// ---------------------------------------------------------------------------

// Created by Tim Eckel - teckel@leethost.com

// Copyright 2016 License: GNU GPL v3 http://www.gnu.org/licenses/gpl-3.0.html

//

// See "TimerFreeTone.h" for purpose, syntax, version history, links, and more.

// ---------------------------------------------------------------------------

#include "ABLocks\_TimerFreeTone.h"

uint8\_t \_tft\_volume[] = { 255, 200, 150, 125, 100, 87, 50, 33, 22, 2 }; // Duty for linear volume control.

void TimerFreeTone(uint8\_t pin, unsigned long frequency, unsigned int duration, uint8\_t volume) {

if (frequency == 0 || volume == 0) { // If frequency or volume are zero, just wait duration and exit.

delay(duration);

return;

}

frequency = 1000000 / frequency; // Calculate the square wave length (in microseconds).

uint32\_t duty = frequency / \_tft\_volume[min(volume, 10) - 1]; // Calculate the duty cycle (volume).

#ifdef \_\_AVR\_\_

uint8\_t pinBit = digitalPinToBitMask(pin); // Get the bitmask for the pin.

volatile uint8\_t \*pinOutput = (uint8\_t \*) portOutputRegister(digitalPinToPort(pin)); // Get the port register for the pin.

volatile uint8\_t \*portMode = (uint8\_t \*) portModeRegister(digitalPinToPort(pin)); // Get the port mode register for the pin.

\*portMode |= pinBit; // Set pin to output mode.

#else

pinMode(pin, OUTPUT); // Set pin to output mode.

#endif

uint32\_t startTime = millis(); // Starting time of note.

while(millis() - startTime < duration) { // Loop for the duration.

#ifdef \_\_AVR\_\_

\*pinOutput |= pinBit; // Set pin high.

delayMicroseconds(duty); // Square wave duration (how long to leave pin high).

\*pinOutput &= ~pinBit; // Set pin low.

#else

digitalWrite(pin,HIGH); // Set pin high.

delayMicroseconds(duty); // Square wave duration (how long to leave pin high).

digitalWrite(pin,LOW); // Set pin low.

#endif

delayMicroseconds(frequency - duty); // Square wave duration (how long to leave pin low).

}

}