# PCW 8256/8512

## JOYSTICK/SOUND CONTROLLER

### INSTRUCTIONS

CONTENTS PAGE


Introduction ..... 2
Installation ..... 3
Register summary ..... 4
Register selection ..... 4
Joystick ..... 5
Output Control ..... 6
External Circuits ..... 7
Tone Control ..... 8
Noise Control ..... 9
Mixer - I/O Control ..... 9
I/O Truth Table ..... 9
Amplitude Control ..... 10
Envelope Generator Control ..... 10
Note Generation ..... 13
Music Generation ..... 13
Chords ..... 13
Frequency Tables ..... 14


dk'tronics®

## INTRODUCTION

The Amstrad PCW 8256/8512 joystick/sound controller uses the popular AY-3-8912 Programmable Sound Generator (PSG) to produce complex sounds under software control. The joystick is connected to an I/O port within the PSG chip.

In order to perform sound effects while the host processor continues executing other tasks, the PSG will maintain sound output after the initial commands have been issued. Also the PSG has 3 independent programmable channels which allows the creation of realistic sounds.

All the functions of the PSG are controlled through its 16 registers, access to which is gained by first writing to the register select port the desired register number, then data is either written to or read from the data port.

The joystick can be interrogated under software control from within CP/M or BASIC via a simple piece of code. Alternatively, by running the 'DEFJOY.COM' program supplied, the joystick can be made to emulate any of the keyboard keys. Eg the joystick could be set up to emulate the cursor keys with the fire button acting as the 'RETURN' or 'ENTER' keys, etc.

The interface will accept any standard joystick that has a 9-pin 'D' TYPE plug and will allow the use of the auto-fire facility as supported by the QUICKSHOT II joystick.

The joystick I/O port can also be programmed to operate in output mode which means that the controller can 'drive' external circuits, ie relay circuits for controlling mains powered equipment.

The interface has a through-connector to allow other peripherals to be connected at the same time, thus eliminating the need for continually removing peripherals.

The joystick controller is supplied with a disc which contains the DEFJOY.COM program which should be run from within CP/M, and a demonstration program which can be run within basic. Both programs are menu driven and are therefore self expanatory.

CP/M is a registered trademark of Digital Research Inc.





## AMSTRAD 8256/8512 JOYSTICK-SOUND CONTROLLER

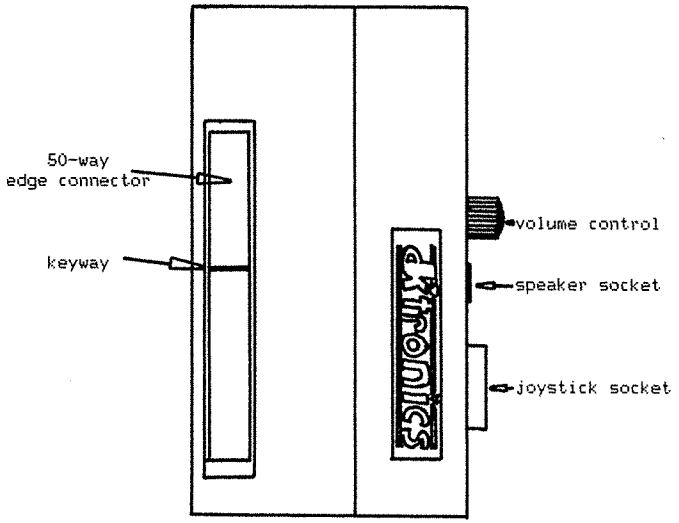
### WARNING

**Ensure that the power to your Amstrad computer is switched off before you fit the interface to the expansion socket. Failure to comply with these instructions may cause permanent damage to the interface or the computer.**

### Installation

Power down your computer. Plug the controller into the expansion socket on the back of the computer so that the speaker jack-socket, volume control and joystick socket are on the left hand side as viewed from the rear of the computer. Plug in the speaker cable and the joystick to the relevant sockets. Above the speaker socket is a volume control which is factory set but can be adjusted to the users requirements.

The computer should now power up as normal. If it fails to do so, check that the joystick / sound controller is correctly fitted. Note that all DK'tronics products have a keyway location on the connector to ensure that there can be no alignment problems, other interfaces may not have this keyway, ( the Amstrad Serial Interface is an example).



## REGISTER SUMMARY

The PSG chip has 16 internal registers, a summary of their functions are as follows :-

| REGISTER | FUNCTION | REMARKS | BITS |
|---|---|---|---|
| R0 | Channel A Tone Period | 8-Bit fine tune | 0-7 |
| R1 | | 4-Bit coarse tune | 0-3 |
| R2 | Channel B Tone Period | 8-Bit fine tune | 0-7 |
| R3 | | 4-Bit coarse tune | 0-3 |
| R4 | Channel C Tone Period | 8-Bit fine tune | 0-7 |
| R5 | | 4-Bit coarse tune | 0-3 |
| R6 | Noise period | 5-Bit control | 0-4 |
| R7 | Control Register | 2-bit A tone/noise | 0/3 |
| | | 2-bit B tone/noise | 1/4 |
| | | 2-bit C tone/noise | 2/5 |
| | | 2-bit I/O | 6-7 |
| R8 | Channel A Amplitude | 5-Bit | 0-4 |
| R9 | Channel B Amplitude | 5-Bit | 0-4 |
| R10 | Channel C Amplitude | 5-Bit | 0-4 |
| R11 | Envelope Period | 8-Bit fine tune | 0-7 |
| R12 | | 8-bit coarse tune | 0-7 |
| R13 | Envelope Shape/Cycle | 1-Bit HOLD | 0 |
| | | 1-Bit ALTERNATE | 1 |
| | | 1-Bit ATTACK | 2 |
| | | 1-Bit CONTINUE | 3 |
| R14 | I/O Port A | 8-Bit Bi-directional | 0-7 |
| R15 | I/O Port B | Not used in AY-8912 | |

## REGISTER SELECTION

Before you can write data to a given register you must first select that register. This is achieved by writing the register number to the register select port, (170). Data for the selected register can then be written to the data port, (171).

Example : set the amplitude control register for channel A, register 8, with a value of 15 to produce maximum volume.

BASIC PROGRAM

```
10 OUT 170,8          REM select register 8
20 OUT 171,15         REM maximum amplitude
```





M/CODE PROGRAM

```
LD A,8              ; register 8
LD C,170            ; register select port
OUT (C),A           ; select register 8
INC C               ; point to data port
LD A,15             ; maximum amplitude
OUT (C),A           ; send data
```

## JOYSTICK

The joystick is connected to the I/O port of the PSG chip, it is therefore necessary to initialise the port to input mode before the joystick can be used. This is achieved by clearing the relevant bit in the I/O register 7.

### I/O INITIALISATION

BASIC PROGRAM

```
10 OUT 170,7          REM select register 7
20 OUT 171,63         REM bits 6-7 zero, other bits are for
                      sound effects.
```

M/CODE PROGRAM

```
LD A,7              ; register 7
LD C,170            ; register select port
OUT (C),A           ; select register 7
INC C               ; point to data port
LD A,63             ; bits 6-7 set to zero
OUT (C),A           ; send data
```

### JOYSTICK STATUS

The joystick status can now be read by the user using the following code:-

BASIC

```
10 OUT 170,14         REM select register 14
20 LET J=INP(169)     REM get the joystick data, 169 is the
                      data read port
```

M/CODE

```
LD A,14             ; register 14
LD C,170            ; register select port
OUT (C),A           ; select register 14
DEC C               ; point to data read port (169)
IN A,(C)            ; get joystick status
```

Once the joystick status has been obtained the relevant movement can be obtained by checking to see which bits are set to a logic '0'. The bits are as follows:-

BIT 2 ≡ LEFT
BIT 3 ≡ RIGHT
BIT 4 ≡ DOWN
BIT 5 ≡ UP
BIT 6 ≡ FIRE

Bits 0,1 and 7 are not used and should therefore be ignored by the software.

Thus a BASIC program might be :-

```
                Initialisation

20  OUT 170,7  :REM select PSG control register
30  OUT 171,63 :REM I/O port set to input mode, Bits 7,6 = 0

            Read joystick routine

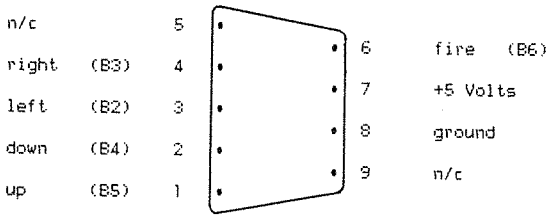
60  OUT 170,14 :REM Select I/O port
70  J=INP(169) :REM Get data

    Joystick sorting routine, Bits are active low

90  IF (J AND 4)=0 THEN PRINT "LEFT"
100 IF (J AND 8)=0 THEN PRINT "RIGHT"
110 IF (J AND 16)=0 THEN PRINT "DOWN"
120 IF (J AND 32)=0 THEN PRINT "UP"
130 IF (J AND 64)=0 THEN PRINT "FIRE"
140 GOTO 60   :REM Loop back for another read.
```

## OUTPUT CONTROL

In certain applications the user may wish to control external equipment,the I/O port can be programmed to operate in the output mode to accommodate this, 5 control lines, ground and +5 Volt references are available via the 9-way 'D' type connector as detailed below.

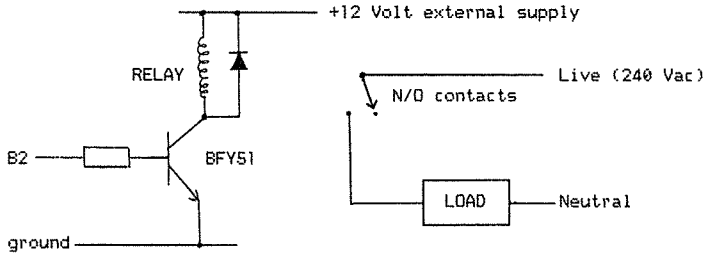







## EXTERNAL CIRCUITS

### RELAY CIRCUITS

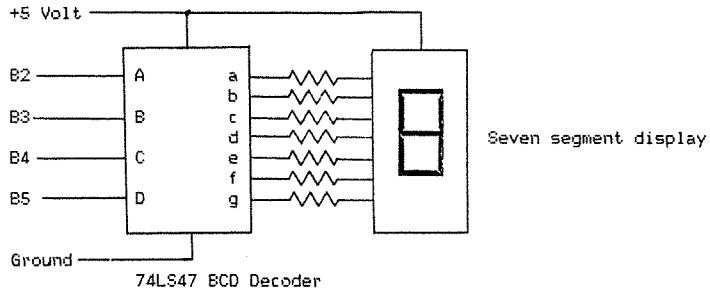



**Control Program**

```
10 OUT 170,7          select control register
20 OUT 171,255        output mode

30 OUT 170,14         select I/O register
40 OUT 171,0          turn relay off
50 OUT 171,4          turn relay on
```

### SEVEN SEGMENT DISPLAY




**Control Program**

```
10 OUT 170,14         select I/O register
20 OUT 171,3*4        display number 3
30 OUT 171,7*4        display number 7
```

note that the number required on the display is multiplied by 4 because the bits start from B2 which has a value of 4.
These circuits are only intended as a guide to the use that the DK'tronics joystick/sound controller may be put to. Care must be taken when connecting any external circuits to your computer.

DK'tronics cannot be held responsible for users own circuits.

## TONE CONTROL (registers 0-5)

The frequency of each square wave generated by the three tone generators (A,B,C) is obtained by the PSG by first dividing the clock frequency by 16 then by further counting down the result by the value held in the relevant tone period registers. The 12-bit Tone Period value is obtained by combining the contents of the relative coarse and fine tune registers. The tone period has a range from 0 to 4095

ie, TP = 256*CT + FT  where  TP = Tone Period
CT = Coarse Tune value
FT = Fine tune value

The equation describing the relationship between the tone frequency, the input clock frequency and the Tone Period is as follows:-

TFR = CF/(16*TP)  where  TFR = the desired Tone Frequency
CF = the Clock Frequency ( 1 MHz )
TP = Tone Period as above

EXAMPLE 1  TFR = 500 Hz
CF = 1 MHz

TFR = CF/(16*TP)
∴ TP = CF/(16*TFR)

TP = 1000000/(16*500)
TP = 125

CT = 0 ≡ 0000 (B3-0)
FT = 125 ≡ 01111101 (B7-0)

EXAMPLE 2  TFR = 100 Hz
CF = 1 MHz

TFR = CF/(16*TP)
∴ TP = CF/(16*TFR)

TP = 1000000/(16*100)

TP = 625

CT = 2 ≡ 0010 (B3-0)
FT = 113 ≡ 01110001 (B7-0)

Note that CT and FT are integer values where FT has a range of 0 to 255 and CT has a range of 0-15. Also that CT is in multiples of 256.





## NOISE CONTROL (register 6)

The frequency of the noise source is obtained by the PSG by first dividing the clock frequency by 16 then by further counting down the result by the value held in the noise period registers. The Noise Period is only 5 bits and so can range from 0 to 31.

The noise frequency equation is :-

NF = CF/(16*NP)  where  NF = the Noise Frequency
CF = the Clock Frequency ( 1 MHz )
NP = the Noise Period

From the above equation it can be seen that the noise frequency can range from 2 kHz to 62.5 kHz.

## MIXER / I/O CONTROL (register 7)

Register 7 controls the three Noise/Tone mixers as well as defining the direction of the I/O port which in this case is connected to the joystick. It therefore must be set to input mode, see page 7 for the I/O port truth table.

**Tone Enable Truth Table**

| B0 | B1 | B2 | Tone enabled on Channel |
|---|---|---|---|
| 0 | 0 | 0 | A B C |
| 1 | 0 | 0 | - B C |
| 0 | 1 | 0 | A - C |
| 1 | 1 | 0 | - - C |
| 0 | 0 | 1 | A B - |
| 1 | 0 | 1 | - B - |
| 0 | 1 | 1 | A - - |
| 1 | 1 | 1 | - - - |

**Noise Enable Truth Table**

| B3 | B4 | B5 | Noise enabled on Channel |
|---|---|---|---|
| 0 | 0 | 0 | A B C |
| 1 | 0 | 0 | - B C |
| 0 | 1 | 0 | A - C |
| 1 | 1 | 0 | - - C |
| 0 | 0 | 1 | A - C |
| 1 | 0 | 1 | - B - |
| 0 | 1 | 1 | A - - |
| 1 | 1 | 1 | - - - |

**I/O Port Truth Table**

| B6 | B7 | I/O Port Status A | B |
|---|---|---|---|
| 0 | 0 | Input | Input |
| 1 | 0 | Output | Input |
| 0 | 1 | Input | Output |
| 1 | 1 | Output | Output |

Note that Port B does not exist on the AY-3-8912.

## AMPLITUDE CONTROL (registers 8-10)

The signal amplitude for the three channels, A,B and C is controlled by the value held in the registers 8,9 and 10 respectively.

The amplitude mode is set by bit 4 in each register. If this bit is set to a logic '0' then the lower bits 0-3 define 1 of 16 fixed levels of amplitude, if this bit is set to a logic '1' then the amplitude is variable at 16 levels as determined by the output of the envelope generator.

| B4 | B3 | B2 | B1 | B0 | AMPLITUDE | OUTPUT |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | Channel turned off |
| 0 | 0 | 0 | 0 | 1 | 1 | Minimum amplitude (volume) |
| . | . | . | . | . | . | |
| 0 | 1 | 0 | 0 | 0 | 8 | Mid way setting ** |
| . | . | . | . | . | . | |
| 0 | 1 | 1 | 1 | 1 | 15 | Full amplitude |
| 1 | X | X | X | X | 0-15 | Set by Envelope generator |

NOTE (X=Don't care), a value of 0 turns channel off.

** The D/A conversion is performed in logarithmic stages with a normalized voltage range of 0 to 1 volt, therefore the mid way setting will not produce half amplitude (volume).

## ENVELOPE GENERATOR CONTROL (registers 11-13)

To accomplish fairly complex envelope patterns, two independent methods of control are provided: first, it is possible to vary the frequency of the envelope using registers 11 and 12, and second, the shape and cycle pattern of the envelope can be varied using register 13.

### Envelope period control.

The frequency of the envelope is obtained by the PSG by first dividing the clock frequency by 16 then by further counting down the result by the value held in the envelope registers. The 16-bit envelope period value is obtained by combining the contents of the Envelope coarse and fine tune registers.

The envelope frequency equations are:-

EP = 256*CT + FT where EP = Envelope Period
CT = Coarse Tune value
FT = Fine Tune value





EFR = CF/(256*EP) where EFR = desired Envelope FRequency
CF = the Clock Frequency ( 1 MHz )
EP = Envelope Period as above

EXAMPLE EFR = 0.5 Hz
CF = 1 MHz

EFR = CF/(256*EP)
EP = CF/(256*EFR)

EP = 1000000/(256*0,5)
EP = 7812,5

CT = 30 ≡ 00011110 (B7-0)
FT = 132 ≡ 10000100 (B7-0)

Note again that we are only interested in whole numbers so FT could be either 132 or 133 depending if you round up TP.

### Envelope shape/cycle control

The Envelope Generator further counts down the envelope frequency by 16, producing a 16-state per cycle envelope pattern as defined by its 4-bit counter output. The particular shape and cycle pattern of any desired envelope is accomplished by controlling the count pattern (up/down) of the 4-bit counter and by defining a single-cycle or repeat-pattern.

The envelope shape/cycle control is contained in the lower 4-bits (B0-3) of register 13. Each of these 4 bits controls a function in the envelope generator as illustrated below :-

**Bit-0 HOLD** When set to a logic '1' limits the envelope to one cycle, holding the last count of the envelope counter to 0000 or 1111, depending on whether the envelope counter was in a count-down or count-up mode respectively.

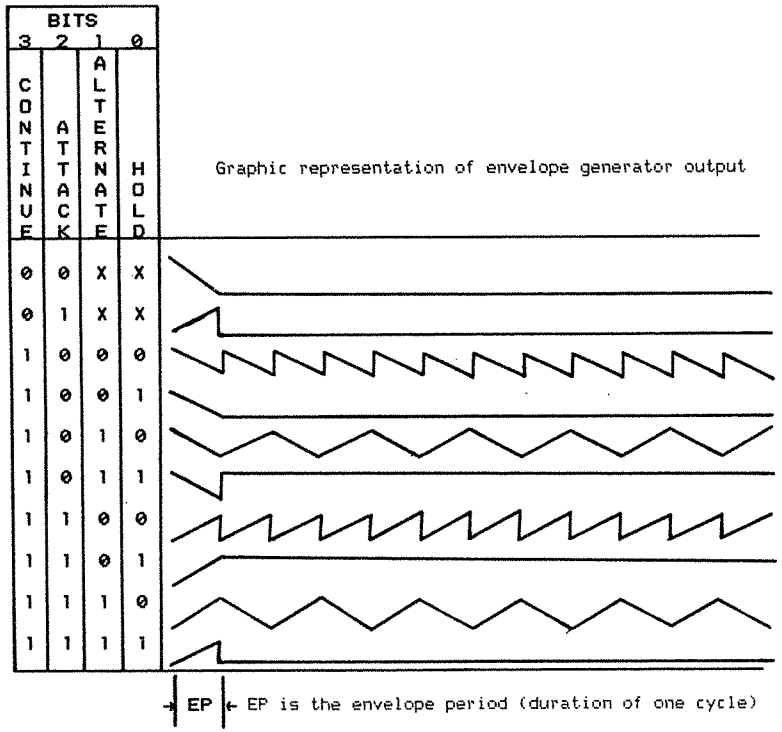
**Bit-1 ALTERNATE** When set to a logic '1' the envelope counter reverses count direction (up-down) after each cycle.

NOTE: when both the hold bit and the alternate bit are '1' s, the envelope counter is reset to its initial count before holding.

**Bit-2 ATTACK** When set to a logic '1' the envelope counter will count up (attack) from 0000 to 1111; when set to a logic '0' the envelope counter will count down (decay) from 1111 to 0000.

**Bit-3 CONTINUE** When set to a logic '1' the cycle pattern will be as defined by the hold bit; when set to a logic '0', the envelope generator will reset to 0000 after one cycle and hold at that count.

The following diagram illustrates the effects of the Envelope Shape/Cycle control register 13.








## NOTE GENERATION

Since notes are produced by sustaining a particular frequency for a given time at a varying amplitude, the PSG performs this function with a series of simple register loads.
The method used in many cases is to obtain register load values for first octave notes and to 'shift' them to the correct octave at play time.

## MUSIC GENERATION

The generation of music involves stringing together a series of frequencies (notes), which have mathematical relationships. This makes this application ideal for computer generation. For example, the shifting up or down in octaves is a multiplication or division by a power of 2, which is a simple shift operation for microprocessors.

Although it is easy to construct recognizable tunes using only one note at a time, the simultaneous sounding of more than one note to produce chords vastly increases the quality of the sound. This feature is easily achieved by the PSG since three independently programmable channels are provided.

## CHORDS

There are certain combinations of notes which when played simultaneously produce 'chords'. These chords can be easily formed from a base note by performing octave and key changes on two notes, which are played with the main note. These relationships are illustrated in the following table, which lists the various note constants which will produce musical chords. A chord with a particular quality may be formed by playing its root, a 3rd Minor or Major, and other notes from the chord chart. Eg, a C Major chord is formed from C(÷2),E(÷2) and G(÷2).

| CHORD SELECTION | ROOT | 3rd MINOR | 3rd MAJOR | 4th | 5th | 6th | 7th |
|---|---|---|---|---|---|---|---|
| C | C (÷2) | D#(÷2) | E (÷2) | F (÷2) | G (÷2) | A (÷2) | A#(÷2) |
| C# | C#(÷2) | E (÷2) | F (÷2) | F#(÷2) | G#(÷2) | A#(÷2) | B (÷2) |
| D | D (÷2) | F (÷2) | F#(÷2) | G (÷2) | A (÷2) | B (÷2) | C (÷1) |
| D# | D#(÷2) | F#(÷2) | G (÷2) | G#(÷2) | A#(÷2) | C (÷1) | C#(÷1) |
| E | E (÷2) | G (÷2) | G#(÷2) | A (÷2) | B (÷2) | C#(÷1) | D (÷1) |
| F | F (÷2) | G#(÷2) | A (÷2) | A#(÷2) | C (÷1) | D (÷1) | D#(÷1) |
| F# | F#(÷2) | A (÷4) | A#(÷4) | B (÷4) | C#(÷2) | D#(÷2) | E (÷2) |
| G | G (÷2) | A#(÷4) | B (÷4) | C (÷2) | D (÷2) | E (÷2) | F (÷2) |
| G# | G#(÷2) | B (÷4) | C (÷2) | C#(÷2) | D#(÷2) | F (÷2) | F#(÷2) |
| A | A (÷2) | C (÷2) | C#(÷2) | D (÷2) | E (÷2) | F#(÷2) | G (÷2) |
| A# | A#(÷2) | C#(÷2) | D (÷2) | D#(÷2) | F (÷2) | G (÷2) | G#(÷2) |
| B | B (÷2) | D (÷2) | D#(÷2) | E (÷2) | F#(÷2) | G#(÷2) | A (÷2) |

The following chart lists a full 8 octaves from low C1 to a high B8. With a clock frequency of 1 MHz and applying the previous formulas for calculating Tone Periods, register load values are as shown in the following tables.

Note that the nature of the PSG divider system produces a high degree of accuracy for low frequencies, less for high frequencies.

The frequency values given are ideal frequencies based on the Equal Tempered Chromatic Scale.

| NOTE | FREQUENCY | REGISTER VALUE | |
|---|---|---|---|
| C | 32.703 | 1911 | |
| C# | 34.648 | 1804 | |
| D | 36.708 | 1702 | |
| D# | 38.891 | 1607 | |
| E | 41.203 | 1517 | |
| F | 43.654 | 1431 | Octave -3 |
| F# | 46.249 | 1351 | |
| G | 48.999 | 1275 | |
| G# | 51.913 | 1204 | |
| A | 55.000 | 1136 | |
| A# | 58.270 | 1072 | |
| B | 61.735 | 1012 | |

| NOTE | FREQUENCY | REGISTER VALUE | |
|---|---|---|---|
| C | 65.406 | 955 | |
| C# | 69.296 | 902 | |
| D | 73.416 | 851 | |
| D# | 77.782 | 803 | |
| E | 82.407 | 758 | |
| F | 87.307 | 716 | Octave -2 |
| F# | 92.499 | 675 | |
| G | 97.999 | 638 | |
| G# | 103.826 | 602 | |
| A | 110.000 | 568 | |
| A# | 116.541 | 536 | |
| B | 123.471 | 506 | |





| NOTE | FREQUENCY | REGISTER VALUE | |
|---|---|---|---|
| C | 130.813 | 478 | |
| C# | 138.591 | 451 | |
| D | 146.832 | 425 | |
| D# | 155.564 | 402 | |
| E | 164.814 | 379 | |
| F | 174.614 | 358 | Octave -1 |
| F# | 184.997 | 338 | |
| G | 195.998 | 319 | |
| G# | 207.652 | 301 | |
| A | 222.000 | 284 | |
| A# | 233.082 | 268 | |
| B | 246.942 | 253 | |

| NOTE | FREQUENCY | REGISTER VALUE | |
|---|---|---|---|
| C | 261.626 | 239 | Middle C |
| C# | 277.451 | 225 | |
| D | 293.665 | 213 | |
| D# | 311.127 | 201 | |
| E | 329.628 | 189 | |
| F | 349.228 | 179 | Octave 0 |
| F# | 369.994 | 169 | |
| G | 391.995 | 159 | |
| G# | 415.305 | 150 | |
| A | 440.000 | 142 | International A |
| A# | 466.164 | 134 | |
| B | 493.883 | 126 | |

| NOTE | FREQUENCY | REGISTER VALUE | |
|---|---|---|---|
| C | 523.251 | 119 | |
| C# | 554.365 | 112 | |
| D | 587.330 | 106 | |
| D# | 622.254 | 105 | |
| E | 659.255 | 95 | |
| F | 698.457 | 89 | Octave +1 |
| F# | 739.989 | 84 | |
| G | 783.991 | 79 | |
| G# | 830.609 | 75 | |
| A | 880.000 | 71 | |
| A# | 932.328 | 67 | |
| B | 987.767 | 63 | |

| NOTE | FREQUENCY | REGISTER VALUE | |
|---|---|---|---|
| C | 1046.502 | 59 | |
| C# | 1108.731 | 56 | |
| D | 1174.659 | 53 | |
| D# | 1244.508 | 50 | |
| E | 1318.510 | 47 | |
| F | 1396.913 | 44 | Octave +2 |
| F# | 1479.978 | 42 | |
| G | 1567.982 | 40 | |
| G# | 1661.219 | 37 | |
| A | 1760.000 | 35 | |
| A# | 1864.655 | 33 | |
| B | 1975.533 | 31 | |

| NOTE | FREQUENCY | REGISTER VALUE | |
|---|---|---|---|
| C | 2093.004 | 30 | |
| C# | 2217.461 | 28 | |
| D | 2349.318 | 26 | |
| D# | 2489.016 | 25 | |
| E | 2637.021 | 23 | |
| F | 2793.826 | 22 | Octave +3 |
| F# | 2959.955 | 21 | |
| G | 3135.963 | 20 | |
| G# | 3322.438 | 19 | |
| A | 3520.000 | 18 | |
| A# | 3729.310 | 17 | |
| B | 3951.066 | 16 | |

| NOTE | FREQUENCY | REGISTER VALUE | |
|---|---|---|---|
| C | 4186.009 | 15 | |
| C# | 4434.922 | 14 | |
| D | 4698.636 | 13 | |
| D# | 4978.032 | 12 | |
| E | 5274.041 | 12 | |
| F | 5587.652 | 11 | Octave +4 |
| F# | 5919.911 | 10 | |
| G | 6271.927 | 10 | |
| G# | 6644.875 | 9 | |
| A | 7040.000 | 9 | |
| A# | 7458.621 | 8 | |
| B | 7902.133 | 8 | |