ILP32 Performance on AArch64

Chris Tyler
Hong (Michael) Huang
Glaser Lo

Seneca College
Centre for Development of Open Technology
Who
What
Where
Why
Results
Who

Chris Tyler
Industrial Research Chair
Open Source Technology for Emerging Platforms (OSTEP)

Seneca College
Centre for Development of Open Technology (CDOT)
Who

We do applied research into the potential of energy-efficient datacentres based on ARM hardware and open source software.
<table>
<thead>
<tr>
<th></th>
<th>LP64</th>
<th>ILP32</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integer</td>
<td>32 bits</td>
<td>32 bits</td>
</tr>
<tr>
<td>Long Integer</td>
<td>64 bits</td>
<td>32 bits</td>
</tr>
<tr>
<td>Pointer</td>
<td>64 bits</td>
<td>32 bits</td>
</tr>
</tbody>
</table>
ILP32 affects the syscall and procedure interfaces, and therefore touches:

- kernel
- gcc
- binutils
- libraries
Where

LEAP Project
leapproject.ca

A distribution we built for ARM64 testing; based on CentOS7 with local changes. Both the distro and access to the buildsystem are publicly available.
Where

Seneca EHL

A rack and a half of ARM64 servers, 10Gbe networking, SSD storage, power conditioning, and test equipment.

Two ARMv8 enterprise platforms with different microarchitectures were selected for testing.
Why?!

“They” Want ILP32
Who are they, and why do they want ILP32?
Why?!  

1. To look good on benchmarks. Want improved performance.  

2. To run old code unmodified. Want minimal performance loss.
The Pieces

Linaro gcc 4.9.4 (+2 patches)

Kernel 4.3 (+2015.11 ILP32 LKML patches – Cavium via LKML)
The Pieces

Docker Container
The Pieces

SPECint 2006
The Results

Every enterprise system we have was received under a benchmark embargo.

Even the “Commercially Available” ones.

What I'm going to share is therefore the relative performance of ILP32 compared to an LP64 baseline, on each of the two selected platforms.
The Results

ILP32 Performance Impact

Improvement of ILP32 over LP64 (%)

Benchmark
The Results

ILP32 wins on memory utilization and cache packing when there are a lot of pointers.

ILP32 loses when these advantages are overwhelmed by munging overhead.
Why?! 

1. To look good on benchmarks. Want improved performance.

2. To run old code unmodified. Want minimal performance loss.
The Results

1. To look good on benchmarks. ILP32 looks slightly better than LP64 on benchmarks.

2. To run old code unmodified. ILP32 will allow you to use most of the performance of AArch64 without changes.
Do We Care? Enough?

A lot of code to maintain for little gain.

(OTOH: The pieces have all been written)
Chris Tyler
chris.tyler@senecacollege.ca
http://leapproject.ca