INTRODUCTION TO 8-BIT AVR (ATMEGA8)

Week 1

OVERVIEW

- Atmega8 Power Supply Requirements
- 7805 Regulator
- Atmega8 Power Supply Decoupling
- Reading Datasheet
- Interfacing with Push Buttons
- Interfacing with LED
- ISP Programming
- Troubleshooting Example

Power Supply Requirements

Features

- High-performance, Low-power Atmel®AVR® 8-bit Microcontroller
- Advanced RISC Architecture
  - 130 Powerful Instructions – Most Single-clock Cycle Execution
  - 32 × 8 General Purpose Working Registers
  - Fully Static Operation
  - Up to 16MIPS Throughput at 16MHz
  - On-chip 2-cycle Multiplier
- High Endurance Non-volatile Memory segments
  - 8Kbytes of In-System Self-programmable Flash program memory
  - 512Bytes EEPROM
  - 1Kbyte Internal SRAM
  - Write/Erase Cycles: 10,000 Flash/100,000 EEPROM
  - Data retention: 20 years at 85°C/100 years at 25°C[1]
  - Optional Boot Code Section with Independent Lock Bits
    In-System Programming by On-chip Boot Program
    True Read-While-Write Operation
  - Programming Lock for Software Security
- Peripheral Features
  - Two 8-bit Timer/Counters with Separate Prescaler, one Compare Mode
  - One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
  - Real Time Counter with Separate Oscillator
  - Three PWM Channels
  - 8-channel ADC in TQFP and QFN/MLF package
    Eight Channels 10-bit Accuracy
  - 6-channel ADC in PDIP package
    Six Channels 10-bit Accuracy
  - Byte-oriented Two-wire Serial Interface
  - Programmable Serial USART
  - Master/Slave SPI Serial Interface
  - Programmable Watchdog Timer with Separate On-chip Oscillator
  - On-chip Analog Comparator
- Special Microcontroller Features
  - Power-on Reset and Programmable Brown-out Detection
  - Internal Calibrated RC Oscillator
  - External and Internal Interrupt Sources
  - Five Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, and Standby
- I/O and Packages
  - 23 Programmable I/O Lines
  - 28-lead PDIP, 32-lead TQFP, and 32-pad QFN/MLF
- Operating Voltages
  - 2.7V - 5.5V (ATmega8L)
  - 4.5V - 5.5V (ATmega8)
- Speed Grades
  - 0 - 8MHz (ATmega8L)
  - 0 - 16MHz (ATmega8)
- Power Consumption at 4Mhz, 3V, 25°C
  - Active: 3.6mA
  - Idle Mode: 1.0mA
  - Power-down Mode: 0.5µA

8-bit AVR® with 8KBytes In-System Programmable Flash

ATmega8
ATmega8L

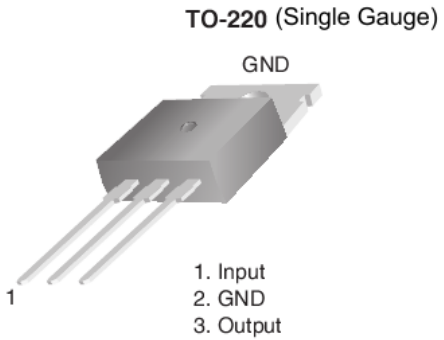
Rev.2486Z–AVR–02/11

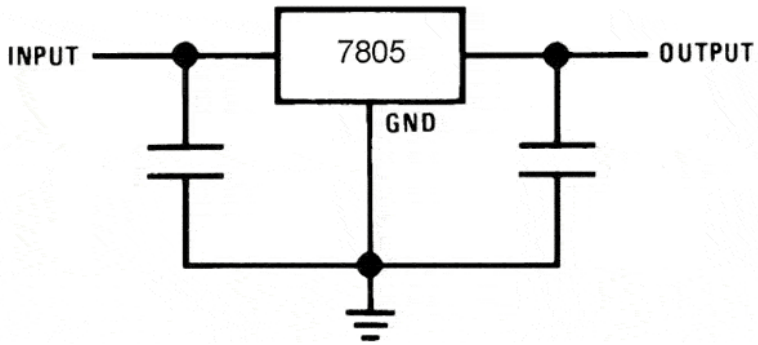
- Operating Voltages
  - 2.7V - 5.5V (ATmega8L)
  - 4.5V - 5.5V (ATmega8)
- Speed Grades
  - 0 - 8MHz (ATmega8L)
  - 0 - 16MHz (ATmega8)
- Power Consumption at 4Mhz, 3V, 25°C
  - Active: 3.6mA
  - Idle Mode: 1.0mA
  - Power-down Mode: 0.5µA

Atmega8 Datasheet: Page 1.

7805 Regulator

- Features:
  - Maximum input voltage: 35V
  - Maximum current: 1A
- Design Considerations:
  - Minimum input voltage?
  - Capacitors
  - Power dissipation





LM7805 Datasheet: Page 2

Atmega Power Supply Decoupling

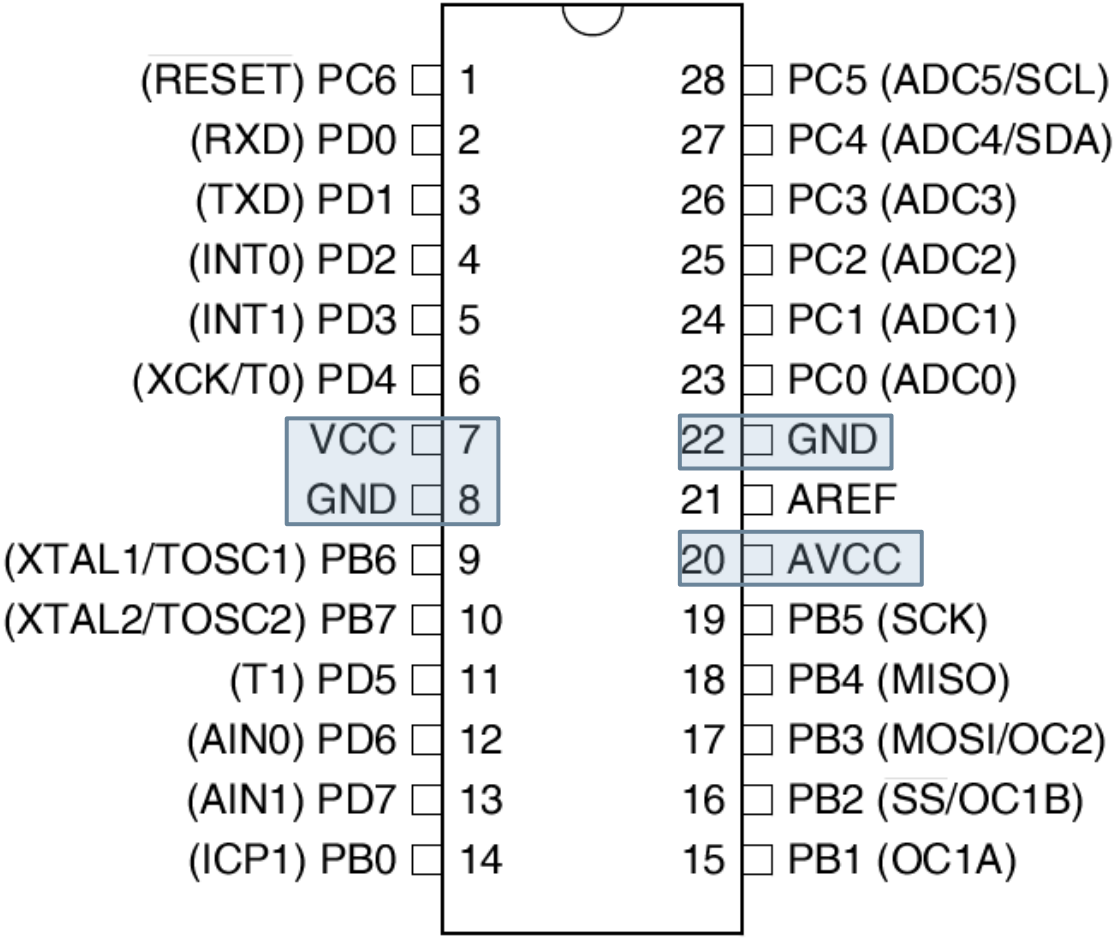
- Requires decoupling capacitor for each supply connection
- Should be placed as close as possible to MCU
- Digital circuits require "fast" capacitors

  → Use ceramic capacitors!

- See AVR042 Application Note for more information.

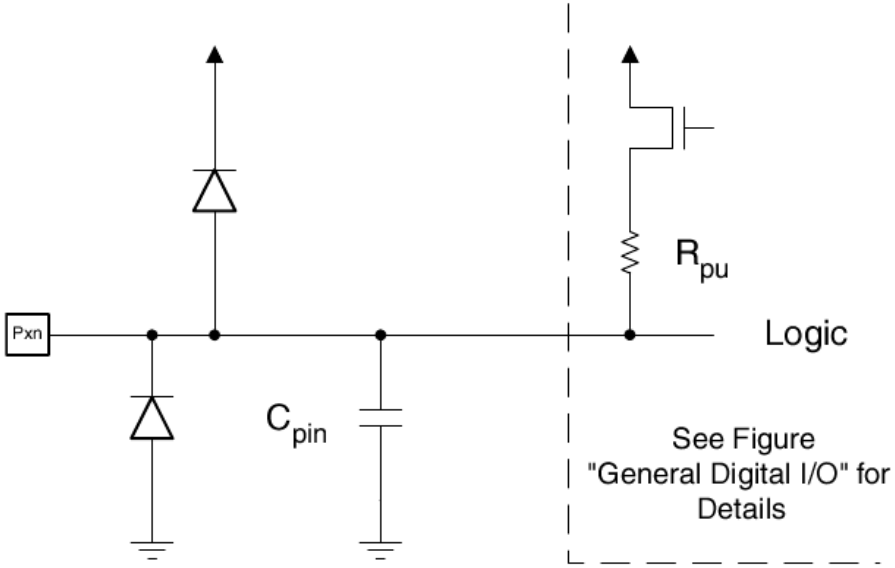
PIN LAYOUT

Power Supply Connections



Atmega8 I/O Pin

- Features:
  - Current handling 20 mA/pin (max 40mA)
  - Internal Pull-up resistor
  - Protection Diodes



Atmega8 Datasheet Figure 21

Reading Datasheet Parameters

DC Characteristics

T_A = -40°C to +85°C, V_{CC} = 2.7V to 5.5V (unless otherwise noted)

Symbol	Parameter	Condition	Min	Typ	Max	Units
	Input Low Voltage except					

- Define the function of the pin (in/out)
- Find a corresponding description (Parameter)
- Check the testing conditions (Condition)
  - Range (2.7-5.5V)
  - Fixed value (s) (5V, 20mA)
- Get the value(s) (Min-Typ-Max Units)

Electrical Characteristics

Note: Typical values contained in this datasheet are based on simulations and characterization of other AVR microcontrollers manufactured on the same process technology. Min and Max values will be available after the device is characterized.

Absolute Maximum Ratings*

Operating Temperature ..... -55°C to +125°C

Storage Temperature ..... -65°C to +150°C

Voltage on any Pin except $\overline{\text{RESET}}$ with respect to Ground .....-0.5V to V_{CC}+0.5V

Voltage on $\overline{\text{RESET}}$ with respect to Ground.....-0.5V to +13.0V

Maximum Operating Voltage 6.0V

DC Current per I/O Pin 40.0mA

DC Current V_{CC} and GND Pins 300.0mA

*NOTICE: Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Atmega8 Datasheet p.235

DC Characteristics

T_A = -40°C to +85°C, V_{CC} = 2.7V to 5.5V (unless otherwise noted)

Symbol	Parameter	Condition	Min	Typ	Max	Units
V_{IL}	Input Low Voltage except XTAL1 and $\overline{\text{RESET}}$ pins	V_{CC} = 2.7V - 5.5V	-0.5		0.2 V_{CC} (1)	V
V_{IH}	Input High Voltage except XTAL1 and $\overline{\text{RESET}}$ pins	V_{CC} = 2.7V - 5.5V	0.6V_{CC} (2)		V_{CC} + 0.5	
V_{IL1}	Input Low Voltage XTAL1 pin	V_{CC} = 2.7V - 5.5V	-0.5		0.1V_{CC} (1)	
V_{IH1}	Input High Voltage XTAL 1 pin	V_{CC} = 2.7V - 5.5V	0.8V_{CC} (2)		V_{CC} + 0.5	
V_{IL2}	Input Low Voltage $\overline{\text{RESET}}$ pin	V_{CC} = 2.7V - 5.5V	-0.5		0.2 V_{CC}	
	Input High Voltage					

DC Characteristics

T_A = -40°C to +85°C, V_{CC} = 2.7V to 5.5V (unless otherwise noted)

Symbol	Parameter	Condition	Min	Typ	Max	Units
V_{IL}	Input Low Voltage except XTAL1 and $\overline{RESET}$ pins	V_{CC} = 2.7V - 5.5V	-0.5		0.2 V_{CC} (1)	V
V_{IH}	Input High Voltage except XTAL1 and $\overline{RESET}$ pins	V_{CC} = 2.7V - 5.5V	0.6V_{CC} (2)		V_{CC} + 0.5	
V_{IL1}	Input Low Voltage XTAL1 pin	V_{CC} = 2.7V - 5.5V	-0.5		0.1V_{CC} (1)	
V_{IH1}	Input High Voltage XTAL 1 pin	V_{CC} = 2.7V - 5.5V	0.8V_{CC} (2)		V_{CC} + 0.5	
V_{IL2}	Input Low Voltage $\overline{RESET}$ pin	V_{CC} = 2.7V - 5.5V	-0.5		0.2 V_{CC}	
V_{IH2}	Input High Voltage $\overline{RESET}$ pin	V_{CC} = 2.7V - 5.5V	0.9V_{CC} (2)		V_{CC} + 0.5	
V_{IL3}	Input Low Voltage $\overline{RESET}$ pin as I/O	V_{CC} = 2.7V - 5.5V	-0.5		0.2V_{CC}	
V_{IH3}	Input High Voltage $\overline{RESET}$ pin as I/O	V_{CC} = 2.7V - 5.5V	0.6V_{CC} (2) 0.7V_{CC} (2)		V_{CC} + 0.5	
V_{OL}	Output Low Voltage (3) (Ports B,C,D)	I_{OL} = 20mA, V_{CC} = 5V I_{OL} = 10mA, V_{CC} = 3V			0.9 0.6	
V_{OH}	Output High Voltage (4) (Ports B,C,D)	I_{OH} = -20mA, V_{CC} = 5V I_{OH} = -10mA, V_{CC} = 3V	4.2 2.2			
I_{IL}	Input Leakage Current I/O Pin	Vcc = 5.5V, pin low (absolute value)			1	µA
I_{IH}	Input Leakage Current I/O Pin	Vcc = 5.5V, pin high (absolute value)			1	
R_{RST}	Reset Pull-up Resistor		30		80	kΩ

T_A = -40°C to +85°C, V_{CC} = 2.7V to 5.5V (unless otherwise noted) (Continued)

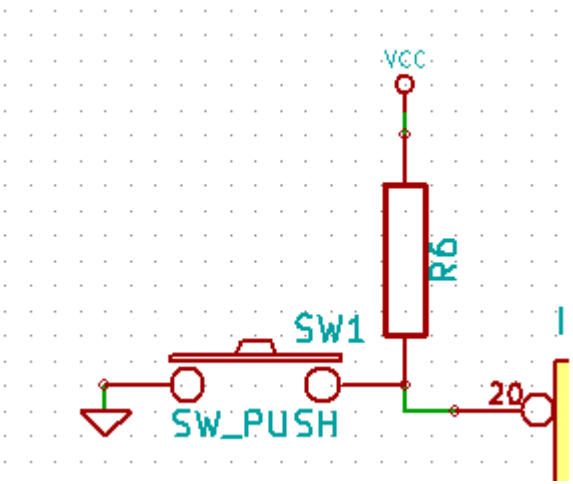
Symbol	Parameter	Condition	Min	Typ	Max	Units
R_{pu}	I/O Pin Pull-up Resistor		20		50	kΩ
I_{CC}	Power Supply Current	Active 4MHz, V_{CC} = 3V (ATmega8L)		3	5	mA
		Active 8MHz, V_{CC} = 5V (ATmega8)		11	15	
		Idle 4MHz, V_{CC} = 3V (ATmega8L)		1	2	
		Idle 8MHz, V_{CC} = 5V (ATmega8)		4.5	7	
	Power-down mode(5)	WDT enabled, V_{CC} = 3V		< 22	28	µA
		WDT disabled, V_{CC} = 3V		< 1	3	
V_{ACIO}	Analog Comparator Input Offset Voltage	V_{CC} = 5V V_{in} = $V_{CC}/2$			40	mV
I_{ACLK}	Analog Comparator Input Leakage Current	V_{CC} = 5V V_{in} = $V_{CC}/2$	-50		50	nA
t_{ACPD}	Analog Comparator Propagation Delay	V_{CC} = 2.7V V_{CC} = 5.0V		750 500		ns

Notes:
1. "Max" means the highest value where the pin is guaranteed to be read as low
2. "Min" means the lowest value where the pin is guaranteed to be read as high
3. Although each I/O port can sink more than the test conditions (20mA at Vcc = 5V, 10mA at Vcc = 3V) under steady state conditions (non-transient), the following must be observed:
   PDIP, TQFP, and QFN/MLF Package:
   1] The sum of all IOL, for all ports, should not exceed 300mA.
   2] The sum of all IOL, for ports C0 - C5 should not exceed 100mA.
   3] The sum of all IOL, for ports B0 - B7, C6, D0 - D7 and XTAL2, should not exceed 200mA.
   If IOL exceeds the test condition, VOL may exceed the related specification. Pins are not guaranteed to sink current greater than the listed test condition
4. Although each I/O port can source more than the test conditions (20mA at Vcc = 5V, 10mA at Vcc = 3V) under steady state conditions (non-transient), the following must be observed:
   PDIP, TQFP, and QFN/MLF Package:
   1] The sum of all IOH, for all ports, should not exceed 300mA.
   2] The sum of all IOH, for port C0 - C5, should not exceed 100mA.
   3] The sum of all IOH, for ports B0 - B7, C6, D0 - D7 and XTAL2, should not exceed 200mA.
   If IOH exceeds the test condition, VOH may exceed the related specification. Pins are not guaranteed to source current greater than the listed test condition
5. Minimum V_{CC} for Power-down is 2.5V

Atmega8 Datasheet p.236

Interfacing with a Push Button

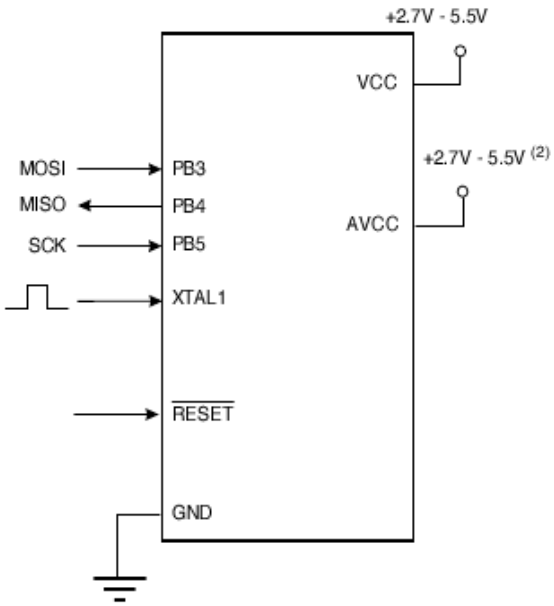
- Input pin requires defined logic levels
  - → See Electrical Characteristics in Atmega8 Datasheet
- Design Considerations
  - Choosing resistor value
  - Alternative: using internal pull-up



Interfacing with an LED

- Small LED can connect directly to I/O pin using a series resistor.
- Choose appropriate series resistor value based on
  - LED voltage drop
  - LED current
- Verify Operation for
  - Worst case scenario
  - Do not exceed maximum ratings of pin, LED or resistor.

ISP Programming



Atmega8 Datasheet p.230

Notes:
1. If the device is clocked by the Internal Oscillator, it is no need to connect a clock source to the XTAL1 pin
2. $V_{CC} - 0.3 < AV_{CC} < V_{CC} + 0.3$, however, AV_{CC} should always be within 2.7V - 5.5V

When programming the EEPROM, an auto-erase cycle is built into the self-timed programming operation (in the Serial mode ONLY) and there is no need to first execute the Chip Erase instruction. The Chip Erase operation turns the content of every memory location in both the Program and EEPROM arrays into 0xFF.

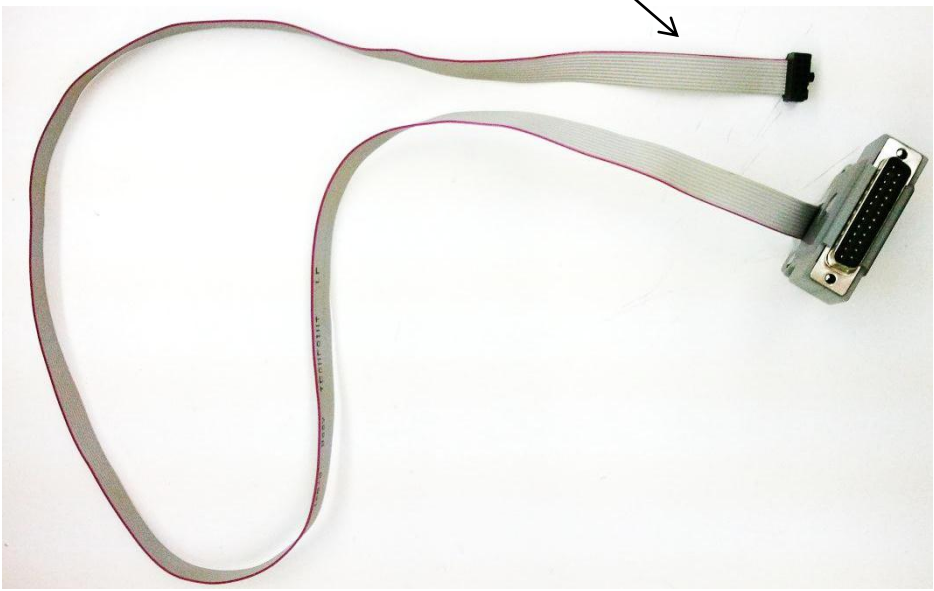
Depending on CKSEL Fuses, a valid clock must be present. The minimum low and high periods for the Serial Clock (SCK) input are defined as follows:

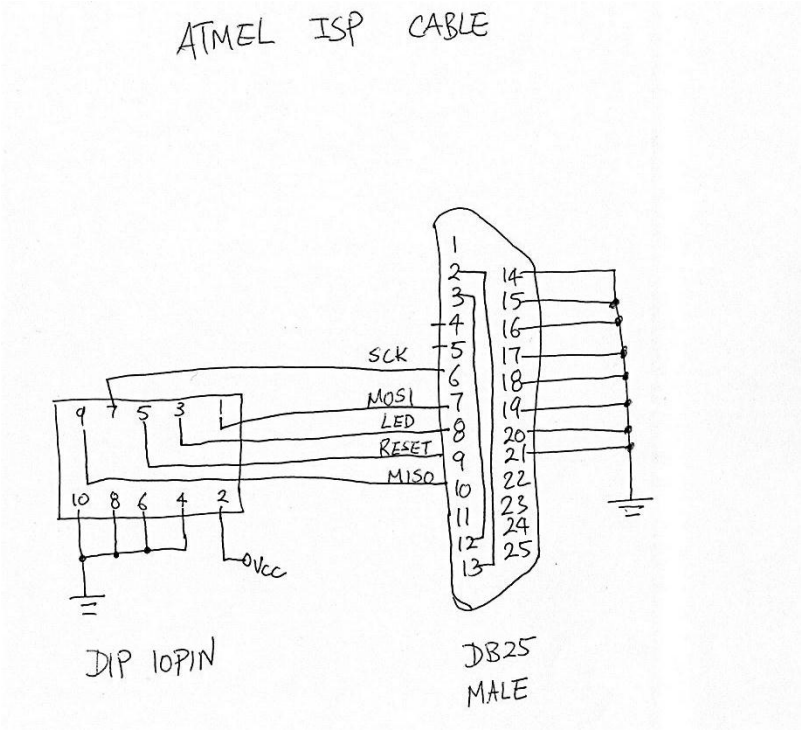
Low:> 2 CPU clock cycles for f_{ck} <12MHz, 3 CPU clock cycles for f_{ck} >=12MHz

High:> 2 CPU clock cycles for f_{ck} <12MHz, 3 CPU clock cycles for f_{ck} >=12MHz

ISP Programming Cable

Red wire denotes pin 1



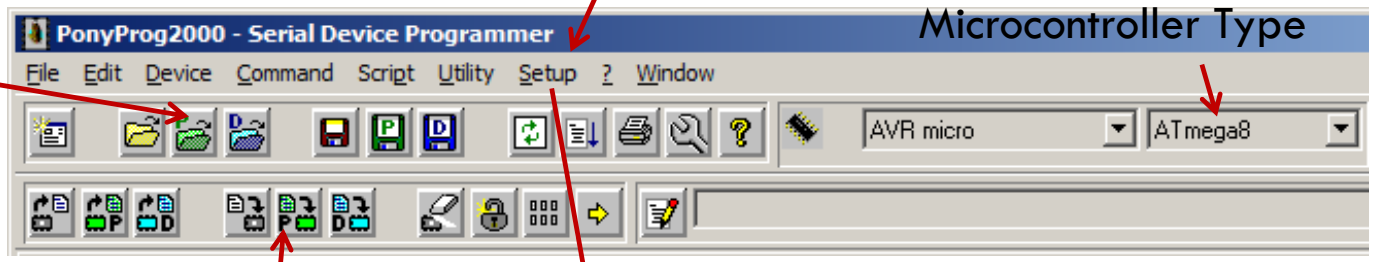


PonyProg 2000

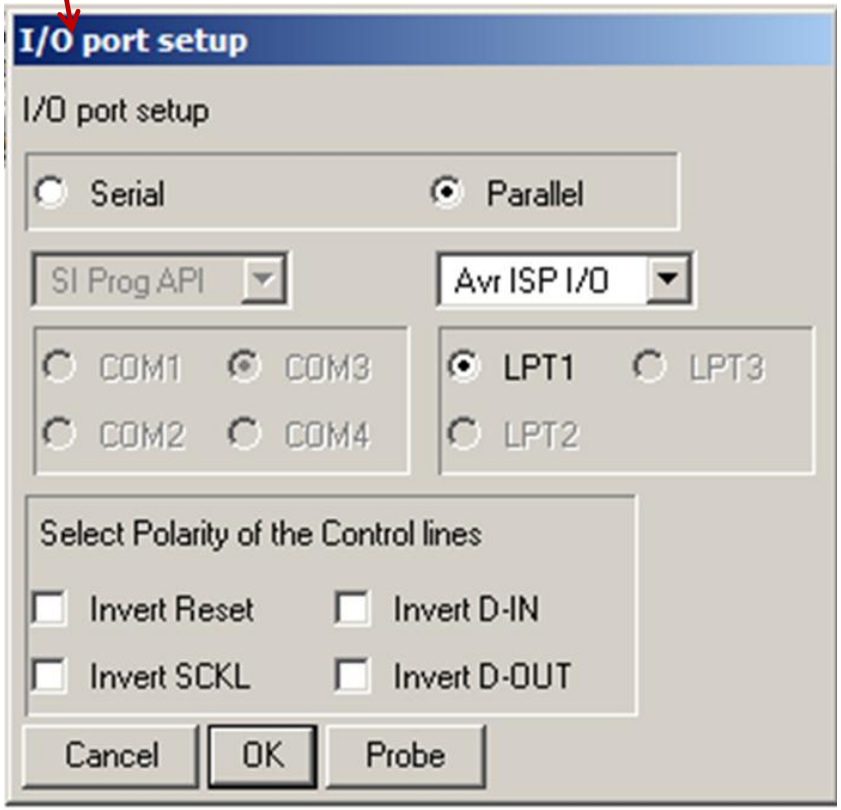
Setup for the ISP cable

Open hex file

Microcontroller Type



Program hex file



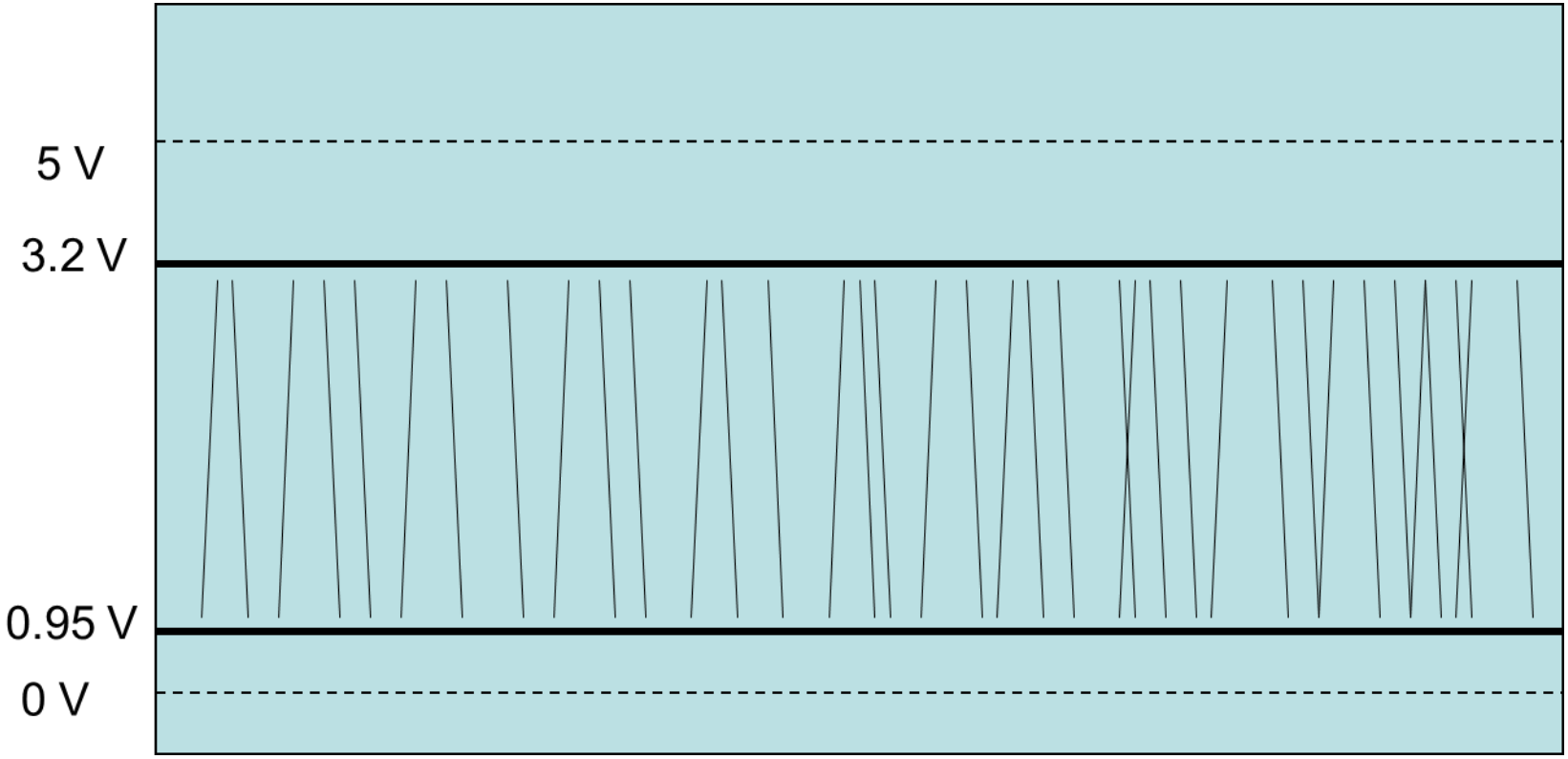
1- Setup the I/O PonyProg
2- Run calibration if requested
3- Select Microcontroller type in main window
4- Open your compiled HEX file
5- Program your file

Troubleshooting Steps

- Learn to use the instruments in the lab, especially the <u>Oscilloscope</u>
- Probe MCU pins directly
- Verify
  - Power Supply voltages
  - Signal levels
  - Signal data

Oscilloscope Example

- Verify that the signals are within bounds according to the datasheet. Vcc = 5V
- Reset Pin, SCK Pin, MISO Pin, MOSI Pin (Hint: See Datasheet p.230)



INTRODUCTION TO 8-BIT AVR (ATMEGA8)

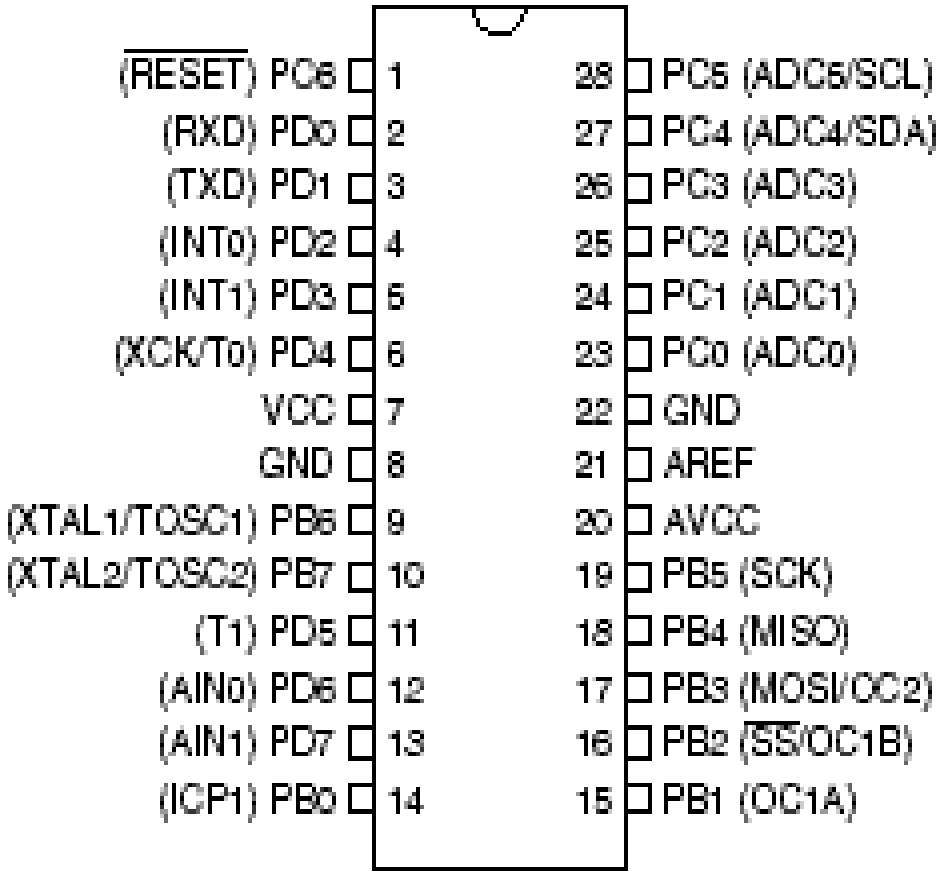
Week 1

OVERVIEW

- Pin Layout
- R/W: PORT
- R/W: PIN
- Interrupts
- External Interrupts
- Programming Environment
- References

PIN LAYOUT

- Pin name header file



R/W: PORT (1/2)

- Before read from/write to a PORT need to set the port direction (DDRx)

- Read = Input, Write = Output
- Direction = 0: Input; 1: Output

R/W: PORT (2/2)

- Example – READ from port B:
  - int myPort;

    DDRB = 0x00; // set port B as input – 0b00000000

    myPort = PINB; // read contents of port B

- Example – WRITE to port B;
  - DDRB = 0xff; // set port B as output – 0x11111111

    PORTB = 0b1111111; // make port B high

R/W: PIN (1/3)

- Bit Operators
  - | bit OR
  - & bit AND
  - ~ bit NOT
  - ^ bit EXLUSIVE OR (XOR)
  - << bit LEFT SHIFT
  - >> bit RIGHT SHIFT

R/W: PIN (2/3)

- Ex – Write high to pin B1:
  - PORTB = (1 << PB1); OR
  - PORTB |= (1 << PB1); OR
  - PORTB |= _BV(PB1);
- Ex – Write low to pin B1:
  - PORTB &= ~(1 << PB1); OR
  - PORTB &= ~_BV(PB1);
- Ex – Write to multiple pins in port B:
  - PORTB |= _BV(PB1) | (1 << PB3); // set PB1 & PB3 to high

R/W: PIN (3/3)

- Ex – Toggle pin B1:
  - PORTB ^= (1<<PB1);
- Ex – Test if bit is clear:
  - bit_is_clear(PINB, PB1); // returns F if set, T if clear
- Ex – Test if bit is set:
  - bit_is_set(PINB, PB1); // returns F if clear, T if set

INTERRUPTS (1/2)

- Status Register – SREG (pg 11-2 in ATmega8 ds)
  - I-bit: Global interrupt enable – this bit must be set in order for any interrupt to function
  - sei(); // used to set I-bit in SREG
  - cei(); // used to clear I-bit in SREG
- Interrupt Vectors (pg 46-9 in ATmega8 ds)
  - See Table 18 for list of vector names

INTERRUPTS (2/2)

- Ex - Interrupt Vector INT0
  - ISR(INT0_vect)

```
{
    // TODO: keep interrupt code short!
}
```

- Ex - Interrupt Vector TIMER0_OVF
  - ISR(TIMER0_OVF_vect)

```
{
    // TODO: keep interrupt code short!
}
```

EXTERNAL INTERRUPTS (1/2)

- External Interrupt Checklist:
  - Configure interrupt sense control in MCUCR
  - Enable external interrupt request in GICR
  - Implement code to execute once interrupt triggered
  - Enable all interrupts in SREG

EXTERNAL INTERRUPTS (2/2)

- MCUCR (pg 66-7 ATmega8 Datasheet)
  - Set interrupt sense control to indicate when interrupt triggers
- GICR (pg 67 ATmega8 Datasheet)
  - Set external interrupt request to enable the interrupt
- GIFR (pg 67-8 ATmega8 Datasheet)
  - The external interrupt flag is set when pin is triggered
  - MCU jumps to Interrupt Vector if I-bit in SREG && INTx bit in GICR is set
  - Flag is cleared when interrupt routine is executed

REFERENCES

- R/W: PORT
  - http://elecrom.wordpress.com/2008/02/12/avr-tutorial-2-avr-input-output/
- R/W: PIN
  - http://www.avrfreaks.net/index.php?name=PNphpBB2&file=viewtopic&t=37871

INTRODUCTION TO 8-BIT AVR (ATMEGA8)

Week 2

OVERVIEW

- Timers!
- System Clock
- Timer clock prescaler
- Calculating timer count values
- Timer in Normal Mode / Overflow interrupt
- Timer in CTC Mode / Compare match interrupt
- Timer in CTC Mode / Compare match output
- Datasheet Navigation
- References

Timers

- What is a timer?
  - Hardware counter that increments at periodic intervals
- Can be programmed to
  - Raise interrupts on certain value
  - Change state of output pins
  - Time-stamp external input
- ATMega8 has 3 Timers
  - One 16 bit (max count = 2^{16}-1 = 65535)
  - Two 8 bit (max count = 255)

- Examples are based on timer1, operation and register names are similar for other timers.

TIMER Checklist

- Timer Checklist
  - Set System Clock frequency
  - Set Prescaler
  - Set Overflow/Compare Interrupt
  - Set Timer Counter
  - Turn on global interrupt enable

Setting System Clock Frequency

- Controlled via Fuse Bits
  - **WARNING:** Invalid fuse bit settings can **make it impossible to reprogram ATmega** unless using an HV programmer!!!
  - Changed using PonyProg 2000 or other programmer
  - Default clock is 1 MHz

Table 9. Internal Calibrated RC Oscillator Operating Modes

CKSEL3..0	Nominal Frequency (MHz)
0001[1]	1.0
0010	2.0
0011	4.0
0100	8.0

Note: 1. The device is shipped with this option selected

Atmega8 Datasheet p.30

Timer Prescaler

- Divides System clock
- Allows longer delays at the expense of resolution
- /8, /32, /64, /128, /256, /1024
- Each timer can have a different pre-scaler selected
- Controlled by Timer/Counter Control Register of each timer ex. TCCR1B for timer 1

Calculating Timer Values

- Timer Equations
  - $$Resolution\ (seconds) = \frac{Prescaler\ Setting}{System\ Clock\ Frequency}$$
    - Resolution is the smallest delay measured by the timer (the delay of one count)
  - $$Target\ count = \left[\frac{\frac{System\ Clock\ Frequency}{Prescaler}}{Target\ Frequency}\right] - 1$$
    - Note: If toggling a pin, target frequency should be doubled in above equation because you need to toggle twice to make one cycle.

Setting up a timer

- Basic Setup:
  - Select clock source in TCCR1B

Table 40. Clock Select Bit Description

CS12	CS11	CS10	Description
0	0	0	No clock source. (Timer/Counter stopped)
0	0	1	$clk_{I/O}/1$ (No prescaling)
0	1	0	$clk_{I/O}/8$ (From prescaler)
0	1	1	$clk_{I/O}/64$ (From prescaler)
1	0	0	$clk_{I/O}/256$ (From prescaler)
1	0	1	$clk_{I/O}/1024$ (From prescaler)
1	1	0	External clock source on T1 pin. Clock on falling edge
1	1	1	External clock source on T1 pin. Clock on rising edge

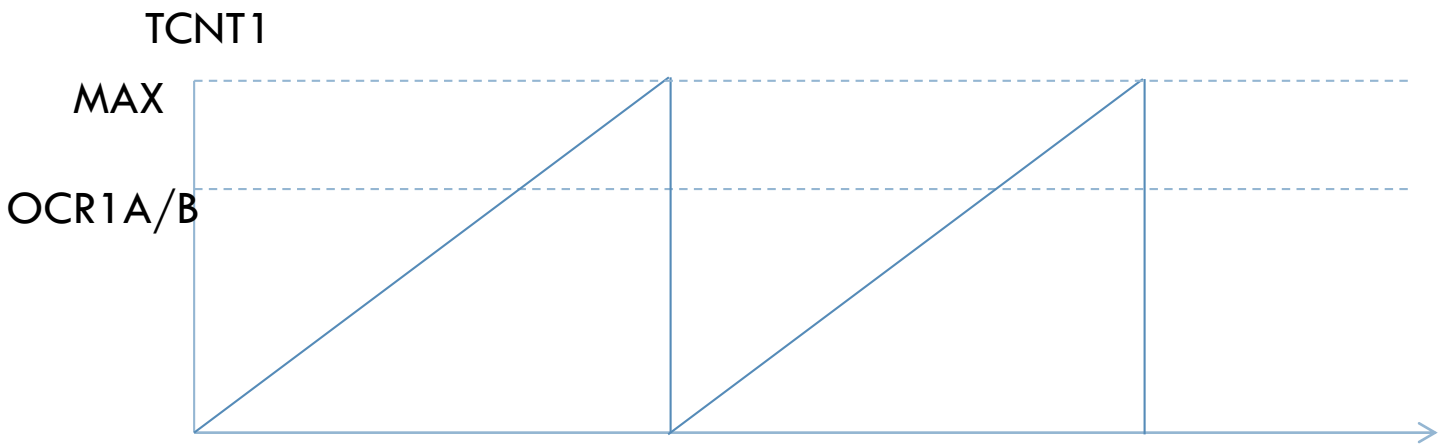
If external pin modes are used for the Timer/Counter1, transitions on the T1 pin will clock the counter even if the pin is configured as an output. This feature allows software control of the counting.

Atmega8 Datasheet p.99

  - Count is stored in TCNT1and starts as soon as clock source is configured

Timer – Normal Mode

- Timer overflow occurs when count exceeds max value.
  - Counter will simply restart counting from zero
- Interrupt can be triggered:
  - On overflow
  - On compare-match with OCR1A/B



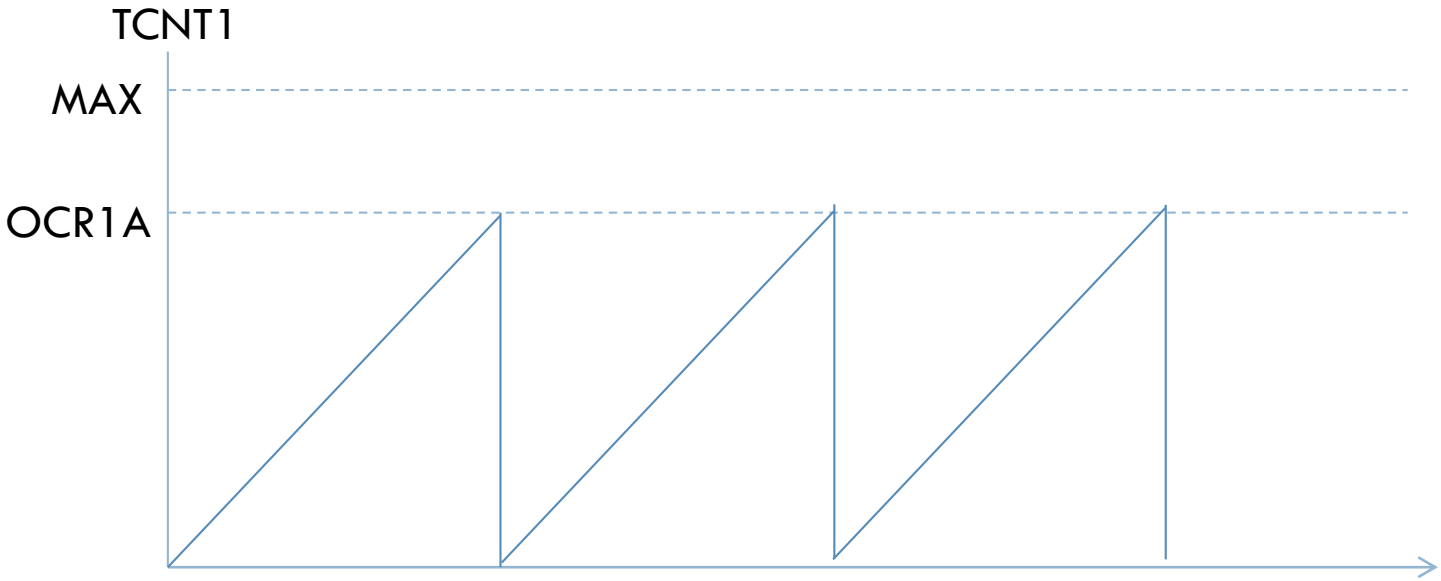
Using Timer – Normal Mode

- Setup Timer clock
  - TCCR1B |= (1 << CS10) ***See Table 40 in datasheet
- Turn on Overflow interrupt
  - TIMSK |= (1 << TOIE1)
- Setup ISR routine
  - ISR Vector is TIMER1_OVF_vect

- Note: Starting value other than zero can be assigned manually at start of program and inside of the ISR by overwriting current count TCNT1

Timer - CTC mode

- CTC - Clear on Timer Compare
- Used to trigger interrupt when specific counter value is reached in OCR1A

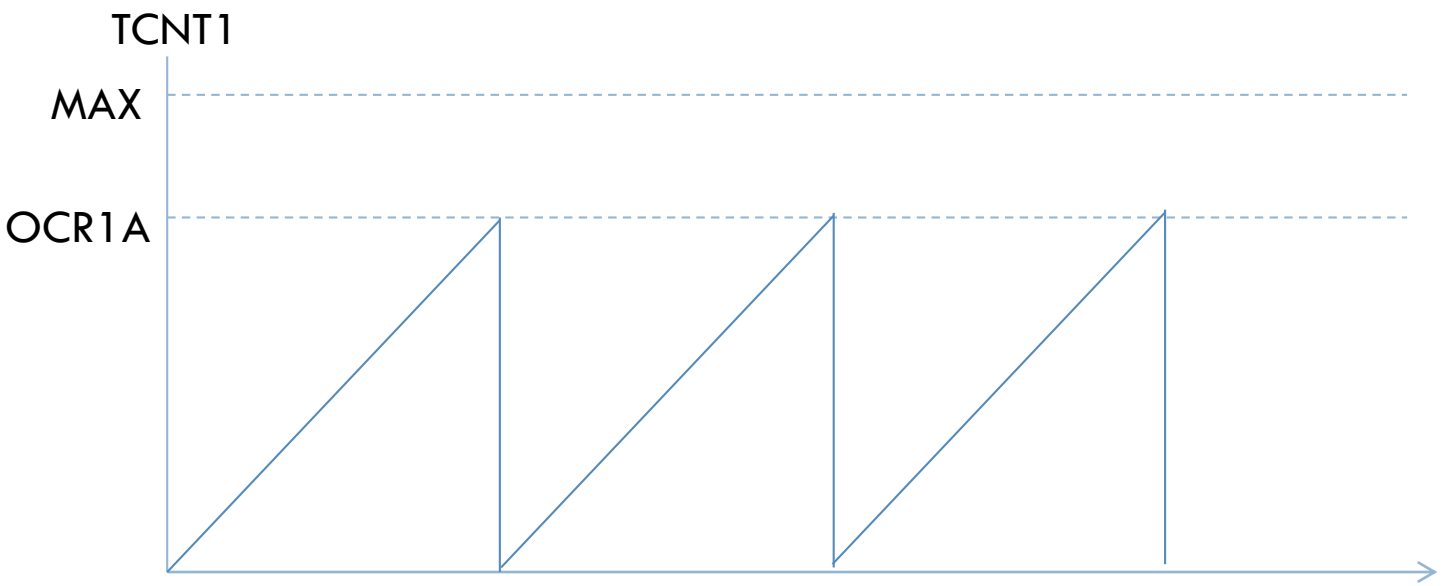


Using Timer – CTC Interrupt

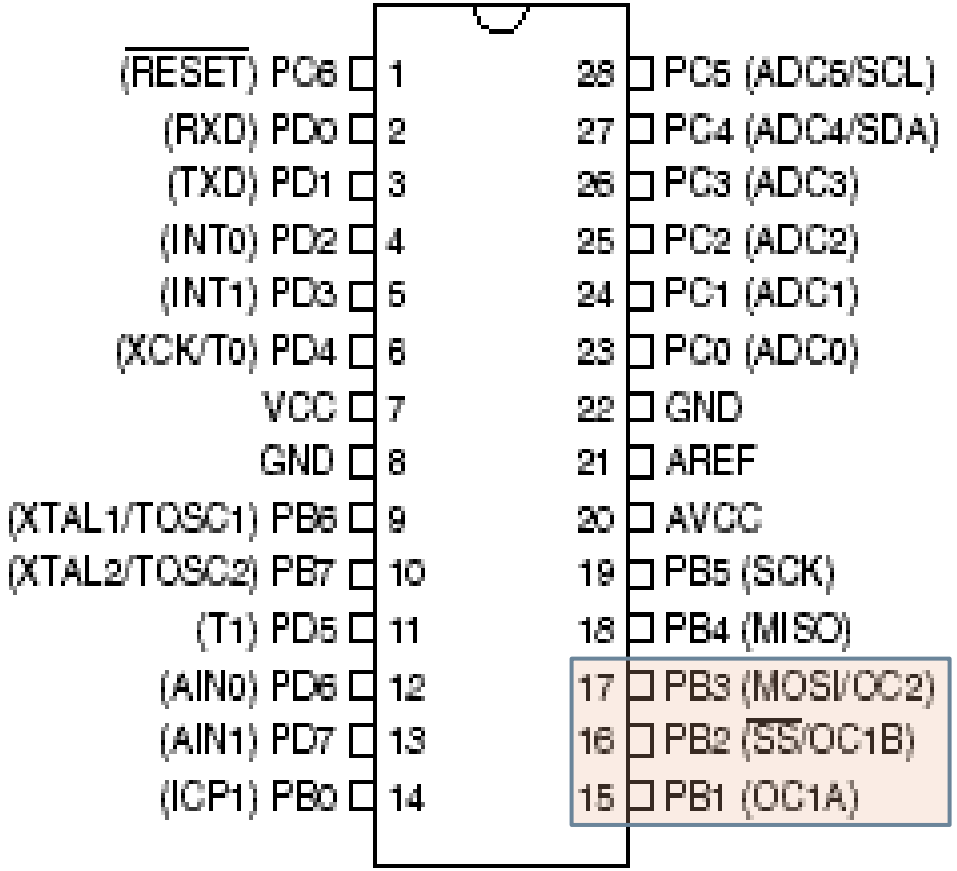
- Configure timer for CTC Mode
  - TCCR1B |= (1 << WGM12)
- Enable compare interrupt with value in OCR1A
  - TIMSK |= (1 << OCIE1A)
- Set CTC compare value
  - OCR1A = 15000
- Setup Timer clock
- Setup ISR
  - ISR Vector is TIMER1_COMPA_vect

Timer - CTC mode with Output Compare

- Same as previous mode but directly controls pins on ATmega instead of raising an interrupt.
- Output Compare mode allows timer1 to directly control pins OC1A/B and Timer2 controls OC2



Timer – CTC mode with Output Compare



- On compare match, choice of:
  - Toggle pin
  - Set pin High
  - Set pin Low
- See table 36 in ATmega8 datasheet.

Using Timer – CTC with Output Compare

- Setup Pin as output
- Configure timer for CTC Mode
  - TCCR1B |= (1 << WGM12)
- Enable Output compare **See table 36 in ATmega8 datasheet
  - TCCR1A |= (1 << COM1A0);
- Set CTC compare value
  - OCR1A = 15000
- Setup Timer clock

Datasheet Navigation

- Timer/Counter1 Control Register B **TCCR1B** (pg 98-9)
  - CS12, CS11, CS10 used to select clock prescaler
- Timer/Counter1 TCNT1H & TCNT1L
- Timer/Counter1 Interrupt Mask Register **TIMSK** (pg 100)
  - OCIE1A OR OCIE1B → set to enable interrupt

REFERENCES (1/1)

- Timers
  - http://deans-avr-tutorials.googlecode.com/svn/trunk/Timers/Output/Timers.pdf
  - http://www.avrfreaks.net/index.php?name=PNphpBB2&file=viewtopic&t=68302&start=all&postdays=0&postorder=asc
  - http://extremeelectronics.co.in/avr-tutorials/avr-timers-an-introduction/
  - http://extremeelectronics.co.in/avr-tutorials/pwm-signal-generation-by-using-avr-timers-part-ii/

INTRODUCTION TO SENSORS

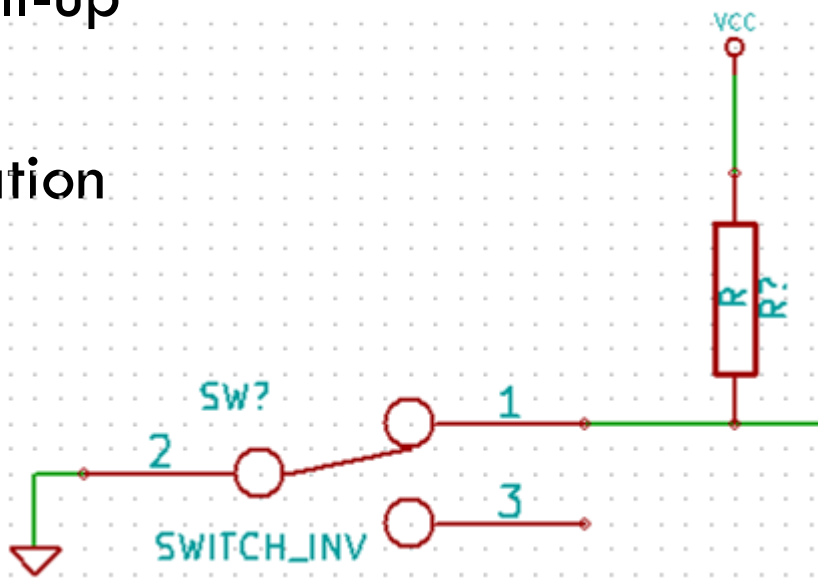
Week 3

OVERVIEW

- Sensors:
  - Contact Switch
  - Ultrasonic distance sensor
  - Infrared distance sensor
  - Reflective object sensor
  - Analog Voltage Reference
- Analog-Digital Conversion
- Universal Synchronous and Asynchronous serial Receiver and Transmitter

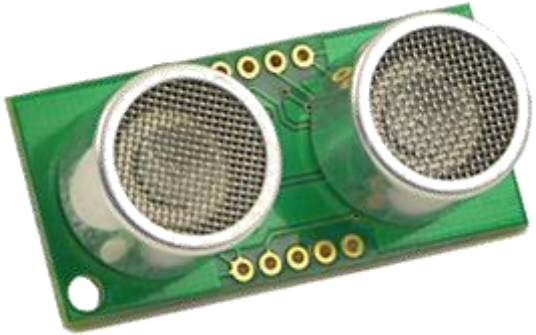
Interfacing with a contact switch

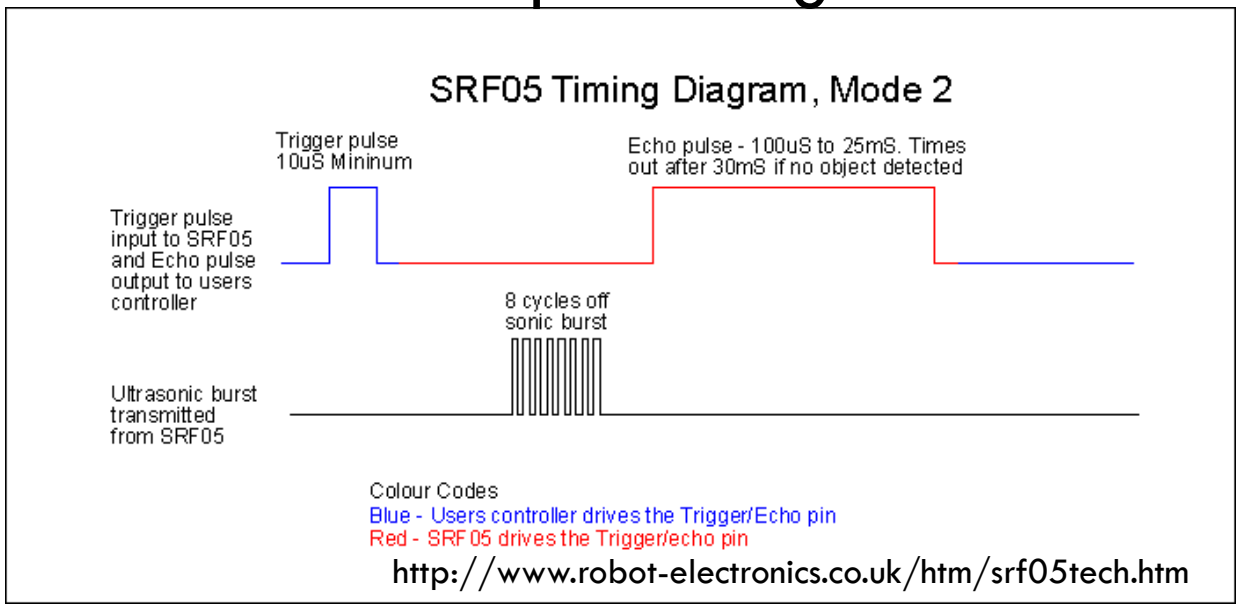
- Input pin requires defined logic levels
  - → Identical to push button setup
- Design Considerations
  - Choosing resistor value
  - Alternative: using internal pull-up
  - Normally Open (NO) or
    Normally Closed (NC) operation



Ultrasonic Distance Sensor

- Example: Devantech SRF05
- Operates by sending an ultrasonic pulse and listening for an echo.
- Output is of the form of a digital pulse, your code measures the width of the echo pulse to get distance.





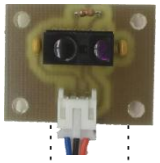
http://www.robot-electronics.co.uk/htm/srf05tech.htm

Sharp Infrared Distance Sensors

- Two types are available:
  - Digital (ex: GP2Y0D810)
  - Analog (ex: GP2D12)

SHARP GP2Y0D810Z0F

Fig. 2 Example of distance measuring characteristics (output)



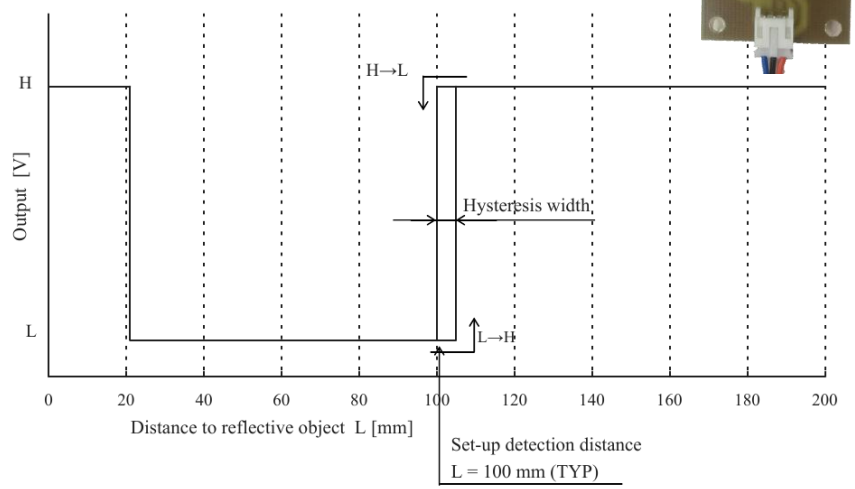
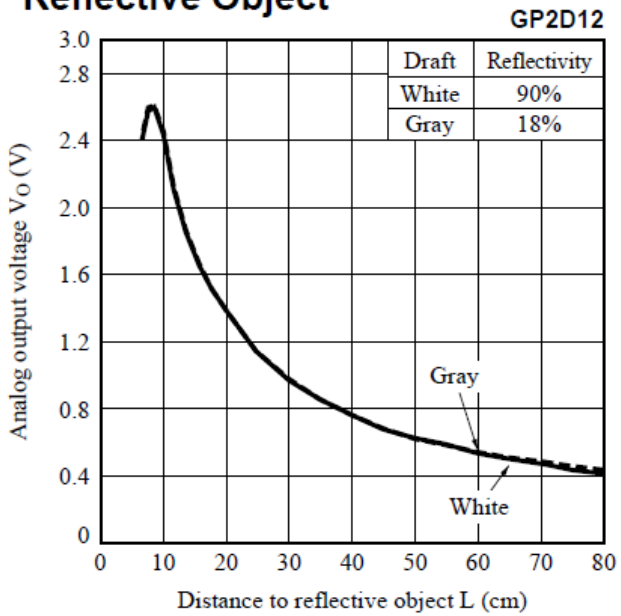


Fig. 2 p. 5 of GP2Y0D810Z0F datasheet

Fig.6 Analog Output Voltage vs.Distance to Reflective Object



Interfacing with GP2Y0D810

- Sensor requires power supply to operate
- Connect Vo to a digital input pin on the Atmega

- Design Considerations:
  - Pull-down resistor may be required
  - Capacitor on power supply line

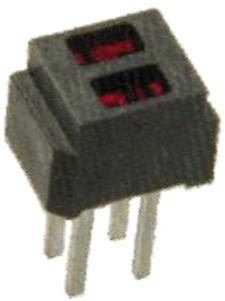
Interfacing with GP2D12

- Sensor requires power supply to operate
- Connect Vo to an ADC pin of the Atmega

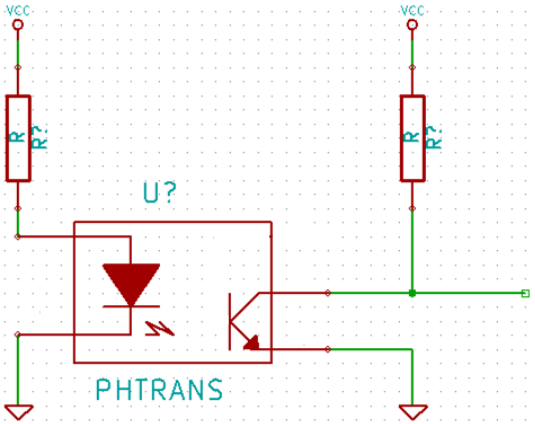
- Design Considerations:
  - Capacitor on power supply line
  - Choose appropriate Aref voltage for this sensor

Reflective Object Sensor

- Example: Optek OPB606A
  - Can be used as a line sensor
  - Analog sensor by nature
  - With proper choice of R_{led} and R_c, it can be directly connected to a digital input.



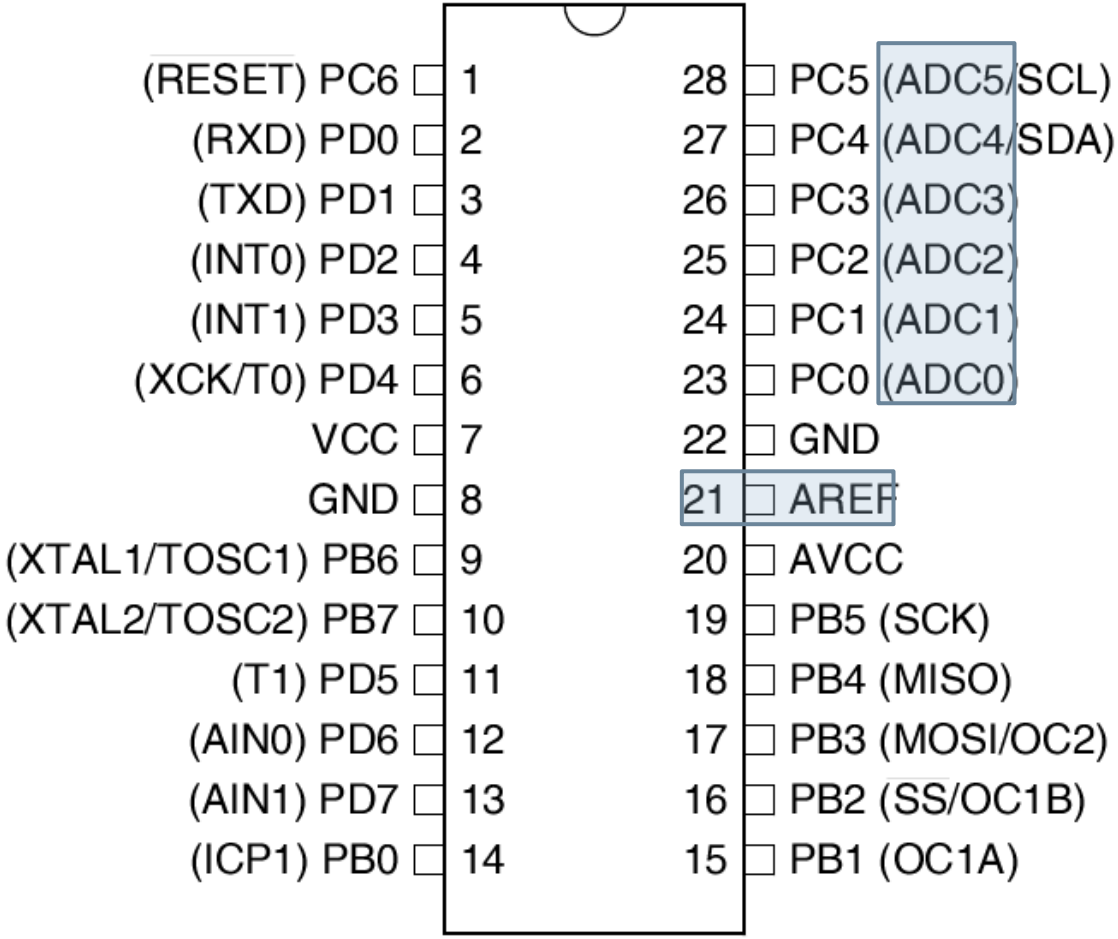
- Design considerations:
  - LED current limiting resistor
  - Collector resistor
  - Surface reflectivity
  - Distance between sensor and surface



PIN LAYOUT

Analog voltages can be measured on pins 23 to 28

If used, an external voltage reference can be connected to Aref



Analog voltage reference

- Analog signals can be read on pins ADC0~ADC5
- Minimum value is GND
- Maximum value that can be read depends on Aref which is selected with ADMUX register:
  - Internally generated: 2.56V or AVcc
  - External
- Design considerations
  - Choose Aref voltage based on maximum voltage you expect to read
  - Internal voltage reference options may not be used when an external voltage is being applied to the Aref pin
  - Add a capacitor to Aref pin for better noise performance

ANALOG-TO-DIGITAL CONVERTER (1/7)

- ADC translates an analog signal to an 8 or 10 bit number that the microcontroller can process
- Needed when interfacing a microcontroller with analog sensors, e.g. GP2D12, OPB606A (optional)
- First conversion takes 25 ADC clock cycles and must be discarded
- Normal conversion takes 13 ADC clock cycles

ANALOG-TO-DIGITAL CONVERTER (2/7)

- ADC Checklist
  - Select Voltage Reference
  - Select ADC Channel
  - Enable ADC
  - Enable ADC Interrupt (if desired)
  - Select Conversion Mode
  - Wait until conversion is complete
  - Read ADC Registers

ANALOG-TO-DIGITAL CONVERTER (3/7)

- Select Voltage Reference: (pg 194, 199)
  - 3 options for ADC voltage reference
    - AREF
    - AVcc
    - Internal 2.56V voltage reference
  - Select by setting REFS1 & REFS0 bits in ADMUX according to Table 74 on pg 199
  - $ADC = \frac{Vin * 1024}{Vref}$

ANALOG-TO-DIGITAL CONVERTER (4/7)

- Select ADC Channel
  - 6 different ADC channels (see Pin Layout)
  - Select specific channel by setting MUX3:0 bits in ADMUX according to Table 75 on pg 199-200

- Enable ADC
  - Set ADEN bit to logical one in ADCSRA
  - ADEN must be set in order for conversion to occur

ANALOG-TO-DIGITAL CONVERTER (5/7)

- Enable ADC Interrupt
  - If set, ADC interrupt triggers when ADC conversion completes
  - Enable ADC Interrupt by setting ADIE to one in ACSRA (note that global interrupts must also be enabled for interrupt to trigger)

ANALOG-TO-DIGITAL CONVERTER (6/7)

- Select Conversion Mode
  - 2 conversion modes
    - Single Conversion – must always set ADSC before conversion
    - Free Running Mode – only set ADSC one before first conversion
- Start Conversion
  - Set ADSC bit in ADCSRA
  - Conversion will start on next ADC clock cycle

ANALOG-TO-DIGITAL CONVERTER (7/7)

- Read ADC Register
  - 10-bit conversion stored in register ADCL and ADCH
  - ADLAR bit in ADMUX register sets how the bits are arranged in the ADC registers
    - ADLAR 0: right adjusted; 1: left adjusted
  - If at most 8-bit precision is required, set bits to left adjusted and only read ADCH

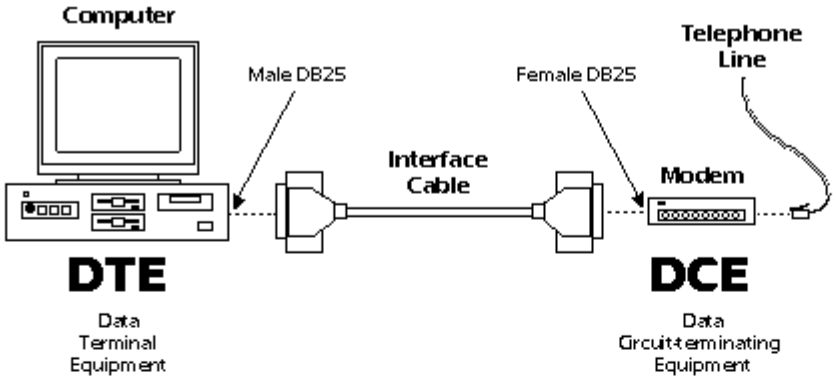
Serial Communications

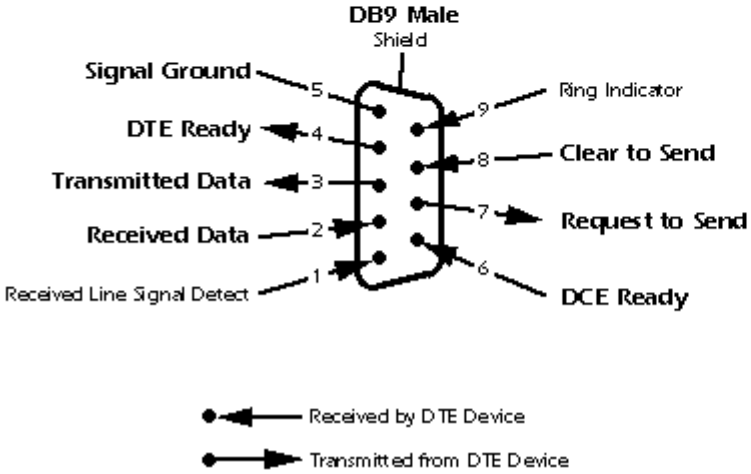
- RS-232-C (Computer Serial Interface)
  - Signal Levels (open circuit): up to ±25V
  - Idles at logic one
  - > +3V = Logic Zero
  - < -3V = Logic One
  - These values are for input, refer to standard for output values
- Serial TTL (Atmega Serial Interface)
  - Signal Levels same as mentioned in datasheet
  - RXD on pin 2, TXD on pin 3

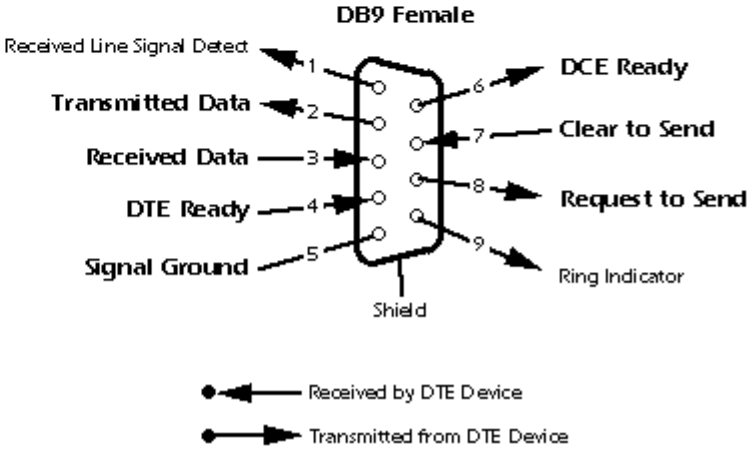
Level Converters

- Simple circuit
  - See “PC (serial port) + transmitter/receiver” on Moodle
- Commercial Chips
  - Example: MAX232

Serial Interface Pinout







http://www.winlab.rutgers.edu/~zhibinwu/html/serial.htm

RS232C Frame

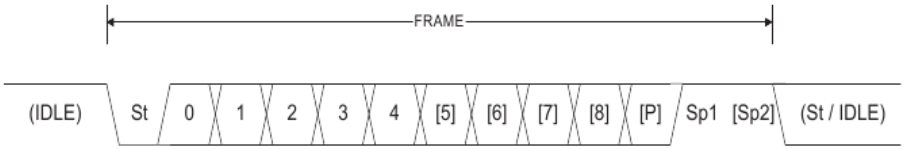
- Many combinations:
  - 1 Start bit

    5 to 9 data bits

    No, even or odd parity

    1 or 2 stop bits

Figure 64. Frame Formats



St	Start bit, always low
(n)	Data bits (0 to 8)
P	Parity bit. Can be odd or even
Sp	Stop bit, always high
IDLE	No transfers on the communication line (RxD or TxD). An IDLE line must be high

The frame format used by the USART is set by the UCSZ2:0, UPM1:0 and USBS bits in UCSRB and UCSRC. The Receiver and Transmitter use the same setting. Note that changing the setting of any of these bits will corrupt all ongoing communication for both the Receiver and Transmitter.

Communicating with Computer

- Need a terminal emulator software
  - Example: Putty, Hyper Terminal
  - Select proper port
    - Example: COM 1
  - Select baud rate and framing
    - Example: 9600-N-1
- Send ASCII characters

USART (1/7)

- Universal Synchronous and Asynchronous serial Receiver and Transmitter – serial communication
- Used to communicate between computers
  - Helpful for collecting data & debugging
- Send ASCII formatted data to ensure compatibility

USART (2/7)

- USART Checklist
  - Set Baud Rate
  - Set Frame Format
  - Select operation mode (Tx/Rx)
  - Enable USART specific interrupt (if desired)
  - Read USART data

USART (3/7)

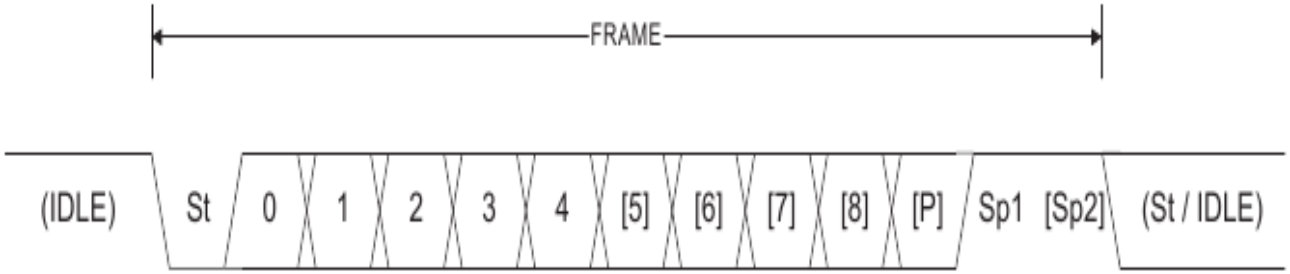
- Set Baud Rate
  - Calculate Baud Rate according to formula
  - Load answer into UBRRH[11:8] and UBRRL[7:0]
  - * Note that to access the UBRRH register you must first set UBSEL to zero (pg 146)

- $BAUD = f_{OSC}/16(UBRR + 1)$
- $UBRR = (f_{OSC}/16*BAUD) - 1$
  - BAUD: baud rate in bits per second (bps)
  - f_{OSC}: system oscillator clock frequency
  - UBRR: contents of the UBRRH and UBRRL registers

USART (4/7)

- Frame Format
  - 1 start bit
  - 5, 6, 7, 8 or 9 data bits
  - No, even or odd parity – used to for error checking (see pg 134 for parity calculation)
  - 1 or 2 stop bits

Figure 64. Frame Formats



USART (5/7)

- Set Frame Format
  - * Note that to access the UCSRC register you must first set UBSEL to one (pg 146)
  - Set character size by setting bit UCSZ2 in UCSRB and bits UCSZ1:0 in UCSRC according to Table 58 (pg 151)
  - Select parity by setting UPM1:0 in UCSRC according to table 59 (pg 151)
  - Select stop bit USBS – 0: 1 bit; 1: 2 bits

USART (6/7)

- Enable Operation Mode
  - 2 modes of operation: Transmitter OR Receiver
  - Set Transmitter by setting TXEN bit to one in UCSRB
  - Set Receiver by setting RXEN bit to one in UCSRB

- Read/Write USART data
  - Read Received data from UDR
  - Load data to UDR to be transmitted

USART (7/7)

- USART Interrupts
  - USART Transmitter has 2 interrupts that can be used
    - Data Register Empty (UDRE): indicates transmit buffer ready to receive new data
    - Transmit Complete (TXC): triggers when entire frame shifted out and no new data present in buffer
  - USART Receiver
    - Receive Complete (RXC): triggers when unread data present in receive buffer
  - Set interrupt enable bits in UCSRB

INTRODUCTION TO PWM & MOTORS

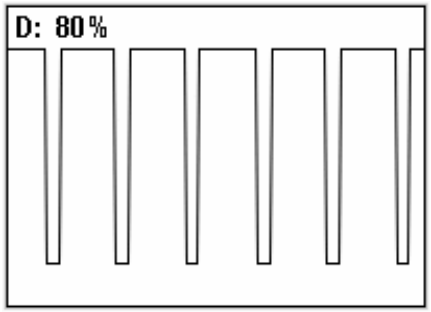
Week 4

OVERVIEW

- Pulse Width Modulation:
  - Fast PWM
- Motors:
  - DC Brush Motor
  - Motor Driver
  - Other Motor Types
- Q&A

What is Pulse-Width Modulation?

- Pulse-Width Modulation (PWM) is a modulation technique that generates a square wave with variable duty-cycle to represent the amplitude of an analog signal.
- Applications include:
  - Generation of analog voltages
  - Dimming LEDs
  - Controlling speed of motors
  - Generation of sound

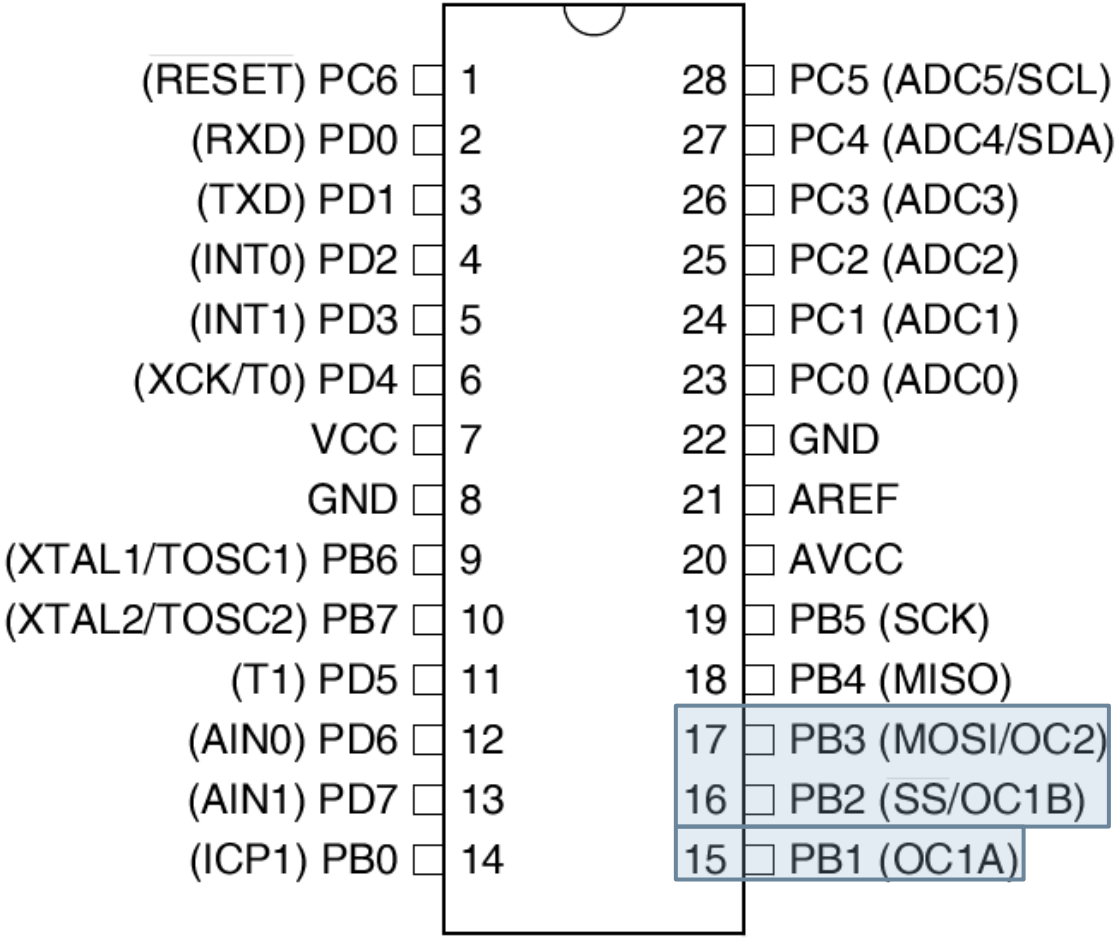


http://en.wikipedia.org/wiki/Duty_cycle

- Duty Cycle $D = \frac{t_{ON}}{t_{ON}+t_{OFF}} = \frac{t_{ON}}{T}$

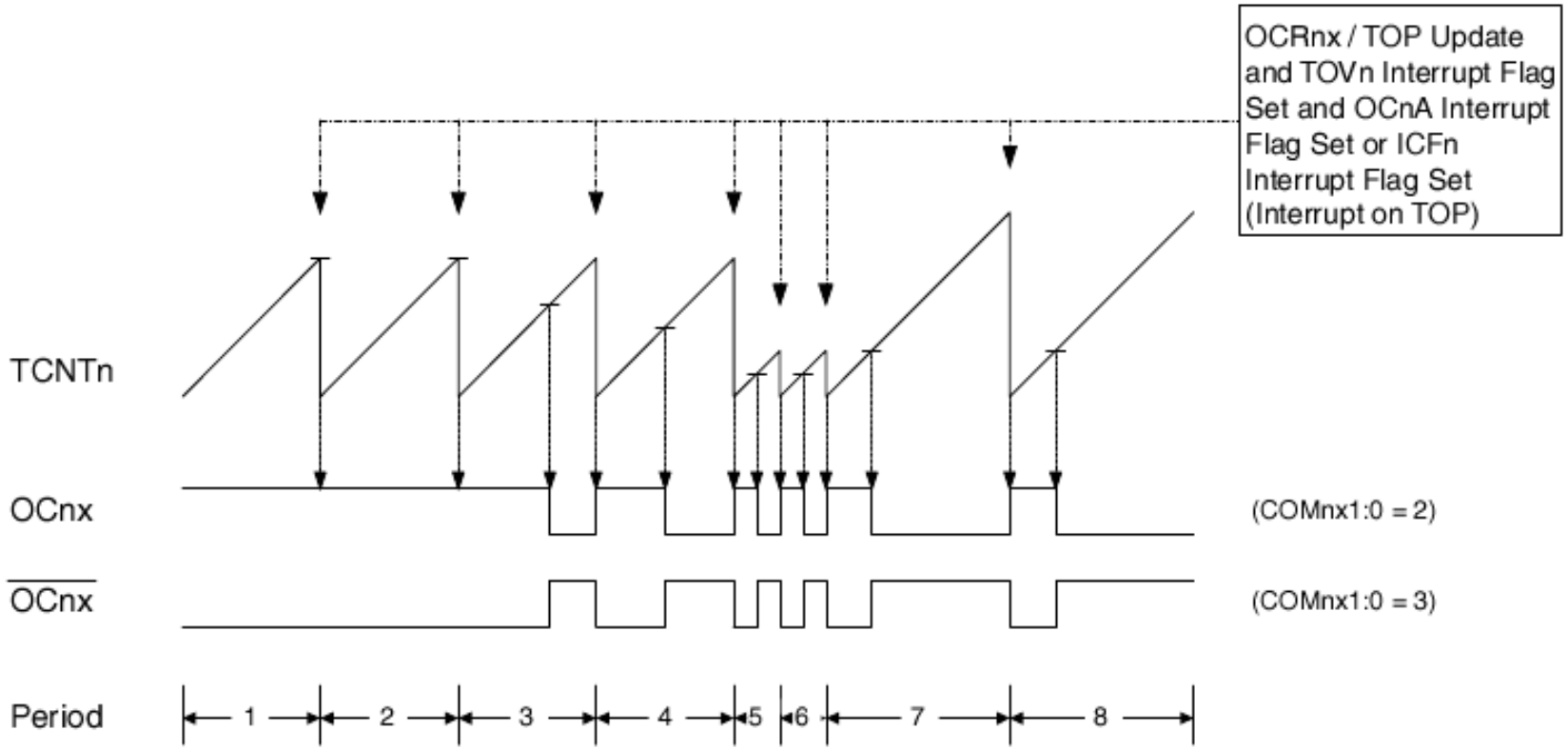
PWM on ATmega8

- Uses Timers
- PWM Output available on pins
  - OC1A & OC1B for timer1
  - OC2 for timer2



Fast PWM Timing Diagram

Figure 38. Fast PWM Mode, Timing Diagram



ATmega8 Datasheet, page 89.

Fast PWM – Checklist

- Timer Checklist
  - Set Compare Output Mode
    - TCCR1A
    - See Table 37 on p.97 of datasheet
  - Set Waveform Generation Mode Bits to Fast PWM
    - TCCR1A & TCCR1B
    - See Table 39 on p.97 of datasheet
  - Set Output Compare Compare Value (OCR1A/B)
  - Set Timer Prescaler
    - See Table 40 on p.99 of datasheet
  - Set Pin as output

Fast PWM

Table 39. Waveform Generation Mode Bit Description

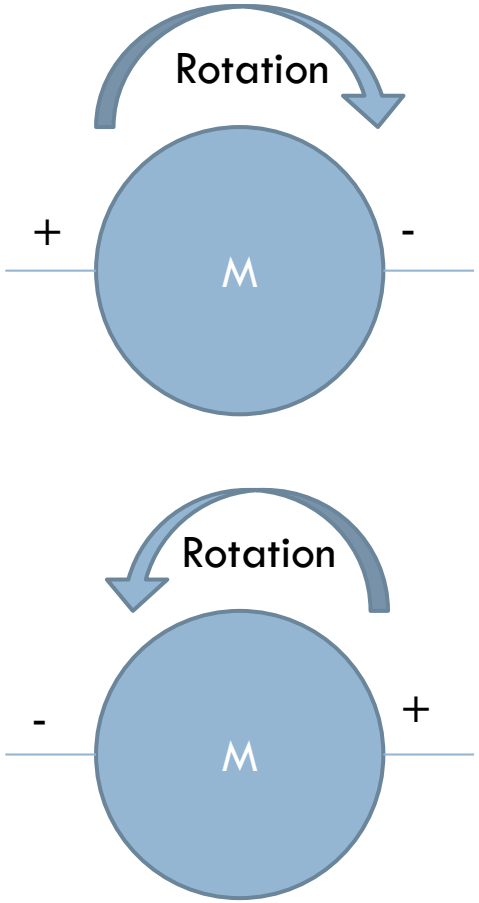
Mode	WGM13	WGM12 (CTC1)	WGM11 (PWM11)	WGM10 (PWM10)	Timer/Counter Mode of Operation[1]	TOP	Update of OCR1x	TOV1 Flag Set on
0	0	0	0	0	Normal	0xFFFF	Immediate	MAX
1	0	0	0	1	PWM, Phase Correct, 8-bit	0x00FF	TOP	BOTTOM
2	0	0	1	0	PWM, Phase Correct, 9-bit	0x01FF	TOP	BOTTOM
3	0	0	1	1	PWM, Phase Correct, 10-bit	0x03FF	TOP	BOTTOM
4	0	1	0	0	CTC	OCR1A	Immediate	MAX
5	0	1	0	1	Fast PWM, 8-bit	0x00FF	BOTTOM	TOP
6	0	1	1	0	Fast PWM, 9-bit	0x01FF	BOTTOM	TOP
7	0	1	1	1	Fast PWM, 10-bit	0x03FF	BOTTOM	TOP
8	1	0	0	0	PWM, Phase and Frequency Correct	ICR1	BOTTOM	BOTTOM
9	1	0	0	1	PWM, Phase and Frequency Correct	OCR1A	BOTTOM	BOTTOM
10	1	0	1	0	PWM, Phase Correct	ICR1	TOP	BOTTOM
11	1	0	1	1	PWM, Phase Correct	OCR1A	TOP	BOTTOM
12	1	1	0	0	CTC	ICR1	Immediate	MAX
13	1	1	0	1	(Reserved)	–	–	–
14	1	1	1	0	Fast PWM	ICR1	BOTTOM	TOP
15	1	1	1	1	Fast PWM	OCR1A	BOTTOM	TOP

Note: 1. The CTC1 and PWM11:0 bit definition names are obsolete. Use the WGM12:0 definitions. However, the functionality and location of these bits are compatible with previous versions of the timer

See Table 39 on p.97 of datasheet

DC Brush Motor

- Features:
  - Most simple motor to use
  - Available with gears
  - Speed can be controlled using PWM
- Design Considerations:
  - Noise suppression
  - Requires motor driver
  - Inductive kickback

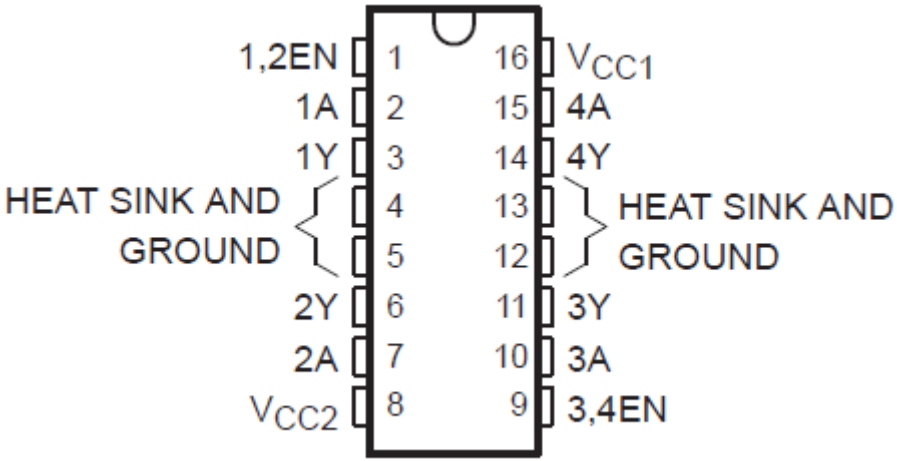


Motor Driver

- H-Bridge motor driver allows a motor with microcontroller
- Example: SN754410
  - Economical
  - Supports 2 motors
  - Separate motor supply
  - 1A current handling per motor

SN754410 Driver

- Simple Connections:
  - Motor leads connected to 1Y/2Y and 4Y/3Y pairs
  - Vcc2 is the motor power supply
  - Connect A and EN lines to Atmega



FUNCTION TABLE
(each driver)

INPUTS†		OUTPUT
A	EN	Y
H	H	H
L	H	L
X	L	Z

H = high-level, L = low-level
X = irrelevant
Z = high-impedance (off)
† In the thermal shutdown mode, the output is in a high-impedance state regardless of the input levels.

Other Motor Types

- Stepper motor
  - Allows precise angular movements in discrete steps
  - Appropriate motor driver required
  - Multiple independent coils “phases” which must be cycled in specific order to induce movement
- Servo motor
  - Allows precise angular movements
  - Built-in motor driver and control system
  - Controlled via width/timing of pulses directly from Atmega
  - Depending on model, may not make complete rotations

Q&A

- Questions about Tech Assignment 2