Button based delay

The button based delay in this lab first designates P1 for the output and BIT1 for the LED wire connection.

As the button is pressed the databoard will process the amount of time in seconds the button is pressed and convert that value to clock cycles at a rate of 1 second = 250 hz. For a maximum value of 65000 hz (16 seconds).

|  |
| --- |
| #pragma vector=PORT1\_VECTOR |
|  |

|  |
| --- |
| \_\_interrupt void Port\_1(void){ |
|  |

|  |
| --- |
| if ((P1IN&BIT1)==0x0000) //if button pressed |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| P1IES &= ~BIT1; //sets interupt to trigger on rising edge (button release) |
|  |

|  |
| --- |
| TB0CTL = TBSSEL\_2 + ID\_3 + MC\_2 + TBCLR; // SMCLK, divide clock by 8, continous mode, clear TAR |
|  |

|  |
| --- |
| P1IFG &= ~BIT1; //clear interupt flag |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| else //if button released |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| P1IES |= BIT1; //sets interupt to trigger on falling edge (button press) |
|  |

|  |
| --- |
| TB0CCR0 = TB0R; //changes ccr0 to value of timer |
|  |

|  |
| --- |
| TB0CTL = TBSSEL\_2 + ID\_3 + MC\_1 + TBCLR; // SMCLK, divide clock by 8, upmode, clear TAR |
|  |

|  |
| --- |
| P1IFG &= ~BIT1; //clears interupt flag |
|  |

|  |
| --- |
| } |
|  |

}

This code shows the relationship between button pressed time, to the conversion rate, then to the output toggle of the LED.