Evaluating DSP with Dynamic Reconfigurable Processing Units

Dresden University of Technology
Department of Computer Science
Institute of Computer Engineering

Steffen Köhler, Sergej Sawitzki, Rainer G. Spallek
{stk,sawitzki,rgs}@ite.inf.tu-dresden.de

Schloß Dagstuhl, 26th to 30th June 2000

Contents

- Background and Motivation
- Characterization of modern DSP architectures
- Acceleration of DSP using reconfigurable processing units
- Design exploration using retargetable compiler
- Prototype implementation aspects
- Conclusions and further work

Background

- The processing of streaming data can be efficiently done by application specific signal processors
- Performance of these DSP is not always acceptable to be only sufficient for a very limited scale of special purpose applications
- Another fact is the problem of low average utilization possibilities of multiple DSP units (e.g. VLIW or SIMD architectures)
- Dynamic reconfigurable DSP processing units provide a more flexible and efficient way for implementation of special purpose application system designs

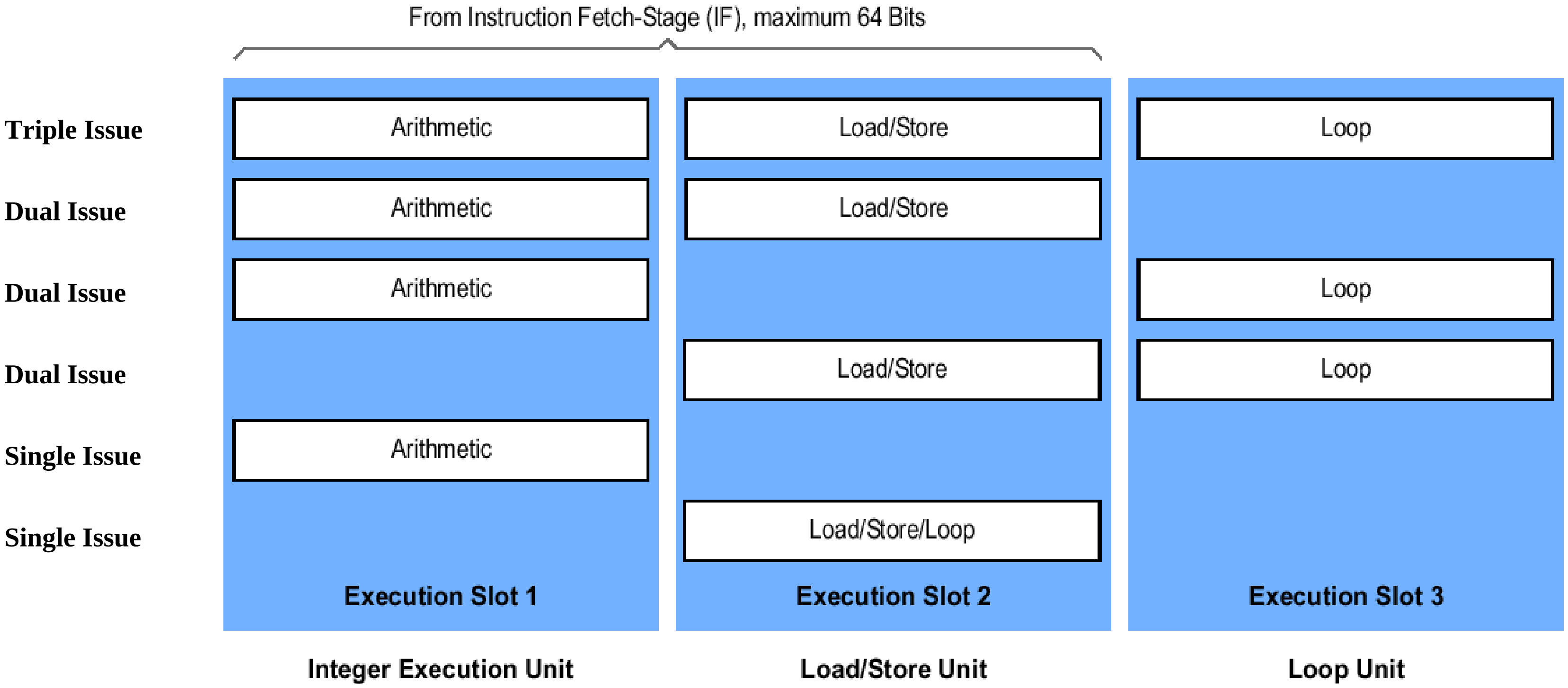
Limits of today's DSP

- DSP are designed for a strait segment of special purpose applications
- The application domain may change and may not be considered as fixed at the beginning of the design cycle
- An acceptable functional processing density can be achieved only in very rare cases.
- Performance enhancements are only possible through hardware exchange.

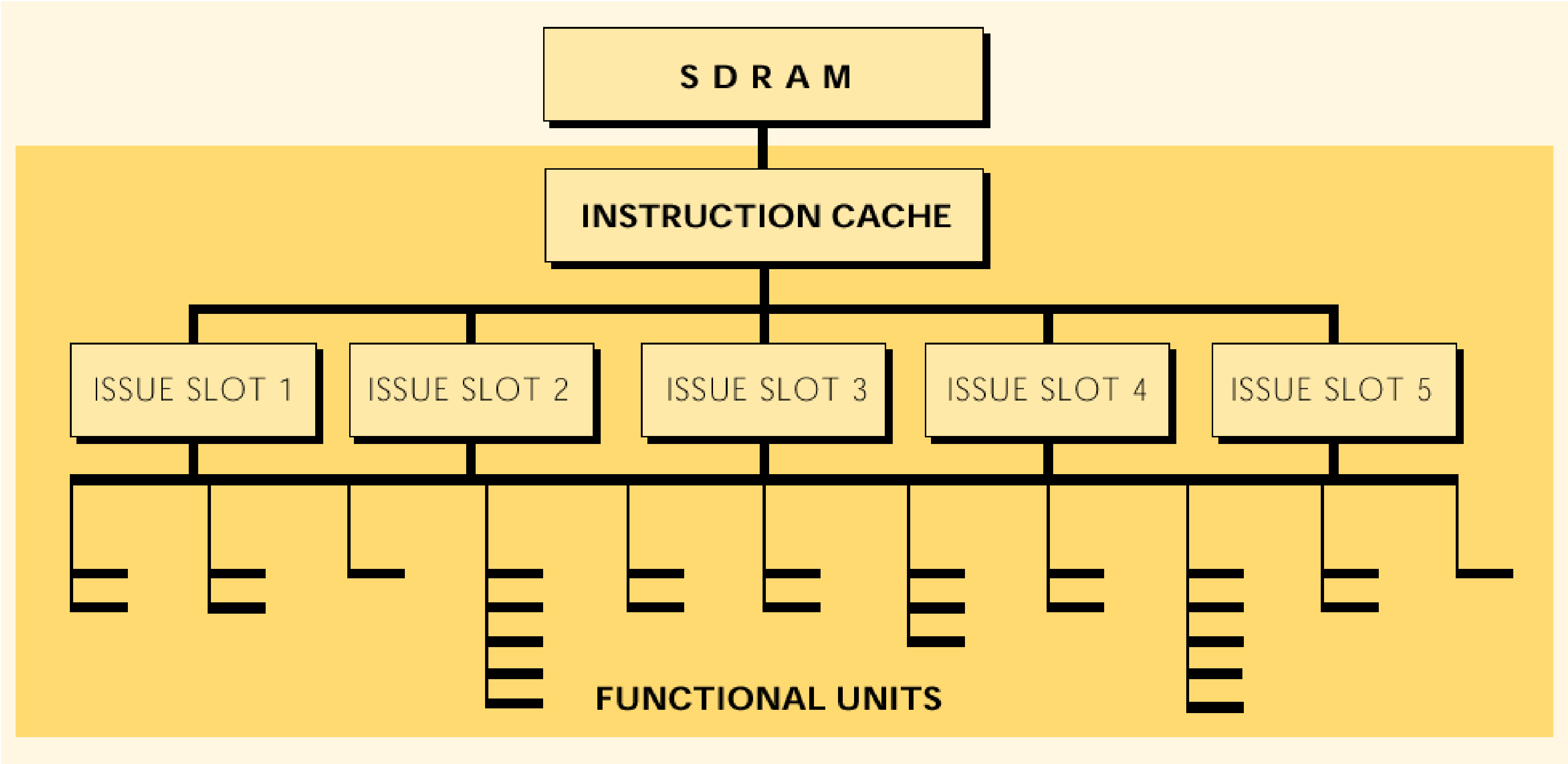
Possible Solutions

- Increasing the number of processing units and instruction level parallelism (VLIW, SIMD)
- Improvement of physical design parameters (clock rate)
- Introduction of new computing technologies
- Component based DSP hardware and software design evaluation

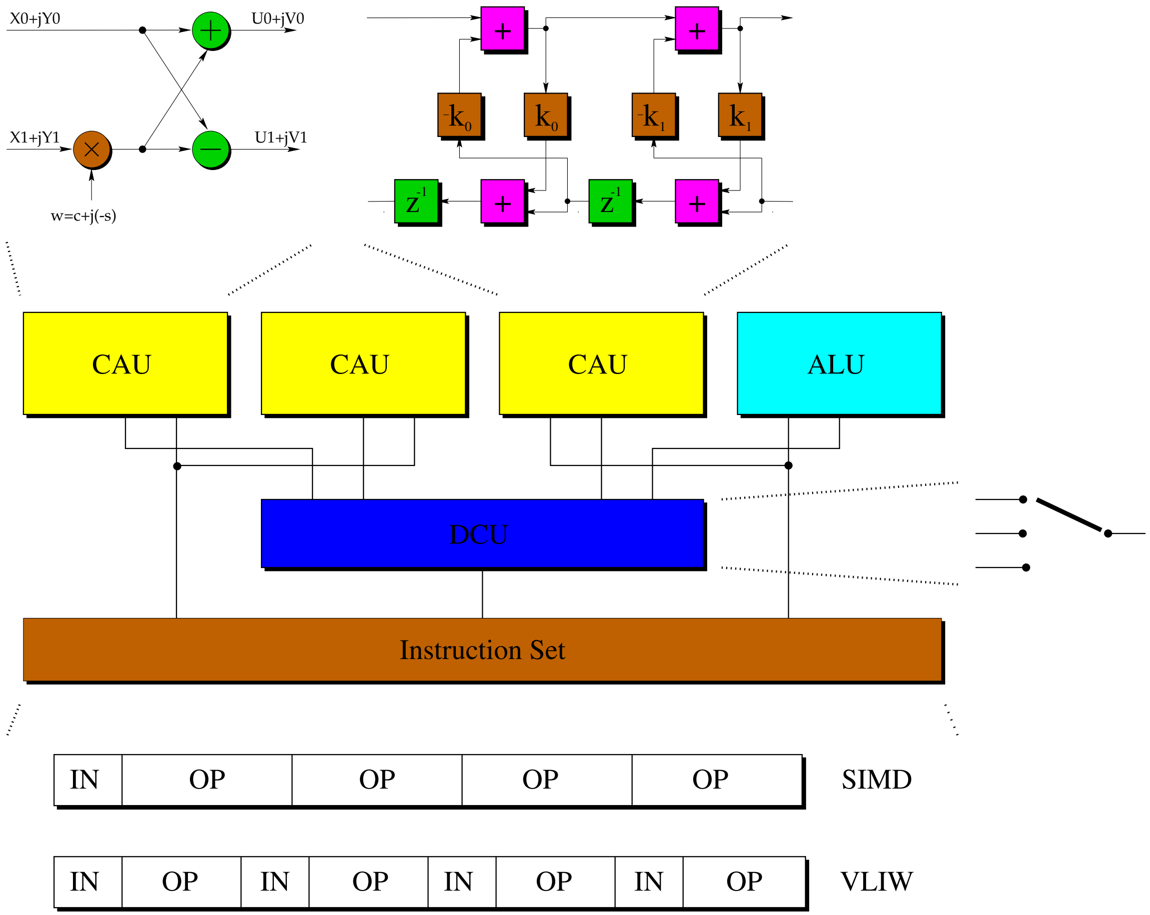
TriCore DSP Execution



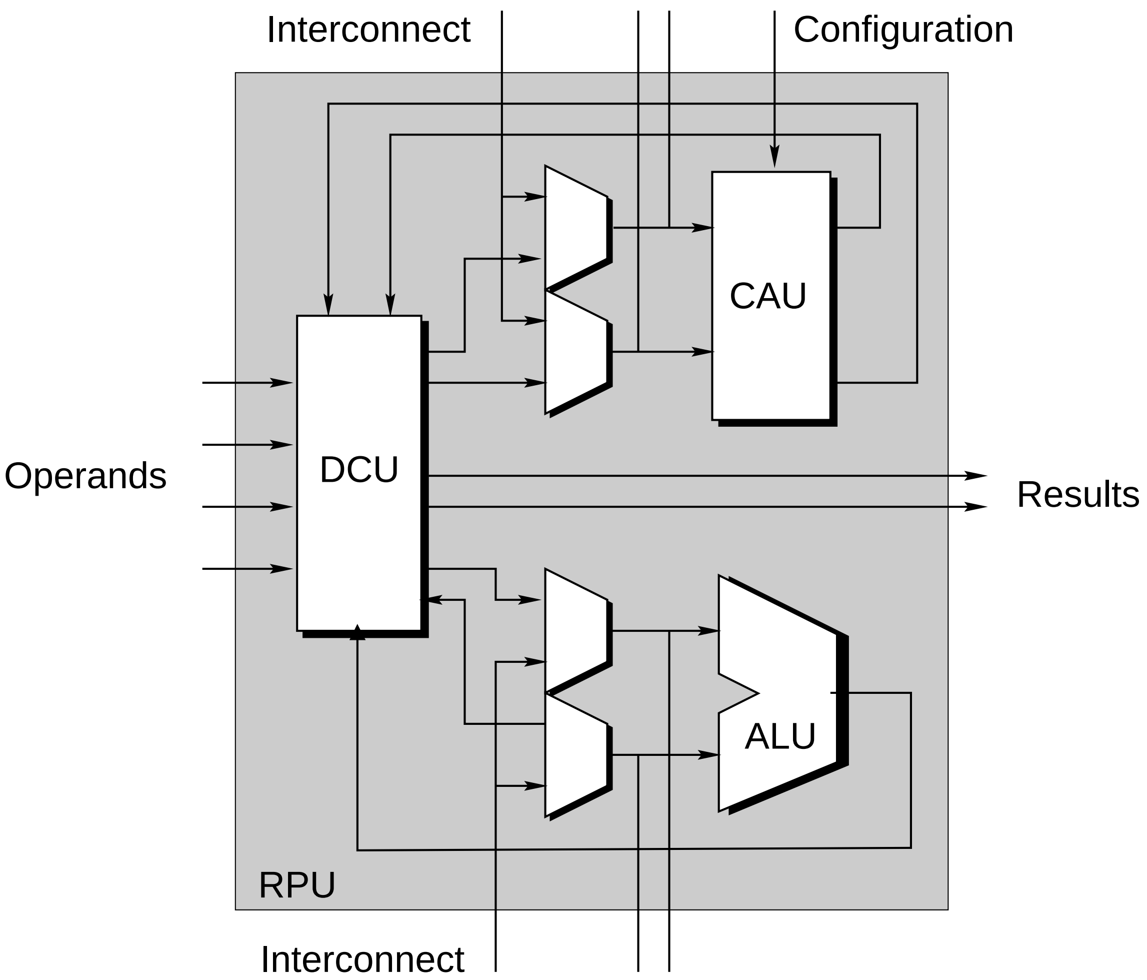
TriMedia Processor



Reconfigurable DSP Architecture



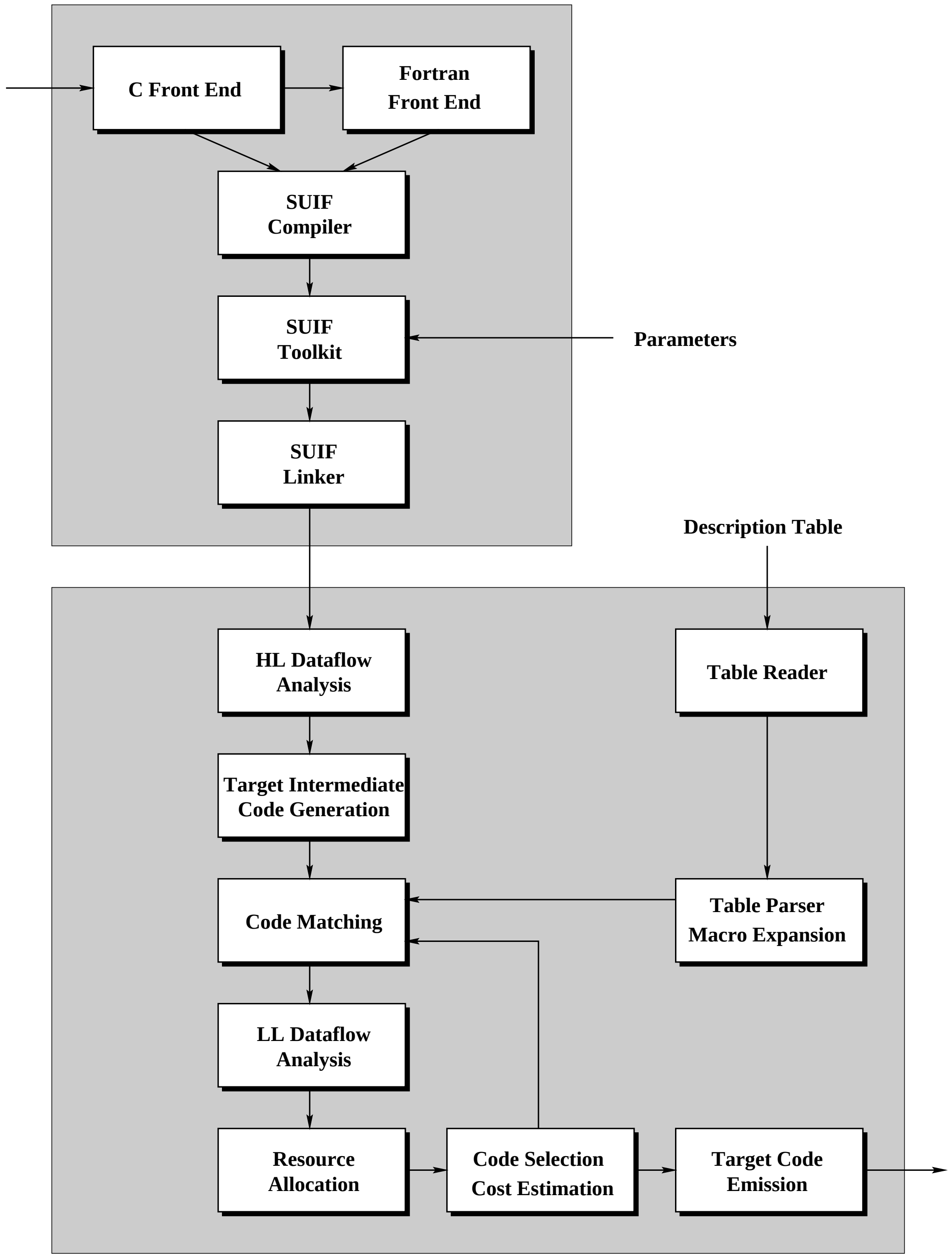
Reconfigurable Processing Unit



Profiling Analysis

Execution time %	Number of calls	ms/call	Function
37.00	578022	0.21	quantize
13.36	23944	1.84	fft1024
10.69	430992	0.08	L3_subband_filt
8.96	71832	0.41	fft256
5.98	766208	0.03	dct

Retargetable Compiler



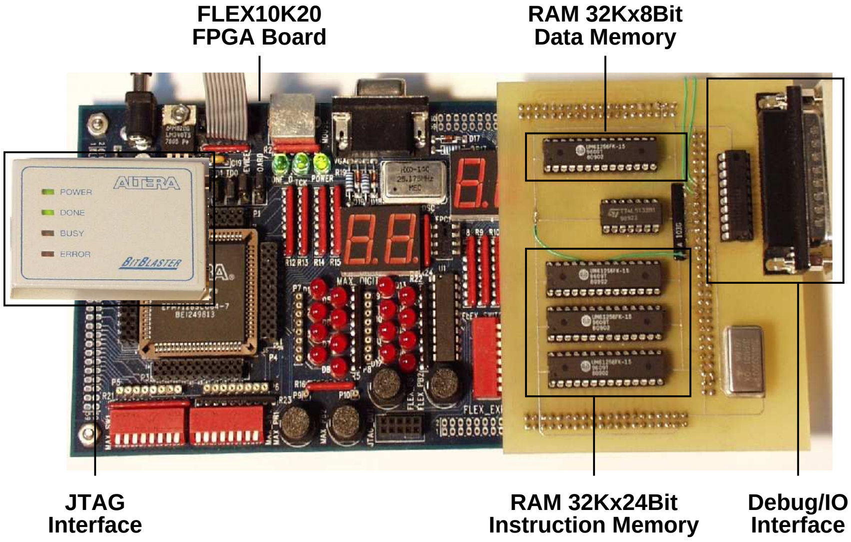
SUIF Instruction Tree

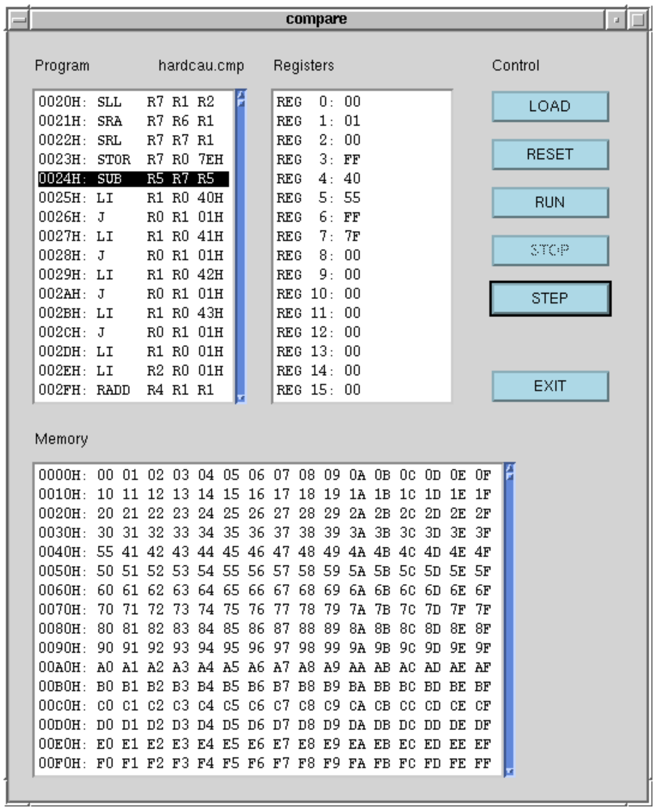
```
add t:g4 (i.32) fft.e2 = fft.e1, fft.dist
sub t:g10 (f.32) fft.re_tmp = e1, e2
e1: mul t:g10 (f.32) fft.re_twiddle, e3
  e3: lod t:g10 (f.32) e4
    e4: array t:g49 (p.32) e5+0, size(32), index(fft.e2),
          bound(e6)
      e5: ldc t:g74 (p.32) <fft.re,0>
      e6: ldc t:g4 (i.32) 512
e2: mul t:g10 (f.32) fft.im_twiddle, e7
  e7: lod t:g10 (f.32) e8
    e8: array t:g49 (p.32) e9+0, size(32), index(fft.e2),
          bound(e10)
      e9: ldc t:g74 (p.32) <fft.im,0>
      e10: ldc t:g4 (i.32) 512
```

Translation Table

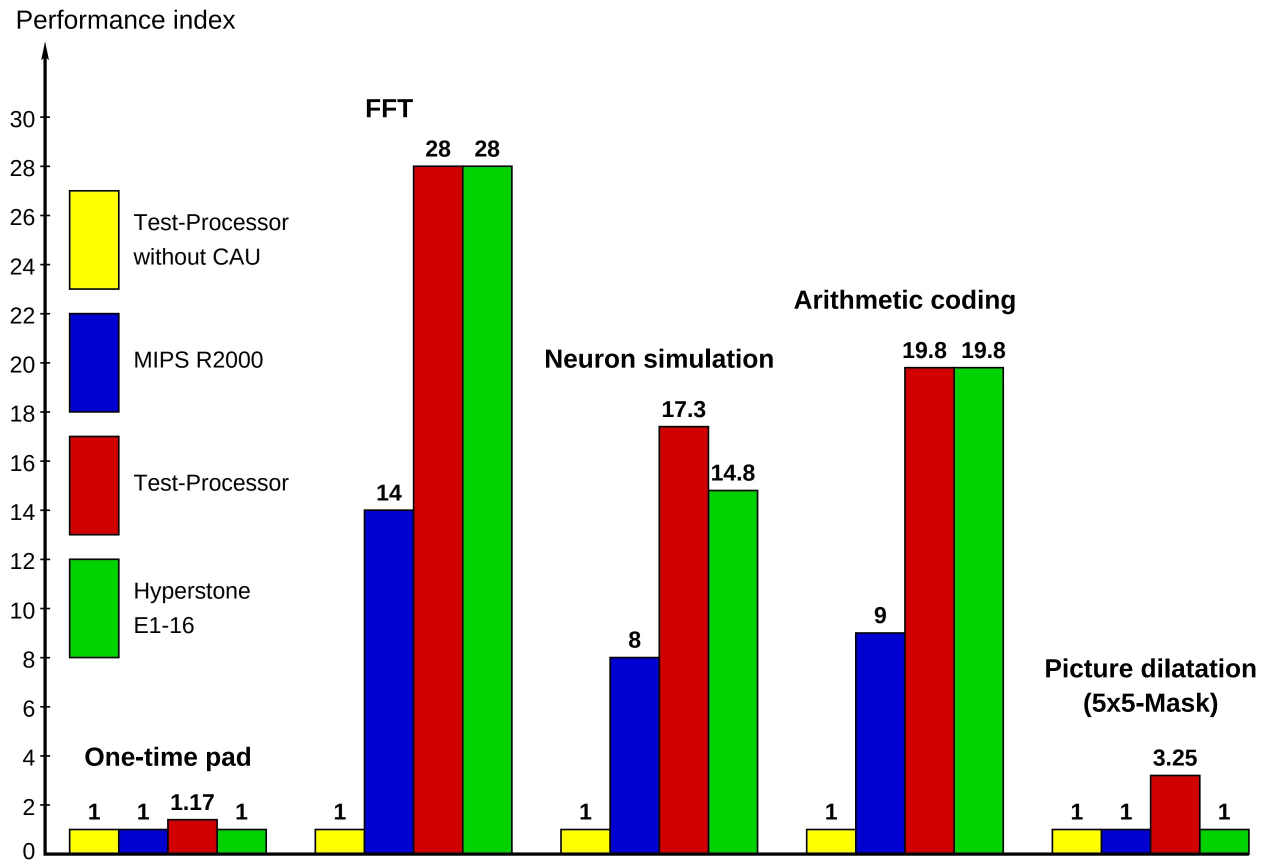
```
mul (drtu16 drtu16 slu16 slu16 slu16 slu16)
{
        ldw R1 &S2 BP    :3:R1:S2,BP
        ldw R2 &S3 BP    :3:R2:S3,BP
        mul D0 R1 R2     :5:D0:R1,R2
        ldw R2 &S0 BP    :3:R2:S0,BP
        ldw R3 &S1 BP    :3:R3:S1,BP
        mul D1 R2 R3     :5:D1:R2:R3
}
mul (drtu16 drtu16 slu16 slu16 slu16 slu16)
{
        mul D0 D1 &S1 &S0 &S3 &S2    :10:D0,D1:S1,S0,S3,S2
}
```

Prototype Environment

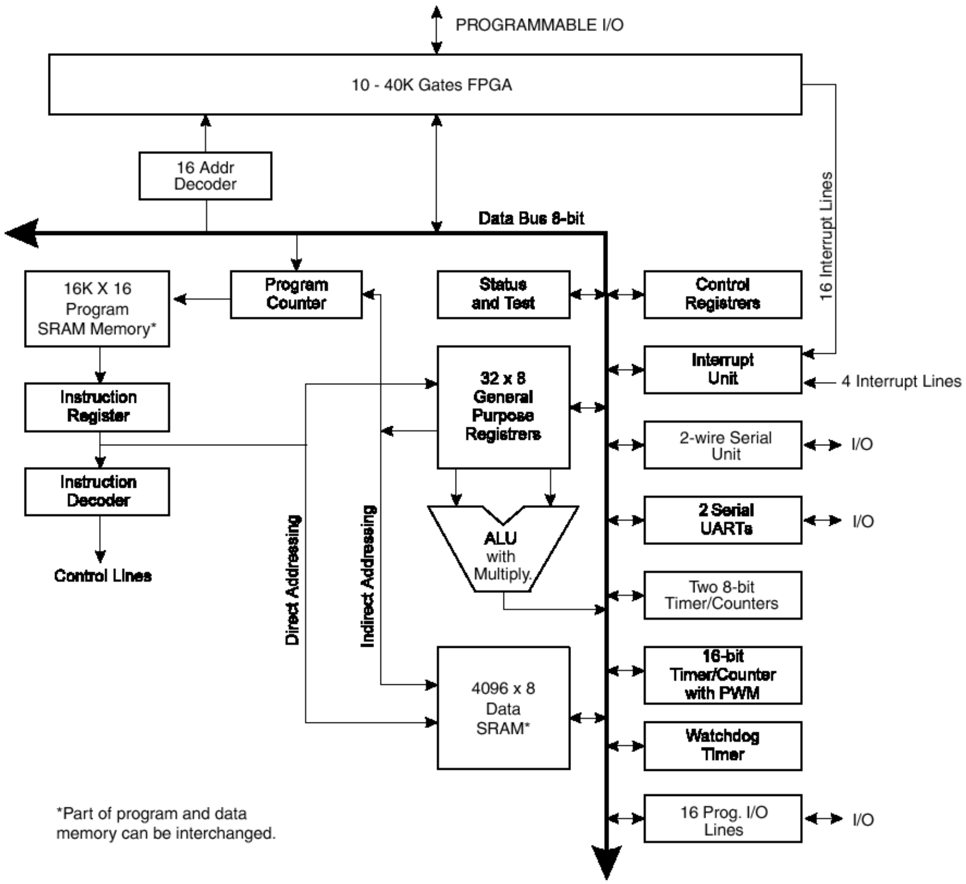




Performance Results



FPSLIC Prototype



Conclusions and Further Work

- Concepts for Reconfigurable Prototype Processors
  - Usability and utilization of multiple reconfigurable processing units (VLIW, SIMD)
  - Reconfigurable data paths as interconnect structures for functional units
  - Reconfigurable peripheral communication interfaces
  - Take advantage of real dynamic reconfigurability

- Hardware/Software DSP component design
  - Creation of hypothetic processor architectures through definition of instruction set/hardware execution translation table
  - Target system evaluation using interactive simulator/profiler (behavior level)