TimerA with the EasyWeb2 Hardware Platform

by Alex Milenkovich, milenkovic@computer.org

Objective: This tutorial discusses the use of MSP430’s TimerA.

Note: Required are all previous tutorials.

1. Blink a LED Using TimerA ISR

Let us consider the following program shown in Figure 1. We want to periodically blink the status LED on the EasyWeb2 board. Rather than using software delay we want to learn how to use the MSP430’s TimerA peripheral device. Note: to learn more about TimerA read the user manuals.

The MSP430 clocks MCLK, SMCLK, and ACLK are default (MCLK=SMCLK=800KHz). TimerA is configured to use the SMCLK as its clock input and to operate in the continuous mode. The CCR0 register is initialized with 50,000 and the TimerA is configured to generate an interrupt when the timer’s counter reaches the value in CCR0. After the initializing the timer the processor is put in the low power mode 0. What happens in the low-power mode 0? Is the DCO still active? What about SMCLK?

When the interrupt is received the processor exits the low-power mode. Explain how? The TimerA interrupt service routine toggles the port P2.1 connected to the status LED, and 50,000 is added to the CCR0 register. What is going to be the new value in the CCR0? How long it will take until a new interrupt is received? Explain. How many times the LED is toggled in a second? Explain.
Figure 1. C program for Blinking the Status LED using TimerA ISR.
2. **Assignments**

**Assignment #1: Blinking the Status LED.**
Write a C program that toggles the status LED on the EasyWeb2 development board with frequency of 0.5 Hz (approx. 1s is on, and 1s is off). Use the high frequency crystal for MCLK (8 MHz) and ACLK (1 MHz). TimerA is clocked by ACLK. Write two versions of the program (i) TimerA is in the UP mode, and (ii) TimerA is UP/DOWN mode. Explain how have you determined values to be stored in counters?

**Assignment #2: Buzzer**
Write a C program that controls the buzzer on the EasyWeb2 development board. The buzzer is controlled by P4.2 and P4.3 ports and it needs a square wave input of 4.0 +/- 0.5 KHz on both ports, but with opposite values, that is, when P4.2 is a logic one, P4.3 should be a logic zero. The buzzer should repeat the sequence: 1 second off, 2 seconds on. Use TimerA to generate square waves.