Sound and Controller Extensions for Turbo Pascal 3

Coleco Adam Computer CP/M 2.2

SOUND EXTENSIONS

The SOUND.PAS module needs to be included in your source code in order to use the sound extensions:

{$I SOUND.PAS}

This module allows for up to 3 simultaneous tone channels and 1 noise channel. <u>When you generate a tone or noise event, the sound will play continuously until you manually turn it off.</u>

Important: In order to maximize performance, no boundary checks are made for the argument values or type and it is the programmer's responsibility to make sure any arguments comply with the requirements as detailed below.

- **TONE(channel,frequency,volume) –** Play a tone
  - Channel = 1 – 3
  - Frequency = 0 – 1023. The actual frequency is determined by the following formula: clock/32*frequency
  - Volume = 0 (muted) to 15 (loudest)
- **NOISE(type,volume) –** Play a noise
  - Type:
    - 0 – 2 = white noise (hiss)
    - 3 = white noise tied to tone channel 3
    - 4 – 6 = periodic noise (motor)
    - 7 = periodic noise tied to tone channel 3
  - Volume = 0 (muted) to 15 (loudest)
- **SOUNDOFF(channel) –** Turn off sound
  - Channel
    - 0 = all channels (tone and noise)
    - 1 – 3 = corresponding tone channel
    - 4 = noise channel

CONTROLLER EXTENSION

The CONTROL.PAS module needs to be included in your source code in order to use the controller extension:

{$I CONTROL.PAS)

This module allows for full polling of the standard Coleco Adam controllers attached to the computer. It has not been tested with other third-party controllers.

Important: In order to maximize performance, no boundary checks are made for the argument values or type and it is the programmer's responsibility to make sure any arguments comply with the requirements as detailed below.

- **CONTROLLER(port,left button,right button,keypad,joystick) –** Polls the controller attached to the corresponding port and returns the appropriate values into the corresponding variables to the calling program. The variables are of type BYTE.
  - Port **=** 1 or 2 for joysticks 1 or 2
  - Left button: returns 64 when the left fire button is pressed
  - Right button: returns 64 when the right fire button is pressed
  - Keypad: returns the following values for the corresponding key pressed
    - 0 = no key pressed
    - 1 = key 6
    - 2 = key 1
    - 3 = key 3
    - 4 = key 9
    - 5 = key 0
    - 6 = key *
    - 8 = key 2
    - 9 = key #
    - 10 = key 7
    - 12 = key 5
    - 13 = key 4
    - 14 = key 8
  - Joystick: returns the position of the joystick
    - 1 = N
    - 3 = NE
    - 2 = E
    - 6 = SE
    - 4 = S
    - 12 = SW
    - 8 = W
    - 9 = NW

Credits:
Walid Maalouli
September 2019