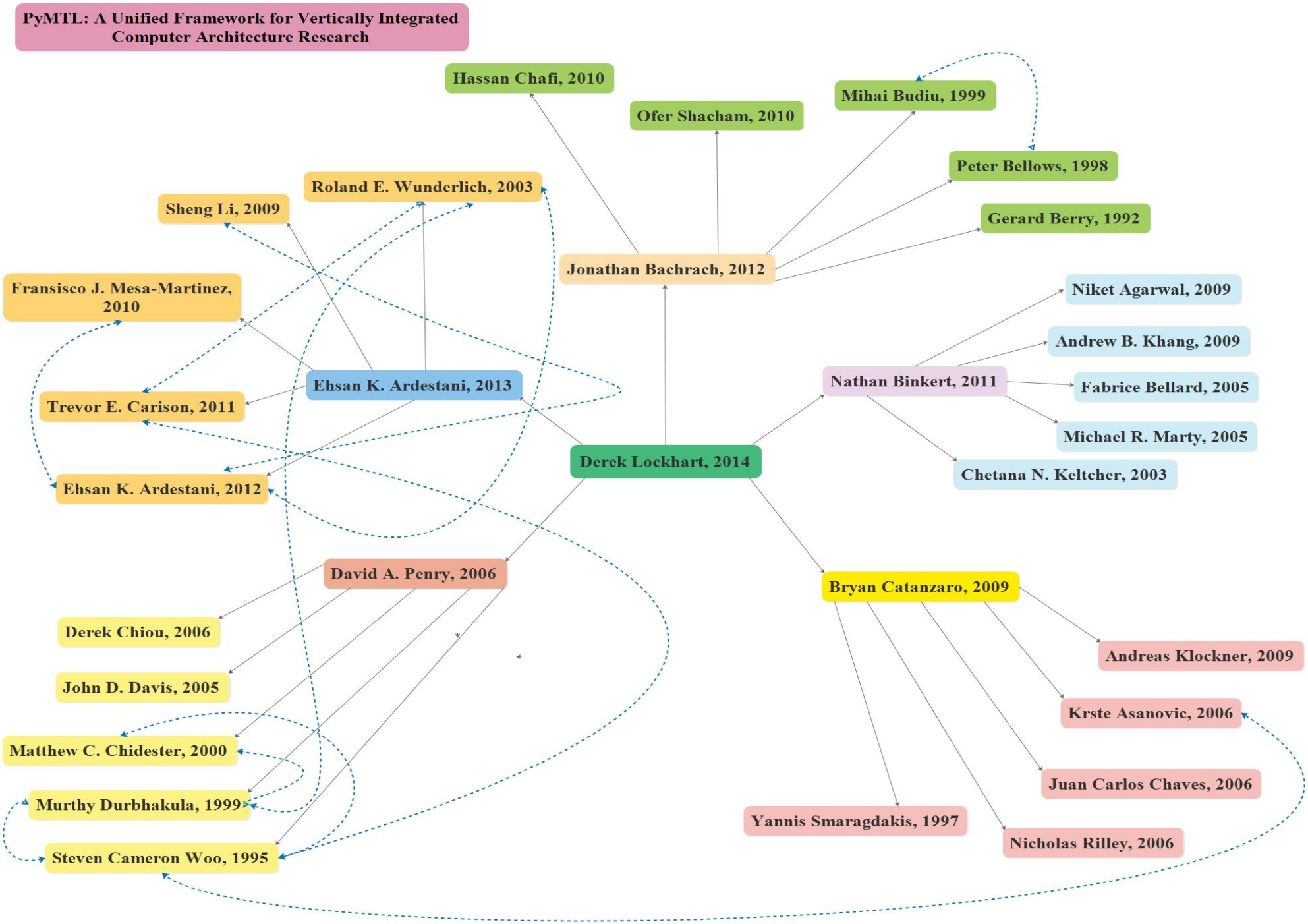
TEKNIK PENULISAN KARYA ILMIAH

Nama : Fitriah Wulandari

NIM : 09011181520022

Kelas : SK2A

FAKULTAS ILMU KOMPUTER
JURUSAN SISTEM KOMPUTER
UNIVERSITAS NEGERI SRIWIJAYA


PyMTL: A Unified Framework for Vertically Integrated Computer Architecture Research
Hassan Chafi, 2010
Ofer Shacham, 2010
Mihai Budiu, 1999
Peter Bellows, 1998
Gerard Berry, 1992
Roland E. Wunderlich, 2003
Sheng Li, 2009
Jonathan Bachrach, 2012
Fransisco J. Mesa-Martinez, 2010
Niket Agarwal, 2009
Andrew B. Khang, 2009
Nathan Binkert, 2011
Fabrice Bellard, 2005
Michael R. Marty, 2005
Chetana N. Keltcher, 2003
Ehsan K. Ardestani, 2013
Trevor E. Carison, 2011
Derek Lockhart, 2014
Ehsan K. Ardestani, 2012
David A. Penry, 2006
Bryan Catanzaro, 2009
Derek Chiou, 2006
John D. Davis, 2005
Andreas Klockner, 2009
Krste Asanovic, 2006
Matthew C. Chidester, 2000
Juan Carlos Chaves, 2006
Murthy Durbhakula, 1999
Yannis Smaragdakis, 1997
Nicholas Rilley, 2006
Steven Cameron Woo, 1995

PyMTL: A Unified Framework for Vertically Integrated Computer Architecture Research

Dari hasil paper yang saya cari dengan menggunakan google scholar, saya mengambil judul "PyMTL: A Unified Framework for Vertically Integrated Computer Architecture Research". Penulis paper pertama yang saya ambil adalah Derek Lockhart (2014). Penulis menjelaskan bahwa, teknologi cenderung mendorong arsitektur untuk mempertimbangkan hal yang tidak sejenis dan khusus perangkat keras ada peningkatan kebutuhan metodologi penelitian yang terintegrasi secara vertikal itu bisa efektif menilai hasil, tempat, dan energi metrik untuk masa depan arsitektur. Tetapi, membangun metodologi seperti itu dengan alat yang ada merupakan sebuah tantangan yang penting untuk bahasa yang unik, bentuk pola, dan alat-alat bekas di tingkat fungsional (FL), tingkat siklus (CL), dan tingkat pemindahan daftar pemodelan (RTL). Kami memperkenalkan kerangka baru yang disebut PyMTL yang bertujuan untuk menutup arsitektur komputer ini metodologi penelitian dengan menyediakan lingkungan desain terpadu untuk pemodelan FL, CL, dan RTL. PyMTL memanfaatkan bahasa pemrograman phyton untuk membuat bahasa domain tertentu yang sangat produktif bagi struktur yang sama pemodelannya dan desain perangkat keras. Sementara itu, penggunaan phyton sebagai model dan kerangka implementasi bahasa memberikan manfaat yang cukup besar dalam hal produktivitas. Ia datang dengan harga yang mantap untuk meniru waktu yang lebih panjang. Kami mengatasi kesenjangan produktivitas kinerja ini dengan kompilasi hybrid JIT dan pendekatan JIT khusus. Kami memperkenalkan SimJIT, kebiasaan mesin JIT khusus yang secara otomatis mengoptimalkan C++ untuk model CL dan RTL. Untuk mengurangi dampak kinerja kode terspesialisasi yang tersisa, kami menggabungkan SimJIT dengan off-the-shelf phyton dengan JIT compiler-meta-tracing (PyPy). SimJIT + PyPy memberikan kecepatan tinggi hingga 72× untuk model CL dan 200× untuk model RTL, kami membawa 4-6× dioptimalkan C++ kode sambil memberikan manfaat yang penting dalam hal produktivitas dan kegunaan. Paper ini mensitasi 5 paper dari referensinya sendiri, 5 paper tersebut yaitu :

1. Paper dari Ehsan K. Ardestani (2013). Dia menjelaskan bahwa, Arsitek mengandalkan simulasi dalam eksplorasi mereka dari ruang desain. Namun, lambat

kecepatan simulasi produktivitas mereka dan membatasi kedalaman eksplorasi mereka. Pengambilan sampel biasanya telah menjadi obat yang umum digunakan. Sementara contoh itu terbukti menjadi teknik yang efektif untuk pengolah inti itu sendiri, penerapannya telah terbatas pada simulasi program multi, hanya sepanjang aplikasi . Karya ini menyajikan Time-Based Sampling (TBS), kerangka yang pertama untuk memungkinkan pengambilan sampel dalam simulasi prosesor multicore dengan tidak ada batasannya dalam hal jenis aplikasi (multiprogrammed atau multithreaded), jumlah core, homogenitas atau heterogenitas dari konfigurasi simulasi (rata-rata 4,99% error di semua konfigurasi yang dievaluasi). TBS adalah benda yang pertama untuk mengaktifkan kekuatan terpadu dan suhu yang dilakukan dalam evaluasi simulasi statistik sampel dari sistem multicore (dengan masing-masing 5,5% dan rata-rata 2,4% error). Kami menerapkan simulator arsitektur berdasarkan TBS, yang disebut ESESC, yang menyediakan satu set alat holistik untuk evaluasi yang cukup dari arsitektur yang berbeda.

2. Paper dari Jonathan Bachrach (2012). Dalam paper ini kami memperkenalkan *pahat*, perangkat keras baru konstruksi bahasa yang mendukung desain perangkat keras canggih menggunakan generator yang sangat parameter dan bahasa perangkat keras domain khusus berlapis. Dengan menempelkan pahat di bahasa pemrograman, kita bisa meningkatkan desain perangkat keras dengan memberikan konsep abstraksi termasuk orientasi objek, pemrograman fungsional, jenis parameter, dan contoh kesimpulan. Pahat bisa menghasilkan kecepatan tinggi C ++ -berdasarkan siklus-simulator perangkat lunak yang akurat atau tingkat rendah Verilog dirancang untuk memetakan baik FPGA atau ASIC standar untuk aliran sintesis. Paper ini menyajikan pahat, ia menempelkan di Scala, contoh perangkat keras, dan hasil untuk simulasi C ++ , emulasi Verilog dan sintesis ASIC.

3. Paper dari Nathan Binkert (2011). Dia menjelaskan bahwa infrastruktur simulasi gem5 adalah penggabungan aspek terbaik dari M5 dan GEMS simulator. M5 sangat menyediakan kerangka kerja yang dapat dikonfigurasi simulasi, banyaknya ISA, dan bermacam-macam model CPU. Komplemen GEMS fitur ini dengan rinci dan fleksibelnya sistem memori, termasuk dukungan untuk beberapa hubungan protokol dan model interkoneksi. Saat ini, gem5 mendukung sebagian komersial ISA (ARM, ALPHA, MIPS, Power, SPARC, dan x86), termasuk boot Linux pada tiga dari mereka (ARM, ALPHA, dan x86).

Proyek ini merupakan hasil upaya gabungan dari banyak akademis dan lembaga industri, termasuk AMD, ARM, HP, MIPS, Princeton, MIT, dan Universitas Michigan, Texas, dan Wisconsin. Selama sepuluh tahun terakhir, M5 dan GEMS telah digunakan di ratusan publik dan telah puluhan ribu kali di download. Tingginya tingkat kolaborasi pada proyek gem5, dikombinasikan dengan keberhasilan sebelumnya dari komponen dan lisensi BSD seperti liberal, membuat gem5 menjadi alat penuh sistem simulasi yang berharga.

4. Paper dari Bryan Catanzaro (2009). Dia menjelaskan bahwa "Produktivitas Tinggi" bahasa pemrograman ini seperti Python yang kekurangan kinerja program bahasa "efisiensi" (CUDA, Cilk , C dengan OpenMP) yang dapat memanfaatkan pengetahuan programmer sejajar dengan arsitektur perangkat keras. Kami menggabungkan kinerja efisiensi bahasa dengan program abilitas daya produksi bahasa menggunakan bahasa selektif khusus di waktu yang tepat (SEJITS). Di perjalanan waktu, kami mengkhusus pada (menghasilkan, mengumpulkan, dan mengeksekusi kode sumber bahasa untuk efisiensi) aplikasi tertentu dan podium terntentu dari bahasa produktivitas, sebagian besar tak terlihat untuk aplikasi programmer. Sebab mesin khusus itu menerapkan daya produksinya sendiri, itu adalah hal yang mudah bagi programmer untuk efisiensi secara bertahap menambahkan hal khusus untuk gambaran umum daerah, perangkat keras baru, atau keduanya. SEJITS memiliki potensi untuk penghubung penelitian lapisan daya produksinya dan penelitian lapisan efisiensi, memungkinkan ahli domain untuk memanfaatkan arsitektur perangkat keras sejajar dengan pecahan waktu dari programmer dan biasanya diperlukan usaha.

5. Paper dari David A. Penry (2006). Dia menjelaskan bahwa Simulasi merupakan cara penting untuk mengevaluasi mikro arsitektur baru. Tren saat ini menuju chip multipro (CMPS) mencoba mengembangkan kemampuan perancang untuk mencukupi simulator yang efisien. Kecepatan simulasi CMP dapat ditingkatkan dengan memanfaatkan kesejajaran dalam simulasi model CMP. Hal ini dapat dilakukan dengan baik untuk menjalankan simulasi pada prosesor ganda atau dengan mengintegrasikan beberapa prosesor ke dalam simulasi untuk menggantikan prosesor simulasi. Jadi biasanya kategori itu membutuhkan paralelisasi manual yang membosankan atau bentuk untuk merangkum prosesor.

Kedua masalah itu dapat dihindari dengan menghasilkan simulator dari model

struktural bersamaan dengan CMP. Model seperti itu tidak hanya menyerupai perangkat keras, sehingga penggunaannya mudah untuk dimengerti dan juga menyediakan informasi yang cukup secara otomatis sejajar dengan simulator tanpa memerlukan perubahan model manual. Selanjutnya, komponen-komponen individu dari model itu seperti prosesor dapat digantikan dengan perangkat keras tanpa memerlukan partisi ulang.

Paper ini menyajikan teknik untuk melakukan paralelisasi simulator otomatis dan integrasi perangkat keras untuk model struktural CMP. Kami menunjukkan bahwa paralelisasi otomatis dapat mencapai kecepatan tinggi 7.60 untuk model CMP 16-prosesor pada 4-prosesor memori bersama multiprosesor konvensional. Kami menunjukkan tenaga dari perangkat keras dengan mengintegrasikan delapan inti perangkat keras PowerPC menjadi model CMP, mencapai kecepatan tinggi hingga 5,82.

Layer ke 2 juga mensitasi paper pada layer ke 3. Masing-masing paper pada layer ke 3 diambil dari referensi paper layer ke 2. Pertama paper Ehsan K. Ardestani (2013) mensitasi paper Ehsan K. Ardestani (2012), Trevor E. Carison (2011), Fransisco J. Mesa-Martinez (2010), Sheng Li (2009), dan Roland E. Wunderlich (2003). Kedua paper Joanthan Bachrach (2012) mensitasi paper Hassan Chafi (2010), Ofer Shacham (2010), Mihai Budiu (1999), Peter Bellows (1998), dan Gerard Berry (1992). Ketiga paper Nathan Binkert mensitasi paper Niket Agarwal (2009), Andrew B. Khang (2009), Fabrice Bellard (2005), Michael R. Marty (2005), dan Chetana N. Keltcher (2003). Keempat paper Bryan Catanzaro (2009) mensitasi paper Andreas Klockner (2009), Krste Asanovic (2006), Juan Carlos Chaves (2006), Nicholas Rilley (2006), dan Yannis Smaragdakis (1997). Kelima paper David A. Penry (2006) mensitasi paper Derek Chiou (2006), John D. Davis (2005), Matthew D. Chidester (2000), Murty Durbhakula (1999), dan Steven Cameron Woo (1995).

Kesimpulan yang dapat saya ambil dari paper diatas adalah paper ini telah memperkenalkan PyMTL, mempersatukan, kerangka kerja yang terintegrasi secara vertikal untuk pemodelan FL, CL, dan RTL. Penyelidikan kasus kecil yang digunakan untuk menggambarkan bagaimana PyMTL dapat menutup komputer arsitektur metodologi dengan memungkinkan pembangunan produktif model FL, CL, dan RTL menggunakan desain struktural secara bersamaan dan desain sensitif. Sementara contoh-contoh kecil menunjukkan beberapa tenaga dari PyMTL, kami

percaya PyMTL hanya langkah pertama menuju eksplorasi ruang angkasa desain cepat dan konstruksi template perangkat keras fleksibel untuk melunasi usaha desain. Masa depan pekerjaan merencanakan untuk mengeksplorasi memperluas PyMTL dengan abstraksi desain tingkat tinggi yang lebih meningkatkan produktivitas desainer.

Selain itu, pendekatan hybrid untuk just-in-time optimasi diusulkan untuk menutup kesenjangan kinerja yang diperkenalkan dengan menggunakan Python untuk pemodelan perangkat keras. SimJIT, sebuah JIT khusus kustom untuk model CL dan RTL, dikombinasikan dengan PyPy meta-tracing JIT untuk membawa PyMTL simulasi dari jaringan dalam 4×-6× dioptimalkan kode C ++. Kita berharap untuk lebih mengembangkan SimJIT untuk mendukung lebih banyak jenis PyMTL membangun dan mengeksplorasi spesialisasi optimasi lebih maju lagi..