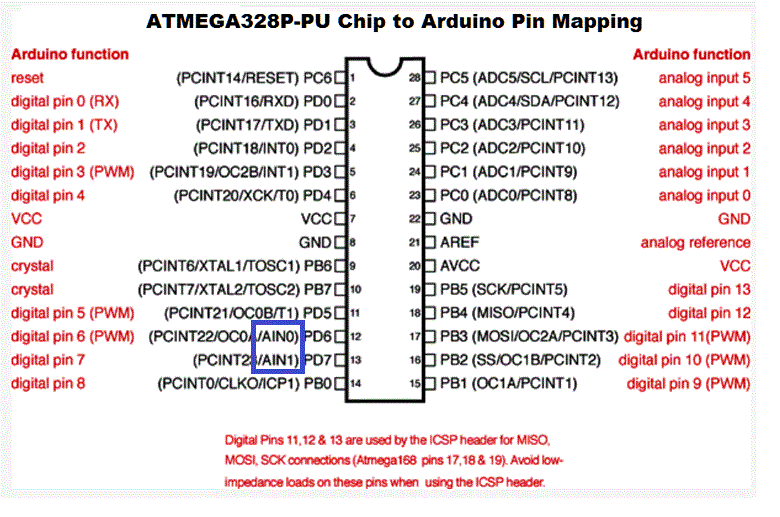
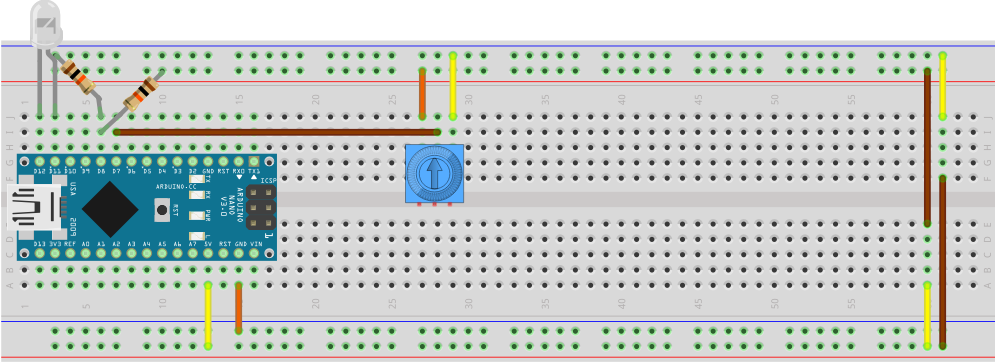
The ATmega328p’s onboard analog comparator pins (AIN0 and AIN1) are highlighted below.



A possible prototype to explore this peripheral appears below.

* A fixed voltage divider (Vcc/2) is impressed upon D7, the negative (inverting) pin AIN1.
* A variable voltage from a potentiometer appears on D6, the non-inverting (positive) pin AIN0.
* A bicolor LED spanning D11 and D12 acts as a status indicator for the ACO flag of the ACSR.



Register-Level and Assembly code appears on the next page.

// PROJECT  :AnalogComparatorRegisterLevel

// PURPOSE  :Register level intro to the 328p's Analog Comparator peripheral with bicolor LED

// COURSE   :ICS4U-E

// AUTHOR   :C. D'Arcy

// DATE     :2025 01 24

// MCU      :328P

// STATUS   :Working

// REFERENCE:[http://darcy.rsgc.on.ca/ACES/Datasheets/ATmega328PBFull.pdf#page=335](http://darcy.rsgc.on.ca/ACES/Datasheets/ATmega328PBFull.pdf%23page=335)

#define DDR     DDRB          // active data direction register for bicolor LED

#define PORT    PORTB         // active port

#define RED     (1 << PB3)    // anode pin for red led (cathode for green)

#define GREEN   (1 << PB4)    // anode pin for green led (cathode for red)

void setup() {

  DDR = RED | GREEN;          // set direction for LED leads to output

  ACSR = 0;                   // clear all bits in the Analog Comparator Status Register

}

void loop() {

  PORT = ACSR & (1 << ACO) ? RED : GREEN;   // respond appropriately to analog comparator output

}

// PROJECT :AnalogComparator

// PURPOSE :AVR Assembly intro of the 328P's Analog Comparator to the 2024/25 12s

// REFERENCE:[http://darcy.rsgc.on.ca/ACES/Datasheets/ATmega328PBFull.pdf#page=335](http://darcy.rsgc.on.ca/ACES/Datasheets/ATmega328PBFull.pdf%23page=335)

#include "avr/io.h"

.equ DDR, DDRB-0x20 ;

.equ PORT, PORTB-0x20 ;

.equ RED, 3 ;

.equ GREEN, 4 ;

.global main

main:

rcall initIO ;

rcall initAC ;

wait:

lds r16,ACSR ;

sbrs r16,ACO ;

rjmp around ;

rcall doRed ;

rjmp wait ;

around:

rcall doGrn ;

rjmp wait ;

ret

initIO:

sbi DDR,RED ;

sbi DDR,GREEN ;

ret

initAC:

clr r16 ;

sts ACSR,r16 ;all bits are cleared to function

ret

doRed:

sbi PORT,RED ;

cbi PORT,GREEN ;

ret

doGrn:

sbi PORT,GREEN ;

cbi PORT,RED ;

ret