Imperial College London

julia

High-Level Synthesis using the Julia Language

Benjamin Biggs, Ian McInerney, Eric Kerrigan, and George Constantinides

Imperial College London, The University of Manchester

{bb2515, e.kerrigan, g.constantinides}@imperial.ac.uk, ian.mcinerney@manchester.ac.uk

Julia: Two Language Problem

Algorithm Development

THE C PROGRAMMING LANGUAGE

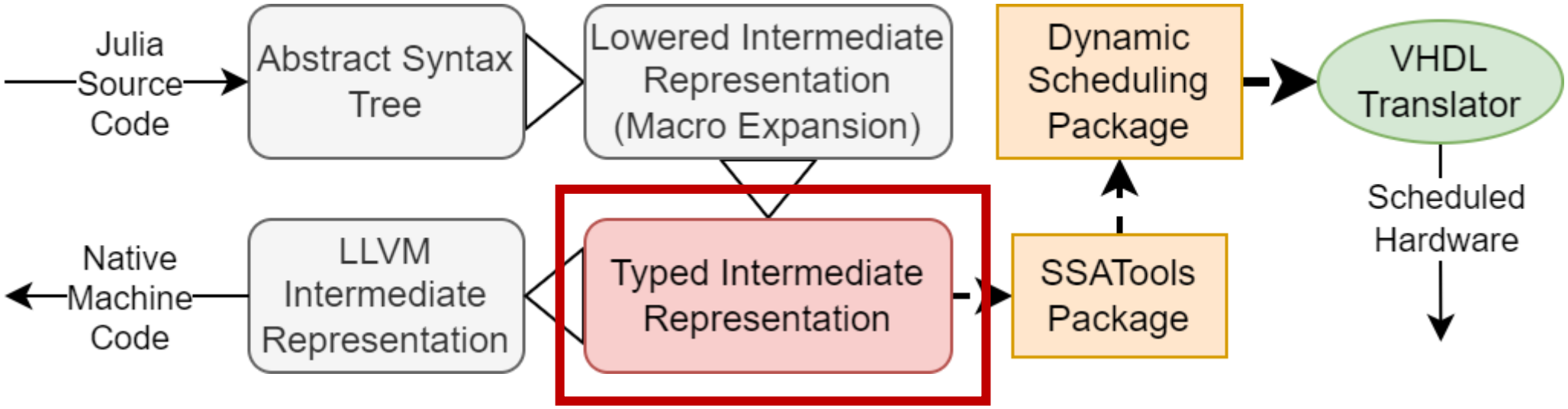
Optimised Code

julia

Algorithm Development

Readable, Optimised Code

Compilation Flow



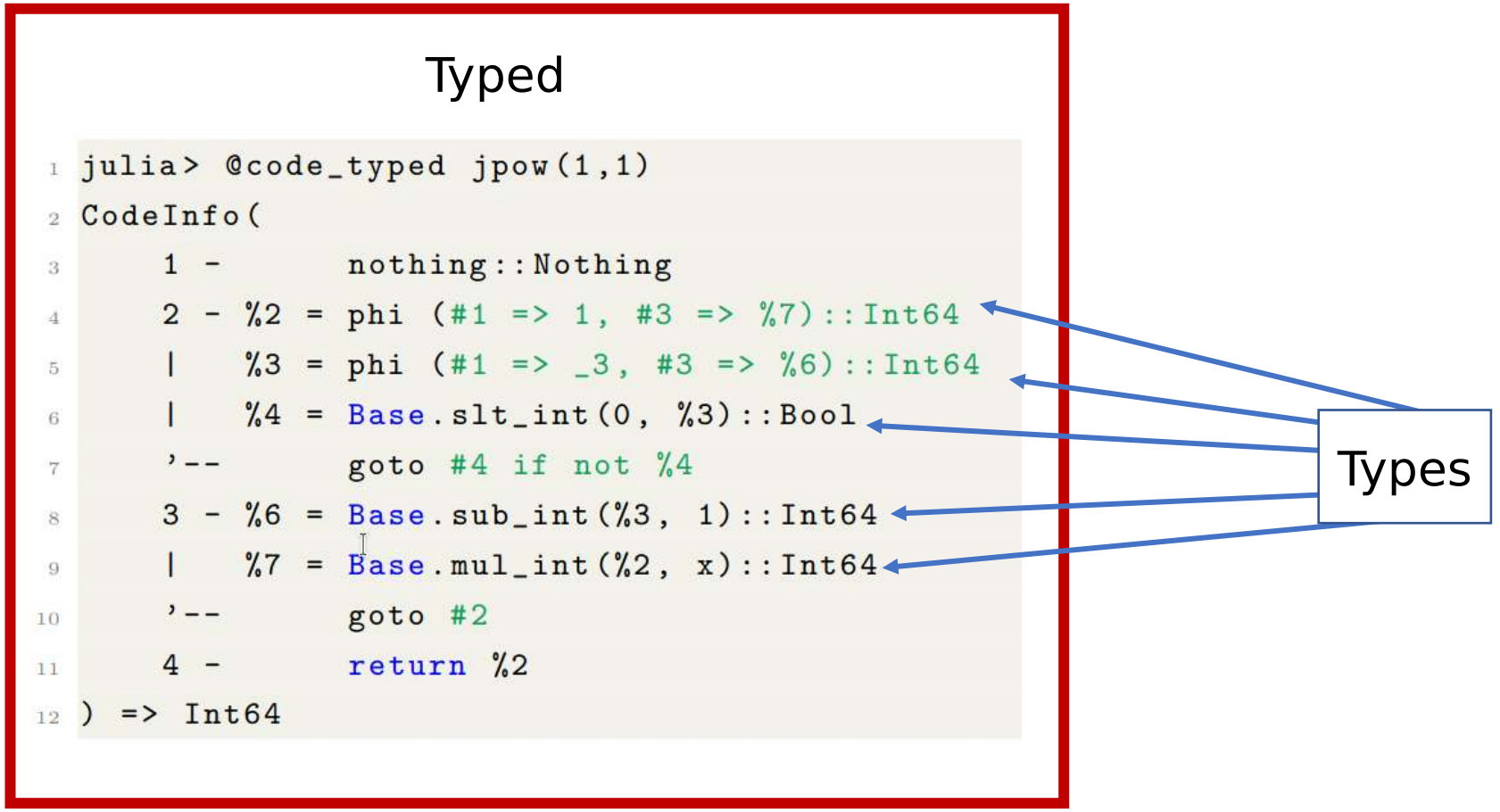
Dataflow Type Inference

Source

```
#basic power function
function jpow(x, n)
    r = 1
    while n > 0
        n -= 1
        r *= x
    end
    return r
end
```

Typed

```
julia> @code_typed jpow(1,1)
CodeInfo(
    1 -       nothing::Nothing
    2 - %2 = phi (#1 => 1, #3 => %7)::Int64
    |   %3 = phi (#1 => _3, #3 => %6)::Int64
    |   %4 = Base.slt_int(0, %3)::Bool
    '--       goto #4 if not %4
    3 - %6 = Base.sub_int(%3, 1)::Int64
    |   %7 = Base.mul_int(%2, x)::Int64
    '--       goto #2
    4 -       return %2
) => Int64
```



Multiple Dispatch

- Improves readability
- Type-optimised methods

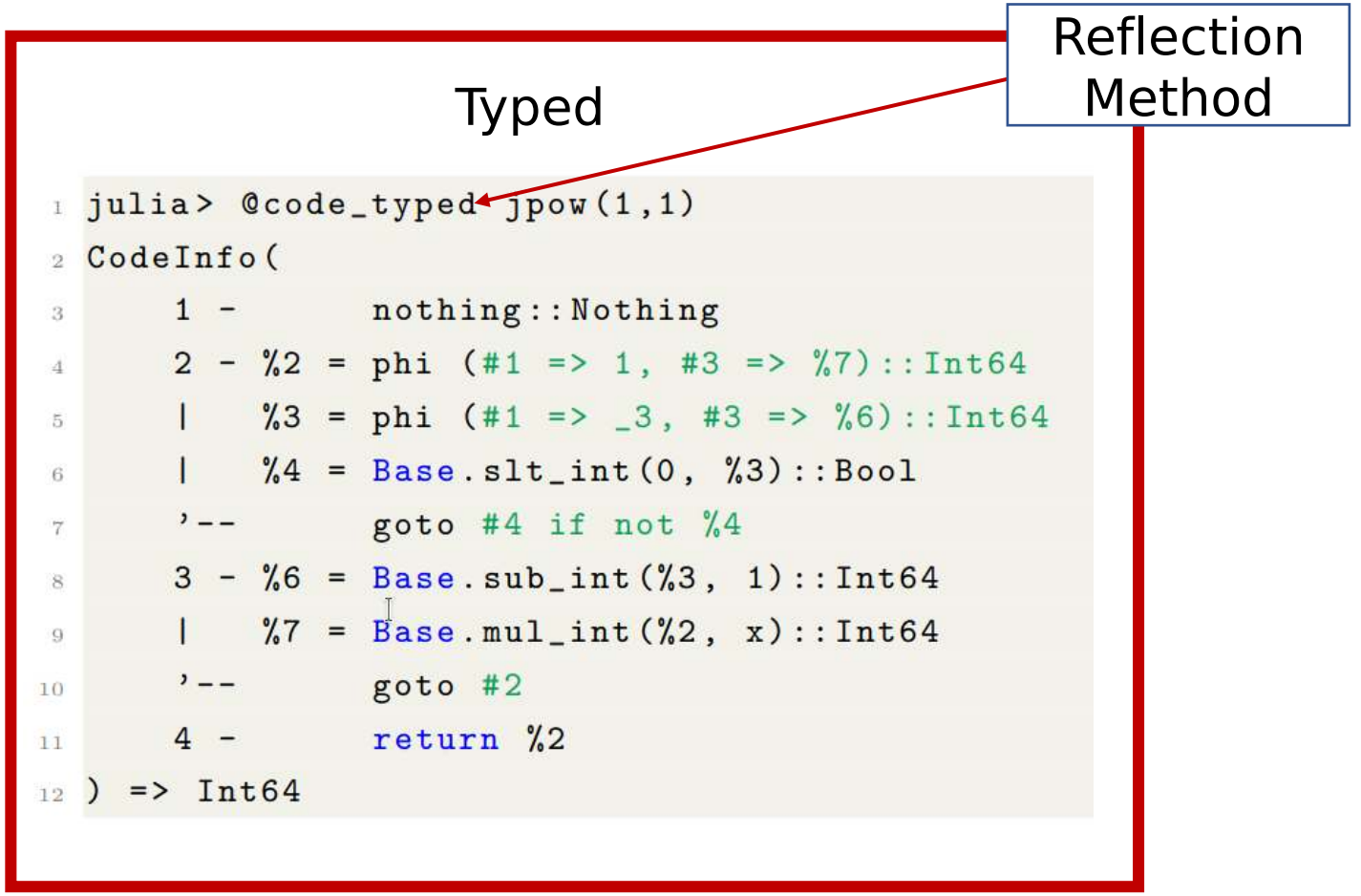
Typed

```
julia> @code_typed jpow(1,1)
CodeInfo(
    1 -       nothing::Nothing
    2 - %2 = phi (#1 => 1, #3 => %7)::Int64
    |   %3 = phi (#1 => _3, #3 => %6)::Int64
    |   %4 = Base.slt_int(0, %3)::Bool
    '--       goto #4 if not %4
    3 - %6 = Base.sub_int(%3, 1)::Int64
    |   %7 = Base.mul_int(%2, x)::Int64
    '--       goto #2
    4 -       return %2
) => Int64
```

Dispatched Functions

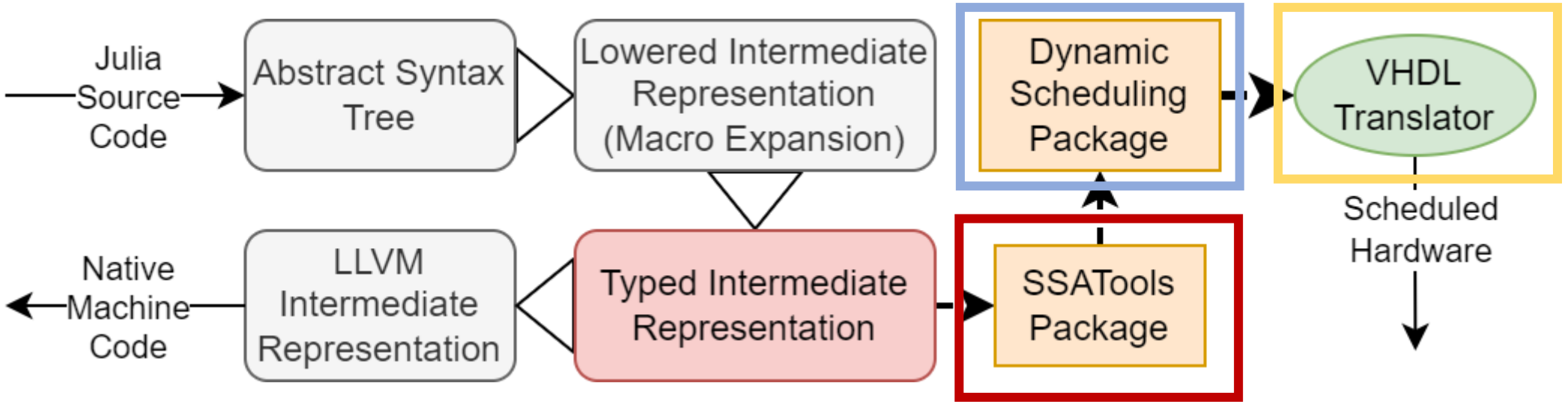
Meta-programming

- Code insertion
- Code reuse
- Reflection methods



Current Toolflow

- SSATools.jl package
  - *CDFG*
- DynamicScheduling.jl package
  - *DOT Graph*
- dot2vhdl (Dynamatics) translator



Testing

- Comparable resource usage to Dynamatics
- Lacking basic block reduction

Program	Basic Blocks		Components		LUTs		FFs		DSPs	
	Julia	C++	Julia	C++	Julia	C++	Julia	C++	Julia	C++
`if_else`	5	3	41	32	511	399	595	460	6	6
`power`	4	4	61	64	555	571	671	710	3	3
`newton_raphson`	10	6	225	147	4006	3319	3456	2961	18	12

Future Work

- Memory support – on/off chip
- Basic block optimization passes
- Static Scheduling
- Support larger subset of language and libraries
- Support programming host and accelerator

Imperial College London

julia

Questions

Benjamin Biggs, Ian McInerney, Eric Kerrigan, and George Constantinides

Imperial College London, The University of Manchester

{bb2515, e.kerrigan, g.constantinides}@imperial.ac.uk, ian.mcinerney@manchester.ac.uk