Measurement issues
From gathering numbers to gathering knowledge

Dr. A.P. Kostelijk (Ton), PDSL
Contents

• Discussion
• Pieces of the puzzle
• Multi-level performance modelling
Example. Discussion.

```c
int main() {
    int t1, t2, n=10;
    t1 = OS_time();
    for (i=0; i<n; i++)
        f(0);
    t2 = OS_time();
    return (t2 - t1)/n;
}
```
Pieces of the puzzle

Aim
Why?

Analysis
Verification

Context

How?
Precision

Expectation
Aim

• Why measure something?
• What is the underlying model?
  – Without a model one is collecting meaningless figures.
• Guestimate, estimate, measure, …?
• Min / max / typical?
• Feasibility: min / max reasoning (close in)
Determine context / state (static)

Version of
• source-code
• operating system
• compiler(s)
• hardware

• dataset operated on
• compiler settings
• linker settings
• hardware settings (Board-support package)
• other HW bus traffic
• memory configuration
Determine context / state (dynamic)

- Cache
- Network
- Disks
- Higher level other processing

Looping optimism.
Silence or busy.
Jitter due to move & rotate.
Other threads, or interrupts (Profilers)
Expectation

• Experience with and understanding of the old system is key
• Use a model
• Balance effort by granularity, guided by analysis and relative importance.
• Simplify.
Pitfall
Measuring

- Measuring overhead ignored
- OS-timer ticks: unsave
- Forget asynchronous elaboration
Overhead

- Bus protocol
- Bus congestion
- Array indexing overhead
  (usually solved by compiler)
- Loop overhead
- Function-call
- Thread / interrupt switch
- Measurement
Measurement precision

- Use HW timer register (Limit range!)
- IO-pin toggling with Logic Analyser
- OS-timer
- Profilers
- Precision: 1 instruction, or cache-line updates
- Idem, plus Logic An. Precision
- 1 OS-tick ≈ 10 ms
- Diverse
Measurement precision

When subtracting measurement results...
• $X = 10.0 \pm 5\%$, $Y = 8.0 \pm 5\%$
• $X - Y = 2.0 \pm 45\%$

Statistics out of scope.

Last but not least: m/$\mu$s, byte or bit, clock cycles of what? Be explicit on units.
Example

Idle process measurements:
• OS-based profiler that counts the time each task has run?

Missing (part of) task-switch overhead, interrupts (5 - 25 % load).

Instruction i++ Counter in lowest priority task might be a better idea.
Analysis & Verification

Modelling is the heart of the matter

and therefore subject of the rest of this presentation.
System levels of execution

1. Task and priority assignment
2. Algorithms, source code
3. Machine code, CPU
5. Device access
Software Latency models

- **Soft RT tasks**
- **Hard RT tasks**

Dependencies:

- **Action / Function**
- **OS-call**

- **‘Data’ Transfer**
Typical iteration in time

- SoftRT
- HardRT
- Function
- Transfer

Time
‘Data’ transfer latency model

Instruction and registers

CPU

Device

i

k

Memory dependent

Memory
‘Data’ transfer latency model

Instruction and registers

CPU

Cache-line

I$

D$

Device

Memory dependent

Memory A

Memory B

Drive

Sector
Data transfer latency model

• Major input to understand HW/SW bottleneck
• Indicates for a given cpu-instruction, and context, the cpu-delay + cpu-stall cycles.
• Derive model by measuring a forced predefined number of bus-transactions.
  
  Example:
  
  $I$$_{flush}; measure; 1000$ nops; measure;

• Focus on typical transfers.
Data transfer - pitfalls

- Incorrect HW-settings.
- Unnoticed bus transfers.
- CPU-instruction pipe-line stalls.
- Overhead due to measurement.
- Precision is delicate. Subtraction of timing overhead is often required.
- Effect of large cache-lines
Function / OS-call

• Indicates for a given function, and context, the total run-time (latency).
• This includes interrupt routines and -overhead.
• Major input for e.g. RMA.
Function / OS-call & Data Transfer

Action / Function | OS-call
--- | ---
c cpu cycles
x cache-misses
y page-faults

‘Data’ Transfer

\[ T = c + x.f + y.h \]

\[ T = 1000 + 50 \times 10 + 5 \times 1000 \]
Function / OS-call / Transfer

• Simplified model:
• CPI: cycles per instruction
  – statistical number of cache-misses
  – For a MIPS: 3 - 10 typically for control code
Function / OS-call pitfalls

- Timer precision.
- Cache-line trashing.
- Inconvenient compaction of data
  (e.g. char / int / char, iso. Char/char/int)
- Link-address may influence 5 - 15%
- Additional busload interference.
- Forget write-backs in a loaded system.
- Hidden taskswitches and interrupts.
Task switching

Task X

CPI

time

F
Task switching

1: Overhead
2: Cache pollution
Task interference

![Diagram showing task interference]

- CPI
- Task X
- Switch
- Time

Switch

Switch

Task X

Time

F

F
Task interference

For small interruptions y
context switch and cache-pollution

• Resulting load = (switch + f_{load}) \times 2.
Hard RT tasks

- From high-priority (interrupts) to middle-priority (threads), successive function processing measurements and analysis (e.g. RMA) are done to check the timing requirement feasibility.
- Include Thread switches, Interrupts pollution, Mutex-latencies.
- Example: RT-streaming platform
Soft RT tasks

• For middle to low priority threads, do response-time measurements where the rest of the system is also fully active.
• Measure relevant jitter in response-time by understanding the underlying causes.
• Trace substantial losses due to overhead.
Conclusion

• Measurements serve to build a reliable predicting multi-level time model of a system.

• Analysis, verification and simplification are major means to reach this goal.

• Without aim, context, expectation and precision, the actual outcome of a measurement is useless.