The 2025 Double Dabble Project will require a Timer1 Overflow Event to trigger the start of an ADC conversion every second.

// PROJECT :ADCAssemblyTimer1OVFTriggerSource

// PURPOSE :ADC in Assembly. Conversions triggered by Timer1 Overflow

// COURSE :ICS4U-E

// AUTHOR :C. D'Arcy

// DATE :Created 2025 03 29. Confirmed: 2025 04 15.

// MCU :328P

// STATUS :Working

// REFERENCE:http://darcy.rsgc.on.ca/ACES/TEI4M/Assembly/images/InterruptVectorTable.png

// :http://www.rjhcoding.com/avr-asm-tutorials.php

// NOTE :No need to build an IVT. Just define the expected ISR names as .global

#include <avr/io.h> //required to reference ports by

#include "Timerprescalers.h" //local Timer defines

#include "ADCprescalers.h" //local ADC defines

.global main ;bypass Arduino C requirement of setup & loop

.global TIMER1\_OVF\_vect ;sufficient for handling this interrupt

.global ADC\_vect ;sufficient for handling this interrupt

;.section .data ;for (initialized) data to be placed in SRAM

;.section .bss ;for (uninitialized) data to be placed in SRAM

.section .text ;for Program Flash (.section is optional)

.equ plHI, 0x00 ;optional preload (0xFDDD yields 120ovf/s=60Hz)

.equ plLO, 0x00 ;optional preload for TCNT1L (0xFD70=50Hz, 0xFCCC=40Hz)

util = 16 ;good practice to name registers

;configure the Interrupt Vector Table

;NOTE: It's not necessary to write this vector table since we're using avr-libc's pre-defined vector table.

;Just define your ISR as a .global with the appropriate name as above...

.org 0x00 ;Reset Interrupt #0

 jmp main

.org 0x1A ;Timer1 Overflow Interrupt #13

 jmp TIMER1\_OVF\_vect

.org 0x2A ;ADC Complete Interrupt #21

 jmp ADC\_vect

.org \_VECTORS\_SIZE ;add Program Memory data starting here, if required

data:

 .word 1023 ;for example

.org 0x100 ;locate PC well beyond the IVT & data storage

main: ;code starts here...

 rcall IOSetup ;configure the IO port(s)

 rcall TIMER1Setup ;Timer 1 Mode 0 with preload

 rcall ADCSetup ;

 rcall USARTSetup ;

 sei ;enable global interrupts

hold:

 rjmp hold ;hold. Should blink...

 ret

USARTSetup:

 ret

IOSetup:

 sbi DDRB-0x20,DDB5 ;pinMode(13,OUTPUT);

 ret

ADCSetup:

 ser util ;r16 <- 0xFF

 sts DIDR0, util ;disable digital pins on PORTC when using ADC

 ldi util, ADCAVCC ;AVCC as voltage reference (0.1uF on AREF)

 ori util, ADC5 ;choose (arbitrarily) analog pin A5

 sts ADMUX,util ;do it

 ldi util, 1<<ADEN ;enable the ADC utility

 ori util, 1<<ADSC ;start a first dummy conversion

 ori util, ADCps128 ;recommended prescaler to get to 125kHz

 sts ADCSRA, util ;do it

dummy:

 lds util,ADCSRA ;wait for it to complete

 sbrs util,ADIF ;is the first 25-cycle conversion complete?

 rjmp dummy ;keep waiting....

 lds util, ADCSRA ;prepar to clear the ADIF flag

 ori util, 1<<ADIF ;ADIF flag is cleared by writing a 1 to it

 ori util, 1<<ADATE ;enable external trigger to start a conversion

 sts ADCSRA,util ;

; clr util ;Free Running Mode (not in this instance)

 ldi util, ADCtsT1OVF ;select Timer1 Overflow as Trigger Source

 sts ADCSRB,util ;do it

 lds util,ADCSRA ;need to enable ADC Complete Interrupt...

 ori util, 1<<ADIE ;set the interrupt Enable bit...

 sts ADCSRA,util ;do it

 ret

TIMER1Setup:

 clr util ;prepare for Normal Mode

 sts TCCR1A, util ;set Normal Mode, now configure the prescaler...

 ldi util, T1ps64 ;T1:2^24/2^8/2^16 (prescale)>1 ovf/s > 0.5Hz

 sts TCCR1B, util ;Timer1 clock = system clock / prescale

 ldi util,plHI ;load TCNT1 (Timer1's 2-byte counter)

 sts TCNT1H,util ;

 ldi util,plLO ;

 sts TCNT1L,util ;at 60Hz there appears to be some flicker

 ldi util,1<<TOIE1 ;enable Timer/Counter1 Overflow Interrupt

 sts TIMSK1,util ;enable Timer1 Overflow interrupts

 ret

TIMER1\_OVF\_vect: ;destination for TCNT1 overflow interrupt

 ldi util,plHI ;load TCNT1 (Timer1's 2-byte counter)

 sts TCNT1H,util ;

 ldi util,plLO ;

 sts TCNT1L,util ;at 60Hz there appears to be some flicker

 reti

ADC\_vect: ;ISR handler for ADC Complete

 sbi PINB-0x20,DDB5 ;writing a 1 to PINx toggles the in PORTx

 reti

# Timer1prescalers.h

// Prescale constants for ATmega328P Timers

#define T0Stopped 0b00000000 ; Timer0 stopped

#define T0psNone  0b00000001 ; T0:2^24/2^8  (no prescale)> 2^? ovf/s = ? Hz

#define T0ps8     0b00000010 ; T0:2^24/2^3/2^8 (prescale)> 2^? ovf/s = ? Hz

#define T0ps64    0b00000011 ; T0:2^24/2^6/2^8 (prescale)> 2^? ovf/s = ? Hz

#define T0ps256   0b00000100 ; T0:2^24/2^8/2^8 (prescale)> 2^? ovf/s = ? Hz

#define T0ps1024  0b00000101 ; T0:2^24/2^10/2^8(prescale)> 2^? ovf/s = ? Hz

#define T1Stopped 0b00000000 ; Timer1 stopped

#define T1psNone  0b00000001 ; T1:2^24/2^16  (no prescale)>2^8 ovf/s> 128Hz

#define T1ps8     0b00000010 ; T1:2^24/2^3/2^16 (prescale)> 2^5 ovf/s> 16Hz

#define T1ps64    0b00000011 ; T1:2^24/2^6/2^16 (prescale)> 2^2 ovf/s> 2Hz

#define T1ps256   0b00000100 ; T1:2^24/2^8/2^16 (prescale)> 1 ovf/s> 0.5Hz

#define T1ps1024  0b00000101;T1:2^24/2^10/2^16(prescale)>0.25ovf/s> 0.125Hz

#define T2Stopped 0b00000000 ; Timer2 stopped

#define T2psNone  0b00000001 ; T2:2^24/2^8  (no prescale)> 2^? ovf/s > ? Hz

#define T2ps8     0b00000010 ; T2:2^24/2^3/2^8 (prescale)> 2^? ovf/s = ? Hz

#define T2ps32    0b00000011 ; T2:2^24/2^5/2^8 (prescale)> 2^? ovf/s = ? Hz

#define T2ps64    0b00000100 ; T2:2^24/2^6/2^8 (prescale)> 2^? ovf/s = ? Hz

#define T2ps128   0b00000101 ; T2:2^24/2^7/2^8(prescale)> 2^? ovf/s = ? Hz

#define T2ps256   0b00000110 ; T2:2^24/2^8/2^8(prescale)> 2^? ovf/s = ? Hz

#define T2ps1024  0b00000111 ; T2:2^24/2^10/2^8(prescale)> 2^? ovf/s = ? Hz

# ADCprescalers.h

;Library of ADC-related equates for the ATmega328P

;------------------------------------------------------------------

; ADC (Chapter 24)  Voltage Reference Selection for the ADC

.equ    ADCVREF   , 0<<REFS1 | 0<<REFS0  ;AREF, Interval Vref turned off

.equ    ADCAVCC   , 0<<REFS1 | 1<<REFS0  ;AVcc, ext. capacitor at AREF pin

.equ    ADCRSVD   , 1<<REFS1 | 0<<REFS0  ;Reserved

.equ    ADCINTR   , 1<<REFS1 | 1<<REFS0  ;Interval 1.1V Voltage Reference

;------------------------------------------------------------------

; ADC (Chapter 24)  Recommended ADC run between 50k and 200k

.equ    ADCps2    , 0<<ADPS2 | 0<<ADPS1 | 1<<ADPS0  ;Division Factor 2

.equ    ADCps4    , 0<<ADPS2 | 1<<ADPS1 | 0<<ADPS0  ;Division Factor 4

.equ    ADCps8    , 0<<ADPS2 | 1<<ADPS1 | 1<<ADPS0  ;Division Factor 8

.equ    ADCps16   , 1<<ADPS2 | 0<<ADPS1 | 0<<ADPS0  ;Division Factor 16

.equ    ADCps32   , 1<<ADPS2 | 0<<ADPS1 | 1<<ADPS0  ;Division Factor 32

.equ    ADCps64   , 1<<ADPS2 | 1<<ADPS1 | 0<<ADPS0  ;Division Factor 64

.equ    ADCps128  , 1<<ADPS2 | 1<<ADPS1 | 0<<ADPS0  ;Division Factor 128

;------------------------------------------------------------------

; ADC (Chapter 24)  Trigger Source

.equ    ADCtsFR   , 0<<ADTS2 | 0<<ADTS1 | 0<<ADTS0  ;Free Running Mode

.equ    ADCtsAC   , 0<<ADTS2 | 0<<ADTS1 | 1<<ADTS0  ;Analog Comparator

.equ    ADCtsEI   , 0<<ADTS2 | 1<<ADTS1 | 0<<ADTS0  ;Ext. Int. Request 0

.equ    ADCtsT0CM , 0<<ADTS2 | 1<<ADTS1 | 1<<ADTS0  ;TCounter0 Comp Match A

.equ    ADCtsT0OVF, 1<<ADTS2 | 0<<ADTS1 | 0<<ADTS0  ;TCounter0 Overflow

.equ    ADCtsT1CM , 1<<ADTS2 | 0<<ADTS1 | 1<<ADTS0  ;TCounter1 Comp Match A

.equ    ADCtsT1OVF, 1<<ADTS2 | 1<<ADTS1 | 0<<ADTS0  ;TCounter1 Overflow

.equ    ADCtsT1CAP, 1<<ADTS2 | 1<<ADTS1 | 1<<ADTS0  ;TCounter1 Capt. Event

;------------------------------------------------------------------

; ADC (Chapter 24)  Single-Ended Input

.equ    ADC0      , 0<<MUX3  | 0<<MUX2  | 0<<MUX1  | 0<<MUX0  ;

.equ    ADC1      , 0<<MUX3  | 0<<MUX2  | 0<<MUX1  | 1<<MUX0  ;

.equ    ADC2      , 0<<MUX3  | 0<<MUX2  | 1<<MUX1  | 0<<MUX0  ;

.equ    ADC3      , 0<<MUX3  | 0<<MUX2  | 1<<MUX1  | 1<<MUX0  ;

.equ    ADC4      , 0<<MUX3  | 1<<MUX2  | 0<<MUX1  | 0<<MUX0  ;

.equ    ADC5      , 0<<MUX3  | 1<<MUX2  | 0<<MUX1  | 1<<MUX0  ;

.equ    ADC6      , 0<<MUX3  | 1<<MUX2  | 1<<MUX1  | 0<<MUX0  ;

.equ    ADC7      , 0<<MUX3  | 1<<MUX2  | 1<<MUX1  | 1<<MUX0  ;

.equ    ADC8      , 1<<MUX3  | 0<<MUX2  | 0<<MUX1  | 0<<MUX0  ; Temp.Sensor

.equ    ADC1V     , 1<<MUX3  | 1<<MUX2  | 1<<MUX1  | 0<<MUX0  ; 1.1V

.equ    ADCGND    , 1<<MUX3  | 1<<MUX2  | 1<<MUX1  | 1<<MUX0  ; 0V (GND)