BLG351E Experiment 6 “Playing with Time” REPORT	CRN	12635
	Group	Group 11
	Name #1	Fatih Baskın
	Name #2	Mehmet Eymen Ünay
	Name #3	Nada Malek
	Name #4	Rojen Arada Şeşen

Q1) (30 pts.) Explain the TA0CTL register. How the bits of the register were set?

TA0CTL is the control register for Timer A0 and there are several bits that should be set in order to set timer action correctly. This register is 16 bits read/write memory space labeled as TA0CTL.
Bits: 15-10 -> unused

- 9-8 -> TASSELx, this is the clock source, we set these bits to 10 to select SMCLK.
- 7-6 -> These bits were used for input divider. We set it to 00 to divide by 1.
- 5-4 -> These bits are for mode control. We set this to 01, which means up mode.
- 3 -> Unused
- 2 -> TACLR, this bit is for clearing the counter, we set it as 0
- 1 -> TAIE, set this to 1 for enabling interrupt
- 0 -> TAIFG, Timer A intterupt flag, set this to 0.

According to information above, we set the TA0CTL with following instruction:

```
mov.w   #0000 0010 0001 0010 b, &TA0CTL
```

Q2) (40 pts.) Explain how to stop the timer interrupts?

Bit 1 of the 16 bit TACTL (Timer_A Control Register) is the interrupt enable bit, TAIE. If we set TAIE, timer interrupt is enabled. If we set it to 0, interrupts are disabled. Since we are setting TAIFG to 0 every time an interrupt occurs to prevent calling interrupt subroutines infinitely many times, it is unnecessary to set it to 0.

```
mov.w   #xxxx xxxx xxxx xx1xb, &TA0CTL -> Interrupts enabled
mov.w   #xxxx xxxx xxxx xx0xb, &TA0CTL -> Interrupts disabled
```

Q3) (30 pts.) Explain your constructed mechanism for the features of the chronometer. How did you handle the events? (such as reset, start etc.)

All these actions would be mapped to certain interrupts and those interrupt subroutines would be called upon those interrupt flags are set.

Four registers would be kept to hold data of centiseconds(10^-2), deciseconds(10^-1), seconds and decaseconds(10^1). Centiseconds will be incremented when counter flag is set. When a register reaches ten, it will send carry to next register (increment) and set itself to 0.

Inside the main loop, value inside four registers will be used to access 7 segment display output of correct number from the array for ex:

mov.w R1(array), &P2OUT

Reset: In reset, bit 2 of TA0CTL should be set to 1 to clear counter, also four registers that holds time should be set to 0.

Start: In start, if stop interrupt is also set, program should jump to save best time, otherwise bits 5 and 4 of TA0CTL should be set to 01 to continue upcounting.

Stop: In stop, if start if stop interrupt is also set, program should jump to save best time, otherwise bits 5 and 4 of TA0CTL should be set to 00 to stop counting.

Save Best Time: Since seconds and centiseconds are saved as 1-byte values, with using BCD conversion subroutine we converted decaseconds and seconds into seconds and deciseconds and centiseconds into seconds. Then this returned values must be saved into memory and both interrupt flags should be reseted and program should return to main.

BCD conversion: There are two variables, register10 and register1, using a basic loop to multiply value of register10’s value by 10 and adding register 1’s value end returning the result in order to convert BCD into binary data.