# Hints for performance optimizations when using the MPC8xx family:

Revision 1.0
January 15, 1997

- Make sure the memory controller programming is tuned to the memory and frequency used. E.g. do not use UPM code designed for 50Mhz with 20Mhz system since it will cause unnecessary wait states.
- Make sure caches are enabled
- Make sure MMU is enabled
- Avoid "Guarded" storage for anything but FIFOs
- Make sure not to work in "serialized mode" (ICTRL register)
- Make sure to avoid show cycles if not debugging (ICTRL register)
- Make sure all relevant pages are cache enabled
- Use copy back policy for all "software only" data
- Use D-Cache enabled and "write through" for transmit data
- Use D-Cache inhibit for receive data
- Use D-Cache inhibit for data used only once.
- Prefer cache usage optimization over core optimization techniques.

Example:

Sometimes, loop unrolling will cause a greater instruction miss ratio by using more instructions to do a routine.

- Check how many interrupts per second are activated. Interrupts affect performance in two ways:

1. Interrupt overhead. Does the interrupt handler save/restore only the necessary state?
2. Interrupts can trash the caches, especially the I-Cache.

**Note:** The caches are two way set associative. That is, each memory location can be located in just TWO places in cache. You can analyze your code by blocks.

Example:

Typical code is composed of task calling C library routines and interrupted by an interrupt handler. If the library routines and interrupt handler use the same cache sets, then the task will see larger than necessary miss ratio. Locating the frequently used C library and frequently activated interrupt handler in different sets allows always one set free for the task.

- For the case of a frequently activated interrupt handler, consider locking in (if the same code runs in subsequent interrupts) or cache inhibit (if it is typically different code for subsequent interrupts) the interrupt handler.