PIC Activity #7

Interrupt Service Routine (ISR)

① processor stops → pause
② jumps to a function ISR
③ Run code w/ ISR
④ go back to main → Resume

TMR0IF (Timer 0 Interrupt flag)

① activates when timer expires
② Enable timer system to respond automatically to this flag!

Focus

① ISR
② Enable interrupt systems and individual interrupts
③ Managing interrupt flags

Timer 2 (1ms system tick) → interrupts (1ms)

Timer 1 (audio generation) → interrupts (1:8 prescaler) 0.5µs per tick

↳ choose the next sin value from the table

GIE (Global interrupt enable)

① GIEH → high priority interrupts
② GIEL → Low priority interrupts
③ If off, then no individual interrupts will trigger the processor

Add two lines of code in the event handler section of timer1 ISR to grab the next sin value for DAC1DATL and increment index by 4

```
DAC1DATL = Gu8UserAppSinTable[u8Index];
u8Index += 4;
```

timer runs the program in 16µs intervals until 1 period (1ms) is complete and the TMR0IF flag is enabled.

call InterruptTimeXus( ) ← 1000, true ↳ so that its continuous

$1\text{KHz time} = \frac{1}{1\times10^{+3}} = 1\text{ms} = 1000\mu s$

256 sine values increment by 4] - $256/4 = 64$ steps in one period. → # of times the index updates

$\frac{10^{-3}\text{s}}{64} = 16\mu s$ per step] - 1 period → already determined by timer's configuration

takes 16µs to update.

16 ~~from~~ ~~[illegible]~~ µs ← [illegible] the time ~~per~~