QSOUND / QPRINT

This upgrade adds to the good characteristics of the QL a 'real' sound generation facility and a parallel printer interface.

Fitting the board
Ensure that the QL is disconnected from the mains. If you do not you may damage the QL, the QSOUND/QPRINT board or both.
Remove the rectangular plastic cover on the left hand side of your QL which covers the expansion bus slot. This may require some effort.
Now push the QSOUND/QPRINT board firmly in the expansion slot. This may take also a little effort. You should be able to feel when the board is firmly in place.

To test your new upgrade connect the QL to the mains. After the usual memory test screen, the TV/Monitor selection screen will appear with an additional copyright message from the QSOUND/QPRINT board. If you have connected an amplifier you should hear a short sound like a bell: QSOUND/QPRINT tells you that it is OK! After pressing F1 (or F2) you can use the full power of your QSOUND/QPRINT expansion.
To connect an amplifier to the QSOUND/QPRINT card, there is a 3.5 mm cinch socket on the left hand side of the card. You will also find a centronics compatible printer interface there.

PARALLEL INTERFACE

Use
After powering up and initialisation you can use a new QL device named PAR.
You can connect a centronics compatible printer to the parallel interface and use it via the device name PAR.
A procedure to list the actual SuperBASIC program to a printer connected to the PAR port could look like this:

```
30000 DEFine PROCedure llist
30001   OPEN #3,PAR : LIST #3 : CLOSE #3
30002 END DEFine
```

Buffers
You can use a part of the QL's RAM as a buffer for efficient print spooling. Its size in blocks of 512 bytes must be specified in the OPEN command as in the following examples:

```
OPEN #3, PAR_1          512 byte buffer
OPEN #3, PAR_32         16 kbyte buffer
```

The maximum buffer size is 63 kbytes.

Options
Two options are accepted as part of the device name.

PARC_<bfsize> The 'C' flag is used if the <LF> character (chr$(10)) should be converted to a <CR> character (chr$(13)).

PARF_<bfsize> The 'F' flag is used if a <FF> character should be sent when the channel is closed.

PARCF_<bfsize> Combination of the two flags is allowed.

SER emulation
Using the new PAR_USE command you can get the benefits of your new parallel interface without changing existing programs. This command accepts a three character device name (with or without string quotes) as a parameter.

PAR_USE SER will emulate the SER ports of the QL. All output sent to the serial port will be send to the new PAR device.

Psion software
If you want to use the PAR interface with ABACUS, ARCHIVE EASEL and QUILL you may include the PAR_USE command in the corresponding boot program.
Alternativly, you can also change the printer driver with the

INSTALL_BAS program supplied with the Psion software package: Load and run the program and choose the Microdrive as required. Choose the PAR port option by pressing the space bar. Get the list of printer parameters by pressing F2. Change the PORT by pressing either the left or right cursor key and then type 'PAR' as the valid device name.
Don't use a buffer unless you have a memory expansion as the Psion programs use the entire RAM of an unexpanded QL.
Don't add the flags 'C' or 'F'. You can obtain these options specifying the END OF LINE code as CR and by including FF in the POSTAMBLE CODE, respectively.

ADDITIONAL SUPERBASIC COMMANDS

The QSOUND/QPRINT firmware consists of 30 new SuperBASIC commands which allow you to utilize the PAR interface and the full capability of the sound chip. A few useful procedures/functions are also included.
The commands are described below. Parameters enclosed in < > are optional. The default window is #1.

BELL — procedure
causes the sound chip to emit a short bell like sound

CONTROL — procedure
creates a job which opens a window in the upper right-hand corner of the screen shows the number of the job that is currently ready to receive input (i.e. whose cursor is flashing)

CUR_FLASH f — procedure
changes the flashing rate of the cursor
f = 0 normal
f = 1 medium
f = 2 high

CURDIS <#n> — procedure
disables the cursor in window #n

CURSEN <#n> — procedure
enables the cursor in window #n
The INKEY$ command doesn't show a cursor. The following function also returns the character corresponding to the key pressed but shows a flashing cursor for the number of frames specified in wait:

```
DEFine FUNction getkey$(chn, wait)
  CURSEN #chn
  r$=INKEY$(#chn, wait)
  CURDIS #chn
  RETurn r$
END DEFine
```

m = D_MODE — function
returns the current display mode
m = 4 high resolution (4 colours)
m = 8 low resolution (8 colours)

t = D_TYPE	function returns the current display type (which determines the default windows) t = 0 monitor t = 1 TV
DOWN <#n>	procedure moves the cursor in window #n one row down
EXPLODE	procedure causes the sound chip to emit an explosion like noise
HOLD	procedure stops all interrupt sound lists
HOLD n	procedure stops the interrupt sound list n (1 .. 3)
LEFT <#n>	procedure moves the cursor in window #n one column to the left
LIST_AY r0 .. r13	procedure sends the value of r0 .. r13 to the registers 0 .. 13 of the sound chip
nno = NET_NR	function returns the network station number of your machine which was assigned by the SuperBASIC command NET
NEW_FONT <#n>, add	procedure assigns the new character font at address add to window #n
OLD_FONT <#n>	procedure assigns the standard QL font to window #n
PAR_STOP	procedure clears the PAR device buffer and stops printing via the PAR port
PAR_USE ddd	procedure renames the PAR device to the new three character name ddd. Useful for SER emulation. PAR_USE SER emulates the serial port 1.
v = PEEK_AY (r)	function returns the contents of the register r of the sound chip

PLAY n, sound$ procedure
puts the string sound$ into the interrupt list of the sound channel n (1 .. 3).

sound$ may contain various characters (case is not distinctive) to denote

notes: C D E F G A H
(H corresponding to B, HB to Bflat)
sharps: #
flats: b
rests: p (one length unit)

change octave: o0 o1 .. o7
(default: o2)
change volume: v0 v1 .. v15
(default: v0)
v16 switches to wrap control
duration of note in 1/50 sec: 10 .. 1255
(default: 15)
change noise frequence: n0 n1 .. n31
(default: n0)
determine wrap curve: w0 w1 .. w15
(default: w0)
change length of wrap: x0 x1 .. x32767
(default is x0)

synchronisation stop: S
causes a sound channel to wait
activate a waiting channel: r1 r2 r3

Sound example (try it):

```
PLAY 1,'pr15o4sCDEFGAHo5CDEFGAHp'
PLAY 2,'pv15o2r1CDEFGAHo3CDEFGAHp'
```

POKE_AY r,v procedure
sets one of the registers (0 .. 13) of the sound processor to value v (0 .. 255)

vno$ = QDOS$ function
returns the version number of your QDOS operating system

RELEASE procedure
causes all interrupt sound lists to be played

RELEASE n procedure
causes the sound list n (1 .. 3) to be played

RIGHT <#n> procedure
moves the cursor in window #n one column to the right

SHOOT procedure
causes the sound chip to emit a noise like a shot

SOUND procedure
clears all sounds played by the sound chip. All tunes in the sound interrupt lists created whith the PLAY command are cleared.

SOUND n procedure
clears the sound channel n (1 ..3) and the corresponding interrupt list

SOUND n,f,v procedure
sets the sound output to sound channel n to the frequency f (400 ... 5000 Hz) and the volume v (0 .. 15)

UP <#n> procedure
moves the cursor in window #n one row up

MACHINE CODE PROGRAMMING WITH THE AY-3-8910 SOUND PROCESSOR

The QSOUND/QPRINT card could theoretical occupy any of the 16 expansion slots recognized by QDOS. To find the actual base address, read the system variable SV.AYBAS. The address to jump to is in the sustem variable SV.AYJMP.
The sound procrssor control routines are called with a code in the register D0 of the MC68000 (like the QDOS trap routines).
A call to the routine AY.RDREG (which reads a register of the AY-3-8910) should look like this:

```
MOVEQ     #AY.RDREG,D0        code for AY-routine
MOVEQ     #0,D2               read register $00
MOVE.L    SV.AYJMP,A0         get address to jump to
JSR       (A0)                do it
```

A QDOS error code will be returned in D0.

ROUTINE AND ADDRESS SUMMARY

ADDRESS DEFINITIONS

Name	Address	Description
SV.AYBAS	$28160 long	Base address of the firmware
SV.AYJMP	$28164 long	Start address for machine code routines

ROUTINES FOR SOUND CONTROL

D0	Name	Description
$00	AY.RESET	clears sound, stops noise
$01	AY.WRREG	writes one AY-3-8910 register
$02	AY.RDREG	reads one AY-3-8910 register
$03	AY.WRALL	writes all (0 .. 13) registers
$04	AY.RDALL	reads all (0 .. 15) registers
$05	AY.PLAY	plays a tune
$06	AY.TSTPL	status query
$07	AY.HOLD	causes a sound channel to wait
$08	AY.RELSE	releases a waiting channel
$09	AY.NOISE	emits a predefined noise
$0A	AY.SOUND	emits a user-defined sound

HARDWARE KEY

Name	Address	Description
AY.PORTA	$8000	PIA-Dataport A
AY.CTRLA	$8001	Control port A
AY.PORTB	$8002	PIA-Dataport B
AY.CTRLB	$8003	Control port B

D0 = $00 AY.RESET

Clears the sound.

Call parameters	Return parameters
D1	D1 undefined
D2	D2 undefined
D3	D3 preserved
A0	A0 preserved
A1	A1 undefined
	A5 undefined

ERROR RETURNS

None

NOTES

Also clears all sound interrupt lists.

D0 = $01 AY.WRREG

Writes a value to AY-3-8910 register.

Call parameters	Return parameters
D1.B value	D1 preserved
D2.B register	D2 preserved
D3	D3 preserved
A0	A0 preserved
A1	A1 preserved
	A5 undefined

ERROR RETURNS

ERR.OR Invalid register number (>13)

NOTES

Please note, that only registers 0 .. 13 are used. The other ports are used for the PAR interface. Bit 6 and 7 of register 7 remain unchanged. They are also used for port managing.

D0 = $02 AY.RDREG

Reads a AY-3-8910 register.

Call parameters	Return parameters
D1	D1 value read
D2.B register	D2 preserved
D3	D3 preserved
A0	A0 preserved
A1	A1 preserved
	A5 undefined

ERROR RETURNS

ERR.OR	Invalid register number (>15)

D0 = $03 AY.WRALL

Writes all registers (0 .. 13) of the AY-3-8910.

Call parameters	Return parameters
D1	D1 undefined
D2	D2 undefined
D3	D3 preserved
A0	A0 preserved
A1.L pointer to datablock	A1 undefined
	A5 undefined

ERROR RETURNS

None

NOTES

The datablock must contain 14 bytes with the values for the registers in ascending order ($00 = r0 .. $0C = r13).

D0 = $04 AY.RDALL

Reads all registers (0 .. 15) of the AY-3-8910.

Call parameters	Return parameters
D1	D1 undefined
D2	D2 undefined
D3	D3 preserved
A0	A0 preserved
A1.L pointer to buffer	A1 undefined
	A5 undefined

ERROR RETURNS

None

NOTES

The 16 byte buffer contains the values of the registers in ascending order ($00 = r0 .. $10 = r15).

D0 = $05 AY.PLAY

Puts a string to the interrupt sound list.

Call parameters	Return parameters
D1.B AY-channel	D1 undefined
D2	D2 preserved
D3	D3 preserved
A0.L pointer to the string	A0 undefined
A1	A1 preserved
	A5 undefined

ERROR RETURNS

ERR.OR	AY-channel was not 1, 2 or 3.
ERR.BP	String contains undefined sound items.

NOTES

The string must be preceded by a word containing the string length.

D0 = $06 AY.TSTPL

Returns the status of a AY-3-8910 channel buffer.

Call parameters	Return parameters
D1.B AY-channel	D1 status
D2	D2 preserved
D3	D3 preserved
A0	A0 preserved
A1	A1 preserved
	A5 undefined

ERROR RETURNS

ERR.OR	AY-channel was not 1, 2 or 3.
ERR.NO	Sound list doesn't exist.

NOTES

Status return in D0: 0 waiting
1 playing

D0 = $07 AY.HOLD

Suspends playing a sound list.

Call parameters	Return parameters
D1.B AY-channel	D1 undefined
D2	D2 undefined
D3	D3 undefined
A0	A0 undefined
A1	A1 undefined
	A5 undefined

ERROR RETURNS

ERR.OR	AY-channel was not 0,1,2 or 3.
ERR.NO	Sound list doesn't exist.

NOTES

AY-channel number 0 stops playing on all channels.

D0 = $08 AY.RELSE

Releases a suspended sound list.

Call parameters	Return parameters
D1.B AY-channel	D1 undefined
D2	D2 undefined
D3	D3 undefined
A0	A0 undefined
A1	A1 undefined
	A5 undefined

ERROR RETURNS

ERR.OR	AY-channel was not 0,1,2 or 3.
ERR.NO	Sound list doesn't exist.

NOTES

AY-channel number 0 causes all channels to continue.

D0 = $09 AY.NOISE

Causes the sound processor to emit predefined noises.

Call parameters	Return parameters
D1.B noise	D1 undefined
D2	D2 undefined
D3	D3 preserved
A0	A0 preserved
A1	A1 undefined
	A5 undefined

ERROR RETURNS

ERR.BP noise >2

NOTES

Values of noise:

0 explosion
1 gunshot
2 bell

D0 = $0A AY.SOUND

Emits a sound with a specific frequency and volume on the chosen channel.

Call parameters	Return parameters
D1.B AY-channel	D1 undefined
D2.W frequency	D2 undefined
D3.B volume	D3 undefined
A0	A0 preserved
A1	A1 preserved
	A5 undefined

ERROR RETURNS

ERR.BP	AY-channel was not 1, 2 or 3.
ERR.OR	Frequency was out of range.

NOTES

The valid frequency is 400 .. 5000 Hz.
The valid volume code is 0 .. 15.

REGISTERS OF THE AY-3-8910 SOUND PROCESSOR

The sound processor AY-3-8910 works with a 0.75 MHz time frequency on the QL. It uses 16 read/write registers. The use of the registers is described below.

reg		bit	7	6	5	4	3	2	1	0
$00	chn A tone	LSB	7	6	5	4	3	2	1	0
$01	chn A	MSB	x	x	x	x	B	A	9	8
$02	chn B tone	LSB	7	6	5	4	3	2	1	0
$03	chn B	MSB	x	x	x	x	B	A	9	8
$04	chn C tone	LSB	7	6	5	4	3	2	1	0
$05	chn C	MSB	x	x	x	x	B	A	9	8
$06	noise period		x	x	x	x	3	2	1	0
$07	release		ioB	ioA	nC	nB	nA	sC	sB	sA
$08	chn A amplitude		x	x	x	w	3	2	1	0
$09	chn B amplitude		x	x	x	w	3	2	1	0
$0A	chn C amplitude		x	x	x	w	3	2	1	0
$0B	wrap period	LSB	7	6	5	4	3	2	1	0
$0C	wrap period	MSB	F	E	D	C	B	A	9	8
$0D	wrap curve		x	x	x	x	w3	w2	w1	w0
$0E	I/O port A		7	6	5	4	3	2	1	0
$0F	I/O port B		7	6	5	4	3	2	1	0

NOTES

x: bit not used
ioA: If bit is set then port A is input channel else it is output channel.
ioB: If bit is set then port B is input channel else it is output channel.
nA .. nC: If bit is reset channel is emitting noise.
sA .. sC: If bit is reset channel is emitting sound.
w0 .. w3: Wrap curve (cf. page 22)
w: Bit activates wrap control.

Registers $00 .. $05 define the pitch of the channel. Two registers define a note. The main time is divided by 16. By counting down the 12-bit-counter the output frequency is generated.

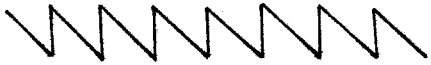
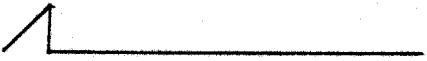
Register $06 defines the noise frequency. This works like the pitch control but with 5 bits only.

Register $07 releases the sources. It selects silence, sound, noise or sound and noise for each channel.

Registers $08 .. $0A define the volume. The four LSBits denote the volume in logarithmic steps. The w-bit activates the wrap control.

Registers $0B and $0C define a 16 bit wrap period.

Register $0D selects the wrap curve as shown below.

w3	w2	w1	w0	wrap curve
0	0	x	x	
0	1	x	x	
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

Registers $0E and $0F describe the state of port A and port B.