Why Do We Need Something New?

Existing parallel programming models all fail

We can’t get the ultimate performance today
Given the hardware trend, we need prepare for scalability to at least 1,000,000,000 threads
Every application and developer is unique with specific needs

All of this needs to be supported
Need something different and better!
World’s Only Unified New Development
WOUND
Open Source (obviously)
A Brave New Parallel World

Open WOUND
Open WOUND

Language

Environment

Runtime

Tools

A Brave New Parallel World
Open WOUND Goals

*Extreme scalability*
*Extremely powerful*
*Always optimal performance*
*High level*
*Explicit language*
*Total control over everything*
*Can express uncertainty*
*Trivial to understand*
What Is Wrong With This?

Traditional pragma based model:

```c
#pragma omp <action>
```

This is an all or nothing model
Real-life is different
Open WOUND Differentiator

Confidence Levels!

This allows the developer to rely on the system to help out, while maintaining full control
Confidence Levels

*Open WOUND adds a second dimension:*

```markdown
#pragma wound [confidence level] <action>
```
How To Define Parallelism?

Start and end parallel execution

```c
#pragma wound [confidence level] open
#pragma wound [confidence level] close
```

Confidence levels supported:

- always
- maybe
- sometimes
- depends
- not (to preserve sequential semantics !)
Open WOUND’s Differentiator

Confidence levels can be combined!
Examples Confidence Levels

```c
#pragma wound maybe
#pragma wound maybe sometimes
#pragma wound maybe sometimes not
#pragma wound depends sometimes
#pragma wound depends not
#pragma wound not always depends
```

*Current limitation is to only support 5 combinations (see the roadmap though)*
Acknowledgement

What follows next has been inspired by the expressive power of some of the more elegant constructs from OpenMP 4.0

For example:

```c
#pragma omp declare simd
aligned(a,b,c) \nlinear(e,f,g:18)
simdlen(27) \ninbranch
```
How About Open WOUND Then?

Non-Expert Rapid Decision (NERD) controls

Successfully marries full control with ease of code development

Initially only support for:

- superscalar level
- pipeline scheduling order
- preferred number of instructions
- number of cache misses allowed: target and max
Simple Example Of NERD Control

```
#pragma nerd wound \\n  superscalar(3) \\n  pipeline(ex0,FPU1,ex1) \\n  instructions(12) \\n  cachemisses(t:0,m:123454321)
```
NERD supports confidence levels!
Simple Example Of NERD Control/2

```plaintext
#pragma nerd wound
maybe superscalar(3) 
pipe(dle(not ex0,always FPU1,ex1) \ 
instructions(sometimes 12) \ 
cachemisses(not t:0,m:123454321)
```
HURT – The Run Time System

*Hybrid Unified Run Time (HURT):*

*Handles everything for the user*

*Extremely powerful*

*Easy to use* *

*) Incorrect use is fatal though*
Open WOUND - Environment Variables

WOUND GO_AHEAD [YES | NO]

HURT_WOUND [ALWAYS | SOMETIMES | NEVER]

NERD_WOUND [[CONFIDENCE LEVEL],TARGET]]

Example:

NERD_WOUND INSTRUCTIONS, NOT PIPELINE
Open WOUND - TEARS

Totally Expressive Advanced Relational System (TEARS)

Symbolic debugger for Open WOUND

Example

(tears) STOP if perhaps not always
(tears) TEARS: program stopped
(tears) print handkerchief
(tears) TEARS: inconsistency detected
(tears) TEARS: conflict with “maybe depends”
Open WOUND – BANDAID

Two component SDK for rapid program development and deployment

BAND – Brave And Naive Developer
AID – Architecturally Independent Dependences
Community Documentation Project *

Connects it all together
Dropped concept of page count
Uses “STAPLES” instead

*) Driven by initial user feedback. Personally I think Open WOUND is so easy, straightforward and obvious that it doesn’t need documentation
for (int i=0; i<n; i++)
    do_whatever_you_like(&i)

this_is_my_job_only()

always_the_same_answer()}
Hello World Example – Prototype!

```c
#pragma wound always open
hurt_always(sometimes)
#pragma nerd wound
  superscalar(8) instructions(4) \\pipeline(FPU0,not FPU1,MMU3) \ cachemisses(t:4,m:666) 
for (int i=0; i<n; i++)
  do_whatever_you_like(&i)
hurt_ends()

#pragma wound never open
this_is_my_job_only()
#pragma wound always close

#pragma wound maybe not
  always_the_same_answer()
#pragma wound sometimes close
```

A Brave New Parallel World
Open WOUND - Status

- WOUND is on ice
- HURT is fleshed out still
- TEARS code is trickling in
- BANDAID 0.9.1.7.3 has been patched for 1.0 release
- STITCHES is at 666 STAPLES (and growing)

Public comment phase expected to start April 1, 2014
Open WOUND – Roadmap/1

- **WOUND**
  - Plan 2 more actions for 1.1 release
  - Target for 1.1 is to support 200 confidence levels
- **Massive increase of HURT at all levels**
Open WOUND – Roadmap/2

• **BANDAID**
  ✓ BAND has been frozen
  ✓ AID is on the cutting table

• **TEARS**
  ✓ Support 200 confidence levels in 1.1 release

• **STITCHES**
  ✓ Effort to automatically insert STAPLES
Open WOUND 1.0 targeted for SC’14
A BOF WOUND planned!
Come and stop by our booth for your free copy of the 1.0 BANDAID kit!
Thank You And ..... Stay Tuned!

ruud.vanderpas@openwound.bandaid