CG1112 Engineering Principles and Practices II for CEG

Tutorial 2

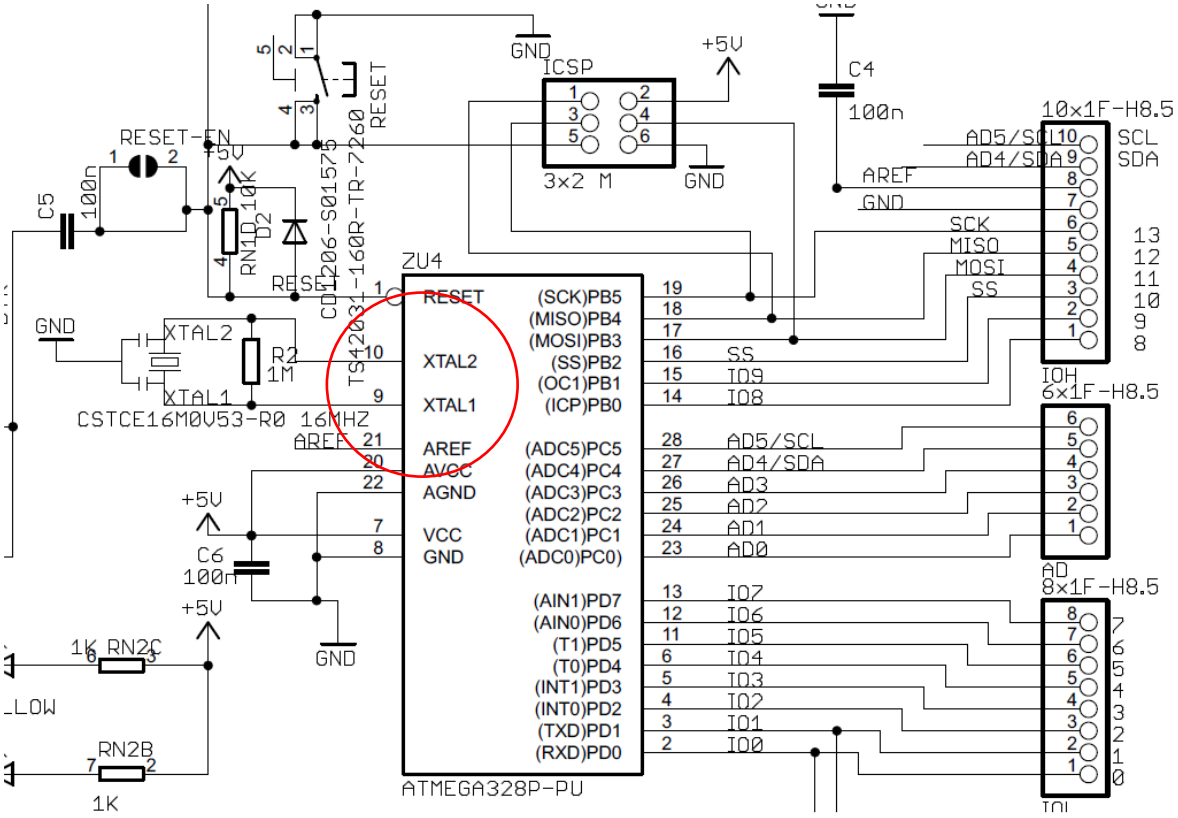
GPIO, Interrupts and PWM

Part I – GPIO

Question 1

Most Microcontroller Pins have multiple functionalities. It is up to the designer to plan ahead and be aware of the peripherals that are needed and select the appropriate pins to be used in the project. In the studio we have used PORT B pins for the LED's and Switches.

PB7 and PB6 are currently mapped to the external crystal oscillator (XTAL1 and XTAL2 for the Uno Board).



You have decided to create your own board using the Atmega328p and want to make use of all 8-bits of PORTB. Is it possible? What are the factors to consider?

(Hint: Refer to "Section 13: System Clock and Clock Options" in the Atmega datasheet)

Question 2.

In our studio we configured LED's in "Active-High" logic, i.e. we applied a Logic '1' to Turn it 'ON' and a Logic '0' to turn it 'OFF'.

a. Draw a circuit connection to connect an LED in "Active-Low" logic to PORTB Pin 1, i.e. you need to apply a Logic '0' to turn it 'ON' and a Logic '1' to turn it 'OFF'. Write the code to set the DDRB register value according to your schematic.

b. In a design such as this, we say that the microcontroller is "sinking" current. That means, that current is now flowing into the pin of the device. The manufacturer specifies a limit to the amount of sinking current. Anything more than this, could lead to a "smoke-effect" + "water-sprinkler" effect in the lab.

   Refer to Table 32-2 in the Atmega328p datasheet and state the maximum sinking current when the device is operating at 5V.

c. The voltage drop across the LED is 0.7V. Choose an appropriate R value to prevent a "smoke-effect".

Question 3

Power Consumption is a critical factor in Embedded Systems and it is important to minimize is to as to extend its usage before recharging the batteries. You read from Section 14 of the datasheet that the AT328P has some energy-saving features that you want to implement for "Alex" your robot. Mainly, you wish to put the robot in "Standby Mode" when the robot is idle and not doing anything useful.

a. Which is the main register that controls these features and what value should be written to it?
b. In Standby Mode, which events can trigger the device to "wake-up" and resume full-functionality?
c. How many clock cycles does it take for the device to come out of Standby mode to full operation?
d. Based on your initial assessment, you feel that you may not need to use the ADC module for this project. As such, you want to disable it so that it doesn't consume any additional power. How can this be achieved?

Part II - Interrupts

Question 4.

What are hardware interrupts? Why are they needed? Give some examples of hardware interrupts in your laptop or PC.

Similarly, what are software interrupts? Give some examples of how software interrupts can be used.

Question 5.

What interrupt request lines are available on the Atmega328P? Describe these lines and how they are.

Question 6.

Discuss used how hardware interrupts are implemented. In particular, talk about how hardware interrupts are detected, how the CPU/MCU decides with ISR to run, how control is handed over to the ISR, and how execution resumes normally after the ISR has completed execution.

Part III – PWM

Question 7.

We saw that the AT328p has 3 Timers capable of generating 6 individual PWM signals. Consider the following scenarios and describe how you will be able to resolve the challenges.

a. Describe a SW approach to generate a PWM signal that is not dependent on the HW PWM peripheral block within the microcontroller. What is the drawback of generating the PWM using this approach?
b. The microcontroller that you have to use for a particular project (not the AT328p) doesn't have any PWM module. How can you still generate a PWM signal WITHOUT relying on the software-based approach that you did in Part A?
   (Recall HW-based solutions from EPP1)
c. You feel that the approach in the earlier part is going to add additional cost to your project and you decide that stick with the AT328P. You require a very low-frequency PWM signal and even with the largest pre-scaler setting, the period is still too high. How will you be able to generate a PWM signal with the required period?

   Propose at least **TWO** different approaches that can involve both HW and SW.

Question 8.

In the studio, you were driving a single motor. If we were to drive both Motors concurrently, one possibility is to use both Timer 0 and 2. Assume that both Timer blocks are configured to use Interrupts and that the microcontroller was executing code in your loop() routine prior to the interrupts.

a. If both these interrupts were to be triggered at the same time, what would be the sequence of execution?
b. If Timer 0 triggered the interrupt just before Timer 2, what would be the sequence of execution?