Power Management - Lab

2016-10-10

Aim

To get familiar with the power management, real-time clock, and sleep modes of the AVR32. To gain practical experience of power consumption of CMOS devices at different clock speeds.

Literature

Data sheet “AT32UC3A” AVR32 32-Bit Microcontroller (the manual)

- Chapter 13. Power Manager (PM)
- Chapter 14. Real Time Counter (RTC)

Brief summary

This exercise gives a basic introduction to the power management features of the AVR32 chip, both in terms of setting different clock speeds for the CPU and using the different sleep modes.

Practical measurements on the power consumption of the CPU and other devices of the EVK1100 board will be performed, to give you a feeling for what consumes power in an embedded device.

Reporting

Demonstrate the programming tasks for the teaching assistants. Make sure you are prepared to answer all the preparatory questions.

Upload (1) the final C files and (2) a brief report about the measurement tables to KTH Social.

1 Setting the Real-time Clock

In this section, you will experiment with the Real-Time Counter (RTC) module of the AVR32 microcontroller, in order to control a functionality that requires real-time periodicity.

Review section “14. Real Time Counter (RTC)” in the microcontroller manual to gain an understanding of the AVR32 RTC’s functionality. The overview of the PM module in the block diagram of Figure 4.1 may also be useful.

Preparatory Questions

1. Describe the clock source(s) that the RTC module can run on? What is the frequency of each of the available sources?
2. When configuring the RTC, how can one decide which of the clock sources is used by the RTC?
3. What is the role of the 16-bit prescaler?
4. For a given source clock, which bitfield is used to control the output frequency of the prescaler?
5. What is the relationship between the input and output frequencies of the prescaler? What is the maximum and minimum value for the resulting frequency?
6. What is the role of the 32-bit counter? Explain how the 32-bit counter operates!
7. What is the role of the bitfields RTC_TOP, RTC_VAL, and TOPI?
8. What is the relationship between the input and output frequencies of the 32-bit counter?

Programming Task

1. To experiment with the RTC module, create a new AVR32 project, based on the example project “RTC Example” for the EVK1100 board.
2. Run the example program. Investigate and understand the program, by observing its behavior, as well as studying its code.
3. Make sure you understand how the RTC module is being used. When necessary, further investigate the used functions as defined in the RTC driver - rtc.h.
4. Device a method to test whether the clock interrupts produced by the RTC correspond to the expected clock values in real time!

5. Explain where in the code do each of the relevant bitfields - discussed in the manual – get modified! For example,
    a. What is the output frequency of the prescalor? Which function in the code sets this frequency? Which bitfields are being set, and what values are they being set to?
    b. What is the output frequency of the RTC? Which function in the code sets this frequency? Which bitfields are being set, and what values are they being set to?
6. Modify the program so that an LED blinks based on a real-time interrupt produced
    a. Every 2½ seconds.
    b. At a rate of 2.5 Hz.

2 Setting the CPU Speed

In this section, you will learn to control the runtime frequency of the AVR32, resulting in different speeds, and power consumptions of the device.

The PM module can be used to set different frequencies for different parts of the device, such as its CPU, and various busses, and peripherals. In this lab, we will focus on setting the CPU speed only.

Review section "13. Power Manager (PM)" in the microcontroller manual to gain an understanding of the AVR32 power management functionality. The overview of the PM module in the block diagram of Figure 4.1 may also be useful.

Preparatory Questions

1. What are the different clock sources available for the PM module?
2. Describe the functionality of a PLL. Describe the main parameters/registers that control the functionality of a PLL.
3. Which of the following clock generators can be used for setting the CPU frequency: (a) the "Synchronous Clock Generator" or (b) the "Generic Clock generator"?
4. Describe the functionality of the Synchronous/Generic Clock Generator, which is of relevance when controlling the CPU frequency. Describe the main parameters/bitfields that control the operation of the generator.

Programming Task

1. To experiment with the PM module, create a new AVR32 project, based on the example project "PM Example 2" for the EVK1100 board.
2. Run the example program. Investigate and understand the program, by observing its behavior, as well as studying its code.
3. Make sure you understand how the main CPU clock is being set. When necessary, further investigate the used functions available as defined in the PM driver - pm.h.
4. What is the resulting CPU frequency being set?
5. Explain where in the code do each of the bitfields - discussed in the manual – get modified in order to reach the desired frequency?
6. Adjust the software delay period so that the LED blinks at a 1 Hz frequency. To obtain an (almost) accurate frequency, blink and compare to another LED that blinks based on a RTC interrupt of 1 Hz.
7. Calculate the appropriate values of the bitfields/parameters that are needed to reach the desired CPU clock speeds in the table below. Start by choosing a suitable frequency to be outputted by the PLL. Note that the PLL does not work with all settings! Specifically, the PLL will fail if you run it outside of its specifications (e.g. allowed frequency range).
8. Modify the program to implement each of the clock speeds in the table below.

9. For each clock speed (C_n):
    a. Measure (e.g. with your wristwatch) the resulting LED blinking frequency (F_n).
    b. Measure the power consumption using an amperemeter. In order to connect it, you need to supply power through the external jack instead of through USB (switch in position "EXT"). There are custom-made cables that you may use– its red cable needs to be connected to +5V and the black one to ground.
10. Explain the relationship between the blinking frequency (F_n) and the clock speed C_n, compared to the reference blinking frequency of 1 Hz (F_o), and its reference clock speed (C_o)?
11. **Hint**: to simplify measurement, you may use the pushbuttons on the EVK board to dynamically switch clock frequency at runtime.

Settings	**Clock speed (C_n) [MHz]**				
	2	12	33	48	66
PLL frequency					
PLLOSC					
PLLMUL					
PLLDIV					
PLLOPT[1]					
PCSEL					
CPUSEL					
CPUDIV					
Works as expected?					
LED blinking frequency (F_n)					
power consumption					

3 Using Sleep Modes

Finally, you will now experiment with the different sleep modes of the processor.

Review section “13.5.7 Sleep modes” in the microcontroller manual to gain an understanding of the AVR32 sleep modes.

Programming Task

1. Investigate the function Sleep(mode) available in the PM driver - pm.h.
2. Modify the program from the “Setting a Real-time Clock” section, such that the software goes to sleep when no code needs to be executed.
3. Experiment with the available sleep modes, as listed below:
    a. With which sleep modes does the software still perform the same desired functionality? With which ones does it fail? Explain!
    b. Measure the power consumption of the board, when running the CPU clock at each of 12 and 66 MHz

Sleep mode	**Software functioning?**	**Power consumption at 12 MHz**	**Power consumption at 66 MHz**
Idle			
Frozen			
Standby			
Stop			
Deep stop			
Static			