

# Embedded system

[http://git.eti.pg.gda.pl/intel-grant/pliki/esa/Embedded\\_Systems\\_Architecture\\_P2.pdf](http://git.eti.pg.gda.pl/intel-grant/pliki/esa/Embedded_Systems_Architecture_P2.pdf)

# Software and hardware minimizing energy consumption

Designers have at least three options available that can be used to minimize system maintenance and energy consumption:

- Hardware energy optimization
- Software energy optimization
- Energy harvesting

- Minimizing power consumption – in practice – the action in order to reach a compromise between the energy consumed by the system, and its performance.

# Software and hardware minimizing energy consumption

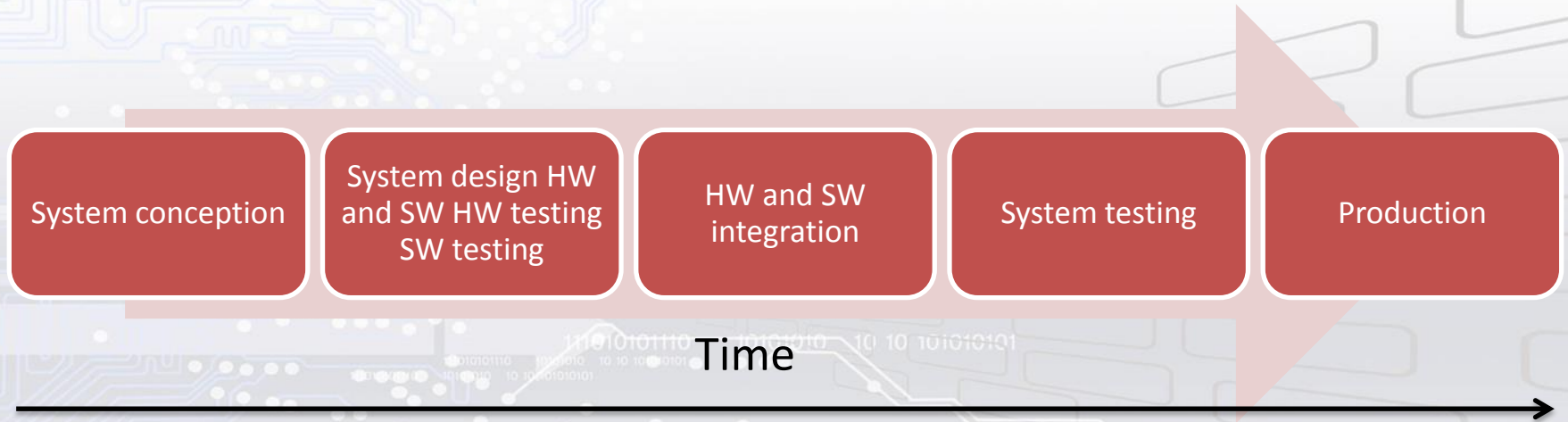
In many applications, it is required at the same time low power consumption and high performance, which is contradictory, forcing designers to seek compromise.

Energy demand should be considered at the concept stage of the system.

In many cases wrong decisions will have irreversible effects

- System excessive energy consumption
- System performance below minimum acceptable level.

# Software and hardware minimizing energy consumption



# Hardware minimizing energy consumption - $\mu$ Controllers

Hardware minimizing energy consumption:

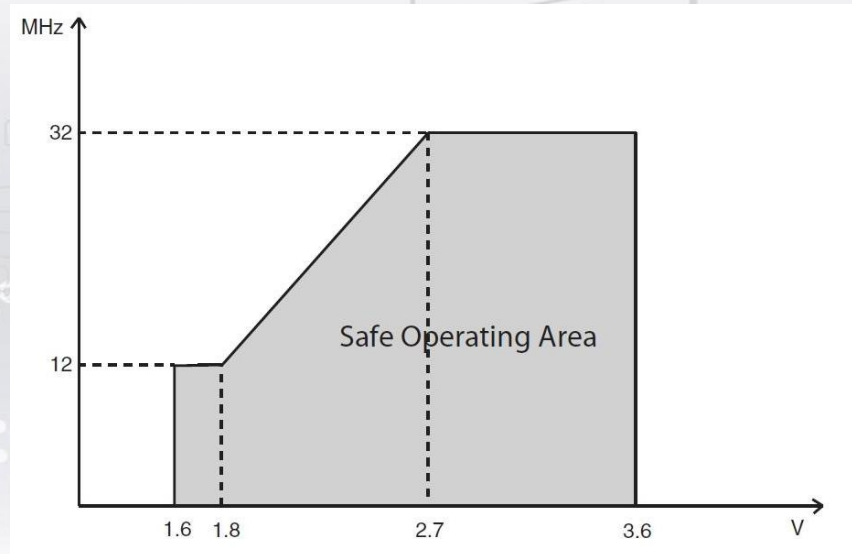
- Work with a supply voltage as low as possible - power consumption is proportional to the square of the supply voltage.

Lower supply voltage and lower system clock frequency gives lower system performance

- Work with a system clock frequency as low as possible - power consumption is proportional to the system clock frequency.

Lower system clock frequency and lower system performance gives extended processing time.

Example - Frequency versus supply voltage – ATxmega family.





# Software minimizing energy consumption - $\mu$ Controllers

How to save energy?

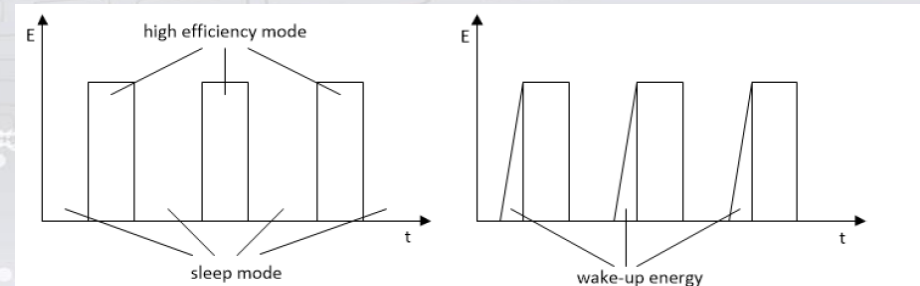
By using of available power-saving modes, as follows:

- $\mu$ Controller should spend as much time as possible in sleep modes.
- In Active mode  $\mu$ Controller should work with maximum performance.
- Active mode should be as short as possible. It is a well known method.

Effectiveness of this method depends on:

- how the designer can deal with the effective use of power-saving modes and shorten the duration of the application with maximum performance.

- Estimating the energy demand of the system using power-saving modes designer has to take into account in the energy balance energy needed to wake up CPU (wake-up energy).



# Software minimizing energy consumption - $\mu$ Controllers

TI MSP430 operating modes

Operating Mode	Description	CPU (MCLK)	SMCLK	ACLK	RAM Retention	BOR	Self Wakeup	Interrupt Sources
<b>Active</b>	CPU, all clocks and peripherals available.	•	•	•	•	•		Timers, ADC, DMA, UART, WDT, I/O, comparator, Ext. Interrupt, RTC, serial communications, other peripherals
<b>LPM0</b>	CPU is shutdown, peripheral clocks available.		•	•	•	•	•	Timers, ADC, DMA, UART, WDT, I/O, comparator, Ext. Interrupt, RTC, serial communications, other peripherals
<b>LPM1</b>	CPU is shutdown, peripheral clocks available. DCO is disabled and the DC generator can be disabled.		•	•	•	•	•	Timers, ADC, DMA, UART, WDT, I/O, comparator, Ext. Interrupt, RTC, serial communications, other peripherals
<b>LPM2</b>	CPU is shutdown, only one peripheral clock available. DC generator is enabled.			•	•	•	•	Timers, ADC, DMA, UART, WDT, I/O, comparator, Ext. Interrupt, RTC, serial communications, other peripherals
<b>LPM3</b>	CPU is shutdown, only one peripheral clock available. DC generator is disabled.			•	•	•	•	Timers, ADC, DMA, UART, WDT, I/O, comparator, Ext. Interrupt, RTC, serial communications, other peripherals
<b>LPM3.5</b>	No RAM retention, RTC can be enabled. (MSP430F5xx generation only)					•	•	Ext. Interrupt, RTC
<b>LPM4</b>	CPU is shutdown and all clocks disabled.				•	•		Ext. Interrupt
<b>LPM4.5</b>	No RAM retention, RTC disabled. (MSP430F5xx generation only)					•		Ext. Interrupt

# Software minimizing energy consumption - $\mu$ Controllers

ATMEL XMEGA operating modes

	Active clock domain			Oscillators		Wake-up sources			
	CPU clock	Peripheral clock	RTC clock	System clock source	RTC clock source	Asynchronous Port Interrupts	TWI Address match interrupts	Real Time Clock Interrupts	All interrupts
<b>Sleep modes</b>									
Idle		X	X	X	X	X	X	X	X
Power-down						X	X		
Power-save			X		X	X	X	X	
Standby				X		X	X		
Extended Standby			X	X	X	X	X	X	

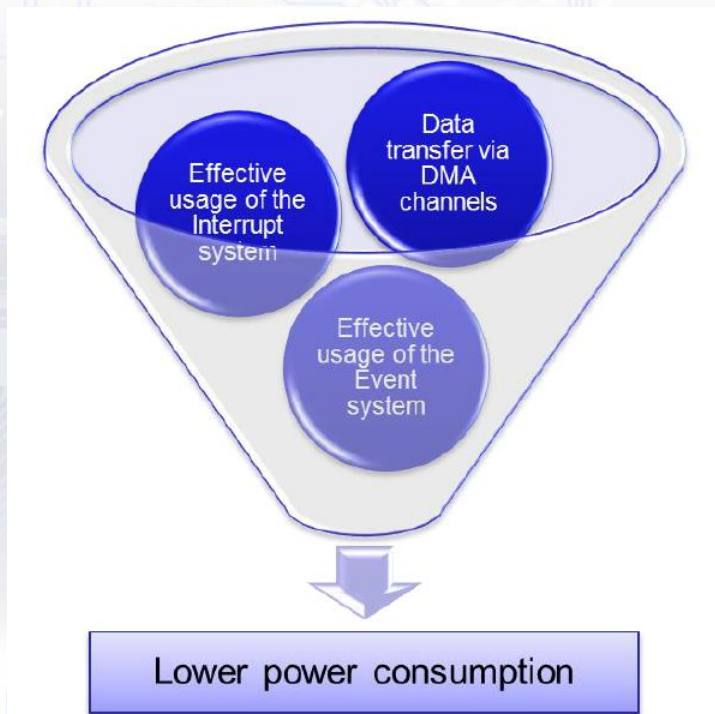


# Software minimizing energy consumption - $\mu$ Controllers

STMicroelectronics STM32L1 operating modes

Mode	Power consumption	CPU	Flash / EEPROM	RAM	DMA & Periph	Clock	LCD	RTC
Sleep	100 $\mu$ A/MHz (R1)	NO	On	On	Active	Any		Enable
Low power run	10,4 $\mu$ A Flash OFF, 32 kHz	YES	On or Off	On	Active	MSI		Enable
Low power sleep	5,1 $\mu$ A (periph. Off)	NO	Off	On	Active	MSI		Enable
Stop with RTC	1,3 $\mu$ A (1,8V)	NO	Off	On	Stopped	LSE, LSI		Enable
Stop	500nA	NO	Off	On	Stopped	LSE, LSI	Off	Off
Standby with RTC	1 $\mu$ A (1,8V)	Off	Off	Off	Off	LSE, LSI	Off	Off
Standby	270nA	Off	Off	Off	Off	LSE, LSI	Off	Off

# Software minimizing energy consumption - $\mu$ Controllers



## Other exemplary methods:

- Repeated explicit substitutions – reduce the number of jump instructions;
- Replacement of a single complex instruction set of simple instructions e.g. modulo instruction (may take from 2 to 12 clock cycles) – may be replaced by a simple control of the variable state.

# Software minimizing energy consumption - $\mu$ Controllers

Software compilation with code optimization:

- optimization option – max. speed

Cost:

- larger code size;

Benefits:

- shorter work time with max. performance;
- smaller number of instructions to execute.

Compilation option	Code size	Energy consumed within 1 sec
Max. speed	5456b (+38%)	41 $\mu$ J
Min. size	3944b	77,6 $\mu$ J (+92%)

# Software minimizing energy consumption - DSP

- Frequency scaling.
- Supply voltage scaling.
- Usage of the power-saving modes:
  - Power STANDBY mode;
  - Power SLEEP mode;
  - Power DEEPSLEEP mode.
- Proper use of the above mentioned options.
- Effective usage of the Interrupt system;
- Data transfer via DMA channels;
- Effective usage of the Event system;
- Code optimization;
- FFT hardware acceleration

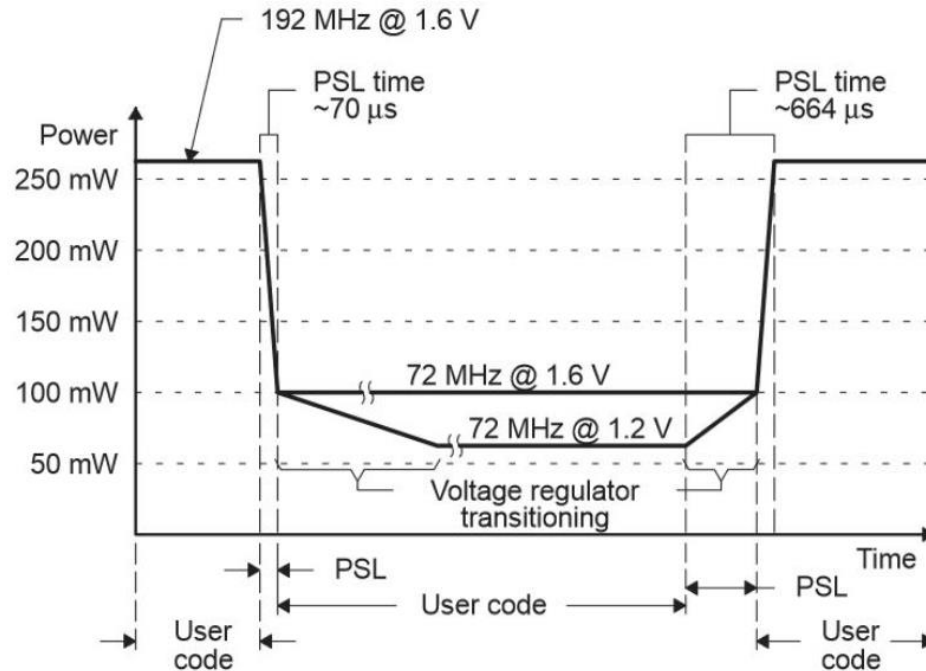
# TI DSP processors C5XXX

## Power Scaling Library:

- Scaling operation to scale frequency and voltage or frequency only;
- Query operations that return current frequency and voltage settings;
- Query operations that return available frequencies settings and the required voltage settings for those frequencies
- Query operation that returns the latencies associated with a scaling operation;
- Callbacks to user code before and after scaling operations. These callbacks will enable users to perform any necessary peripheral modifications that may be required as a result of the upcoming/just completed scaling operation.



# TI DSP processors C5XXX



# TI DSP processors C5XXX

The DSP/BIOS Power Manager, PWRM – module that lets you reduce the power consumption of your application. The PWRM module provides the following capabilities:

- Resource Tracking.
- Scaling Voltage and Frequency.
- Using Sleep Modes.
- Coordinating Sleep and Scaling.

'C55x PWRM additional capabilities:

- Idling Clock Domains.
- Saving Power at Boot Time.
- DSP Device Initialization.

# Analog Devices DSP processors

Dynamic Power Management Controller – consists of two main components which allows flexibility in managing power dissipation on the processor:

- Operating Modes;
- Peripheral Clocking;
- Dynamic Voltage Control.

Operating modes:

- Full-on mode;
- Active mode;
- Sleep mode;
- Deep Sleep mode.

# INTEL processors

Various power states:

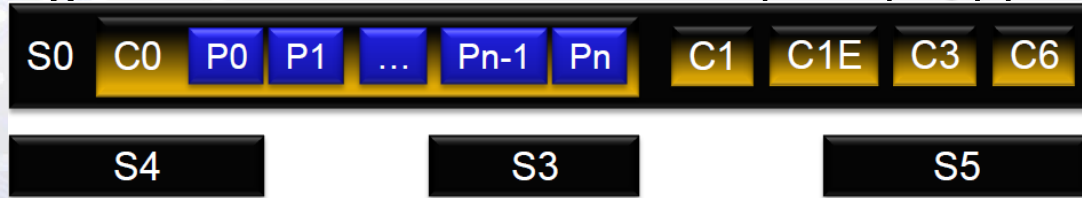
- System Sleep States (S-States)
- Processor Power States (C-States)
- Processor Performance Power States (P-States)

Power Management Technologies:

- C-State Auto Demotion
- Intel® Turbo Boost Technology
- Virtualization Power Management
- Memory Power Management

# INTEL processors

## Advanced Configuration and Power Interface (ACPI) Supported States



P-States are a sub-state of the C0 state

- Offer reduced power consumption while the processor is executing code.

C-States are a sub-state of the S0 state

- Offer reduced power consumption while the system is fully on.



# INTEL processors

System Sleeping Power States (S-States) – S-States allow the system to save a power when not being used.

- S0 – Full on: Processor Operating. Individual devices may be shut down or be placed into lower power states to save power.
- S3 – Suspend-to-RAM: System context maintained in DRAM. Power shut off to non-critical circuits. Memory is retained and refreshes continue. All clocks are stopped except RTC clock.
- S4 – Suspend-to-Disk: System context maintained on disk. All power is then shut off except for that needed to resume.
- S5 – Soft off: System context not maintained. All power is shut off except for that needed to restart. A full boot is required when waking.

# INTEL processors

S-States reduce power consumption significantly. Latency back to S0 is in the order of seconds. Wake event can be triggered by:

- a motion sensor;
- remotely using Intel® Active Management Technology (Intel® AMT);
- a timer or another activity (LAN or GPIO) pins.

# INTEL processors

## Processor Power and Performance Power States

- Processor Power States (C-States):
  - Reduces power consumption by sleeping the processor when it doesn't have code to execute.
  - Entry and Exit delays are much smaller compared to S-States.
- Processor Performance Power States (P-States): Enhanced Intel SpeedStep® Technology
  - Reduces power consumption without preventing the processor from executing code.

# INTEL processors

## Processor C-States:

- C-States ensure lower processor power during idle light workloads;
- C-State limits can be set by BIOS;
- A processor can go into sleep states several thousand times per second;
- OS controls the C-states in its idle process.

# INTEL processors

	Active state	Sleep states			
	<u>C0</u>	<u>C1/C1E</u>	<u>C3</u>	<u>C6</u>	<u>C7</u>
	Operating	Halt	Sleep	Deep Sleep	
Core clock		off	off	off	off
PLL			off	off	off
Core caches			flushed	flushed	flushed
Shared cache					flushed
Wakeup time*	active				
Core Idle power*				~ 0	< C6

\* Rough approximation



# INTEL processors

C-State Auto Demotion:



In general, Latency and Energy entry/exit costs increase with deeper C-States.

C-State Auto Demotion prevents unnecessary excursions into C6 & C3 states to improve latency.

# INTEL processors

Two C-State auto-demotion options.



Requests for C6 or C3 will route to C3 or C1 until sufficient residency is established.

Decision to demote is based on each core's immediate residency history. This feature is disabled by default. See your BIOS vendor to enable this feature.

# INTEL processors

## Processor Performance Power States (P-States):

- P0 - Processor consumes max power and is at max performance.
- P1 - Processor consumes less power and performance capabilities are limited below max.
- Pn - Performance is at minimal level and lowest power consumption. n must not exceed 16.

# INTEL processors

Multiple voltage and frequency operating points:

- Software controlled by writing to MSR;
- The voltage is optimized based on the selected frequency and number of active processor cores;
- All active processor cores share the same frequency and voltage.

Number of supported states is processor dependent.

# INTEL processors

P-states (Where are they useful?):

- P-States are useful when the system runs non-critical workloads that don't require higher performance.
- P States are useful when the system in power sensitive markets that don't care so much about performance. i.e. longer battery life.
- P States is useful under interactive modes where the system is waiting for user inputs.
- When I/O or memory is throttled.



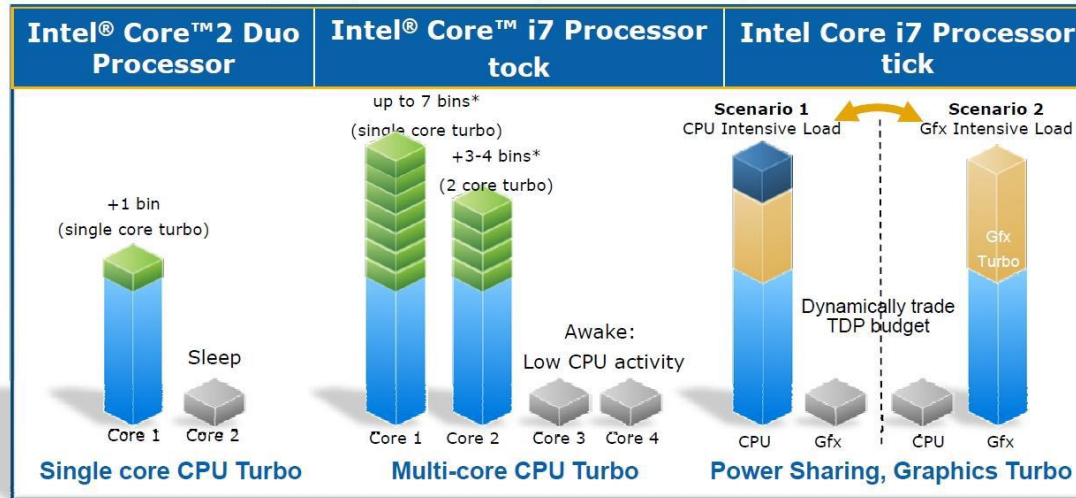
# INTEL processors

The Aggressiveness of P states. Set via OS:

- Linux – P States can be configured based on the end users using governors:
  - “On Demand” (Preferred) –can be customized;
  - Performance – Aggressive Performance;
  - Power save – Aggressive Power;
  - User space- User can set upper and lower limits.
- Windows – has power management schemes to set the values:
  - Balanced;
  - Max Battery;
  - Max Performance;
  - Custom.

# INTEL processors

Virtualization Power Management - Workload consolidation during off peak usage. Memory Power Management. Intel® Turbo Boost Technology.



# INTEL Core™2 Duo Processor

**Intel® Core™2 Duo Processor**  
*Dynamic FSB Switching*

CORE 1	CORE 2
HFM	HFM
P3	P3
P2	P2
P1	P1
LFM	LFM
superLFM	superLFM


**ACTIVE STATE**

**Increased Power Savings In Active Mode**  
Promotes Longer Battery Life In Low Power Active States, Such as When Playing DVD's and Music

**New Lower Active State**  
Delivers Additional Power Savings Through A Lower CPU Core Voltage and Reduced Bus Speed

Virtual Change In Bus Clock Frequency Recognized by CPU and Chipset Enables Lower Voltage While In Active State

**Lower Power In Active State**



- Improved power saving in an Idle state. Enhanced Deeper Sleep state both the CPU cores and chipset will power down without shutting the system down completely.

**Intel® Core™2 Duo Processor**  
*Longer Enhanced Deeper Sleep*

CORE 1	CORE 2
C0	C0
C1	C1
C2	C2
C3	C3
C4	
DC4	


**POWER**

ACTIVE  
HALT  
STOP CLOCK  
DEEP SLEEP  
DEEPER SLEEP  
ENHANCED DEEPER SLEEP

**Promotes Longer Battery Life**  
Prolongs Idle State For Computer

**Better Power Management Features Compared to Previous Generation Platforms**  
New Power C State For Idle State Allows Chipset to Power Down with CPU In LFM State

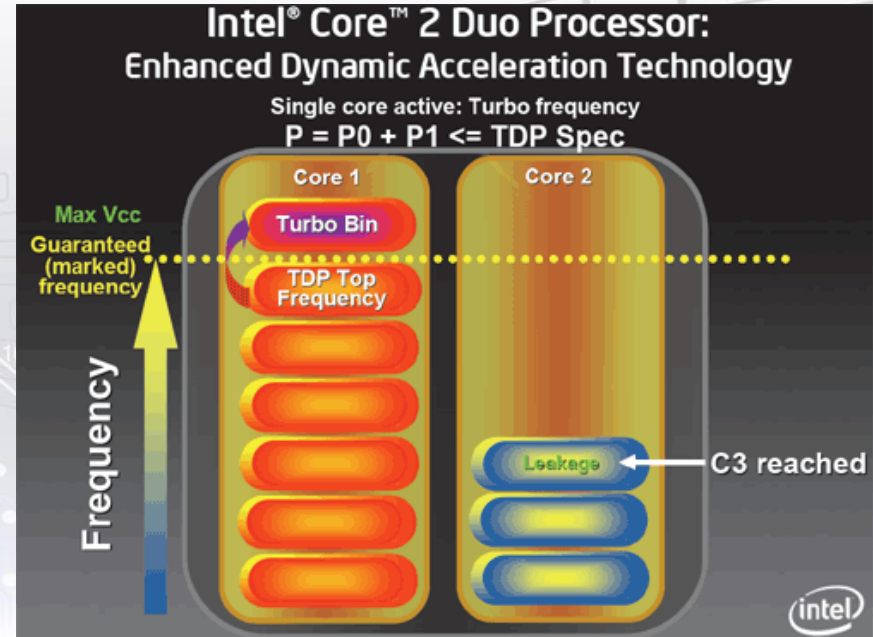
**Lower Power In Idle State**



# INTEL Core™2 Duo Processor

**Intel Dynamic Acceleration Technology** – support applications executed via a single thread.

**Smart Cache Technology** – single pool of memory for both cores – in case that only one core was under load, it had access to all the cache.





# AMD processors

**Enhanced AMD PowerNow!™ Technology** – for reduced power consumption by the entire processor. Native quad-core technology enables enhanced power management across all four cores.

- High Performance mode;
- Power-Saver mode;
- Automatic mode.



# AMD processors

- **Independent Dynamic Core Technology** – allows each core to vary its frequency, based on the specific needs of the system. This allows for more precise power management to reduce data center energy consumption and thereby reduce total cost of ownership (TCO).
- **Dual Dynamic Power Management™** - allows each processor to maximize the power-saving benefits of Enhanced AMD PowerNow! technology without compromising performance. Dual Dynamic Power Management can reduce idle power consumption and allow for per-processor power management in multi-socket systems to decrease power consumption.

# AMD processors

- **AMD CoolCore™ Technology** – To reduce power consumption within each core AMD CoolCore Technology evaluates which parts of the die – the cores, the memory, or both – are needed to support currently running applications. It can cut power to unused transistor areas to reduce power consumption and lower heat generation.
- **AMD Smart Fetch Technology** – AMD Smart Fetch Technology helps reduce power consumption by allowing idle cores to enter a “halt” state, causing them to draw even less power during processing idle times.

# AMD processors

- **Integrated DDR3 DRAM Memory Controller:** low-power memory to help reduce power consumption. AMD's integrated memory controller works with high bandwidth, energy-efficient DDR3 memory, both standard power 1.5v and low voltage 1.35v memory. It incorporates memory RAS for increased fault tolerance to help reduce system downtime and increase system reliability.

# ACPI Advanced Configuration and Power Interface

The principal goals of **ACPI** and **OSPM** are to:

- Enable all computer systems to implement motherboard configuration and power management functions, using appropriate cost/function tradeoffs.
- Enhance power management functionality and robustness.
  - Power management policies too complicated to implement in a ROM BIOS can be implemented and supported in the OS, allowing inexpensive power managed hardware to support very elaborate power management policies.
  - Gathering power management information from users, applications, and the hardware together into the OS will enable better power management decisions and execution.
  - Unification of power management algorithms in the OS will reduce conflicts between the firmware and OS and will enhance reliability.
- Facilitate and accelerate industry-wide implementation of power management.
- Create a robust interface for configuring motherboard devices.



# Preemptive multitasking or multi-threading

Because of these complexities, it is common for organizations to use a real-time operating system (RTOS), allowing the application programmers to concentrate on device functionality rather than operating system services, at least for large systems; smaller systems often cannot afford the overhead associated with a generic real time system, due to limitations regarding memory size, performance, or battery life.

The choice that an RTOS is required brings in its own issues, however, as the selection must be done prior to starting to the application development process.

This timing forces developers to choose the embedded operating system for their device based upon current requirements and so restricts future options to a large extent.



# Preemptive multitasking or multi-threading

The restriction of future options becomes more of an issue as product life decreases. Additionally the level of complexity is continuously growing as devices are required to manage variables such as serial, USB, TCP/IP, Bluetooth, Wireless LAN, trunk radio, multiple channels, data and voice, enhanced graphics, multiple states, multiple threads, numerous wait states and so on. These trends are leading to the uptake of embedded middleware in addition to a real-time operating system.

# Microkernels and exokernels

A microkernel is a logical step up from a real-time OS. The usual arrangement is that the operating system kernel allocates memory and switches the CPU to different threads of execution. User mode processes implement major functions such as file systems, network interfaces, etc.

In general, microkernels succeed when the task switching and intertask communication is fast and fail when they are slow.

Exokernels communicate efficiently by normal subroutine calls. The hardware and all the software in the system are available to and extensible by application programmers.

# Monolithic kernels

In this case, a relatively large kernel with sophisticated capabilities is adapted to suit an embedded environment. This gives programmers an environment similar to a desktop operating system like Linux or Microsoft Windows, and is therefore very productive for development.

It requires considerably more hardware resources, is often more expensive, and, because of the complexity of these kernels, can be less predictable and reliable.

Common examples of embedded monolithic kernels are embedded Linux and Windows CE.

Despite the increased cost in hardware, this type of embedded system is increasing in popularity, especially on the more powerful embedded devices such as wireless routers and GPS navigation systems.

# Monolithic kernels – reasons of popularity

- Ports to common embedded chip sets are available.
- They permit re-use of publicly available code for device drivers, web servers, firewalls, and other code.
- Development systems can start out with broad feature-sets, and then the distribution can be configured to exclude unneeded functionality, and save the expense of the memory that it would consume.
- Many engineers believe that running application code in user mode is more reliable and easier to debug, thus making the development process easier and the code more portable.
- Features requiring faster response than can be guaranteed can often be placed in hardware.



# Additional software components

In addition to the core operating system, many embedded systems have additional upper-layer software components.

These components consist of networking protocol stacks like CAN, TCP/IP, FTP, HTTP, and HTTPS, and also included storage capabilities like FAT and flash memory management systems.

If the embedded device has audio and video capabilities, then the appropriate drivers and codecs will be present in the system.

In the case of the monolithic kernels, many of these software layers are included.

In the RTOS category, the availability of the additional software components depends upon the commercial offering