FPGAs have been used to provide quick, cost effective solutions as they can be re-programmable capabilities. Yet this re-configurable overhead is also the reason why they consume more power than ASIC designs as power management is more complex. Tuan et al. have investigated low power FPGA applications for battery powered devices and have used a variety of techniques to reduce power. They designed a low power called Pika. Voltage scaling was used to scale the core operating voltage to drastically reduce energy. A 1V operating voltage was found to give the best reductions without severely affecting performance. It was found that SRAM cells were a major contributor towards leakage current. Subthreshold leakage was reduced by using a higher voltage threshold (Vt) and gate leakage was reduced by using thicker gate oxides. Though this increases cost and area, the overall energy savings outweigh the cons. Finally, power gating was also used to reduce leakage current. Unused blocks were turned off to save power. NMOS transistors were used as power gates as they are faster. Both NMOS and PMOS were not used in conjunction to save area. Power gating in FPGAs is complex due to the amount of logic that can be gated. Therefore establishing what the smallest block that can be power gated is important. In Pika, a tile was the smallest unit that was power gated. A tile here is used to define a configurable logic block (CLB) along with its relevant programmable switch matrix that connects it to other CLBs. The SRAM cells in the switch matrix are not power gated to enable state retention when the rest of the core is powered down. Power gating individual tiles helps in implementing a partial standby mode wherein some logic elements can be powered off and the rest can still remain active. This feature is implemented by having a programmable bit per tile. The overall power savings for all these schemes is illustrated in Figure x and it can be seen that a 46% in active power reduction and 99% standby power reduction was observed when compared to a normal FPGA with no power management. There was however a trade-off with performance and area to cater.

Adaptive body biasing is another technique that was found to be beneficial. It is based on the simple idea that forward body biasing (FBB) decreases the threshold voltage and therefore increases performance and power while reverse body biasing (RBB) increases threshold voltage and reduces leakage current. This adaptive biasing is only applied to certain areas of the chip to suitably alter performance or power and this can give better power savings without affecting performance too much. Gammie et al. have described a tool, SmartPriMer, which inserts power management modules into a piece of RTL code. These modules include power domains, adaptive body biasing and other such techniques to reduce power consumption. The tool also generates UPF information for the design that can be used on the later stages for design. These techniques were tested on mobile applications and a 37% reduction in active power was observed.

Finally, a healthy interaction of hardware and software policies is a good way to reduce power. Most mobile phones have multi core processors so to make optimum use of these resources parallel computing and task scheduling can be used. Parallel computing involves executing instructions in parallel and an efficient task scheduler can map out a sequence of instructions that can be executed with minimal delay. The use of heterogeneous cores aids schedulers as cores can be catered to different applications and they make good use of the available resources and help in reducing energy as tasks are executed faster.