Arduino, Zigbee and Embedded Development

A helpful blog designed to get beginners up and running with Arduino, Zigbee and general embedded development.

Sunday, 25 April 2010

Sleeping Arduino - Part 2 Wake Up Via An External Interrupt

Overview

In the second entry of this "Sleeping Arduino" series, we will be covering how to wake the Arduino via an external interrupt. We will be using the external interrupt circuit that has been covered in a previous blog Arduino External Interrupts. Please be sure to get the basic external interrupt example working before attempting to follow this entry, this will prove that your hardware setup is correct.

Operation

Our code will operate as follows:

1. Set up the serial port and set pin 2 (INT0) as an input;
2. Run the loop function which will:
    1. Stay awake for 3 seconds;
    2. Once the 3 seconds have elapsed, SLEEP_MODE_PWR_DOWN will be entered;
    3. All code execution stops;
3. The user then pushes the switch and pin 2 (INT0) will become low;
4. The INT0 interrupt will fire and bring the Arduino out of sleep mode;
5. Code execution continues where it had previously stopped.

Circuit

The circuit is set up as specified in the Arduino External Interrupts blog.

Source Code

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


int pin2 = 2;


/****************************************************
 *  Name:        pin2Interrupt
 *
 *  Returns:     Nothing.
 *
 *  Parameters:  None.
 *
 *  Description: Service routine for pin2 interrupt
 *
 ****************************************************/
void pin2Interrupt(void)
{
  /* This will bring us back from sleep. */

  /* We detach the interrupt to stop it from
   * continuously firing while the interrupt pin
   * is low.
```

Pages

- Home
- Projects & Tutorials

Blog Archive

► 2012 (7)
► 2011 (7)
▼ 2010 (5)
▼ April (4)
Sleeping Arduino - Part 3 Wake Up Via the UART
Sleeping Arduino - Part 2 Wake Up Via An External ...
Sleeping Arduino - Part 1
Arduino External Interrupts
► January (1)

About Me

Donal Morrissey

Senior embedded software engineer and team leader with experience in the telecommunications and automotive industries. Android application developer during my spare time. This includes the continued development of 'My Migraine Log', which is an application for tracking a user's migraine attack history. And a similar app called 'My Headache Log'. Both are currently on sale on the Android market. Specialties o Embedded software development o Android application development o SCRUM o Real-Time Operating Systems o Linux software development (embedded & desktop) o Microchip PIC16/PIC24 firmware development o Bootloaders, including fault tolerant remote update of application software o Embedded RF system software o Automotive embedded software o CAN Gateways & networks o Hardware interfacing (I2C, SPI, CAN, Etc) o Zigbee o C, C++, JAVA, Perl, Python o QT o Team Leadership o Recruitment

View my complete profile

Site Map

Site Map

```
   */
  detachInterrupt(0);
}


/***************************************************
 *  Name:        enterSleep
 *
 *  Returns:     Nothing.
 *
 *  Parameters:  None.
 *
 *  Description: Enters the arduino into sleep mode.
 *
 ***************************************************/
void enterSleep(void)
{

  /* Setup pin2 as an interrupt and attach handler. */
  attachInterrupt(0, pin2Interrupt, LOW);
  delay(100);

  set_sleep_mode(SLEEP_MODE_PWR_DOWN);

  sleep_enable();

  sleep_mode();

  /* The program will continue from here. */

  /* First thing to do is disable sleep. */
  sleep_disable();
}


/***************************************************
 *  Name:        setup
 *
 *  Returns:     Nothing.
 *
 *  Parameters:  None.
 *
 *  Description: Setup for the Arduino.
 *
 ***************************************************/
void setup()
{
  Serial.begin(9600);

  /* Setup the pin direction. */
  pinMode(pin2, INPUT);

  Serial.println("Initialisation complete.");
}



/***************************************************
 *  Name:        loop
 *
 *  Returns:     Nothing.
 *
 *  Parameters:  None.
 *
 *  Description: Main application loop.
 *
 ***************************************************/
int seconds=0;
void loop()
{
  delay(1000);
  seconds++;

  Serial.print("Awake for ");
  Serial.print(seconds, DEC);
  Serial.println(" second");

  if(seconds == 3)
  {
    Serial.println("Entering sleep");
    delay(200);
    seconds = 0;
    enterSleep();
  }

}
```

The sketch for this program can be downloaded here.

All parts of this series:

- Part 1 Overview Of Arduino Sleep Modes
- Part 2 Wake Up Via An External Interrupt
- Part 3 Wake Up Via the UART
- Part 4 Wake Up Via Internal Timer
- Part 5 Wake Up Via The Watchdog Timer

Posted by Donal Morrissey at 12:09

25 comments:

Greg Hauptmann 25 June 2010 at 16:08

Hi Donal,

I'm finding that if I continually press the button I have setup to trigger the interrupt with code as it is the program kind of locks up after it goes into sleep.

BUT if I move the "detachInterrupt(0);" statement out of the call back function, and just put it at the bottom of the enterSleep() method than things seems to work robustly.

Any ideas why?

Cheers
Greg

Reply

Replies

Quentin M. 2 June 2013 at 15:22

Hi, first thanks for the article!
And thanks Greg for your comment, I have the same thing here, spent some time to understand why, but can't find anything, and your solution made it work for me, any bug with the detachInterrupt function?
Bye!
Quentin

jlzuri 12 May 2014 at 02:52

It's been a long time since this comment was posted, but I am sure there is still someone out there seeking an answer to this behaviour, which I got here: http://www.gammon.com.au/forum/?id=11488

The short answer is that an interrupt might occur between the call to attachInterrupt(...) and the call to sleep_mode(), in which case the arduino will never wake up.

1: attachInterrupt(...) gets called.
2: interrupt gets triggered.
3: isr callback gets called and so does detachInterrupt(...)
4: sleep_mode() get called.
5: arduino sleeps forever since interrupts are disabled.

Hope it helps
JL

Reply

Giannis 18 March 2011 at 04:19

Hello, What is the best way to set to sleep arduino using a Xbee module?
I need minimum consumption for my project. Thank you and great blog.

Reply

Donal Morrissey 22 March 2011 at 13:07

Hi Giannis,
Apologies for the delay in replying to you. Probably your best bet is to follow the very helpful tutorial here:

http://www.sensor-networks.org/index.php?page=0820520514
Cheers,
Donal

Reply

Mr Mase 31 December 2011 at 16:57

This comment has been removed by the author.

Reply

poona 16 April 2012 at 22:22

This comment has been removed by a blog administrator.

Reply

nb 17 July 2013 at 09:31

hello Donal,

I have tried this code on an Arduino Duemilanove, but I never get out the sleep. I have trie it by short utting GND and digital pin 2.
Do you have an explanation ?

Regards

Reply

nb 17 July 2013 at 11:18

I reply to myself.
I have added "digitalWrite(2,HIGH);" in setup after "pinMode" and it works perfectly now.

Reply

Replies

Tasos Str 19 December 2013 at 12:03

or
pinMode(pin2, INPUT_PULLUP);

or put a pullup resistor 10K

Very Good Example Thank you ! ! ! ! :-)

Reply

jean des îles 20 December 2013 at 04:48

HI and thanks a lot for your help
But where can we find these ones
#include
#include
Seems not so easy to find

Reply

jean des îles 20 December 2013 at 04:50

sorry these ones
avr/sleep.h
avr/power.h

Reply

jean des îles 23 December 2013 at 16:35

Problem is solved ,these files are in arduino directory ,but not in library so because my Arduino main directory was an another name ,the soft did not find the root
So ,"we should never change the arduino name directory"
jm

Reply

Ian Journeaux 17 September 2014 at 18:43

Trying to use this with a Pro Micro. It appears that it goes to sleep but I can not get it to wake up. I did make a change from INPUT to INPUT_PULLUP

Reply

Replies

Ben Lind 25 October 2014 at 16:08

I'm using a pro mini, and I'm experiencing the same problem.

Ben Lind 9 November 2014 at 08:34

This comment has been removed by the author.

Ben Lind 9 November 2014 at 08:35

The problem is that if we attach the interrupt while the interrupt pin is HIGH, then we will immediately go to the interrupt-function and detach the intterupt. Hence, when we go to sleep, there is no way to wake up.

This is solved by adding an if-statement:
if (digitalRead(pin2)){
sleep_mode();}

Also use this for the setup:
pinMode(pin2, INPUT_PULLUP);

I'm using a pro mini.
It might not be 100% fail proof in theory, but seems to work for me all the time.
Cheers :}

Дмитрий Чибышев 16 February 2015 at 08:04

The code written thus works stablly.
void enterSleep(void)
{
set_sleep_mode(SLEEP_MODE_PWR_DOWN);
sleep_enable();
/* Setup pin2 as an interrupt and attach handler. */
attachInterrupt(0, pin2Interrupt, LOW);
sleep_mode();
/* The program will continue from here. */

/* First thing to do is disable sleep. */
sleep_disable();
}

Reply

Andreas Lanz 12 September 2015 at 15:38

sleeb_mod() is a macro....defined in sleep.h:

#define sleep_mode() \
do { \
sleep_enable(); \
sleep_cpu(); \
sleep_disable(); \
} while (0)

so think 10 Sec. now!

Reply

Andreas Lanz 14 September 2015 at 12:54

/*
* Rain Counter
* in Power Down Mode
* Wake short up for count
* Bord Crumb128 with atmega128
*/

#include avr/sleep.h
#include util/delay.h

const int INT_0 =2; //Arduino Interrupt-No. for INT0
const int INT_0_PIN=16; //Arduino Pin-No. for INT0/PD0 Pin on atmega128
int count;

void setup()

```
{
pinMode(INT_0_PIN, INPUT_PULLUP); //Pullup Button Input PD0/INT0

set_sleep_mode ( SLEEP_MODE_PWR_DOWN ); // sleep mode is set here
attachInterrupt ( INT_0, interrupt1, FALLING ); //INT0 PD0

Serial.begin(9600);
}

void loop()
{
Serial.print("Counter: "); Serial.println(count);
delay(500);

sleep_mode(); //Sleep now
}

void interrupt1()
{
count++;

// Some Stuff for debouncing
while(digitalRead(INT_0_PIN)==LOW);
_delay_ms(1000);
while(digitalRead(INT_0_PIN)==LOW);
EIFR |= (1<<INTF0); //Other Interrupt? Clear! Write 1 to clear Interrupt-Flag INT0!
}
```

Reply

Brian Smith 12 October 2015 at 17:41

I'm confused by this bit of code:

```
sleep_mode();
/* The program will continue from here. */

/* First thing to do is disable sleep. */
sleep_disable();
```

Specifically, what is meant by the comment "The program will continue from here"? I'm assuming that once "sleep_mode();" executes, the Arduino is asleep, and will remain asleep until the Interrupt signal is received. When that happens, does the code start to execute right where the comment is? If that's right, it would explain the next comment, "First thing to do is disable sleep", with sleep_disable().

So, inside loop(), when the condition is met to put the Arduino to sleep, call enterSleep(). Immediately after that, put whatever code I want to execute when the Arduino is NOT asleep (or when it comes out of being asleep), right? It's just kind of confusing in this example Sketch, because the very last thing that happens in loop() is enterSleep() - what's the point in turning the Arduino on for 3 seconds, just to put it to sleep? Would the following be correct? (All I did was add the COMMENT line.)

```
if(seconds == 3)
{
Serial.println("Entering sleep");
delay(200);
seconds = 0;
enterSleep();
}

// COMMENT: put whatever code here that you want to execute when the
// Arduino is NOT asleep.
}
```

Thanks so much!

Reply

Replies

Donal Morrissey 17 October 2015 at 05:01

Hi Brian,

> "When that happens, does the code start to execute right where the comment is? If that's right"

Execution of the main loop is 'paused' within the sleep_mode() function. When the pin2Interrupt() interrupt is triggered by the pin chance, execution resumes in the sleep_mode() function.

> So, inside loop(), when the condition is met to put the Arduino to sleep, call enterSleep(). Immediately after that, put whatever code I want to execute when the Arduino is NOT asleep (or when it comes out of being asleep), right?

Correct

> what's the point in turning the Arduino on for 3 seconds, just to put it to sleep?

Purely just an example of 'doing stuff for a few seconds', and then going back to sleep until the external interrupt is triggered again.

> Would the following be correct? (All I did was add the COMMENT line.)

Correct

Hope that helps :-)

Best Regards,
Donal

Reply

wayne chamberlain 11 March 2016 at 09:26

Hi guys,

Do we have any examples of using sleep for PIR motion detection - wake on interrupt & and time delay sensors. I would like to wake when motion is detected and send rf messages and when a timer hits 5 mins, get a temp reading and send it.

My sensors are battery powered so i need to sleep as much as possible.

Thanks
Wayne

Reply

Donal Morrissey 12 March 2016 at 04:36

Hi Wayne,
You should be able to use a combination of External Interrupts (Part 2 of this guide) and waking up via watchdog (Part 5).
BR,
Donal

Reply

firaz peerjade 3 June 2016 at 23:32

Thanku So Much Such A Wonder full Blog On Arduino Sleep TY..

Reply

Enter your comment...

Comment as: Select profil ▼

Publish Preview

Newer Post Home Older Post

Simple theme. Powered by Blogger.