Embedded Systems

Some Power Considerations

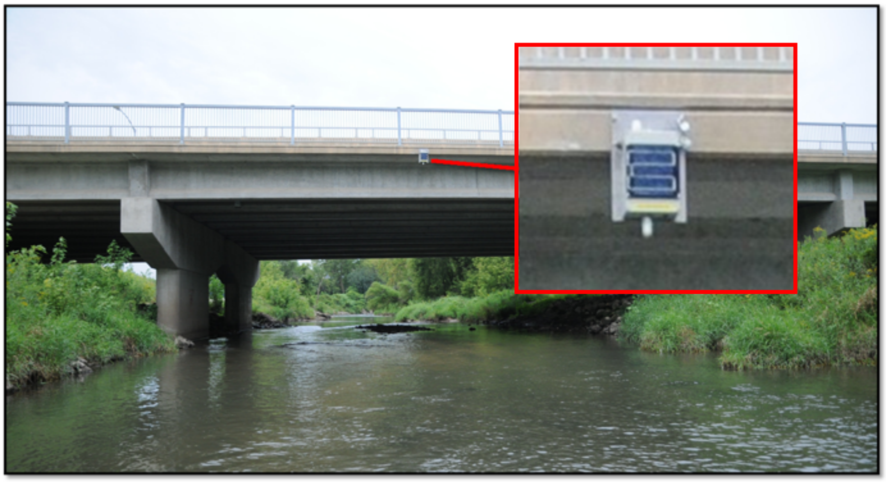
Iowa City – 06/15/08

An Embedded System for Water Level Measurement

An Embedded System

Creek or Small River

An Embedded System

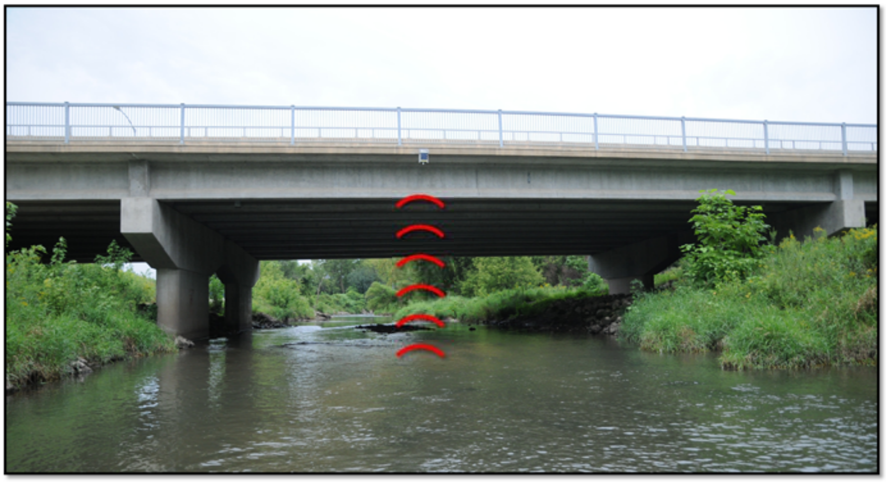


Attach Distance Sensor

An Embedded System

Wake up, Measure Distance to Surface

An Embedded System

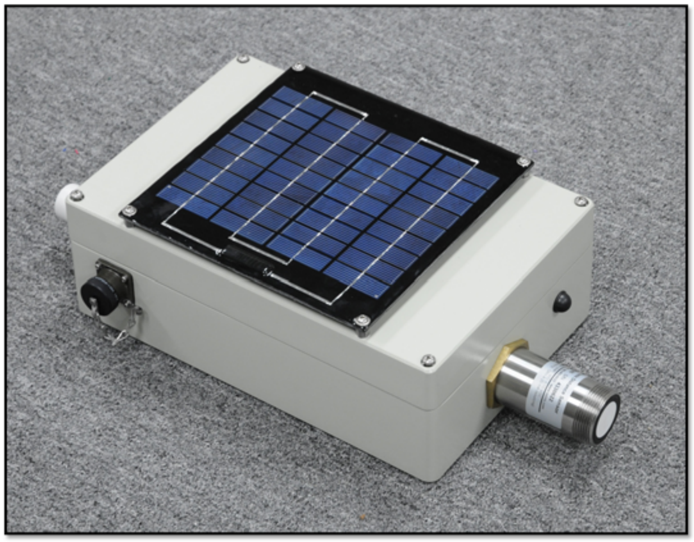


Wake up, Measure Distance to Surface

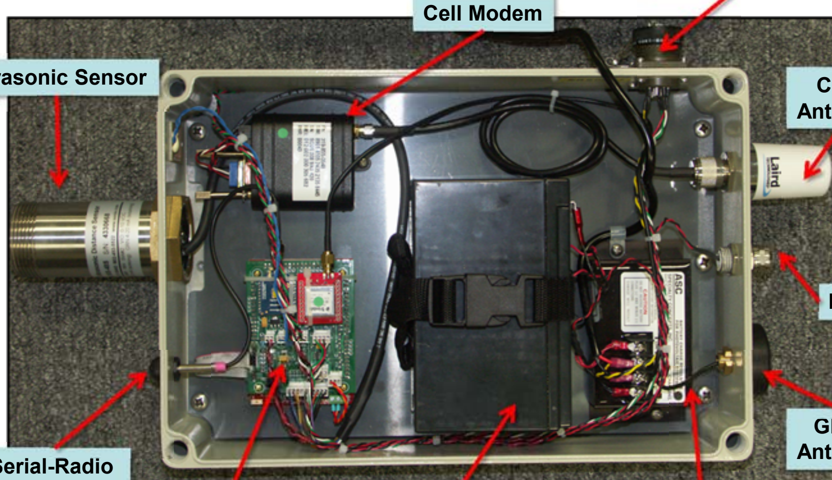
An Embedded System

Relay Data Back to Servers on Internet

An Embedded System



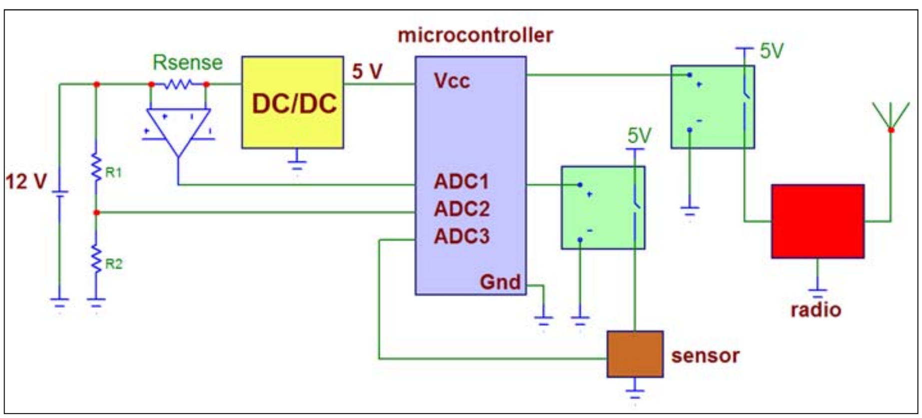
Electronics



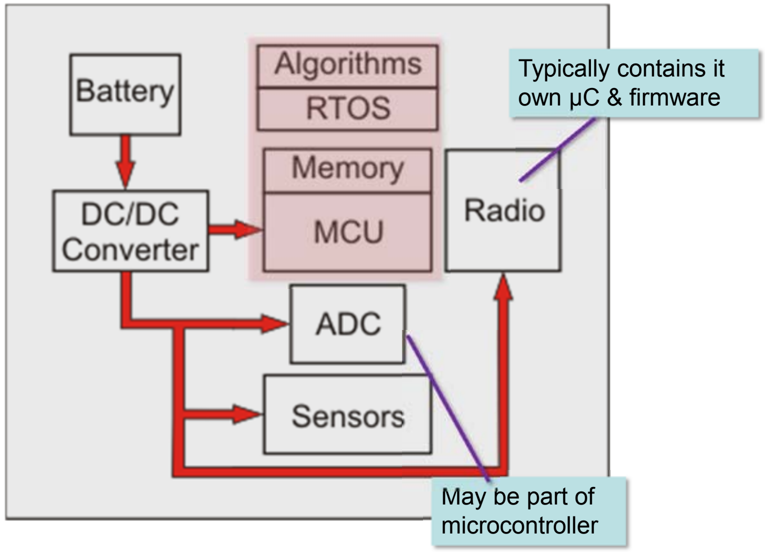
Power – Design Constraints

- Battery operated devices
  - Battery life (runtime)
  - Battery size
  - Weight
  - µC performance
  - Costs
  - …
- High performance applications (e.g., Digital Signal Processing)
  - Cooling
  - Costs (power supply, …)
  - …

Simplified System Diagram



System Components & Power Consumption



CMOS Logic – Power Dissipation

The following is a simple model for (dynamic) average power dissipated by a CMOS-based system

(e.g., use simpler µC) (lower voltage)

Power dissipated

Capacitance

Voltage

$$P = \alpha C V^2 f$$

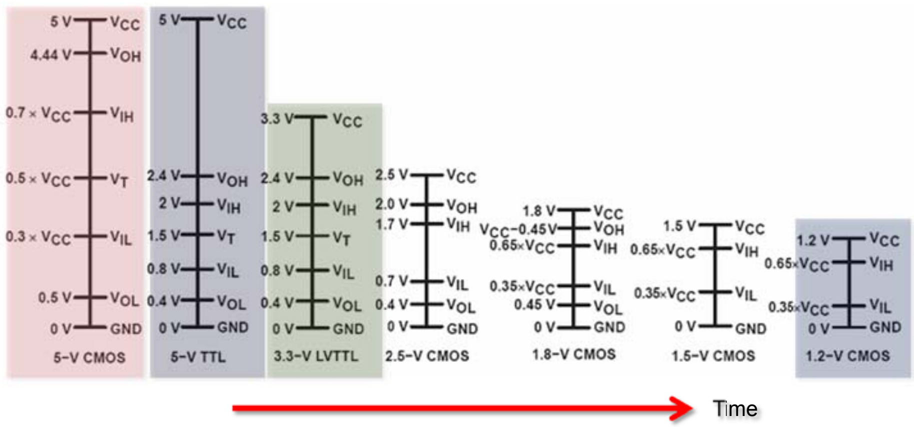
Switching frequency

(lower clock)

Number of active gates

(turn off unused systems)

Logic Families - Signal Voltage Levels



Reducing power consumption is the main reason for a push towards lower voltage levels

Sleep Modes

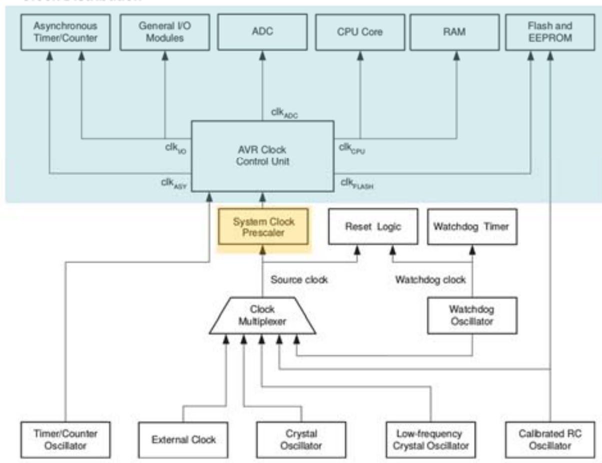
- **µCs typical have several low power modes**
  - **Active, idle, shutdown, deep sleep modes**
    - Various MCU subsystems are turned off
    - Different clock speeds
  - **For some MCUs, in deep sleep modes, the power consumption can almost be negligible**
  - **Takes longer to wake up from a deep sleep mode than just a "nap"**
  - **Utilizing sleep modes → software design**

Microcontroller Unit – ATmega88

Clock Systems and their Distribution

Figure 9-1 presents the principal clock systems in the AVR and their distribution. All of the clocks need not be active at a given time. In order to reduce power consumption, the clocks to modules not being used can be halted by using different sleep modes, as described in "Power Management and Sleep Modes" on page 39. The clock systems are detailed below.

Figure 9-1. Clock Distribution



Radio

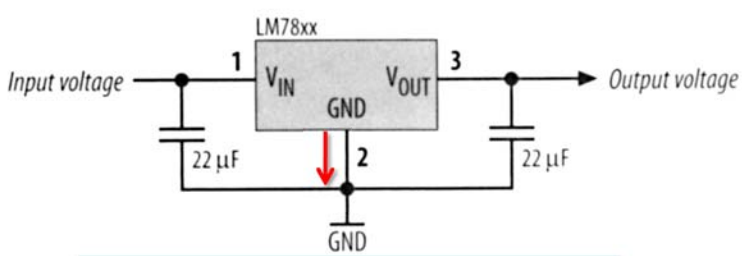
- Radio typically contains an embedded controller that provides many functions
  - Error detection and correction in hardware/firmware, …
- Several modes
  - Receive only, transmit + receive, idle, etc.
- In general, transmit requires most power
  - Use RSSI to adjust transmit power
- Carefully consider/analyze radio spec and modes
  - For example, a mode change can consume a lot of power (shutdown vs. idle mode)

Sensors & ADCs

- **Many sensors are inherently analog, but some sensors have digital interfaces (provided by embedded controllers)**
- **Can have Power save modes**
  - Wake up times need to be considered
- **Analog-Digital Converters (ADC)**
  - Can be a major power consumer
  - More bits and higher conversion rates requires more power
  - → Don’t over-specify

Power Supplies – Linear Voltage Regulator

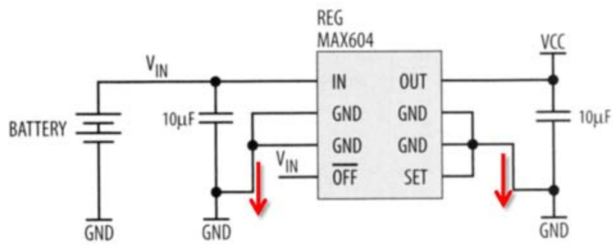
Linear Three-Terminal Regulator



Quiescent current for LM78xx regulators are large, and can waste power
→ not suitable for battery-operated equipment

Typical Quiescent Current: 5 mA (max: 8 mA!)

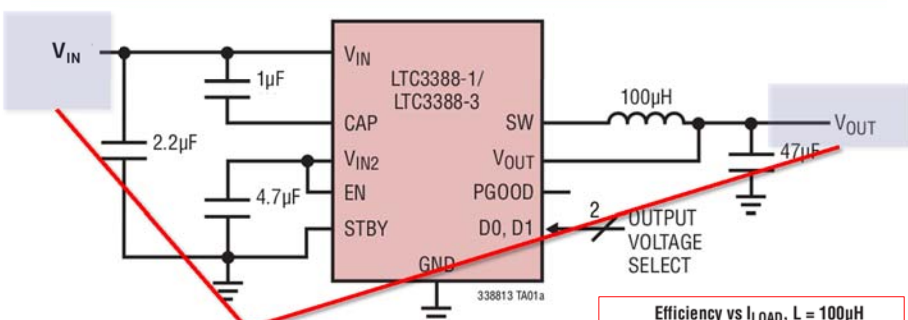
Power Supplies – Linear Voltage Regulator



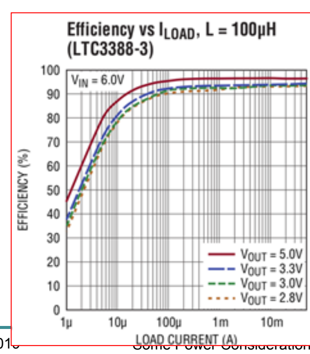
Quiescent current for MAX604 a few micro-amps, thus much more suitable for battery-operated equipment.

Typical Quiescent Current: 15 µA

Power Supplies - Switching Voltage Regulator



Very wide input range, efficient conversion to output voltage



Typical Quiescent Current: <1 µA

Note that some switching regulators **can** step up voltages

Advantage: very efficient
Disadvantage: noise on V_{OUT}

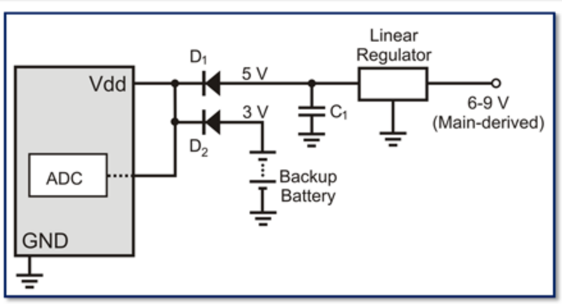
Example – Voltage Regulators & Power Efficiency

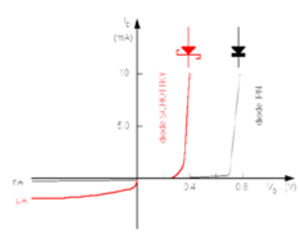
- **An engineer wants to operate an AVR µC with a 9 V battery. The µC requires 15 mA at 5 V.**
- **Compare the following voltage regulators regarding their power efficiency (η = Pout / Pin * 100%).**
  - **a) LM7805**
  - **b) MAX604**
  - **c) LTC3388-3**

→ Solution on whiteboard

Power Supplies - Backup Battery

- With main power, D_1 is forward biased (→ turns on) and powers the controller → D_2 is off.
- Without main power, D_2 is forward biased (→ turns on) and powers the controller.



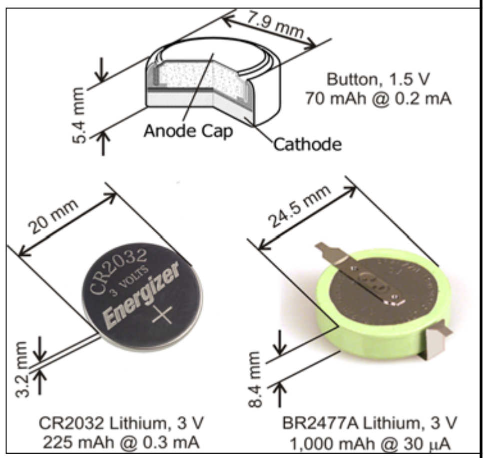


Question: what type of diodes should D1, D2, be, and why?

Answer: Schottky diodes, because they have lower turn-on voltages than Si diodes. Thus, these diodes dissipate less power.

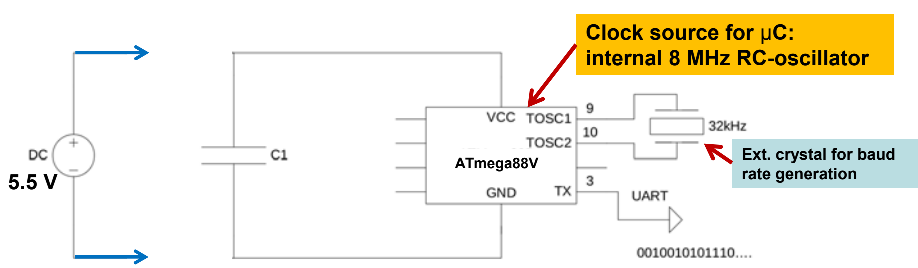
Power Supplies - Coin & Button Cells

- Capacities: 50–300 mAh
- Designed for 3–200 μA or few mA pulsed load
- Enough to power CMOS
- Used for
  - RAM backup
  - powering RTC
- Lithium chemistry
  - single cell
  - very long service



Example – AVR Power Optimization

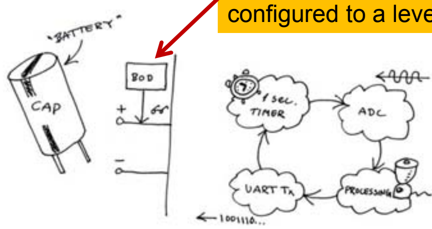
- Based on Application Node AVR4013 “Picopower basics”
- Demonstration of how changes in "software" can extend battery life of an embedded system
  - Change of µC HW configuration by SW
  - Change of code
- HW setup uses a capacitor as power supply:



Example – AVR Power Optimization

- **The code simulates a sensor device that transmits data**
- **After the initialization (which will differ with each power optimization approach), the µC will repeat each second:**
  - **ADC conversion**
  - **Simulated 1000 cycles of processing**
  - **Convert number in to ASCII string**
  - **Send data over UART "More oomph to your amps, picoPower! [iter. #]"**

At 8 MHz the brown out detection (BOD) is configured to a level of 2.7V → µC will stop!



Idea: longer running time → higher power efficiency

Example – AVR Power Optimization

- **The following approaches will be investigated:**

  1. **No optimization**
  2. **Enable pull-ups on unused I/O pins and disable modules not used**
  3. **Pre-scale clock from 8Mhz to 2MHz**
  4. **Use power-save sleep mode while waiting for next transmit**
  5. **Use a higher UART baud rate**

1) No optimization

- 8 MHz internal RC oscillator
- Baud rate generator set to 19.2k baud
- Waiting for the next transmit:
  - poll the TOV2 bit in Timer/Counter 2 Interrupt Flag Register to check if 1 second has elapsed since the last transmit
  - → µC is active all the time

- Runtime: 6 s → defines the baseline

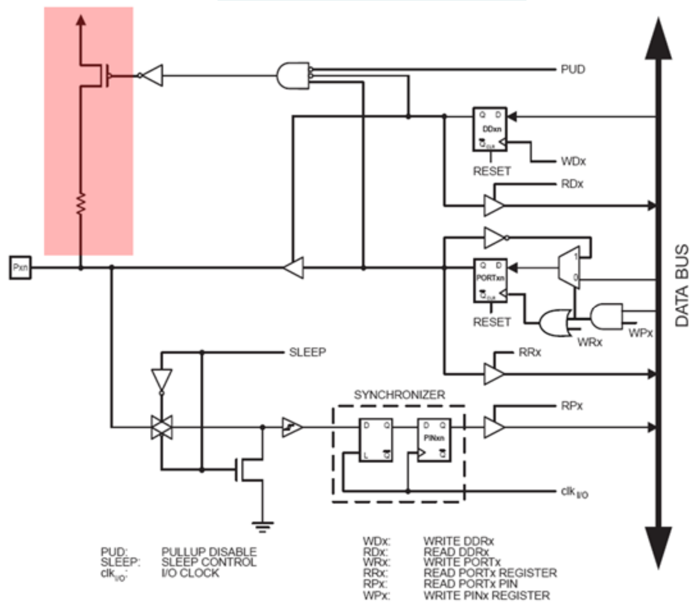
2) Enable pull-ups on unused I/O pins and disable modules not used

- Enable pull-ups on unused I/O pins to get a defined logical level and avoid unnecessary switching
  - Example for I/O port B:
  - DDRB = 0x00; // Set direction to input on all pins
  - PORTB = 0xFF; // Enable pull-ups on pins

- Another power-saving feature is to disable unused on-chip modules in the PRR Power Reduction Register
  - We are not using the TWI, Timer/Counter 0, Timer/Counter 1, and Serial Peripheral Interface (SPI)

- Runtime is increased to 9 s → +50%

ATmega88 – I/O



Power Reduction Register

10.11.3 PRR – Power Reduction Register

Bit	7	6	5	4	3	2	1	0	
(0x64)	PRTWI	PRTIM2	PRTIM0	–	PRTIM1	PRSPI	PRUSART0	PRADC	PRR
Read/Write	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – PRTWI: Power Reduction TWI**

  Writing a logic one to this bit shuts down the TWI by stopping the clock to the module. When waking up the TWI again, the TWI should be re initialized to ensure proper operation.

- **Bit 6 – PRTIM2: Power Reduction Timer/Counter2**

  Writing a logic one to this bit shuts down the Timer/Counter2 module in synchronous mode (AS2 is 0). When the Timer/Counter2 is enabled, operation will continue like before the shutdown.

- **Bit 5 – PRTIM0: Power Reduction Timer/Counter0**

  Writing a logic one to this bit shuts down the Timer/Counter0 module. When the Timer/Counter0 is enabled, operation will continue like before the shutdown.

- **Bit 4 – Reserved**

  This bit is reserved in Atmel® ATmega48PA/88PA/168PA and will always read as zero.

- **Bit 3 – PRTIM1: Power Reduction Timer/Counter1**

  Writing a logic one to this bit shuts down the Timer/Counter1 module. When the Timer/Counter1 is enabled, operation will continue like before the shutdown.

- **Bit 2 – PRSPI: Power Reduction Serial Peripheral Interface**

  If using debugWIRE on-chip debug system, this bit should not be written to one.

  Writing a logic one to this bit shuts down the serial peripheral interface by stopping the clock to the module. When waking up the SPI again, the SPI should be re initialized to ensure proper operation.

- **Bit 1 – PRUSART0: Power Reduction USART0**

  Writing a logic one to this bit shuts down the USART by stopping the clock to the module. When waking up the USART again, the USART should be re initialized to ensure proper operation.

- **Bit 0 – PRADC: Power Reduction ADC**

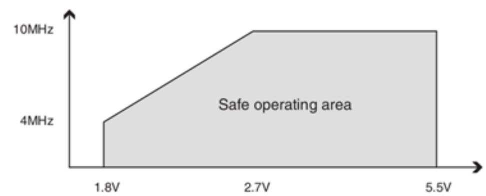
  Writing a logic one to this bit shuts down the ADC. The ADC must be disabled before shut down. The analog comparator cannot use the ADC input MUX when the ADC is shut down.

Not used →
can be turned off

3) Pre-scale clock from 8MHz to 2MHz

- **Reduce system clock from 8 MHz to 2 MHz:**
  - will lower overall power consumption and
  - allow us to change the BOD setting from 2.7 V to 1.8V → better utilization of "battery"
- **Implemented by pre-scaling the RC oscillator output by 4 (→ CLKPR … Clock Prescale Register)**

Figure 29-1. Maximum frequency vs. V_{CC}, ATmega48V/88V/168V.



- **Runtime is now increased to 40 s → +566%**

3) Pre-scale clock from 8MHz to 2MHz

- **Clock division factors of 1, 2, 4, 8, 16, 32, 64, 128, and 256 can be selected with [CLKPS3:CLKPS0] bits**

  → decrease the system clock frequency "on the fly" (→ power consumption) when the requirement for processing power is low

9.12.2 CLKPR – Clock Prescale Register

Bit	7	6	5	4	3	2	1	0	
(0x61)	CLKPCE	–	–	–	CLKPS3	CLKPS2	CLKPS1	CLKPS0	CLKPR
Read/Write	R/W	R	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	See Bit Description				

- **Bit 7 – CLKPCE: Clock Prescaler Change Enable**

The CLKPCE bit must be written to logic one to enable change of the CLKPS bits. The CLKPCE bit is only updated when the other bits in CLKPR are simultaneously written to zero. CLKPCE is cleared by hardware four cycles after it is written or when CLKPS bits are written. Rewriting the CLKPCE bit within this time-out period does neither extend the time-out period, nor clear the CLKPCE bit.

- **Bits 3:0 – CLKPS[3:0]: Clock Prescaler Select Bits 3 - 0**

These bits define the division factor between the selected clock source and the internal system clock. These bits can be written run-time to vary the clock frequency to suit the application requirements. As the divider divides the master clock input to the MCU, the speed of all synchronous peripherals is reduced when a division factor is used. The division factors are given in Table 9-17 on page 34.

4) Use power-save sleep mode while waiting for next transmit

- While waiting for the next 1s-cycle start, it's a good idea to put the device into sleep mode to further reduce power consumption
  - Use power-save sleep mode and
  - Timer/Counter2 interrupt as "wakeup" source
- Brown out detection is not required in the power-save sleep mode (no data can be corrupted)
  - Disable the BOD while in sleep mode
  - It will be automatically re-enabled on wakeup from sleep (see

- Runtime is now 198 s → +3,200%

4) Use power-save sleep mode while waiting for next transmit

Table 10-1. Active Clock Domains and Wake-up Sources in the Different Sleep Modes.

	Active Clock Domains					Oscillators		Wake-up Sources							
Sleep Mode	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 I/O	Software BOD Disable
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

5) Use a higher baud rate

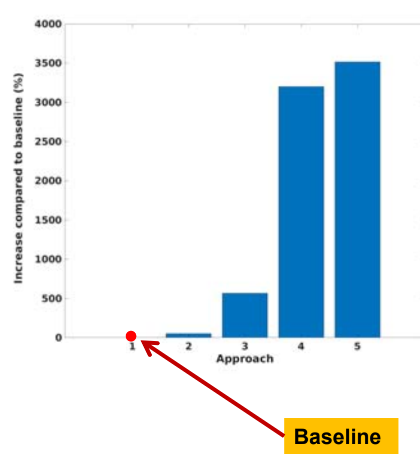
- Idea: shorten time spent in active mode
  - the µC is active while waiting for the UART transmission to end
  - select a faster baud rate: 19.2 k baud → 115.2 k baud
  - → more sleep

- Runtime is now increased to 217 s → +3,517%

AVR Power Optimization – Summary

- Approaches investigated:

1. No optimization (baseline)
2. Enable pull-ups on unused I/O pins and disable modules not used
3. Pre-scale clock from 8Mhz to 2MHz + POD adjustment
4. Use power-save sleep mode while waiting for next transmit
5. Use a higher UART baud rate



Review - Questions

Question 1. Consider a battery-operated consumer electronics device that uses an embedded microcontroller that will accept a 2.7–5.5 V power. The device can also be powered from a power supply that plugs into a mains outlet. Consumers' expectation is that the switchover between battery and mains power is transparent. That is, the instant a user inserts the power supply connector, the device switches to the power supply, and the instant the user unplugs the device, it switched to battery power.

Draw a block diagram/schematic that shows how to implement this functionality using diode(s), linear regulator(s), battery, etc. Explain how the circuit works. Provide as much details as you can. For example, specify the type of diodes, indicate voltages on the diagram, include critical capacitors, and so on. You can assume that unregulated 9 V dc power is available.

Question 2. List and the briefly explain five design considerations that one can employ to reduce the power consumption in an AVR-based embedded system.

... EOL