresses $00000 to $0BFFF will be read from the RAM copy and not from the actual ROM chip. When a SGC is used, the actual ROM chip can be read at address $400000 to $40BFFF. The paging mechanism is used to present any 32k page out of the total ROM capacity at this address. The page can then be read and copied to RAM for execution. At present I do not know how the actual ROM chip can be accessed instead of the shadowed copy on a GC.

2.2 ROM paging register

The ROM paging register is a write-only register at address $18041, which is automatically initialized to 0 at reset. The lower 4 bits contain the ROM page number. Each page is 32 kbytes in size, and there can be up to 16 pages for a total capacity of 512 kbytes. If a ROM smaller than 512 kbytes is used, the pages will repeat, modulo (ROM size). Hence, a 64 kbyte ROM will only have pages 0 and 1 (with 0 repeating in 2,4 and 6, and 1 repeating in 3,5 and 7), a 128 kbyte ROM will have pages 0,1,2,3 (repeated in 4,5,6,7, then in 8,9,A,B and in C,D,E,F) a 256 kbyte ROM will have pages 0 to 7 (repeated in 9 to F), and a 512 kbyte ROM will have all pages.

ADDRESS	D7	D6	D5	D4	D3	D2	D1	D0	
W$18041	////	////	////	////	RP3	RP2	RP1	RP0	RPR

A mechanism should be provided to detect the size of the ROM and limit the number of pages accessed. It is recommended that the contents of the ROM be copied to RAM as needed to form files or executable code. The code to copy the ROM should be atomic especially if the ROM is accessed as a file device, to prevent the contents of the RPR from being smashed by another thread.

3. Additional features

The Aurora has an extra IO area, 15.5 kbytes in size, located at $18100 to $1BEFF. The extended ROM slot header has a decoded select pin which goes low when this area is accessed. The exact use of this area has not been decided upon yet, and for now I recommend that no special measures be taken about it, except to prevent OS code from accessing it in hopes it will find a repeated copy of some control register in the 8302 ULA or the QIMI interface.