LDH
Arduino LowPower Modes

Manuel J. Bellido Díaz
Germán Cano Quiveu

Referencias

- Codigo Base Arduino
  - https://github.com/arduino/Arduino
- Librerias Arduino
  - https://www.arduino.cc/en/Reference/ArduinoLowPower
  - https://github.com/rocketscream/Low-Power
- Datasheet ATMEGA328P
  - http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

BAJO CONSUMO

- En el campo del IoT es vital el diseño de bajo consumo, esto es debido a que normalmente crearemos pequeños nodos que se usarán en cualquier lugar, por lo que resultará en usar una bateria para su alimentación. Por tanto nuestro deber a la hora de diseñar nodos IoT es asegurarnos que estos nodos puedan estar enviando datos durante el máximo tiempo posible antes de reemplazar la bateria.
- En las próximas transparencias observaremos que los MCU tienen modos de baja potencia y veremos algunos ejemplos de cuanto duraría una bateria según estos modos. (En concreto usaremos los valores del MCU ATMEGA328P).

CONSUMO

28.3 DC Characteristics

T_A = –40°C to +125°C, V_{CC} = 2.7V to 5.5V (unless otherwise noted)

Parameter	Condition	Symbol	Min.	Typ.(2)	Max.	Units
Power supply current(1)	Active 4MHz, V_{CC} = 3V	I_{CC}		1.5	2.4	mA
	Active 8MHz, V_{CC} = 5V			5.2	10	mA
	Active 16MHz, V_{CC} = 5V			9.2	14	mA
	Idle 4MHz, V_{CC} = 3V			0.25	0.6	mA
	Idle 8MHz, V_{CC} = 5V			1.0	1.6	mA
	Idle 16MHz, V_{CC} = 5V			1.9	2.8	mA
Power-down mode(3)	WDT enabled, V_{CC} = 3V				44	µA
	WDT enabled, V_{CC} = 5V				66	µA
	WDT disabled, V_{CC} = 3V				40	µA
	WDT disabled, V_{CC} = 5V				60	µA

Notes:
1. Values with Section 9.10 “Minimizing Power Consumption” on page 36 enabled (0xFF).
2. Typical values at 25°C.
3. The current consumption values include input leakage current.

Supongamos que tenemos una bateria LIPO que suministra 500mAh, entonces duraria:

1) Active 8Mhz Vcc=5V Icc=5.2mA => 96 horas duracion LIPO , 4 dias

2) Idle 8Mhz Vcc=5V Icc=1.0mA => 500 horas duracion LIPO , 21 dias

3) Power-Down WDT enable Vcc=5V Icc=66*(10^-3) mA => 7575 horas duracion LIPO, 315 dias

POWER MANAGEMENT

- El ATMEGA328 tiene varios sleep modes para desactivar aquellos perifericos que no sean utiles

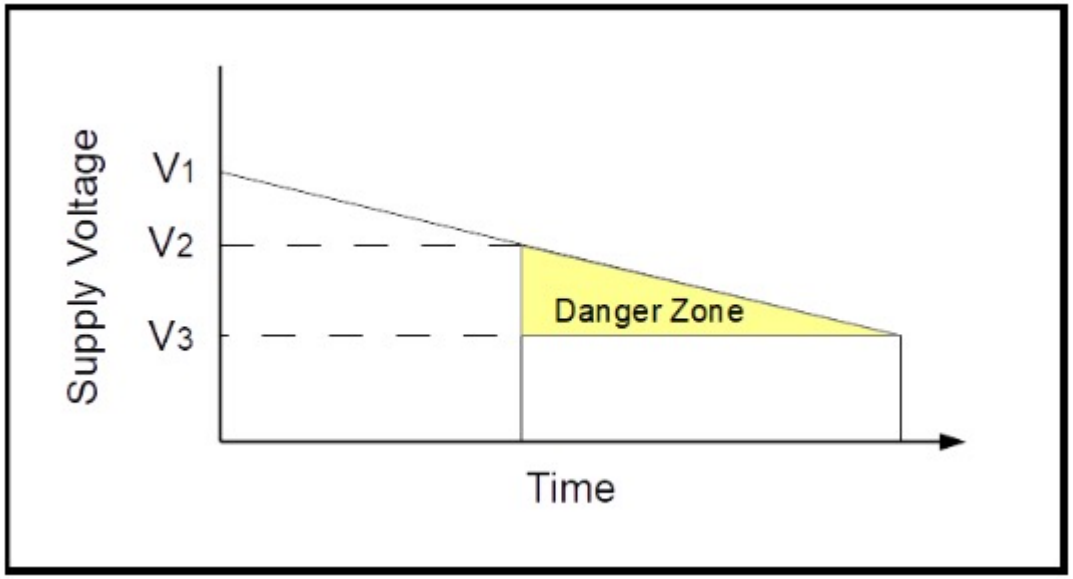
Table 9-1. Active Clock Domains and Wake-up Sources in the Different Sleep Modes.

Sleep Mode	Active Clock Domains					Oscillators		Wake-up Sources							Software BOD Disable
	clk_{CPU}	clk_{FLASH}	clk_{IO}	clk_{ADC}	clk_{ASY}	Main Clock Source Enabled	Timer Oscillator Enabled	INT1, INT0 and Pin Change	TWI Address Match	Timer2	SPM/EEPROM Ready	ADC	WDT	Other/O	
Idle			X	X	X	X	X(2)	X	X	X	X	X	X	X	
ADC noise Reduction				X	X	X	X(2)	X(3)	X	X(2)	X	X	X		
Power-down								X(3)	X				X		X
Power-save					X		X(2)	X(3)	X	X			X		X
Standby(1)						X		X(3)	X				X		X
Extended Standby					X(2)	X	X(2)	X(3)	X	X			X		X

Notes: 1. Only recommended with external crystal or resonator selected as clock source.
2. If Timer/Counter2 is running in asynchronous mode.
3. For INT1 and INT0, only level interrupt.

POWER MANAGEMENT

- BOD : Brown-Out Detector, es un circuito que monitoriza el voltaje suministrado al microcontrolador Vcc. Su funcion es mantener el voltaje del MCU por encima de un cierto valor (V2), si el Vcc es menor que V2 entra el MCU en RST hasta que el suministro se reestablezca. Esto es debido a que por debajo de V3 el MCU no funciona y en el rango de V2 y V3 el Mcu esta en una zona donde no se puede asegurar su comportamiento. Por lo que el BOD es una medida de seguridad.



Libreria Arduino (sleep.h)

- https://github.com/vancegroup-mirrors/avr-libc/blob/master/avr-libc/include/avr/sleep.h
- A continuacion veremos la estructura básica que se encuentra dentro del las librerias avr de arduino

```
#include <avr/interrupt.h>
#include <avr/sleep.h>


set_sleep_mode(<mode>); // configuramos el modo de sleep de MCU
Cli();                          //deshabilita las interrupciones
if (some_condition)
{
  sleep_enable();               // habilitamos la funcion de sleep
  sleep_bod_disable();          //deshabilita BOD
  Sei();                        //habilita las interrupciones
  sleep_cpu();                  //ejecuta las instruccion SLEEP
                                //se para la ejecucion hasta que se reciba la señal de wake up
  sleep_disable();              //deshabilitamos la funcion sleep
}
sei()
```

Libreria LowPower (I)

- A continuacion veremos una Libreria ya creada que gestiona los sleep modes de Arduino
  - https://github.com/rocketscream/Low-Power
  - Su funcionamiento básico se define por dos funciones las cuales siguen la estructura basica que se presento en la anterior diapositiva. Ambas ponen el MCU en modo sleep, la diferencia es que una desactiva el BOD y la otra no.

```
#define lowPowerBodOn(mode)   \
do {                          \
    set_sleep_mode(mode); \
    cli();                \
    sleep_enable();       \
    sei();                \
    sleep_cpu();          \
    sleep_disable();      \
    sei();                \
} while (0);
```

```
#define lowPowerBodOff(mode) \
do {                         \
    set_sleep_mode(mode); \
    cli();                \
    sleep_enable();       \
    sleep_bod_disable(); \
    sei();                \
    sleep_cpu();          \
    sleep_disable();      \
    sei();                \
} while (0);
```

Libreria LowPower (II)

- Esta libreria usa una serie de tipos propios los cuales describiremos a continuacion.
  - period_t : indica un valor ya sea en milisegundos MS, en segundos S, o infinito. Representa cuanto tiempo estara el MCU en el modo sleep. Sus posibles valore son:

```
enum period_t
{
        SLEEP_15MS,
        SLEEP_30MS,
        SLEEP_60MS,
        SLEEP_120MS,
        SLEEP_250MS,
        SLEEP_500MS,
        SLEEP_1S,
        SLEEP_2S,
        SLEEP_4S,
        SLEEP_8S,
        SLEEP_FOREVER
};
```

Libreria LowPower (II)

- bod_t : este tipo indicara si activamos el BOD o lo deshabilitamos, siempre que sea posible dependiendo del modo de sleep. Sus posibles valores son:
  - BOD_ON
  - BOD_OFF
- adc_t : este tipo indicara si activamos o deshabilitamos el convertidor analogico digital, siempre que sea posible dependiendo del modo sleep. Sus posibles valores son:
  - ADC_ON
  - ADC_OFF
- twi_t : este tipo indicara si activamos o deshabilitamos el i2c, siempre que sea posible dependiendo del modo sleep. Sus posibles valores son:
  - TWI_ON
  - TWI_OFF
- usart0_t : este tipo indicara si activamos o deshabilitamos la interfaz serie, siempre que sea posible dependiendo del modo sleep. Sus posibles valores son:
  - USART0_ON
  - USART0_OFF

Libreria LowPower (II)

- timer0_t : este tipo indicara si activamos el timer 0 o lo deshabilitamos, siempre que sea posible dependiendo del modo de sleep. Sus posibles valores son:
  - TIMER0_ON
  - TIMER0_OFF
- timer1_t : este tipo indicara si activamos el timer 1 o lo deshabilitamos, siempre que sea posible dependiendo del modo de sleep. Sus posibles valores son::
  - TIMER1_ON
  - TIMER1_OFF
- timer2_t : este tipo indicara si activamos el timer 2 o lo deshabilitamos, siempre que sea posible dependiendo del modo de sleep. Sus posibles valores son:
  - TIMER2_ON
  - TIMER2_OFF

Libreria LowPower (III)

- La libreria viene con funciones para cada uno de los modos de sleep del ATMEGA328. Asi que veremos su declaracion y ejemplo de cada uno.
  - Idle Mode
    - Void idle(period_t period, adc_t adc, timer2_t timer2, timer1_t timer1, timer0_t timer0, spi_t spi, usart0_t usart0, twi_t twi);
    - LowPower.idle(SLEEP_8S, ADC_OFF, TIMER2_OFF, TIMER1_OFF, TIMER0_OFF, SPI_OFF, USART0_OFF, TWI_OFF);
      - Se despierta a los 8 segundos y desactiva la mayoria de perifericos
    - LowPower.idle(SLEEP_FOREVER);
      - Entra en idle y despertara por interrupciones, los perifericos por defecto estan activados.
  - ADC Noise Reduction Mode
    - void   adcNoiseReduction(period_t period, adc_t adc, timer2_t timer2) __attribute__((optimize("-O1")));
    - LowPower.adcNoiseReduction(SLEEP_FOREVER,ADC_ON,BOD_OFF);
  - Power Save Mode
    - void   powerSave(period_t period, adc_t adc, bod_t bod, timer2_t timer2) __attribute__((optimize("-O1")));
    - LowPower.powerSave(SLEEP_8S,ADC_ON,BOD_OFF,TIMER2_OFF);

Libreria LowPower (IV)

- Power StandBy Mode
  - void powerStandby(period_t period, adc_t adc, bod_t bod) __attribute__((optimize("-O1")));
  - LowPower.powerStandby(SLEEP_30MS);
- Power Extended StandBy Mode
  - void powerExtStandby(period_t period, adc_t adc, bod_t bod, timer2_t timer2) __attribute__((optimize("-O1")));
  - LowPower.powerExtStandby(SLEEP_FOREVER,ADC_OFF,BOD_ON,TIMER2_OFF);
- Power Down Mode
  - powerDown(period_t period, adc_t adc, bod_t bod) __attribute__((optimize("-O1")));
  - LowPower.powerDown(SLEEP_FOREVER, ADC_OFF, BOD_OFF);