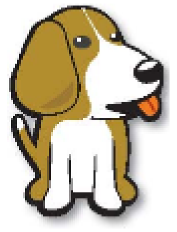
Optimizing the Embedded Platform using OpenCV

February 17, 2012

Matt Weber
(mlweber1@rockwellcollins.com)

- Goals
- Project Approach & Results
- Future Ideas
- References



Goals

- To quantify the effects of the many optimizations available and see what effect, if any power management has

- Most Important Requirements (MIRs)
  - Minimal startup and low latency processing time
  - On-demand Power Management

- Background
  - Utilized a OMAP3 processor for image processing
  - Linux 2.6.39.4 Kernel with OMAP PM patches
  - Buildroot w/ Crosstool-ng toolchain

Project Approach

- Cost/Benefit
  - Compiler → Co-Processor → Power Management → Specialized Cores
  - Supporting software (which kernel, packages, vendor libraries, etc)

- Define benchmarking tool

- Gather metrics for optimization methods applied to
  - Platform (Kernel/rootfs)
  - Application
  - With power management active

Project Approach: Compiler/Toolchain

- Gotchas
  - Are Binary compatibility & architecture (armv5, v6, v7a....) masking a problem?
  - Are your Platform & App using the same toolchain?
  - Are features like VFP (Vector Floating Point) & Advanced SIMD extension (aka NEON) enabled?

- Building your own has some additional benefits
  - Source control & ability to recreate/fix issues
  - Geared towards your CPU arch & hardware FPU
  - Could tailor kernel headers to get a newer feature
  - Possibly incorporate the latest Linaro GCC

Know your toolchain!

Project Approach: Benchmarking Tool

- OpenCv 2.1
  - cvMatchTemplate() algorithm as the test case

  cvMatchTemplate(img, tpl, res, CV_TM_CCORR_NORMED);
  - Lots of matrix math
  - Each of the time measurements were just for the algorithm execution and not the image load time
  - 5.5MB image is searched for the image of a small boat

Project Approach: Metrics Test #1

- Test: Compiler Optimization
- Description: Kernel and Rootfs are built with same flags below and executing off an SDCard.
- Flags:

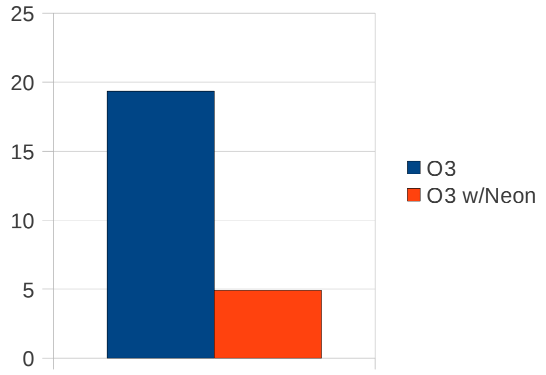
  CFLAGS += -pipe -O3
- Result: ~19.35sec @800Mhz

Project Approach: Metrics Test #2

- Test: Compiler Optimization & use of hardware co-processors
- Description: Kernel and Rootfs are built with same flags below and executing off an SDCard.
- Flags:

  CFLAGS += -pipe -O3 -mfpu=neon -ftree-vectorize -mfloat-abi=softfp
- Result: ~4.91sec @800Mhz

  ~75% increase in performance



Compiler

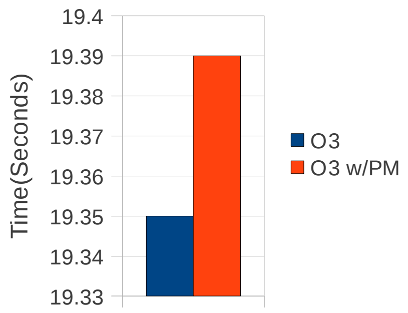
Co-Processor

Project Approach: Metrics Test #3

- Test: Compiler Optimization & Power Management
- Description: Kernel and Rootfs are built with same flags below. Power management is enabled to idle and frequency scale the CPU on-demand between 300 and 800Mhz. It uses the default scaling trigger threshold for the 2.6.39.4 kernel.
  (Note: Purely ARM core instructions.)
- Flags:

  -pipe -O3
- Result: ~19.39sec @300-800Mhz

  ~40msec (2%) increase in processing time w/ PM



- Comment: Solely ARM instructions cause the scheduler to have more demand for a higher clock speed earlier, so it results in a small increase in the additional processing time required.

Compiler

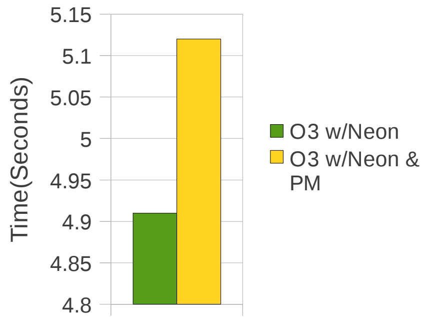
Power Management

Project Approach: Metrics Test #4

- Test: Compiler Optimization, co-processors and Power Management
- Description: Kernel and Rootfs are built with same flags below. Power management is enabled to idle and frequency scale the CPU on-demand between 300 and 800Mhz. It uses the default scaling trigger threshold for the 2.6.39.4 kernel.
  (Note: ARM core and Neon instructions.)
- Flags:

  -pipe -O3 -mfpu=neon -ftree-vectorize -mfloat-abi=softfp
- Result: ~5.12sec @300-800Mhz

  ~210msec (4%) increase in processing time w/ PM



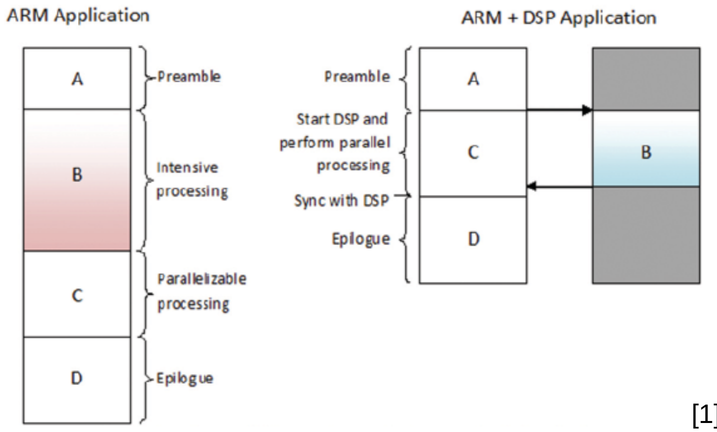
- Comment: Less time spent executing ARM instructions, since the Neon core is offloading some of the processing, causes more execution at 300Mhz and a slight increase in processing time.

Compiler → Power Management → Co-Processor

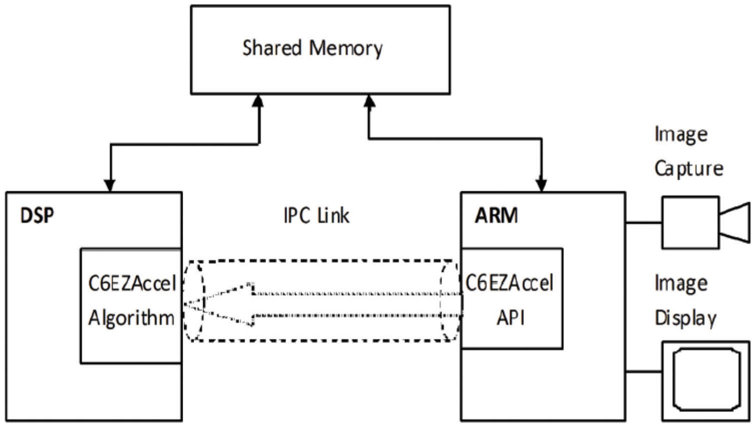
Rockwell Collins

Project Approach: Future Tests

- Finish testing with DSP and TI Codec Engine
  - Initial tests with CMEM, LPM, DSPLINK, TI Codec Engine are working
  - Issues were found with the C6Accel used in SoC OpenCV DSP work
    (newer TI libraries, kernel and compiler issues.....)
  - TI measurements with Integra SOC (floating point DSP) show a 86% speed up for the match template algorithm



[1]



[1]

Compiler | Power Management | Co-Processor | Specialized Cores

Project Approach: Performance Metric Summary

Test	Result (sec)
#1 -O3	19.35
#2 -O3 & Neon	4.91
#3 -O3 w/ PM	19.39
#4 -O3 & Neon w/PM	5.12
#5 -O3 & Neon w/PM & DSP	Est. ~3.07

The key to the next step is controlling offloading overhead

Project Approach: Power Management Test

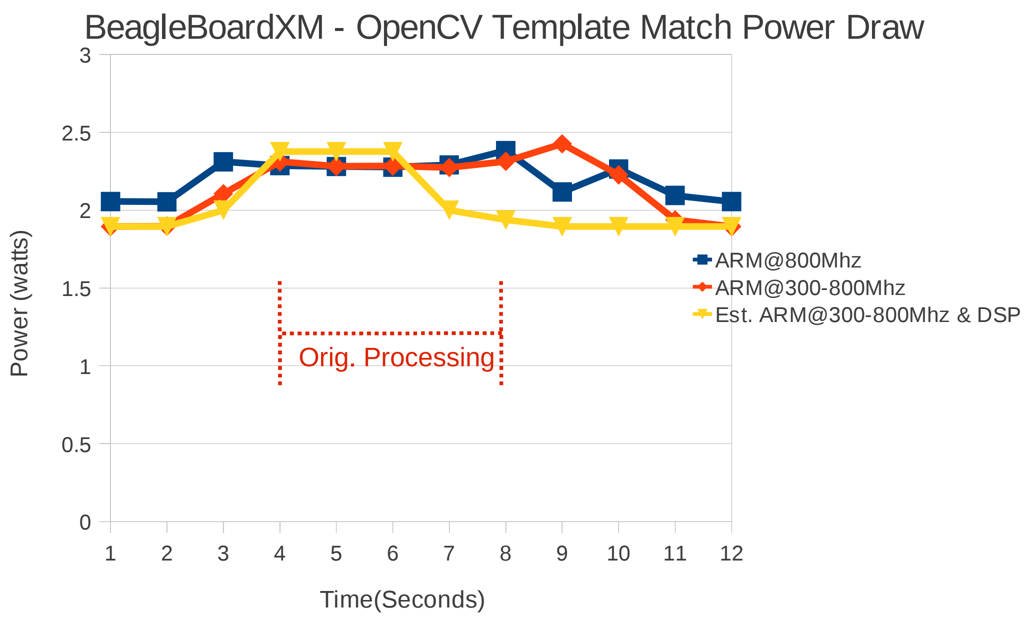
- Tools → bench power-supply and data logging multimeter
- Startup board (power-supply is set to a 1A limit at 5V)
- First test is on-demand

```
[root@buildroot ~]# echo "800000" > /sys/devices/system/cpu/cpu0/cpufreq/scaling_max_freq
[root@buildroot ~]# echo "ondemand" >/sys/devices/system/cpu/cpu0/cpufreq/scaling_governor
cpufreq-omap: transition: 800000 --> 300000
[root@buildroot ~]#  ./opencv_templatematch
WORKING>>>
cpufreq-omap: transition: 300000 --> 800000
5.120000 seconds of processing
 t1: 320000   t2: 5600000
 Clockspersec: 1000000
cpufreq-omap: transition: 800000 --> 300000
[root@buildroot ~]#
```

- Second test is userspace set frequency

```
[root@buildroot ~]# echo "userspace" > /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor
[root@buildroot ~]# echo "800000" > /sys/devices/system/cpu/cpu0/cpufreq/scaling_setspeed
cpufreq-omap: transition: 300000 --> 800000
[root@buildroot ~]# ./opencv_templatematch
WORKING>>>
4.910000 seconds of processing
 t1: 110000   t2: 5020000
 Clockspersec: 1000000
[root@buildroot ~]#
```

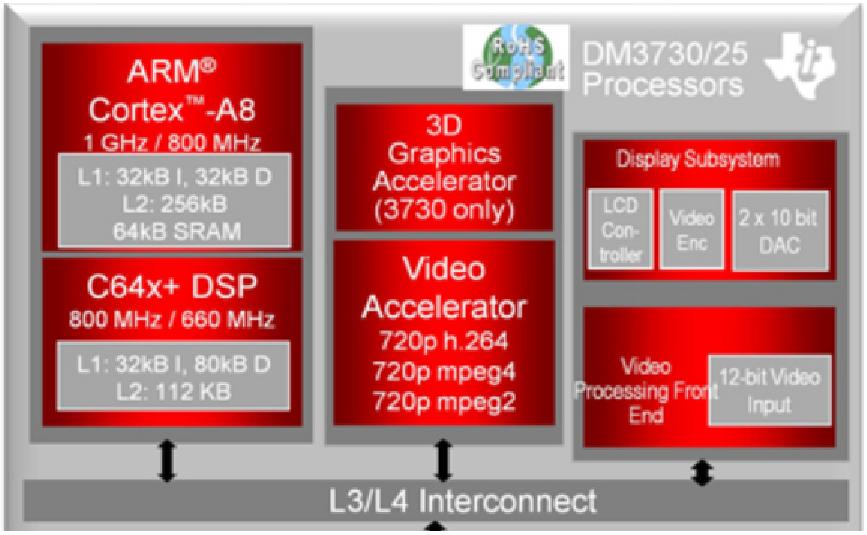
Project Approach: Initial Power Measurements

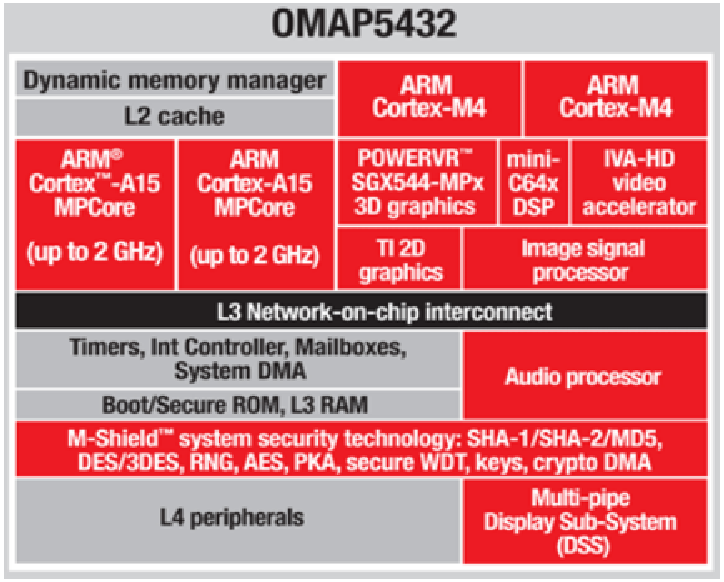


- Note: the DSP adds an additional ~375mW, shown in yellow & prevents the ARM from scaling up to 800Mhz. The chart shows only an estimate of DSP power draw[5] and an approximate timeline from TI whitepaper findings.
- If an OMAP GPU options was added, the approx power draw would increase by ~93mW. We're not sure yet how much overhead this would cause on the ARM...

Future Ideas

- Investigate the new issues of Power Management in a multi-core world
  - How could load statistics be maintained for dynamic power control across cores?
  - Maybe add hooks into existing CPUFreq framework for on-demand based on anticipated completion from other cores? What if Linux on the primary CPU(s) suspended while the offloaded task is being processed?




OMAP5432
Dynamic memory manager
L2 cache
ARM Cortex-M4
ARM Cortex-M4
ARM® Cortex™-A15 MPCore (up to 2 GHz)
ARM Cortex-A15 MPCore (up to 2 GHz)
POWERVR™ SGX544-MPx 3D graphics
mini-C64x DSP
IVA-HD video accelerator
TI 2D graphics
Image signal processor
L3 Network-on-chip interconnect
Timers, Int Controller, Mailboxes, System DMA
Audio processor
Boot/Secure ROM, L3 RAM
M-Shield™ system security technology: SHA-1/SHA-2/MD5, DES/3DES, RNG, AES, PKA, secure WDT, keys, crypto DMA
L4 peripherals
Multi-pipe Display Sub-System (DSS)

[7]

Future Ideas

- GsoC project: OpenCV DSP Acceleration (2010)
  - Investigate OpenCV code issues (lots of floating point and STL)
  - Gather power, timing and latency/IPC overhead numbers using the TI Codec Engine approach
  - Possibly implement custom DSP approach based on results

- GPU
  - Investigate (future) SGX Graphics SDK with OpenCL support
  - Currently the only published vendor supporting OpenCL is ZiiLABS (ZMS SOC) and TI (OMAP5)

Project Information

- Hardware
  - BeagleboardXM
  - (optional) LI-5M03 camera

- Repository & Wiki
  - includes xloader, uboot, sdcard scripts, kernel & rootfs, test sequences

  git://github.com/matthew-l-weber/buildroot.git
  https://github.com/matthew-l-weber/buildroot/wiki

- Buildroot Overview

  http://free-electrons.com/pub/conferences/2011/elce/using-buildroot-real-project.pdf

Credits/References

[1]http://www.ti.com/lit/wp/spry175/spry175.pdf

[2]http://www.ti.com/lit/wp/spry144/spry144.pdf

[3]https://code.google.com/p/opencv-dsp-acceleration/wiki/GettingStarted1

[4]http://old.nabble.com/Request-for-comments-on-packages-for-TI%27s-OMAP3-and-DM365-processors-td29741226.html

[5]http://processors.wiki.ti.com/index.php/OMAP3530_Power_Estimation_Spreadsheet

[6]http://www.sakoman.com/OMAP/an-overiew-of-omap3-power-management-with-2639-pm.html

[7]http://www.ti.com/general/docs/wtbu/wtbugencontent.tsp?templateId=6123&navigationId=11988&contentId=4638