Timer1 Count Verification May 23 23

Jun ha Kim

Advisor: Professor Raul Armendariz

Timer1 explained (1 of 2)

- The Arduino Mega 2560 has a processor that has a clock of 16 Mhz
- This means that it’s clock increments in 1/16,000,000 of a second (which is 62.5 * 10^-9 seconds = 62.5 ns)
- Timer 1 is an interrupt routine, with a 16-bit length counter
- Counts to 65,535 units or $(2^{16} - 1)$ (before overflowing back to 0), every time the Timer1 ISR is called

Timer1 explained (2 of 2)

- There is some flexibility to this statement; Timer1 counts can be used to confirm time elapsed between two outputs in the serial monitor
- When Timer1 reaches 65,535 that means it has counted for 65,535/16,000,000 = 0.004096 seconds or 4.096 ms
- **Objective**: Learn how to use Timer1 to potentially attain higher time resolution

Timer1 Calculation Results (2 ms)

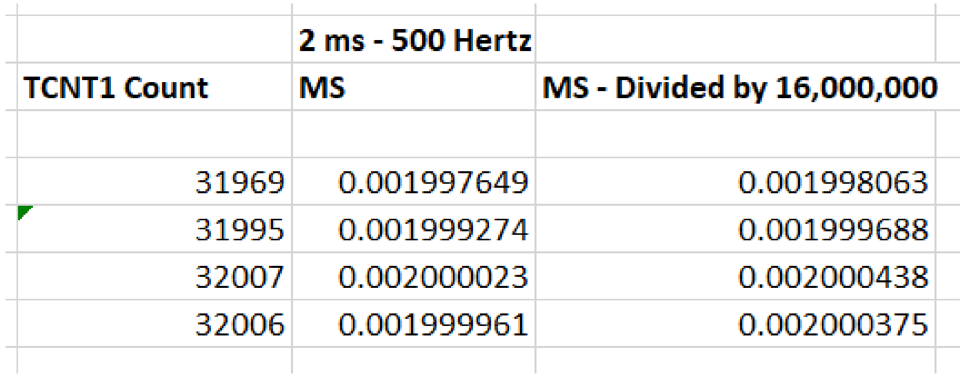
- 2 ms (pulse generator: 500 Hertz)
- Sample raw results:

Windows System Time (Seconds : Milliseconds)	TCNT1 Count
19.468	8713 (pulse 1)
19.468	40682 (pulse 2)
19.468	7142 (pulse 3)

- **Note: The following steps shall take place in Excel:**
- TCNT1 Count Formula*: (Second row result) – (First row result)
- e.g. 40682 – 8713 results in a TCNT1 count of 31969
- 31969/16,000,000 ≈ 0.001999, thus corresponds to 2 ms

Timer1 Calculation Results (2 ms) Warnings

- In the previous slide, the Timer1 Counter overflowed by the time the third pulse came in. To calculate the elapsed time between the 3rd and 2nd pulses: 7142 + (65535 – 40682) = 31995. 31995/16,000,000 = 0.001995 seconds or 2 ms.
- Therefore, if a given row result is on the verge of approaching 65,535; use the following formula: (65,535 - given row result) + (next row result)
- **The steps described above must take place in Excel**



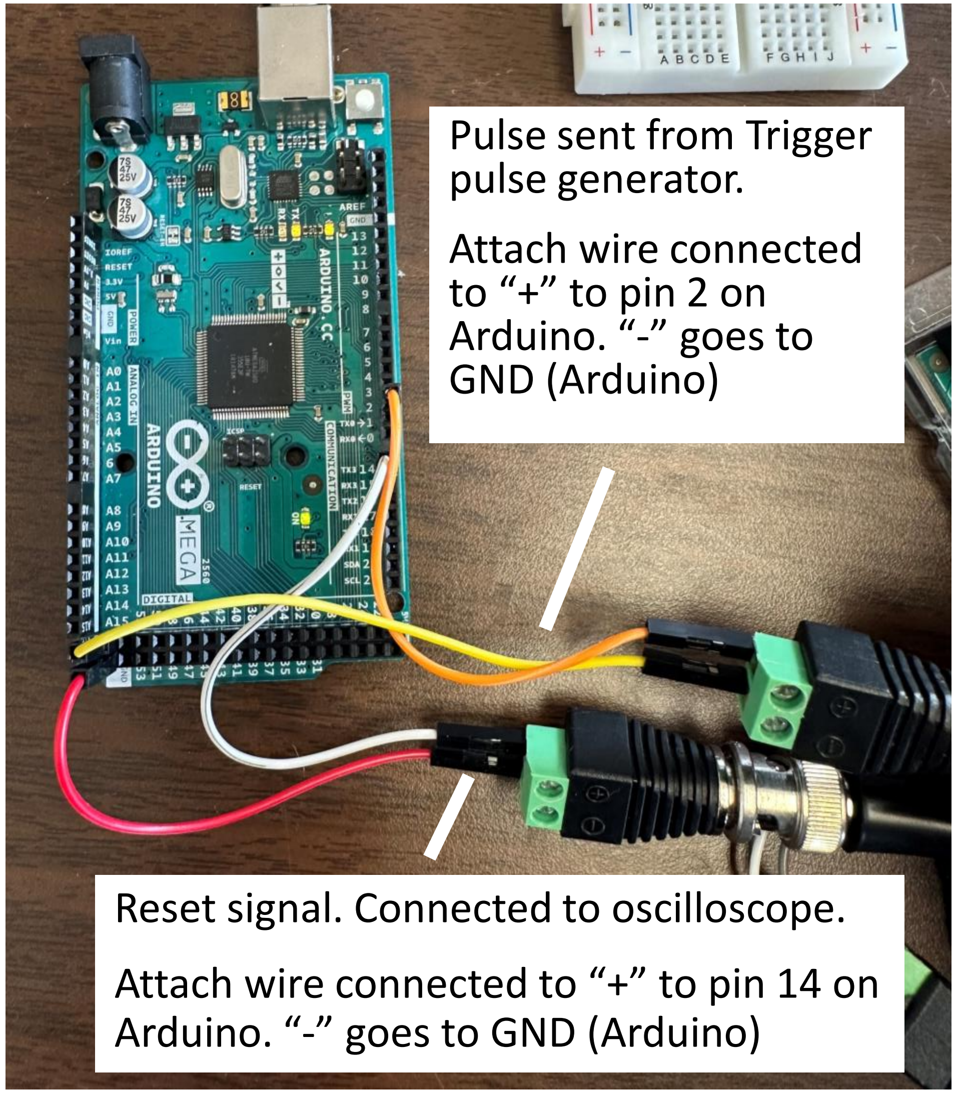
	2 ms - 500 Hertz	
TCNT1 Count	MS	MS - Divided by 16,000,000
31969	0.001997649	0.001998063
31995	0.001999274	0.001999688
32007	0.002000023	0.002000438
32006	0.001999961	0.002000375

- Calculations performed in Excel
- Divide each, individual figure found in TCNT1 Count column by 16,000,000 to acquire result in 2ms column

Timer1 Calculation, Issues at 1ms level

- Repeat the calculations for 1 ms (1000 Hertz) and 3 ms (333 Hertz) using the steps and processes described in the previous slide
- When verifying Timer1 calculations at the 1 ms level, note that the result comes out to 0.00168; a figure that is markedly higher than 0.001 (intended result)
- More research is required to see why the above scenario occurs
- <u>We used an oscilloscope to calibrate the period of the pulse from the pulse generator</u>

Arduino Setup


Pulse sent from Trigger pulse generator.

Attach wire connected to "+" to pin 2 on Arduino. "-" goes to GND (Arduino)

Reset signal. Connected to oscilloscope.

Attach wire connected to "+" to pin 14 on Arduino. "-" goes to GND (Arduino)

- Items required:
  - Pulse generator (Trigger)
    - Period and width settings must be at the correct settings to derive desired frequency
  - Oscilloscope
  - Arduino Mega 2560 (Breadboard is not required for this exercise)

Libraries, Pin Number Names, Global Variables

// Libraries:

#include <SPI.h> // Allows you to communicate with SPI (Serial Peripheral Interface) devices, with the Arduino as the master device

#include <Wire.h> // Enable this line if using Arduino Uno, Mega, etc.

// Signal pins are given a name, Global variables

#define triggerPin 2 // Trigger signal pin

Interrupt Service Routine and setup() functions

```
// Interrupt service routine
// This function must be implemented, so that the TCNT1 counter counts
ISR(TIMER1_OVF_vect)
{
}
```

// All Arduino programs must contain a setup() and loop() functions

```
void setup() {
   Serial.begin(115200); // Starts the serial monitor, sets baudrate to
"115200" BPS
   pinMode(triggerPin,INPUT); // Sets the digital pin 2 as an input
delay(1000); // Pauses the program for one second at the moment of open
```

setup() function *continued*

```
// Initializes the Timer1 registers (16-bit timer -- counts from 0 to 65535 ad nauseam). Timer interrupts/pauses the execution of the loop() function for a predefined number of seconds.

// Timer1 is a 16-bit timer, so the timer will increase its value until it reaches its maximum count before reverting to 0. This enables the program to run a different set of commands. Once executed, the program resumes at the same position.

   TCCR1A = 0; // Sets entire TCCR1A--Timer1 Control Register A--to 0

   TCCR1B = 0; // Timer 1 Control Register B set to 0 (The physical address of timer1)

   TCCR1C = 0; // Timer 1 Control Register C set to 0

   TCNT1 = 0; // Initialize timer/counter 1's value to 0

    TIMSK1 = _BV(TOIE1); // Timer/Counter1's interrupt mask register; TOIE1 is the timer/Counter1 overflow interrupt enable

   TCCR1B = 1; // Timer 1 Control Register B set to 1


   attachInterrupt(digitalPinToInterrupt(triggerPin), Trigger, RISING); // Interrupts execution of the program when a trigger signal is received. The "Trigger" function is subsequently executed

}
```

Trigger() and loop() functions

```
void Trigger(){
   unsigned int temp = TCNT1; // Only positive integers are
required
   Serial.print("TCNT1 value: ");
   Serial.println(temp); // Prints the value stored at temp
}

void loop() {
  // No lines are necessary here
}
```