Projeto 2 - Otimização Energética

Felipe Frid Buniac

24 de Outubro de 2017

1 Introdução

O projeto descrito neste documento tem como objetivo trabalhar com otimização energética em uma aplicação embarcada. Sistemas embarcados são muitas vezes associados a sistemas móveis/portáteis e devem para isso utilizar baterias como fonte de energia. Para garantir a maior economia de bateria deste embarcado a otimização energética é de extrema importância.

2 Descrição do problema

Para garantir uma otimização energética em uma aplicação embarcada este projeto visa fazer um LED externo ao kit controlado por um microcontrolador piscar no mínimo a cada 0.5 segundos utilizando como fonte única de alimentação um super-capacitor eletrolítico de 4F o dispositivo embarcado deve responder a comandos enviados via UART para ativar/desativar o pisca LED e deve informar se o comando foi executado ou não (resposta do microcontrolador).

3 Esquemático

O esquemático inicial fornecido possuia :

- 1x Super capacitor eletrolítico de 4F
- 1x Resistor de 470 ohms para limitar a corrente do LED
- 1x LED azul

Esquema elétrico do projeto

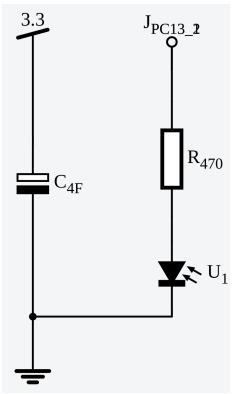

Figure 1: Esquemático do embarcado do projeto

Diagrama de Blocos do projeto com mudanças

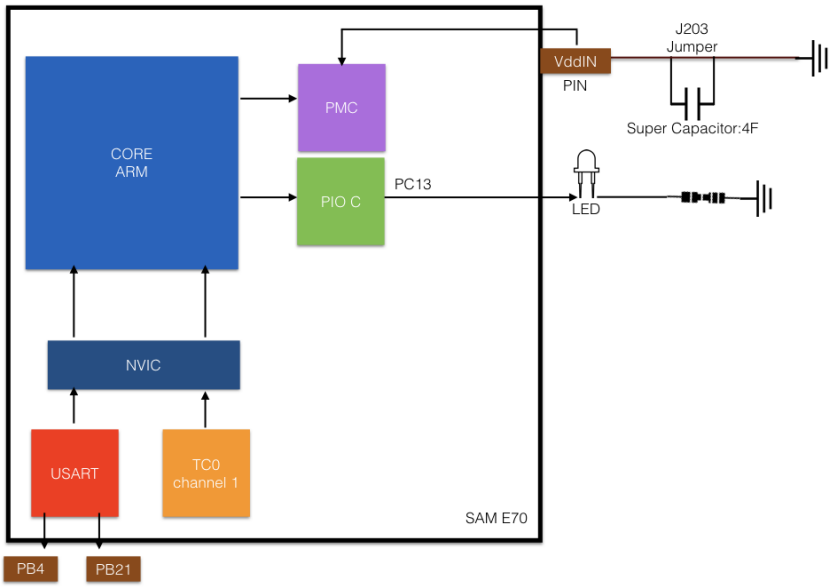

Figure 2: Diagrama de blocos do projeto

4 Mudanças de software

Mudança no clock do processador

A primeira mudança de alta importância para atingir os requisitos do projeto foi uma mudança no prescale do clock do processador.

A conta feita para definir o clock do processador

$$Fhclk = Fsys/(SYSCLK\_PRES)$$

e o prescale inicial definido era de 1 portanto não havia divisão na frequência do clock que o processador trabalha.(#define CONFIG_SYSCLK_PRES SYSCLK_PRES_1)

A mudança feita foi uma maior divisão do clock do processador pelo maior número possível (64) fazendo com que o clock do processador passe de 300mHz para 4,6mHz ganhando muito na eficiência energética da placa que não trabalha a 300 mHz quando não é necessário.(#define CONFIG_SYSCLK_PRES SYSCLK_PRES_64)

Mudanças na main

A mudança inicial do código da main foi a remoção de aspectos que não estavam sendo utilizados no código. Remoção da configuração de um botão (defines, init e handler) e remoção da configuração do RTC (defines,init e handler).

A segunda mudança foi em colocar um sleepmode dentro do $while(1)$ para o processador dormir enquanto o TC Handler não está sendo chamado (pmc_enable_sleepmode(0)). *"The purpose of sleep mode is to optimize power consumption of the device versus response time. In this mode, only the core clock is stopped. The peripheral clocks can be enabled. The current consumption in this mode is application-dependent."* [1].

A terceira mudança foi feita na inicialização do TC onde mudei a frequência do LED para que pisque a cada meio segundo

$$tc\_find\_mck\_divisor(\mathbf{2}, ul\_sysclk, \&ul\_div, \&ul\_tcclks, ul\_sysclk);$$

sendo que o período T = 1/frequência. Portanto 1/2. Mudei o modo de operação

$$tc\_init(TC0, channel, ul\_tcclks|\mathbf{TC\_CMR\_WAVE})$$

Fiz o piscar do LED ser mais rápido aumentando seu divisor

$$tc\_write\_rc(TC0, channel, (ul\_sysclk/ul\_div)/\mathbf{93});$$

Fiz uma mudança na interrupção de RA para que o LED ficasse um tempo sem piscar piscasse rapidamente e depois ficasse desligado e depois voltasse a piscar rapidamente. Este é um compare mode com interrupções em níveis diferentes.

$$tc\_write\_ra(TC0, channel, (ul\_sysclk/ul\_div)/94);$$

Por fim concatenei as interrupções de RA e RC para definir quando ativar a interrupção de RC.

$$tc\_enable\_interrupt(TC0, channel, \mathbf{(TC\_IER\_CPCS)|(TC\_IER\_CPAS)});$$

Mais informações sobre o RC e RA podem ser encontradas em [1] nas páginas (1381-1390).

5 Mudanças de hardware

Foram feitas duas mudanças no hardware.

A primeira mudança foi a troca de um LED azul de alto brilho por um LED vermelho pois o LED azul tem um consumo muito alto de energia.

A segunda mudança foi feita no resistor inicial de 460 Ω por um resistor de maior resistência de 2,2 kΩ para diminuir o brilho do LED.

6 Mudanças que não foram implementadas

Está sessão envolve mudanças que não foram implementadas por falta de tempo que trariam melhora.

- Utilização do RTT para entrar em deep sleep mode deixando o microcontrolador com consumo zero enquanto não faz a interrupção do TC
- Fazer o LED acender e apagar em forma de PWM sendo que nunca estaria 100% acesso trazendo uma efetividade maior na economia de energia.
- Houve mudança no clock do processador e poderia existir uma mudança no clock do sistema
- Ao invés de conectar o LED no PIO C poderia ter conectado no PIO B pois tem menor consumo de energia.
- Poderia fazer com que houvesse interrupção para USART e utilizar a UART ao invés da USART.

7 Resultados

Tempo inicial:1.17 segundos.
Tempo final: 26 minutos.
Otimizei em 25 minutos e 58 segundos.

References

[1] Cortex-M7-SAM-E70.pdf