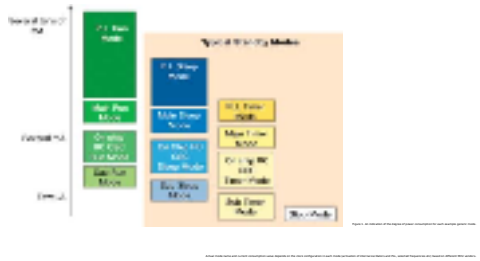
Taking Full Advantage of New, Low Power MCUs

Sandhya Mallikarjun, Fujitsu Microelectronics America

New low power 8-bit, 16-bit and 32-bit microcontrollers designed for portable and battery powered handheld products include an impressive set of new features, including low power capabilities. To optimize performance and take full advantage of these MCUs, designers must understand the flexible clocking systems, resource event-driven features, and — perhaps most important — the way to transition to and from standby and other low power modes.

Low Power Mode Types

Main CPU operating modes are Run Mode (normal operation), Sleep Mode, Timer Mode and Stop Mode. Among these CPU operating modes, Sleep Mode, Timer Mode and Stop Mode are called low power standby modes. The degree of power consumption depends on the active oscillator type and PLL clock source configuration in each mode, as illustrated in Figure 1.



The modes vary the degree to which the processor is aware of its surroundings and the different clocks that the processor must keep running. Increasing the clock speed will not yield similar power savings because faster execution increases power consumption. Similarly, unused peripheral modules on the processor should be de-activated to save power. Thus, the processor lowers power consumption partly by shutting off external and internal oscillators.

Another way to reduce power consumption is to disable the low voltage detection (LVD) circuit. This may reduce about 10 µA to 70 µA of current. Because current consumption increases significantly with higher temperature, decreasing the internal sub-regulator voltage levels of the device also can reduce consumption.

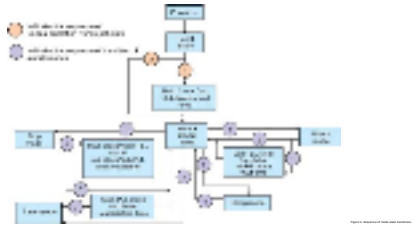
Sleep Mode is the general term where the CPU core, internal memory, the DMA controller and the external bus interface is switched off, and program execution stops. By reducing power consumption, specific clock supply and associated peripherals continue to run. Based on the clock supply either PLL, Main clock, on-chip RC oscillator or subclock, the degree of power consumption of each type of sleep mode varies as shown in Figure 1 above. Decreasing clock speed yields power savings.

Timer Mode allows only the main clock, sub clock oscillations, time base timer or

watch prescaler to work. The operating clock for the CPU and peripheral resources is stopped in this mode to reduce the power consumption. All clocks are halted in Stop Mode, but MCU status, RAM and register contents are preserved. Stop Mode has the lowest current consumption. However, only external interrupts can wakeup from Stop Mode.

Transition to Low Power Modes

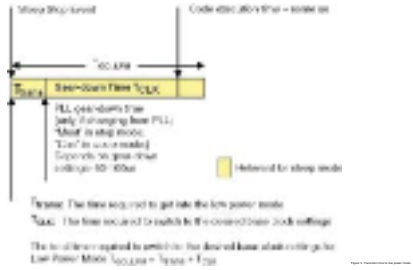
The MCU is switched into low power modes by configuring bits in the status and control registers — that is, by issuing a sleep command over the bus. Figure 2 shows the typical example of sequence of flexible mode state transitions of Fujitsu's low power F2MC-8FX family devices.



The clocking system is critical to MCU power consumption. Applications may enter and exit low-power modes several times or many hundred times a second. Moving into and out of the low-power modes and processing data quickly is crucial because a CPU waiting for clock stability wastes current.

Also, understand which clocks are instantly ON and which are not. Some MCUs have a two-stage clock wake-up providing a low-frequency sub clock (typically 32.768 kHz) to the CPU while a high-frequency clock is being stabilized, which can take up to a millisecond or longer. On these devices, the CPU may be operational in about 10 µs to 15 µs, but they run at a low, inefficient frequency or an incorrect high-speed frequency. Designers need to watch out for such clock state transitions. Be sure to use an MCU that offers a quick startup time, providing a stable, high-speed clock to the CPU in less than 8 µs to 10 µs.

If the MCU clock system provides multiple clock sources for the peripherals, the peripherals can operate while the CPU is asleep. For example, an A/D conversion may require a high-speed clock. If the MCU clock system provides the clock independently of the CPU, the CPU can sleep while the A/D converter is operational, saving CPU current consumption.



Processor Transition Mode to enter into low power modes is another very important factor. Figure 3 shows the example of transition time of Fujitsu's 32-bit

MB91F467DA. The total transition time TGO_LPM is approximately 110 µs with gear down time of the PLL of about 75 µs.

Recovery or Wakeup from Low Power Modes

To recover or wake up from a Standby Mode, an event from a resource is required. The possible resources that can wake up a controller depend on the type of Standby Mode, shown in Figure 4. A good low-power MCU will have extensive interrupt capability, providing interrupts for all its peripherals and many external interrupts for external events.

The wakeup request is similar to interrupt requests. The major difference to interrupts is that the CPU operation is stopped. Unlike interrupt requests, wakeup requests are accepted regardless of processor status in PS register. Hence, wakeup is possible even if interrupts are disabled or interrupt priority is not defined for such interrupts. Consequently, wakeup conditions are enabled by a subset of interrupt settings internal to MCUs, and interrupt configuration is a possible but unnecessary condition. Typically, processing within an interrupt routine will determine when the processor needs to change from a low power mode to normal operation. To achieve this, it alters those same status register bits by directly modifying the memory location where the processor's status register was pushed onto the stack at the start of the interrupt. When the interrupt routine returns, using the RETI instruction, the altered status register value is loaded into the processor status register PS, and the processor continues operation in the newly selected mode. The modern C language compiler tools support an easy method to handle this.
The primary factor of low power modes is that recovering into normal operating modes can impose a significant delay. Therefore a key feature to look for in any MCU is the shortest recovery time. MCU recovery time from some low-power modes should be fast enough to meet the response times of interrupts.

Total $T_{restore}$ is approximately 2.50 ms with T_{wakeup} of 1.4 ms, PLL stabilization time of 1 ms and gear-up time of about 75 µs. This illustrates the ability to optimize the design for low power while having a flexible architecture in the MCU that helps to achieve high performance and lower power consumption.

Sandhya Mallikarjun is a Staff Applications Engineer, working with the Embedded Systems Solutions Group at Fujitsu Microelectronics America. For more information, contact Fujitsu Microelectronics America, 250 E. Arques Blvd., Sunnyvale, CA 94088; 408-737-5600; www.fujitsu.com/us *[1]*

Links:
[1] http://www.fujitsu.com/us