PERCOBAAN 3
Low Power

3.1 TUJUAN

Tujuan dari praktikum bab 3 ini adalah praktikan mampu memahami dan mampu membuat program untuk melakukan penghematan daya yang digunakan selama mikrokontroler beroperasi dengan menggunakan berbagai fitur power management yang telah disediakan oleh mikrokontroler.

3.2 PERANGKAT YANG DIGUNAKAN

Software:

- Arduino IDE

Hardware:

- Perangkat lengkap Arduino UNO
- Modul Regulator 5 volt
- Kabel data
- Push button
- Breadboard
- kabel jumper
- Ampere meter / Multimeter

3.3 DASAR TEORI

Sebelum melakukan percobaan pada praktikum ini, ada beberapa teori tentang hardware dan software yang digunakan yang harus dipahami terlebih dahulu oleh praktikan. Beberapa teori tersebut berkaitan dengan power management yang ada di dalam arduino, cara membangunkan arduino dari sleep mode, library program yang digunakan untuk mengaktifkan sleep mode.

Power Management dan *Sleep Modes*

Sleep mode memungkinkan aplikasi untuk mematikan modul yang tidak digunakan di mikrokontroler, sehingga dapat menghemat daya. Mikrokontroler menyediakan berbagai *Sleep mode* yang memungkinkan pengguna menyesuaikan konsumsi daya sesuai dengan aplikasi yang ingin dirancang.

Arduino Uno atau ATmega328p memiliki 6 *Sleep Mode* yang dapat digunakan untuk melakukan *power management*, antara lain: *Idle, ADC Noise Reduction, Power-down, Power-save, Standby,* dan *Extended Standby.* Untuk masuk ke salah satu dari enam jenis mode tidur, kita harus mengatur bit Sleep Enable di Sleep Mode Control

Register (SMCR.SE) menjadi nilai '1', lalu jalankan perintah SLEEP. Selanjutnya, kita bisa memilih jenis mode tidur yang diinginkan dengan mengatur bit Mode Tidur Pilih (SMCR.SM[2:0]) menggunakan perintah SLEEP.

Jika ada interupsi yang aktif saat mikrokontroler dalam mode tidur, maka mikrokontroler akan bangun dan dihentikan selama empat siklus selain waktu start-up. Setelah itu, rutinitas interupsi akan dieksekusi, dan mikrokontroler akan melanjutkan eksekusi instruksi setelah perintah SLEEP. Selama proses bangun dari tidur, isi Register File dan SRAM tidak berubah. Namun, jika terjadi reset saat mikrokontroler dalam mode tidur, maka mikrokontroler akan aktif dan dieksekusi dari Reset Vector.

Idle Mode

Apabila kita menulis bit SM[2:0] dengan nilai '000', maka perintah SLEEP akan membuat mikrokontroler memasuki mode *Idle*. Dalam mode ini, CPU akan dihentikan tetapi komponen seperti SPI, USART, Analog Comparator, 2-wire Serial Interface, *Timer/Counter*, *Watchdog*, dan sistem interupsi masih dapat beroperasi. Mode tidur ini pada dasarnya akan menghentikan clk_{CPU} dan clk_{FLASH}, tetapi membiarkan clock lain tetap berjalan.

Selain mode Idle, ada juga mode siaga yang memungkinkan mikrokontroler untuk bangun dari interupsi yang dipicu eksternal atau internal, seperti Timer Overflow dan USART Transmit Complete. Jika kita tidak memerlukan interupsi dari Komparator Analog, kita bisa mematikan komponen tersebut dengan mengatur bit ACD pada ACSR. Tindakan ini akan mengurangi konsumsi daya saat mikrokontroler dalam mode Idle.

ADC Noise Reduction Mode

Jika kita menulis bit SM[2:0] dengan nilai '001', maka perintah SLEEP akan membuat mikrokontroler memasuki mode *ADC Noise Reduction*. Dalam mode ini, CPU akan dihentikan tetapi komponen seperti ADC, interupsi eksternal, *2-wire Serial Interface address watch*, *Timer/Counter2*, dan *Watchdog* masih dapat beroperasi jika diaktifkan. Mode tidur ini pada dasarnya akan menghentikan $clk_{I/O}$, clk_{CPU}, dan clk_{FLASH}, tetapi membiarkan jam lain tetap berjalan.

Mode tidur ini mengurangi *noise* yang dapat mengganggu performa ADC sehingga memungkinkan pengukuran resolusi yang lebih tinggi. Jika ADC diaktifkan, konversi akan dimulai secara otomatis saat mode ini dimasukkan. Namun, hanya peristiwa tertentu yang dapat membangunkan mikrokontroler dari mode *ADC Noise Reduction*, yaitu:

- *External Reset*
- *Watchdog System Reset*
- *Watchdog Interrupt*

- *Brown-out Reset*
- *2-wire Serial Interface address match*
- *Timer/Counter interrupt*
- *SPM/EEPROM ready interrupt*
- *External level interrupt on INT*
- *Pin change interrupt*

Power-Down Mode

Ketika bit SM[2:0] ditulis ke '010', instruksi SLEEP membuat mikrokontroler masuk ke dalam mode *Power-Down*. Dalam mode ini, Osilator eksternal dihentikan, sedangkan interupsi eksternal, *2-wire Serial Interface address watch*, dan *Watchdog* terus beroperasi (jika diaktifkan). Hanya satu dari peristiwa ini yang dapat membangunkan MCU:

- *External Reset*
- *Watchdog System Reset*
- *Watchdog Interrupt*
- *Brown-out Reset*
- *2-wire Serial Interface address match*
- *External level interrupt on INT*
- *Pin change interrupt*

Power-save Mode

Apabila nilai bit SM[2:0] adalah 011, ketika instruksi SLEEP dijalankan maka mikrokontroler akan masuk ke dalam mode *Power-save*. Mode ini mirip dengan mode *Power-down*, namun terdapat satu perbedaan penting: Jika Timer/Counter2 sedang diaktifkan, maka meskipun dalam mode tidur, timer tersebut akan tetap berjalan. Perangkat dapat bangun dari mode tidur ketika terjadi overflow atau terjadi perbandingan output dari Timer/Counter2 jika bit yang diatur dalam TIMSK2 diaktifkan dan bit Global Interrupt Enable diatur dalam SREG.

Jika Timer/Counter2 tidak digunakan, disarankan untuk memilih mode Power-down daripada mode Hemat Daya. Timer/Counter2 dapat dijalankan baik secara sinkron maupun asinkron dalam mode Hemat Daya. Jika Timer/Counter2 tidak menggunakan clock asinkron, Timer/Counter tersebut akan berhenti saat mikrokontroler sedang tidur. Jika Timer/Counter2 tidak menggunakan clock sinkron, sumber clock pada perangkat juga akan berhenti saat tidur. Walaupun clock sinkron berjalan dalam mode Hemat Daya, clock tersebut hanya tersedia untuk Timer/Counter2.

Standby Mode

Ketika bit SM[2:0] ditulis ke '110' dan opsi clock kristal/resonator eksternal dipilih, instruksi SLEEP membuat mikrokontroler masuk ke mode *Standby*. Mode ini identik dengan power-Down dengan pengecualian bahwa Oscillator tetap berjalan. Dari mode *Standby*, perangkat bangun dalam enam siklus clock.

Extended Standby Mode

Ketika bit SM[2:0] ditulis ke '111' dan opsi jam kristal/resonator eksternal dipilih, instruksi SLEEP membuat mikrokontroler memasuki mode *Extended Standby*. Mode ini identik dengan Power-Save mode dengan pengecualian bahwa Oscillator tetap berjalan. Dari mode *Extended Standby*, perangkat bangun dalam enam siklus jam.

Low-Power Library

Untuk memudahkan implementasi low power di dalam Arduino dapat digunakan library low-power yang dapat diakses di link github ini dan di import ke dalam Arduino IDE. dengan menggunakan library ini, kita dapat menuliskan fungsi dengan lebih sederhana.

3.4 PROSEDUR PERCOBAAN

PERCOBAAN 3.1: Konsumsi arus listrik untuk program sederhana

Persiapan

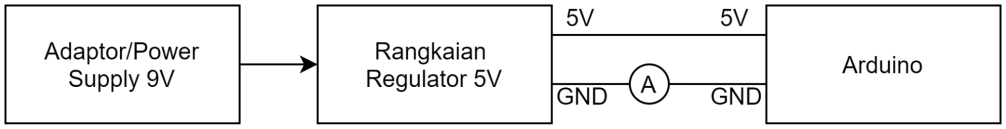
- Tuliskan program untuk menyalakan LED yang ada di Board Arduino Uno seperti ditunjukkan dalam Program 3.1 di bawah ini. Setelah itu *compile* dan *upload* ke dalam board Arduino Uno yang telah dipersiapkan.

Program 3.1

```
1  void setup() {
2    pinMode(LED_BUILTIN, OUTPUT);
3  }
4
5  void loop() {
6    digitalWrite(LED_BUILTIN, HIGH);
7  }
```

- Setalah program berhasil dikompilasi dan diupload, lepas kabel data yang menghubungkan Arduino Uno dengan komputer dan ukur arus yang mengalir ke

Arduino Uno menggunakan skenario seperti ditunjukkan diagram blok dalam Gambar 3.1.



Gambar 3.1. Diagram pengukuran arus yang mengalir dalam arduino

ANALISIS 1.1

1. Mengapa pengukuran arus dilakukan langsung di pin 5V dan GND di arduino dan tidak di bagian jack input power supply arduino?

PERCOBAAN 3.2: Konsumsi arus listrik untuk berbagai mode *Power Management*

- Tuliskan program untuk menyalakan LED yang ada di Board Arduino Uno seperti ditunjukkan dalam Program 3.2 di bawah ini. Setelah itu *compile* dan *upload* ke dalam board Arduino Uno yang telah dipersiapkan.

Program 3.2

```
 1  #include <LowPower.h>
 2
 3  void setup() {
 4    Serial.begin(9600);
 5    pinMode(LED_BUILTIN, OUTPUT);
 6  }
 7
 8  void loop() {
 9    digitalWrite(LED_BUILTIN, HIGH);
10    delay(5000);
11    digitalWrite(LED_BUILTIN, LOW);
12    LowPower.powerDown(SLEEP_8S, ADC_OFF, BOD_OFF);
13  }
```

- Setelah program berhasil dikompilasi dan diupload, lepas kabel data yang menghubungkan Arduino Uno dengan komputer dan ukur arus yang mengalir ke Arduino Uno menggunakan skenario seperti ditunjukkan diagram blok dalam Gambar 3.1.

ANALISIS 3.2

1. Apakah nilai yang ditampilkan oleh amperemeter selalu tetap atau berubah-ubah? Mengapa demikian?
2. Coba ganti mode *Power Management* dari ***Power-Down Mode*** ke mode *Power Management* yang lain (***Idle Mode, ADC Noise Reduction Mode, Power-save Mode, Standby Mode, dan Extended Standby Mode*****)** dan ukur arusnya! Bagaimana perbandingannya?

PERCOBAAN 3.3: Bangun dari mode tidur

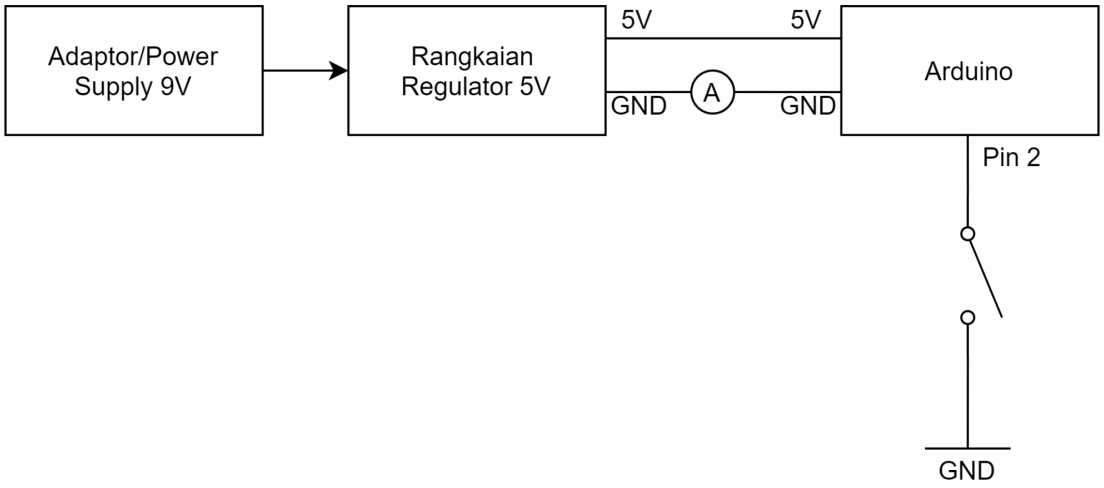
- Tuliskan program untuk menyalakan LED yang ada di Board Arduino Uno seperti ditunjukkan dalam Program 3.3 di bawah ini. Setelah itu *compile* dan *upload* ke dalam board Arduino Uno yang telah dipersiapkan.

Program 3.3

```
1  #include <LowPower.h>
2  const int ledPin = 13;
3  const int interruptPin = 2;
4
5  void setup() {
6      pinMode(ledPin, OUTPUT);
7      pinMode(interruptPin, INPUT_PULLUP);
8  }
9
10 void flash(){
11     digitalWrite(ledPin, HIGH);
12     delay(500);
13     digitalWrite(ledPin, LOW);
14     delay(500);
15     digitalWrite(ledPin, HIGH);
16     delay(500);
17     digitalWrite(ledPin, LOW);
18     delay(500);
19     digitalWrite(ledPin, HIGH);
20     delay(500);
21     digitalWrite(ledPin, LOW);
22     delay(500);
23 }
```

```
24
25  void loop() {
26      attachInterrupt(digitalPinToInterrupt(interruptPin), wakeUp, FALLING);
27      LowPower.powerDown(SLEEP_FOREVER, ADC_OFF, BOD_OFF);
28      detachInterrupt(0);
29      flash();
30  }
31
32  void wakeUp() {
33
34  }
```

- Setelah program berhasil dikompilasi dan diupload, lepas kabel data yang menghubungkan Arduino Uno dengan komputer dan ukur arus yang mengalir ke Arduino Uno menggunakan skenario seperti ditunjukkan diagram blok dalam Gambar 3.2.



Gambar 3.2. Diagram pengukuran arus yang mengalir dalam arduino dan tombol untuk interrupt eksternal

ANALISIS 3.3

1. Tekan push button dan jelaskan apa yang terjadi?
2. Berapa arus yang terukur di amperemeter pada saat tombol tidak ditekan dan pada saat ditekan?

3.5 TUGAS

1. Buatlah program untuk membuat arduino masuk ke dalam salah satu mode penghematan daya dan bangun ketika ada interrupt internal dari timer.
2. Adakah batasan maksimal lama arduino masuk ke dalam mode penghematan daya?
3. Berikan contoh kasus penggunaan mode penghematan daya dalam embedded system dan jelaskan desainnya!

3.6 KESIMPULAN

Berdasarkan percobaan yang telah dilakukan, maka tuliskan kesimpulan percobaan di tempat yang telah disediakan di bawah ini: