システムLSIの設計技術・設計支援技術（CAD技術）に関する問題点

藤田 昌宏／東京大学工学系研究科電子工学専攻

ますます大規模化していくシステムLSI設計をいかに短期間にかつ正しく設計していくかという立場からは，設計の上位レベルからの支援が大切である．そこで，通常，主にソフトウェア・ハードウェア協調設計と呼ばれる分野について，CAD技術の研究開発上の問題点を考察したい．

ソフトウェア研究者とハードウェア研究者の協調が本質

システムLSIとは大規模LSIであり，通常，1つのシステムがそのまま入る．したがって，必然的にLSIによるハードウェアだけでなく，その上で実行されるソフトウェアもいっしょに開発することになる．

従来からシステムLSI開発では，LSI自体を設計している人たちと，その上で実行されるべきソフトウェアを開発している人たちは，別々に作業を進めている．これは，設計の最初の段階で，どの部分をハードウェアで実現し，どの部分をソフトウェアで実現するかを決定してしまっているからこそ可能な開発体制であるといえる．

ところが，世の中の動きが激しくなり，設計の最初の段階でソフトウェアとハードウェアを正しく分割することは，ほとんど不可能になっている．設計がある程度進んでから，ソフトウェアとハードウェアの分担を変えたくなる場合が多いのではないかと想像されるが，実現不可能なので，「まあ，しかたないか」ということで済まさざるを得ない．

したがって，設計の最初の段階でソフトウェアとハードウェアの分担をいろいろ試行錯誤できる環境（英語でいえば，design space exploration）が必要で，場合によっては，複数の分割を念頭において，設計を進めるというようなことも必要になる．設計を詳細化した段階で，当初の予定通りの性能があるモジュール（ソフトウェアのこともハードウェアのこともある）で実現できない場合には，最初に考えておいた複数の分割案を使って，巧みにつじつまを合わせようということである．

もっともな考え方であるが，これを実現しようとすると，ソフトウェア技術者とハードウェア技術者が協調してdesign space explorationを行う必要がある．つまり，両方分かる人が必要ということで，現実には簡単ではない．特にその設計を支援するCAD技術の研究開発に関しては，両方をよく理解した人しか行えない．

もともと，ハードウェアの高位レベル合成と呼ばれる技術と，プログラミング言語のコンパイラ技術は共通点が多い．ただし，コスト関数が大きく異なる．前者は処理時間が長くても（場合によっては数日かかっても）よいから，少しでも高品質な合成結果を求めるのに対し，後者はコンパイル時間があまり長くならないことも必須である．

また，システムLSI用のソフトウェアでは，性能重視の立場から高品質なコードが要求される場合や特殊なハードウェアを利用するコードが要求されることも多く，従来のコンパイラ技術だけでは不十分な場合も多い．しかし，コンパイラ技術の研究者がこのような「特殊な」問題にどれくらい興味を持っているか，疑問である．

一方，近い将来，システムLSIはその規模からいって，複数の専用プロセッサを専用ネットワークで接続したような形の「専用並列計算機システム」になると考えられる．当然のこととして，従来から研究されてきている「汎用並列計算機システム」に対するソフトウェア・ハードウェア設計技術が重要になる．しかし，ここでもハードウェアはケチれるならできるだけケチるとか，低消費電力を最重要と考えて並列計算機を設計するとかという，従来とは異なるコスト関数となり，従来技術がそのまま使えるわけではない．

ここでも，従来の並列計算機研究者とLSI技術研究者の協調が重要となる．このように，システムLSIは研究に対しても，新しい尺度を要求しており，従来の研究者間での協調がとにかく重要で，そのための土台を何らかの方法で確立する必要がある．でもまだ何もできていない．

プログラミング言語による設計記述は高位レベルの記述で，かつ合成可能でないと意味がない

現在，プログラミング言語，特にC言語（あるいはCベースの言語）からのハードウェア自動生成やソフトウェア・ハードウェア協調システムの自動生成が大きな話題となっている．これは，「もともとCを知っている人の人数」は「もともとVerilogやVHDLなどのハードウェア記述言語を知っている人数」よりはるかに多いので，C言語から自動合成できるようになれば，ハードウェア設計技術者以外でも素早く設計できると考えられることが基本となっている（この議論でいくと，近い将来「C」が「Java」に置き換えられることも考えられるが）．

また，組み込みシステムや通信系システムでは，もとも

とCなどのプログラミング言語で仕様が記述されており，現状ではアルゴリズムの検証はCで行われているという事実もある．

この考え方に立つと，RTL（Register Transfer Level）をC言語で記述するという場合に，通常のハードウェア設計とまったく同じように，今までVerilogやVHDLのシンタックスで書いていたものをC言語のシンタックスに直しただけでは，ほとんど意味がない．ハードウェア固有の部分がすべて見えてしまうような書き方をしてしまうと，やはりハードウェア技術者しか使えないことになってしまう．その上，ハードウェアを強く意識して書くと，たとえC言語で記述してもシミュレーション速度はVHDLやVerilogと比較して速くならない（同じことを書き，同じように処理されるのだから当然といえば当然である）．

そうではなくて，できるだけC言語に近い形で記述する必要がある．言い換えると，C言語を知っている人なら，素早くハードウェアの記述法を習得できるか否かがポイントである．

これは，RTLだけではなく，より高位の設計記述が行えることも重要であることを意味する．より上位のレベルから自動合成することにより，設計の効率化が大きく進むと考えられる．

また実際には，C言語の枠内のみで記述するという立場とC言語を少し拡張（あるいは修正）するという立場がある．「ハードウェアは本質的に並列システムである」という事実からすると，順序記述をターゲットとしているC（あるいはC++）を拡張なしで利用するには，何らかの工夫やトリックが必要で，場合によってはかえって分かりにくくなる可能性もある．このあたりは今後の研究次第であるが，記述の分かりやすさを最重要視する立場が望まれる．たとえばC言語を拡張して，ユーザが並列性を陽に指定するのも悪くないと考えられる．

実際には，C言語でシミュレーションのみ可能な記述ができる／できないという議論をしている場合が多く，むなしい．

設計検証は最重要なのに日本では研究されていない

これは言い過ぎであるが，少なくとも形式的検証の研究者間では，一般的にそういう認識になっている．実際，近年目立った研究成果の発表がないのは事実である．

システムLSIの設計では，設計している期間よりも設計の正しさを確認している期間のほうがずっと長いのが普通である．日本でも検証ツールに対する興味や実際の需要が大きいことを考えると，なぜ活発に研究されないのか不思議である（もっとも，これはCAD技術一般にいえることであるが）．

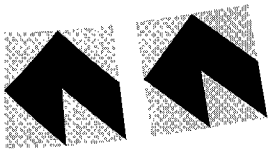
形式的検証技術と呼ばれる，設計の正しさを数学的に証明することで検証を行う技術に関しては，まだ適用可能な設計規模が小さい．そのため，ツールの使い方や，適用範囲の見極め方次第で有用なツールにもなるし，まったく使えないツールにもなる．そこで，形式的検証に関するしっかりした理解を持つエンジニアが重要となり，その意味でも形式的検証を大学などで研究する意義が大きいはずである．なぜ，みなさん研究しないのだろうか？

最後に

以上，いくつかの論点を示した．現在，日本は，システムLSIなどハードウェア用CADシステムに関する技術では，研究面，開発面ともに海外と比較し，大きく遅れている．CADに関してはユーザに徹すればよいという考え方もあると思うが，それで本当にシステムLSIの設計で優位性を出せるのか，大きな疑問である．ツールはどんどん複雑になり，使われ方次第で大きく性能が変わるようになってきている．

また，すでに，システムLSI設計技術やCAD技術の研究開発のための方策はいろいろ施されつつあると思うが，あまり「国産」とか「日の丸」と考えないようにできればと思う．純国産でやろう，あるいはそれがよいことだと思っているのは，世界中でも日本だけなのだから．

（2001.3.8）



システムLSI設計と教育

安浦 寛人／九州大学システムLSI研究センター

藤田さんのご意見に基本的に賛同します．問題点は，ご指摘のとおりですが，では，どうすれば解決できるかに踏み込んだご意見をいただきたいと思います．

私の私見を少し過激に述べさせていただきます．藤田さんがご指摘の

1) ソフトウェア研究者とハードウェア研究者の協調
2) プログラミング言語による設計記述の合成可能性
3) 設計検証の研究の不足
4) CAD技術の研究開発の遅れ

は，現在のシステムLSIの設計技術に関する問題点を鋭く指摘したものと考えます．これらに対し，共通の解決策は，あえて一言でいえば，教育にあると私は思います．

情報関係の学科が日本の大学にできて30年，その当初から，メインフレームの技術を根幹とした技術体系および学問体系の教育が行われてきました．ハードウェアとソフトウェアの分離，汎用計算機のためのソフトウェア技術体

系，汎用計算機アーキテクチャとハードウェア技術などが我が国の情報工学の教育の基本路線でした．このため，電気電子工学との分離が進み，ハードウェアを支える半導体のことが分からない，いや理解しようともしない情報技術者を大量に育て，企業内のシステム部門と半導体部門の構造的な溝とも重なって，ソフトウェア研究者とハードウェア研究者の協調ができにくい土壌を拡大再生産し続けているともいえます．藤田さんがおられるVDEC（東京大学大規模集積システム教育研究センター）の試作サービスの利用状況をみても，本来，情報関係の教育や研究の利用を意図したのに，電気電子系の利用者のほうが多いことが如実にこの事実を物語っていると思います．技術も産業構造も大きく変化した今日でも，30年前と基本的に同じ思想で教育を行っていることが問題の元凶と考えますがいかがでしょうか？　情報工学科ができる前の教育のほうが今のシステムLSI時代によほどマッチしているといえるかもしれません．

言語の話は，情報工学の基本的なテーマですが，ハードウェア設計に関係する技術者の多くは，シンタックスとセマンティクスの差やモデルと現実の違いに無頓着な議論をする人がかなりおられます．これは，多くのハードウェア関係の技術者が情報関係の基礎的な教育を受けてこなかった経歴を持つ結果と考えます．もっといえば，現実の物理現象とそのモデル化の違いをまじめに教えてこなかった日本の理科教育の一般的な問題点の1つの結果といっては過言でしょうか．

検証の問題は，物理現象を相手とする世界では設計通りに動作しないのは当たり前ですが，論理だけの世界では間違えるほうがおかしいとする，旧日本軍にも似たような精神主義が根底にあるような気もします．日本の製造業の強さの秘密をその製造の精度と豪語するなら，設計の精度を高めるための設計検証に価値を見出す思想を若い人にたたき込む教育が必要と思います．組み込みソフトの開発の効率化を，単に「泥臭い仕事」として片づける風潮がある学会や大学では，そのような教育は生まれないような気がします．

CADの研究や開発は，特殊な設計事例から一般的な設計手法を帰納する技術です．泥臭い現場の中に踏み込んで，問題を抽出し，具体的解決策を提示するとともにそれを普遍化し一般化して設計手法やツールに仕上げる作業が設計技術の本質と考えます．我が国の研究や教育の中で，最終的なコスト関数で直接測れるような仕事だけを評価し，設計技術のような一般化・体系化をしていく仕事を評価できない風潮があるとしたら，技術立国の将来は決して明るくないように思われます．

(2001.3.15)

ソフトウェア研究者の立場からのコメント

高田 広章／豊橋技術科学大学 情報工学系

藤田先生のご意見，特にソフトウェア研究者とハードウェア研究者の協調が重要であることにはまったく同感である．ところが，両者の間には大きなギャップがあることも事実である．ソフトウェア研究者の立場からコメントすることで，そのギャップを理解してもらえればと思う．

そもそもソフトウェア研究者が少ない

海外と比べて日本は，ソフトウェア研究者全体の数が少ないことも事実であるが，ここでは，まず協調すべき相手となる組み込みシステムに通じたソフトウェア研究者がきわめて少ないことを強調したい．システムLSIの適用分野の多くは組み込みシステムであり，システムLSI用のソフトウェア技術に取り組むソフトウェア研究者は，（少なくとも）組み込みシステムの現状や要求事項を理解していることが必要である．

組み込みソフトウェア分野の研究者が非常に少ない原因の1つは，安浦先生が指摘されているとおり，汎用システムを中心とした教育の問題であろう．また，産学の連携不足という観点からは，1999年5月号のインタラクティブ・エッセイでも指摘させていただいた．この2年の間に組み込みソフトウェアの重要性に対する認識は広がりつつあり，一部の分野では組み込みシステムを対象とした研究が増えつつあるが，その動きはまだ限定的である．

藤田先生のご意見との関連では，とりわけ，組み込みシステムを対象としたプログラミング言語やそのコンパイル技術に取り組んでいる研究者が，大学や公的研究機関にはほとんど見当たらない（取り組まれている方がおられれば，ご連絡いただけると幸いである）．また形式的検証についても，ソフトウェアの形式的検証の分野の研究者がシステムLSI設計や組み込みシステムに興味を持ってくれれば，協調の可能性が出てくる．

プログラミング言語からのハードウェア生成には期待．でも...

組み込みソフトウェア研究者としても，プログラミング言語からのハードウェアとソフトウェアの協調生成技術には興味を持っているし，期待もしている．しかし，それらの研究・開発がハードウェア研究者に主導されているために，必ずしもソフトウェア側の要求に合致するものになっていないのが現状である．

SpecCやSystemCなど最近話題になっているハードウェア生成のためのCベースの言語をよく調べると，「C言語でソフトウェア記述できるのは自明なので，それをベースにハードウェアを記述可能とすれば要求を満たせる」という安易な考えで設計されているように思える．たとえば，陽に並列性を記述させるのはよいが，その記法がソフトウェアの並行性の記述に十分なものとなっているかの検討が十分でない．並行動作するソフトウェアがC言語で記述できているのは，OSのシステムコールが使えるからであるが，OSのシステムコールを使った記述からハードウェアが生成できるわけではない．

また，安浦先生のコメントを読んで納得してしまったが，言語のセマンティクスに無頓着な議論がされている場面に出会うこともある．たとえば，言語のセマンティクスに非決定性（たとえば，言語仕様上は実行順序が一意に決まらない記述）があるのは，プログラミング言語としては当たり前のことであるが，言語のセマンティクスはシミュレータの動作から分かるといった議論がされることがある．そもそもハードウェアとソフトウェアでは，基本的な実行モデルに違いがある．1つの記述から，ハードウェアを生成した場合にも，ソフトウェアを生成した場合にも，効率的なものとなるためには，言語のセマンティクスに非決定性を持たせて，最適化の余地を広げることが不可欠と思われる．

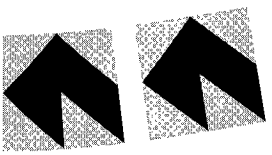
VHDLなどでのハードウェアを意識した記述よりも，ソフトウェアを記述するためのプログラミング言語のほうが抽象度が高い（ハードウェア研究者の言葉でいうと，高位レベルである）．そのため現状では，ハードウェア記述から効率的なソフトウェアを生成しようとすると，抽象度を引き上げる必要がある．逆に，ソフトウェアを素直に表現した記述から効率的なハードウェアを生成しようとすると，いったん抽象度を引き上げてからハードウェアに落とす必要がある場合もある．CADやコンパイラには，抽象度の高い記述から抽象度の低いハードウェアやソフトウェアを生成することは可能であっても，低い抽象度で記述されたものの抽象度を上げることはきわめて難しい．つまり，ハードウェアにもソフトウェアにも素直に落とせるだけの十分に高い抽象度を持った言語が必要なのである．このように考えると，効率的なハードウェアにも効率的なソフトウェアにも落とせる言語は，新しいチャレンジではないだろうか．

ソフトウェア・ハードウェア協調設計

藤田先生は，「ソフトウェア・ハードウェア協調設計」と書かれており，これは通常の用語である「ハードウェア・ソフトウェア協調設計」とは逆順になっている．藤田先生が意識的に逆順にされたものかどうかは分からないが，組み込みソフトウェア研究者としては，ソフトウェア開発側の要求にも応えられる協調設計技術という意味で，より積極的にソフトウェア・ハードウェア協調設計と呼び，研究に取り組みたい．また，SpecC言語の標準化検討が始まろうとしているが，ソフトウェア側からの意見を積極的に出していきたいと考えている．

最後に宣伝になるが，今年の7月には，私が実行委員長を務める組み込みシステム技術に関するサマーワークショップ（SWEST）を，本会のシステムLSI設計技術（SLDM）研究会主催のDAシンポジウムと同日開催する．その場で，システムLSI設計と組み込みソフトウェア設計の協調をテーマにした企画も予定している．

（2001.3.19）



藤田・安浦先生のエッセイへのコメント

吉田 憲司／（株）半導体理工学研究センター

藤田先生，安浦先生の的確な問題分析や解決策の提案を拝見して，まさに同感ですが，産業界に身を置く立場からも，2，3コメントをさせていただきたいと思います．

ソフトウェア研究者とハードウェア研究者の協調

藤田先生による両分野の技術的相違点と両分野の協調の必要性，安浦先生による大学における教育体系と学問体系，それに企業における組織構造に問題があるとする分析には，それぞれ賛同します．しかし，より個人主義の傾向の強い米国でその協調がうまくいっているとすればその理由も考えてみる価値があると思います．その1つには，企業や大学の人材の流動性があると思います．組織の壁を越えた移動により，技術バックグラウンドの異なるメンバからなるチームの結成が容易になると思います．我が国においても企業の終身雇用制の終焉や大学教員に対する規制緩和など，技術者・研究者の流動性を促す社会環境は整いつつあります．また，最近産官学のいろいろな組織から人材を結集して研究プロジェクトを進める試みが増えていますが，異分野の研究者が組織の壁を越えて協力できる場としても，期待できると思います．

プログラミング言語による設計

C言語ベースでハードウェアとソフトウェアが統一的に記述でき，かつハードウェアについては合成可能であることは，高いレベルでの設計最適化のために重要なことだと思います．しかし，安浦先生もご指摘のように，多くのハ

ードウェア技術者（設計者）にとっては，言語のシンタックスやセマンティックスを理解し，また検証ツールや合成ツールの特質を認識した上で設計対象を正しく記述することはきわめて高いハードルです．このハードルを何とか低くすることがCAD技術／ツールの重要な課題ではないかと思います．たとえば，言語の問題を極力隠蔽してしまうのも1つの方法ではないでしょうか．もちろんそのためにはソフトウェア，ハードウェア両方の研究者の協力が必要なわけですが．

我が国のCAD技術・設計技術の研究開発の低迷

1980年代まではCAD技術は半導体やコンピュータ事業の差別化技術との認識で各メーカとも独自開発努力をしてきましたが，CADベンダの成功とともに独自開発への投資効率が疑問になり，さらに90年代の不況が追い討ちをかけました．「メーカにとってCAD技術はコア技術ではないので外部から導入すればよい」「このグローバル化の時代に内製／国産にこだわるのはナンセンス」といった議論のもとに各社のCAD技術研究開発の規模縮小／中止が進み，現在では実質的にすべての設計ツールを海外に依存しているばかりか，設計の作業そのものを外部に委託し，「設計技術の空洞化」が進行しつつあるというのが我が国の愁うべき実状ではないでしょうか．

一方，国内の大学での設計技術・CAD技術の研究も学会発表で見る限り，きわめて低調といわざるを得ません．これは企業側にも責任がありますが，大学の研究に対する考え方にも原因があると思います．安浦先生のご指摘にもありましたが，一般的に「定義された問題を解く」研究が多く，現実の世界から「問題」を抽出すること，あるいは現実の世界を変革する新しい「手法」の提案が少ないと思います．このような研究は論文としてまとまりにくく，また泥沼に陥る危険も少なくありません．しかし，設計技術に要求されている「桁違い」の設計効率向上のためには，これが最も重要であり，また設計技術・CAD技術研究開発の本質的な部分ではないかと思います．ちなみに，最近の米国のDAC（Design Automation Conf.）では，設計ツールやアルゴリズムの論文よりは，設計「手法」に関する論文の方が多くなっています．そしてこれが現実のシステムLSI産業やCADツール産業の隆盛に寄与しているのはいうまでもありません．日本に比べて米国の大学でこのような研究が多いのは，もちろん研究体制の違いもありますが，「研究」に対する価値観の違いがあると思います．

今後なすべきこと

今後のシステムLSI事業，あるいはそれを使ったシステム事業で優位性を確保するために，「設計技術」が「コア技術」であることは議論の余地がありません．「日の丸」にこだわる必要はありませんが，対等以上に「戦う」ためには武器を保有するだけでなく，それをよく理解して完全に使いこなすことが不可欠ですし，時には敵にない秘密兵器も必要となります．また，対等な仲間として「協調」していくためにも，こちら側から提供すべき情報や技術を持つことが必要です．設計技術に関する研究・教育を活性化し，技術者の層を厚くすることが，我が国のエレクトロニクス技術立国にとって不可欠な所以です．幸い我が国の大学や企業の研究・技術の中には，世界のトップレベルも少なくありませんし，また半導体事業も苦戦にせよまだトップグループの座は守っておりますし，応用システムの面でも，今後発展が期待される超小型・低電力，コンシューマ向けの機器はずっと日本のお家芸でした．やり方次第で失地回復のチャンスはいくらでもあると思います．要は将来に対する希望と自信を持ち，いかに力を合わせていくかです．ソフトウェア，ハードウェア各分野の研究者・技術者が，あるいは大学の研究者と企業の技術者が，またCAD技術者とユーザ（設計技術者）がいっしょになって大いに議論を戦わせ，新しい価値を創造していく「場」を，できるだけたくさん作ることが重要であると考えます．

（2001.3.20）

問題点のまとめと１つの提案

藤田 昌宏／東京大学工学系研究科電子工学専攻

安浦先生，高田先生，吉田氏のご指摘はそれぞれ的を得た議論だと思います．誌面の都合上，ここではいくつかの話題に絞って，まとめを行うとともに，私の意見をもう少し書かせていただきます．

安浦先生の「教育が問題である」というご指摘は本質的で，改善すべき点の第一だと思います．ただ，情報工学系と電気電子工学系の分離が教育上進んだことだけではなく，もっと根本的な考え方にも大きな問題があるのではないでしょうか．たとえば，米国の大学ではCADの研究者の多くはComputer Science系の学科に属しています．その学生の多くは電子回路をもともとは理解しないし，物事のすべては0と1を分けるスイッチからスタートするという教育を受けています．しかし一方，そういう人たちが中心となって作ったEDAツールを我々は重宝がって使っているわけです．これは，また異なった意味で教育上の問題であり，要は「実際に適用できる研究成果でなければ，まったく意

味がない」という強い認識を持っているかどうかだと思います．この認識があれば，研究成果を必ずユーザが実際に使えるツールにしようとしますし，そうすれば，必ず電気電子工学系の話も理解せずにはいられないわけです．

では，なぜ「研究成果は使えなければ意味がない」ということを強く意識できるのでしょうか？　理由は，単に「自分が中心になって事業化できる可能性があるから」とか，「その研究成果で“自分が”お金を稼げるから」ではないかと思います．ここでは，吉田氏が指摘されている「米国社会は個人主義で労働市場の流動性が高い」という社会性も重要な要素です．柔軟な労働市場があるので，考えたアイディアをすぐに事業化できる環境が整っているということです．ただし，私は，米国は個人主義で会社のことを考えず，日本は会社中心主義だとはまったく思いません（むしろ，反対ではないかと思っています）．そうではなく，日本は終身雇用・退職金による給料後払い主義で，米国はそうでないだけだと思います（実際，退職金を全廃すれば，即座に労働市場の流動性は桁違いに上がると思います）．このため，結果的に日本では「保身」が最重要になるので，たとえば，すべてのプロジェクトは参加者の評価では必ず成功したなどということになるのです．

一般に，大学の1つの研究室には，複数の研究テーマがあるはずで，そのうちの2つか3つを「実用化できなければ，やった意味は何もない」という意識で研究する必要があると思います．また，高田先生が指摘されている「日本にはもともとソフトウェア技術者が少ない」，特に，「組み込みソフトウェア分野の研究者はほとんどいない」という点も「研究成果とは高貴なもので，実用化を中心に考えてはいけない」という思想の裏返しではないかと思います．「幸いに」といってはいけないのだと思いますが，不景気のおかげで，各種予算は厳しくなり，「実用化する気のない研究はすべきではない」というようになってきているのは，今までの研究態度を考えると，基本的には悪いことではないと思います．

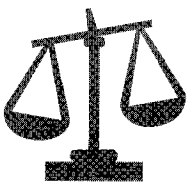
研究成果が学会で評価され，何かの賞をもらうことだけではなく，研究成果を活かしてお金儲けをしようと考える人が増えてくることが重要だと思います．この考えに立つと，大学と企業の関係も自ずとはっきりしてくると思います．さて，安浦先生が指摘された「どうすれば解決できるかの踏み込んだ意見」に関して1つの提案をしたいと思います．CAD技術の主要なもののほとんどは米国で開発されてきました．具体的にどこが開発したかというと，実はそのほとんどは大企業の中央研究所です．大学は貢献している「ふり」はいろいろしていますが，結局のところあまり開発していませんし，ベンチャーは基本的にすでにある研究成果を実用化しているだけです．大企業の中央研究所は，有能な研究員が比較的長期にわたって同一テーマを研究できる唯一の環境でした．大学では学生はどんどん入れ替わるので，研究効率という点であまりよくありません．

次にその研究成果を実用化して，設計者が実際に使えるようにしたのがだれかを考えると，逆にそのほとんどがベンチャーです．彼らはやる気のかたまりです．大企業の人のようにのんびりできません．そこが強みです．今度は，だれが研究開発したかを考えると，そのかなりの部分は米国以外からきた人たちによってなされています．ある一定以上のレベルを確保するには，世界的視野で人材を集めないと一流にはなり得ないのです．

しかし，米国を含め，もはや大企業にも研究成果をばらまく余裕はありません．やはり大学なり公的機関が中心とならざるを得ません．その意味では従来通りなのですが，次の点が重要です．

- 研究成果が熟すまで研究を続けられるような研究員制度の導入
- （日本人ではなく）外国人にとって研究しやすい環境
- 出し抜いて，研究成果をもとにベンチャーを始めることなどに何もいわず，むしろ支援する（あるいは投資して自分も儲ける）環境

(2001.3.22)



議論の続きは，次のURLをご覧ください．http://www.ipsj.or.jp/magazine/interessay.html