volatile int Dac[2];

volatile uint8\_t\* Port[] = { &OCR1A, &OCR1B };

volatile int Cycle = 0;

// Overflow interrupt

ISR (TIMER1\_OVF\_vect) {

static int rem[2];

for (int chan=0; chan<2; chan++) {

int remain;

if (Cycle == 0) remain = Dac[chan]; else remain = rem[chan];

if (remain >= 256) { \*Port[chan] = 255; remain = remain - 256; }

else { \*Port[chan] = remain; remain = 0; }

rem[chan] = remain;

}

Cycle = (Cycle + 1) & 0x0F;

}

void analogWrite12 (int chan, int value) {

cli(); Dac[chan] = value; sei();

}

void setup() {

// put your setup code here, to run once:

// Timer/Counter1 doing PWM on OC1A (PB1) and OC1B (PB4)

TCCR1 = 1<<PWM1A | 1<<COM1A0 | 1<<CS10;

GTCCR = 1<<PWM1B | 1<<COM1B0;

TIMSK = TIMSK | 1<<TOIE1;

pinMode(1, OUTPUT);

pinMode(4, OUTPUT);

}

int value = 0; // lower limit

void loop() {

// put your main code here, to run repeatedly:

value ++;

analogWrite12(0, value);

if (value == 4095 ) { //upperlimit

value = 0;

}

delay(2);

}