EMBEDDED SYSTEMS CMPE-453

Department of Computer Engineering

Pulse Width Modulation

Motivation

- ON or OFF in our AVR World (strictly digital). E.g: LEDs have been either on or off.
- LEDs to fade instead of blink
- A motor to run at half speed instead of being always on or off.
- Volume control.
- PWM is a way to make intermediate voltages from the AVR's logical high and low

- ***PWM*** *toggles the logic output on and off very fast, so quickly that whatever is attached to the output can't react fully.*

- *The result is that the output sees a voltage that is proportional to the average percent of the time that the AVR spends with its output on.*

Ratio of time spent in the on state to period is called the **duty cycle**

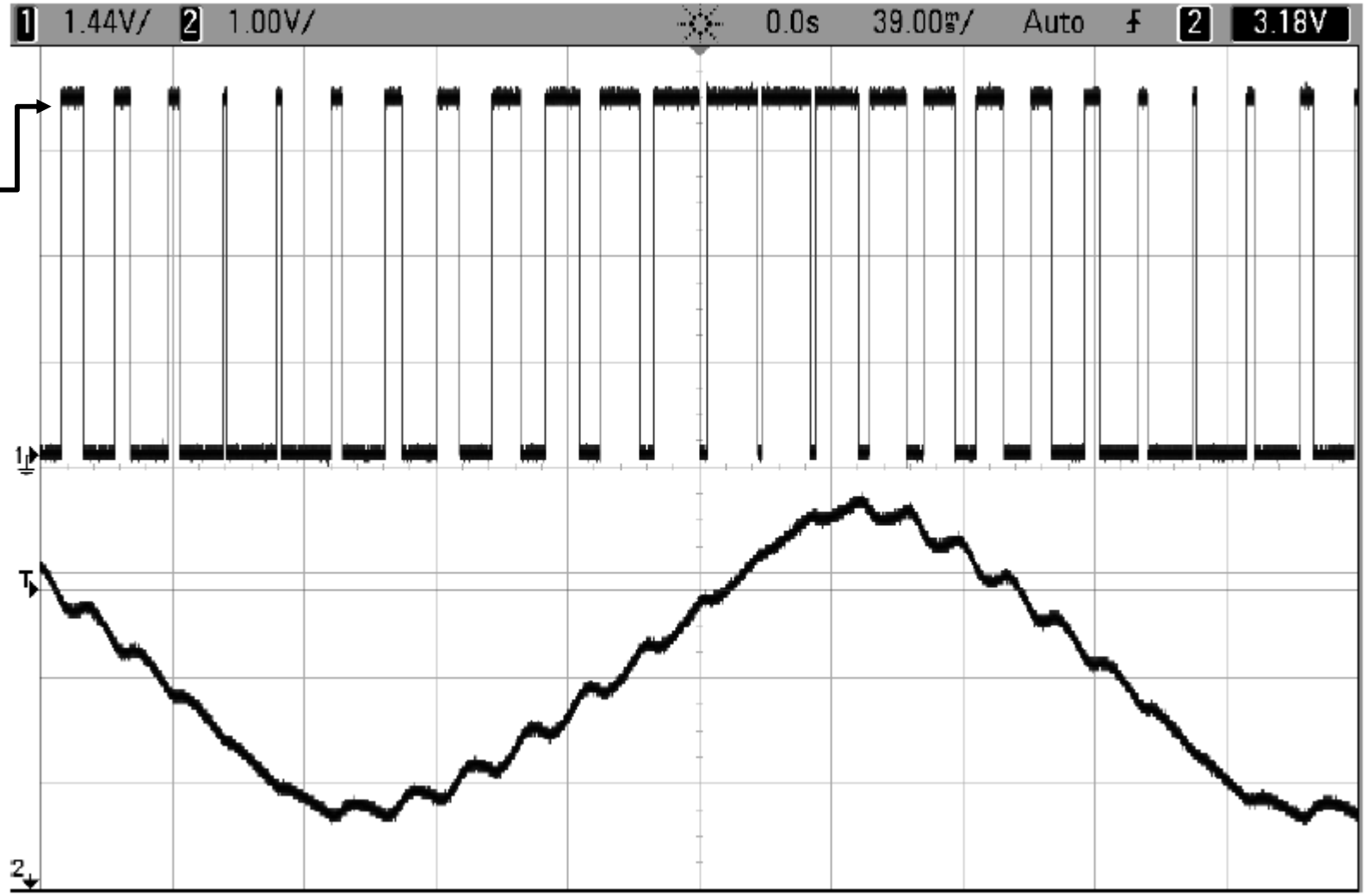


Figure 10-1. *PWM oscilloscope traces*

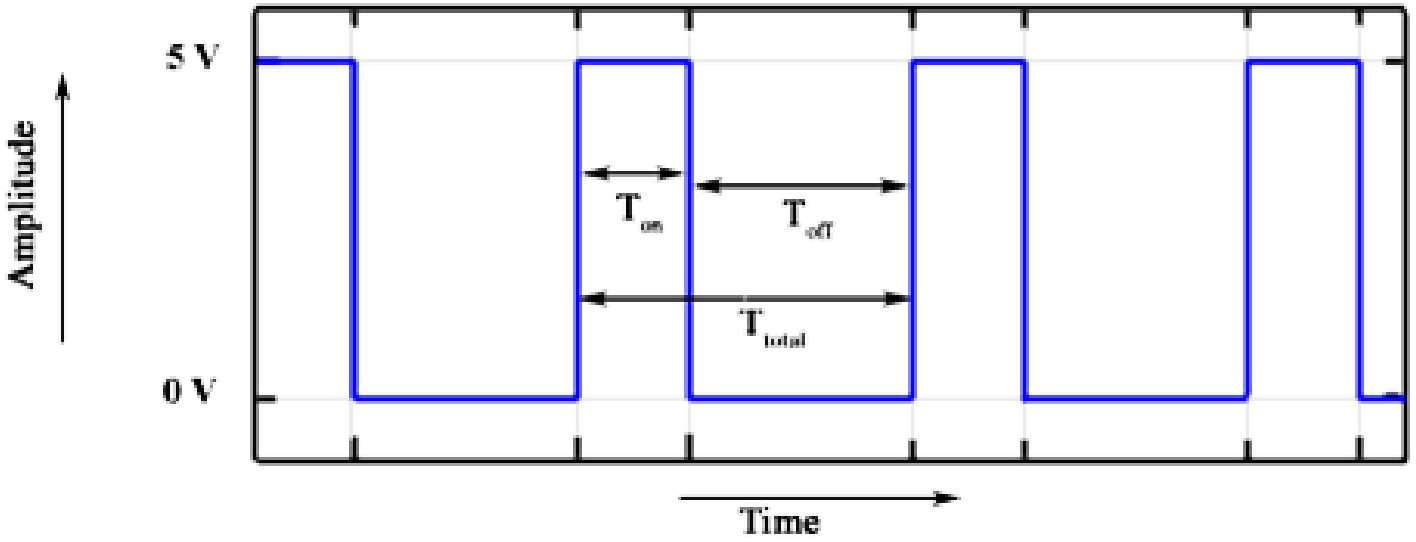
PWM frequency

↑

inversely proportional

↓

Ripples



$$D = \frac{T_{on}}{(T_{on} + T_{off})} = \frac{T_{on}}{T_{total}}$$

50% duty cycle

75% duty cycle

25% duty cycle

$$V_{out} = D \times V_{in}$$

$$\boxed{V_{out} = \frac{T_{on}}{T_{total}} \times V_{in}}$$

Example:

LED brightening and dimming

```
// ------- Preamble -------- //
#include <avr/io.h>                        /* Defines pins, ports, etc */
#include <util/delay.h>                     /* Functions to waste time */
#include "pinDefines.h"

#define LED_DELAY  20                                 /* microseconds */

void pwmAllPins(uint8_t brightness) {
  uint8_t i;
  LED_PORT = 0xff;                                          /* turn on */
  for (i = 0; i < 255; i++) {
    if (i >= brightness) {           /* once it's been on long enough */
      LED_PORT = 0;                                        /* turn off */
    }
    _delay_us(LED_DELAY);
  }
}

int main(void) {

  uint8_t brightness = 0;
  int8_t direction = 1;

  // -------- Inits --------- //

  // Init all LEDs
  LED_DDR = 0xff;
  // ------ Event loop ------ //
  while (1) {
    // Brighten and dim
     if (brightness == 0) {
       direction = 1;
     }
     if (brightness == 255) {
       direction = -1;
     }
     brightness += direction;
     pwmAllPins(brightness);


   }                                              /* End event loop */
   return (0);                            /* This line is never reached */
 }
```

#step per cycle: N
delay: m sec. per step

f_{pwm}=1 / (delay x N)

TinkerCad demo !

Example: Using Timers

```
                    /* PWM Demo with serial control over three LEDs */

// ------- Preamble -------- //
#include <avr/io.h>                         /* Defines pins, ports, etc */
#include <util/delay.h>                      /* Functions to waste time */
#include "pinDefines.h"
#include "USART.h"

static inline void initTimers(void) {
  // Timer 1 A,B
  TCCR1A |= (1 << WGM10);                     /* Fast PWM mode, 8-bit */
  TCCR1B |= (1 << WGM12);                     /* Fast PWM mode, pt.2 */
  TCCR1B |= (1 << CS11);                    /* PWM Freq = F_CPU/8/256 */
  TCCR1A |= (1 << COM1A1);                    /* PWM output on OCR1A */
  TCCR1A |= (1 << COM1B1);                    /* PWM output on OCR1B */

  // Timer 2
  TCCR2A |= (1 << WGM20);                           /* Fast PWM mode */
  TCCR2A |= (1 << WGM21);                     /* Fast PWM mode, pt.2 */
  TCCR2B |= (1 << CS21);                    /* PWM Freq = F_CPU/8/256 */
  TCCR2A |= (1 << COM2A1);                    /* PWM output on OCR2A */
}

int main(void) {

  uint8_t brightness;

  // -------- Inits --------- //

  initTimers();
  initUSART();
  printString("-- LED PWM Demo --\r\n");

              /* enable output on LED pins, triggered by PWM hardware */
  LED_DDR |= (1 << LED1);
  LED_DDR |= (1 << LED2);
  LED_DDR |= (1 << LED3);

  // ------ Event loop ------ //
  while (1) {

    printString("\r\nEnter (0-255) for PWM duty cycle: ");
    brightness = getNumber();
    OCR2A = OCR1B;
    OCR1B = OCR1A;
    OCR1A = brightness;

  }                                                 /* End event loop */
  return (0);                             /* This line is never reached */
}
```

TinkerCad demo !