Science and Technology English I

Exercise 107 Meiji University 2021

(DICS Chapter-1)

EX_107_21.pptx 43 Slides June 6th.,2022

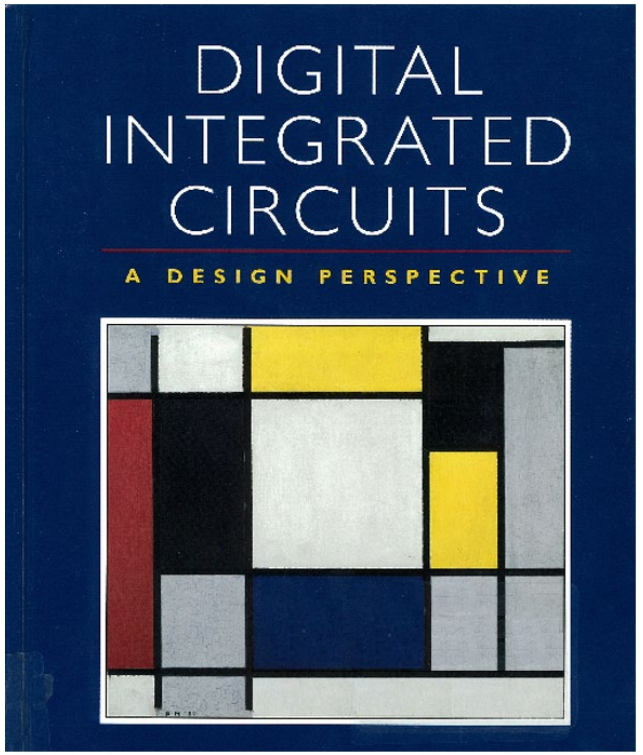
http://mikami.a.la9.jp/mdc/mdc1.htm

Renji Mikami

Renji_Mikami(at_mark)nifty.com [mikami(at_mark)meiji.ac.jp]

Day 106 Review 1

- Day 106 ではマイクロプロセッサの歴史を解説しました。
- Day 107以降はDICS原本の購読に入ります。これまでの基礎知識が活きてきます。
- 春学期はDICS 123章、秋学期は467章を読み進めます。Day 107は第1章から始めます。

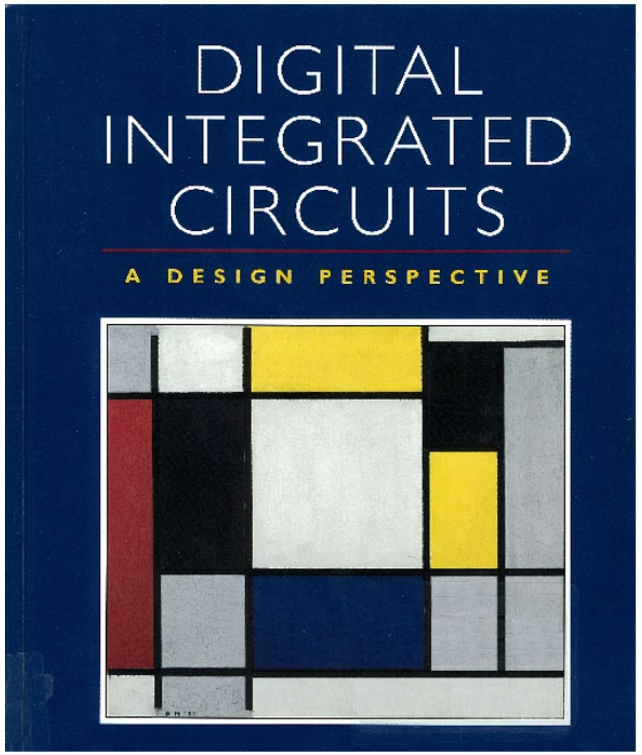


教材(春学期/秋学期) DICS

- National Academy of Engineering (半導体、IC、プロセッサの歴史と概要(英文プリント))
- 教科書(春および秋)

「DIGITAL INTEGRATED CIRCUITS」
JAN M. RABAEY
PRENTICE HALL（1996）

- ISSCC Paper
- US Patent
- WEB NEWS（EE TIMES）
- IEEE Paper



Jan M. Rabaey
Donald O. Pederson Distinguished Professor
Director Gigascale Systems Research Center (GSRC) and Scientific Co-director BWRC University of California, Berkeley

原本のPDFは、HP下の 春学期 備考　DICS Chapter 1, 2, 3 を参考にしてください。

英語４技能の実戦的なスキル

Study 再掲

概要・要旨理解力と質問ができる力

シンプルでよいから正確な表現と論理的な構成(数字、固有名詞は間違わない、展開スタイルを使う)

Inbound

Outbound

Listening

Speaking

分量が多い

言葉だから次々と消えていく

分量は少ない

Reading

Writing

文字だから記録に残る

60~70%の理解でいいから短時間に大量に読み聞き。英語のまま理解。重要点や疑問点をピックアップする。疑問点は相手に聞く。または重要点とともに精読に入る。

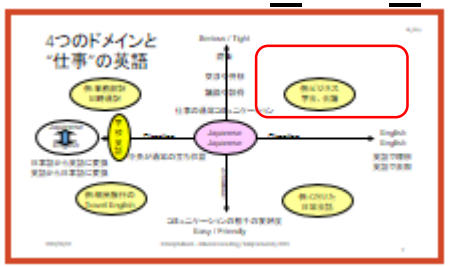
プレゼンテーションとプロポーザルの力

科学技術(仕事)英語 Inbound 演習

- 学校英語:日本語訳の正解が用意されている
  - 日本語訳を書かせ(Outbound)正解と比較して採点
  - 英単語と日本語を対応させ英文法に従い日本文を構成していく
  - 少ない量の英文を時間をかけて正確に訳していく
- 仕事英語:多量の英文からポイントを読みとる
  - 時間をかけず英文を英文のまま(イメージ)理解する(いちいち訳さない)
  - ワンパラグラフ単位で読み切る。内容の理解度は60~70%でよい。
  - ポイントとなるKeywordsやSentencesをつかむ

  (重要点に絞って精読する-あとで)
  - わからないところがあってあたり前、聞けばよい

科学技術(仕事)英語 Outbound 演習



- 科学技術英語や仕事英語:内容の理解がポイント
  - 学校英語や翻訳では内容の理解なしで正しい日本語訳を書ける人がいる(翻訳プロ-左上ドメインの人たち)
- 英文のままで内容を理解するのがベスト(右上第一象限ドメイン)
  - 和訳や概要を和文で書けといわれたら、内容イメージとポイントから日本文で作文する (訳ではなく作文)
  - 英文で概要を書けといわれたら本文中からSentencesを選んで抜き出すのではなく、内容イメージとポイントから英作文する (抜き出してコピペしない)
- すぐにはできなくても、これらを目指すことをイメージしよう (できるようになったら Native と同じ Reading Y^。^Y)

Inbound (Reading) 演習のすすめ方

- Chapter 1 では技術的内容に深入りせず概要の理解に集中します。
- 1.制限時間を決めて(基本4分/A4で1ページ)配布テキスト原文(赤ナシ)を最後まで読み切る。(最低でもワンパラグラフ単位 – 30~50行程度)
- 2.わからない単語があったら(下線を引いて)パス、英単語のままで読み進む
- 3.最後まで読み切ったら(できれば)もういちど繰り返す
  - ホントは何回も繰り返しているうちに内容がわかってきます
  - これは**人間の脳に備わる能力**です。*読書百篇義自ずから通ず*

ねらい
逐次和訳しようとする
日本語脳をブロックします

- 4.続いてEX1_1XX –このテキストの(赤入れ)単語訳つきの英文を読んでみます
- 5.これでだいたいわかったら Outbound 演習にすすみます。
- まだ不明単語が多いと思う場合は、このテキストの打ち直しテキストを自動翻訳サイトにコピペして訳させてみます。納得いかない単語がある場合は辞書引いてみます。そして Outbound にすすんでください。Chapter2以降はポイントになる技術点などは、資料で解説します

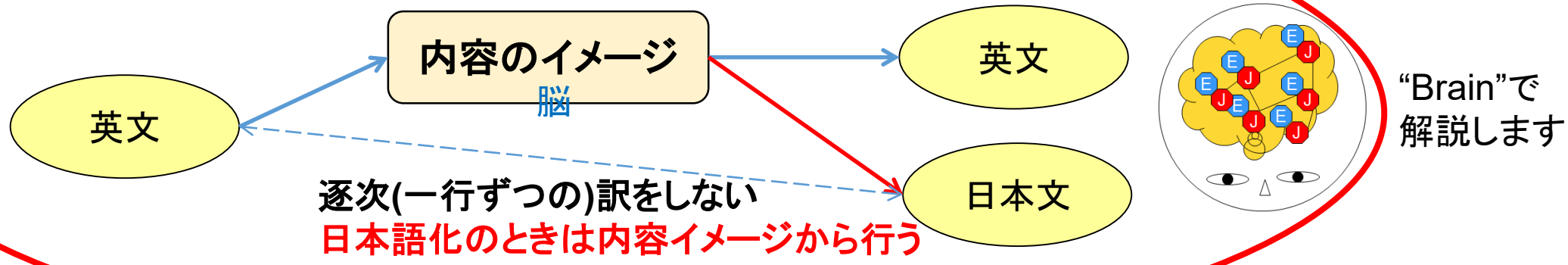
Outbound (Writing) 演習のすすめ方

ねらい
行単位での逐次英訳和訳
ではなく、内容をまとめて
書き出せるようにします

- EX_課題 和文または英文で概要をまとめよ　の場合
- 和文でまとめる場合は、読み取った英語理解イメージから、日本語で作文をするようなイメージで書き出してください。(書いた日本文を和文英訳サイトを使ってみて、どのように英訳されるかを試してもいいです。)
- 英文でまとめる場合は、読み取った英語理解イメージから、英作文をするようなイメージで Sentences を書き出してください。日本語訳を間に入れないようにするのがポイントです。(日本語翻訳脳をブロックするイメージです。これは Native 同様の英語->英語プロセスです)
- 完成した英文は自動翻訳(和訳)サイトの英文入力側に張り付けて、エラーチェックして、赤い波線が出るところを直します。Wordなども使えます。このときチェック機能は完全なものではありませんから技術用語などはエラーのままでかまいません。
- 注意：一行単位で和訳しないように。英語本文をそのまま抜き出してコピペしないように。パラグラフやページ単位で理解したものを書き出していくイメージ。

英文をそのまま理解することを目標にしよう

- テクニカル タームを無理に日本語に訳さなくてもよい
- 英語のニュアンスを正確に伝える日本語がなければ、英単語のままでよい
- 英文パラグラフをそのまま理解し、英語のままで考え英語でまとめる
  英単語や文単位で日本語に訳し、日本語で考えてそれをまた再英訳しない



- 60～70%程度のアバウトな内容理解でよい。全文を素早く通読、不明単語は英語のままでよい。仕事-わからないところは聞けばよい。
- 慣れてくると、まわりの英文から不明単語の意味を類推できるようになる

Introduction 1/2

60秒くらいで読む-途中で止まらない。気になる、わからない単語があったらサッと下線をひいてもよい。とにかく止まらず先に読みすすんでいく。

- 1.1 A Historical Perspective
- The concept of digital data manipulation has made a dramatic impact on our society. One has long grown accustomed to the idea of digital computers. Evolving steadily from mainframe and minicomputers, personal and laptop computers have proliferated into daily life. More significant, however, is a continuous trend towards digital solutions in all other areas of electronics. Instrumentation was one of the first noncomputing domains where the potential benefits of digital data manipulation over analog processing were recognized, Other areas such as control were soon to follow. Only recently have we witnessed the conversion of telecommunications and consumer electronics towards the digital format. Increasingly, telephone data is transmitted and processed digitally over both wired and wireless networks. The compact disk has revolutionized the audio world, and digital video is following in its footsteps.

行(センテンス)単位で訳さない。パラグラフ単位で何が書いてあるか、見当をつける(60%理解でよい)

この原本の打ち直しテキストはコピペできますが特殊な数式やギリシャ文字などには正確に対応しませんから、これらは次の赤文字入りグラフィックス・スライドや配布原本を参照してください

Introduction 1/2

STE-101-101

消費者（家電製品）ＴＶ

大型電子計算機

遠隔通信

解決

進化する

操作

展望

1.1 A Historical Perspective

広がる

The concept of digital data manipulation has made a dramatic impact on our society. One has long grown accustomed to the idea of digital computers. Evolving steadily from mainframe and minicomputers, personal and laptop computers have proliferated into daily life. More significant, however, is a continuous trend towards digital solutions in all other areas of electronics. Instrumentation was one of the first noncomputing domains where the potential benefits of digital data manipulation over analog processing were recognized. Other areas such as control were soon to follow. Only recently have we witnessed the conversion of telecommunications and consumer electronics towards the digital format. Increasingly, telephone data is transmitted and processed digitally over both wired and wireless networks. The compact disk has revolutionized the audio world, and digital video is following in its footsteps.

計測

収束

革命

伝達する

有線/無線

Introduction 2/2

Introduction 1/2 と2/2 では、何が書いてあるかを本文から離れて考えてみる。そのあとまた見直す。

- The idea of implementing computational engines using an encoded data format is by no means an idea of our times. In the early nineteenth century, Babbage envisioned largescale mechanical computing devices, called Difference Engines [Swade93]. Although these engines use the decimal number system rather than the binary representation now common in modern electronics, the underlying concepts are very similar. The Analytical Engine, developed in 1834, was perceived as a general-purpose computing machine, with features strikingly close to modern computers. Besides executing the basic repertoire of operations (addition, subtraction, multiplication, and division) in arbitrary sequences, the machine operated in a two-cycle sequence, called "store" and "mill" (execute), similar to current computers. It even used pipelining to speed up the execution of the addition operation! Unfortunately, the complexity and the cost of the designs made the concept impractical. For instance, the design of Difference Engine I (part of which is shown in Figure I. 1) required 25,000 mechanical parts at a total cost of £ 17,470 (in 1834!).

Figure 1.1 Working part of Babbage's Difference Engine I (1832), the first known automatic calculator (from [Swade93], courtesy of the Science Museum of London).

Introduction 2/2

STE-101-101

The idea of implementing computational engines using an encoded data format is by no means an idea of our times. In the early nineteenth century, Babbage envisioned large-scale mechanical computing devices, called *Difference Engines* [Swade93]. Although these engines use the decimal number system rather than the binary representation now common in modern electronics, the underlying concepts are very similar. The Analytical Engine, developed in 1834, was perceived as a general-purpose computing machine, with features strikingly close to modern computers. Besides executing the basic repertoire of operations (addition, subtraction, multiplication, and division) in arbitrary sequences, the machine operated in a two-cycle sequence, called "store" and "mill" (execute), similar to current computers. It even used pipelining to speed up the execution of the addition operation! Unfortunately, the complexity and the cost of the designs made the concept impractical. For instance, the design of Difference Engine I (part of which is shown in Figure 1.1) required 25,000 mechanical parts at a total cost of £17,470 (in 1834!).

１０進数

差動エンジン

特徴

非現実的な

加算、減算、乗算、除算

汎用

任意の順序

実行する

複雑性

pound=
240円
420万円

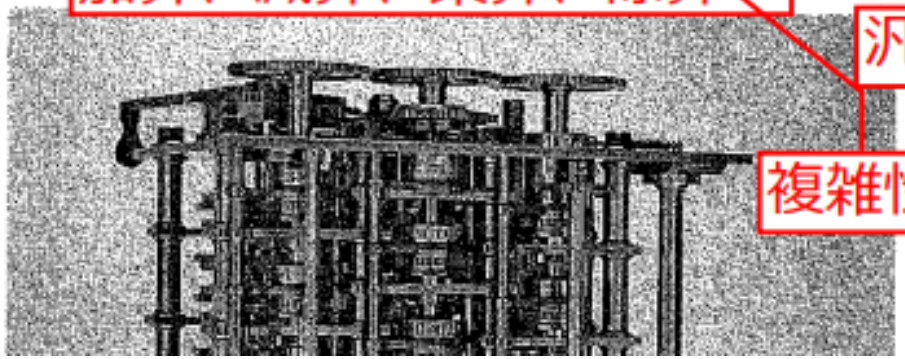


Figure 1.1 Working part of Babbage's Difference Engine I (1832), the first known automatic calculator (from [Swade93], courtesy of the Science Museum of London).

Introduction 1/2 と2/2ではどんなことが書かれていただろうか

この内容90秒くらいで読む-途中で止まらない。

- The electrical solution turned out to be more cost effective. Early digital electronics systems were based on magnetically controlled switches (or relays). They were mainly used in the implementation of very simple logic networks. Examples of such are train safety systems, where they are still being used at present. The age of digital electronic computing only started in full with the introduction of the vacuum tube. While originally used almost exclusively for analog processing, it was realized early on that the vacuum tube was useful for digital computations as well. Soon complete computers were realized. The era of the vacuum tube based computer culminated in the design of machines such as the ENIAC (intended for computing artillery firing tables) and the UNIVAC I (the first successful commercial computer). To get an idea about integration density, the ENIAC was 80 feet long, 8.5 feet high and several feet wide and incorporated 18,000 vacuum tubes. It became rapidly clear, however, that this design technology had reached its limits. Reliability problems and excessive power consumption made the implementation of larger engines economically and practically infeasible.

長くてたいへんなときは、ここまで(パラグラフ単位で進んでもよい)

- All changed with the invention of the transistor at Bell Telephone Laboratories in 1947 [Bardeen48], followed by the introduction of the bipolar transistor by Schockley in 1949 [Schockley49]. It took till 1956 before this led to the first bipolar digital logic gate, introduced by Harris [Harris56], and even more time before this translated into a set of integrated-circuit commercial logic gates, called the Fairchild Micrologic. family [Norman60]. The first truly successful IC logic family, TTL (Transistor-Transistor Logic) was pioneered in 1962 [Beeson62]. Other logic families were devised with higher performance in mind. Examples of these are the current switching circuits that produced the first subnanosecond digital gates and culminated in the ECL (Emitter-Coupled Logic) family. [Masaki74], which is discussed in more detail in this textbook. TTL had the advantage, however, of offering a higher integration density and was the basis of the first integrated circuit revolution. In fact, the manufacturing of TTL components is what spear-headed the first large semiconductor companies such as Fairchild, National, and Texas Instruments.

不明な単語で止まらない

- The family was so successful that it composed the largest fraction of the digital semiconductor market until the 1980s.

リレー 真空管 STC-101-102

The electrical solution turned out to be more cost effective. Early digital electronics systems were based on magnetically controlled switches (or relays). They were mainly used in the implementation of very simple logic networks. Examples of such are train safety systems, where they are still being used at present. The age of digital electronic computing only started in full with the introduction of the vacuum tube. While originally used almost exclusively for analog processing, it was realized early on that the vacuum tube was useful for digital computations as well. Soon complete computers were realized. The era of the vacuum tube based computer culminated in the design of machines such as the ENIAC (intended for computing artillery firing tables) and the UNIVAC I (the first successful commercial computer). To get an idea about *integration density*, the ENIAC was 80 feet long, 8.5 feet high and several feet wide and incorporated 18,000 vacuum tubes. It became rapidly clear, however, that this design technology had reached its limits. Reliability problems and excessive power consumption made the implementation of larger engines economically and practically infeasible.

All changed with the invention of the *transistor* at Bell Telephone Laboratories in 1947 [Bardeen48], followed by the introduction of the bipolar transistor by Schockley in 1949 [Schockley49][1]. It took till 1956 before this led to the first bipolar digital logic gate, introduced by Harris [Harris56], and more time before this translated into a set of integrated-circuit commercial logic gates, called the Fairchild Micrologic family [Norman60]. The first truly successful IC logic family, *TTL (Transistor-Transistor Logic)* was pioneered in 1962 [Beeson62]. Other logic families were devised with higher performance in mind. Examples of these are the current switching circuits that produced the first subnanosecond digital gates and culminated in the *ECL (Emitter-Coupled Logic)* family [Masaki74], which is discussed in more detail in this textbook. TTL had the advantage, however, of offering a higher integration density and was the basis of the first integrated circuit revolution. In fact, the manufacturing of TTL components is what spear-headed the first large semiconductor companies such as Fairchild, National, and Texas Instruments. The family was so successful that it composed the largest fraction of the digital semiconductor market until the 1980s.

排他的に
大砲発火表
1feet=30cm
信頼性
バイポーラ（両極性）電子・正孔
ns以下
革命
製造
部分
部品
先鋒になる
利点
考案する
最高点に達する
集積密度
組み込む
実現できない
4-14 :
4-15 :

- Ultimately, bipolar digital logic lost the battle for hegemony in the digital design world for exactly the reasons that haunted the vacuum tube approach: the large power consumption per gate puts an upper limit on the number of gates that can be reliably integrated on a single die, package, housing, or box. Although attempts were made to develop high integration density, low-power bipolar families (such as 12L—Integrated Injection Logic [Hart72]), the torch was gradually passed to the MOS digital integrated circuit approach.
- The basic principle behind the MOSFET transistor (originally called IGFET) was proposed in a patent by J. Lilienfeld (Canada) as early as 1925, and, independently, by O. Heil in England in 1935. Insufficient knowledge of the materials and gate stability problems, however, delayed the practical usability of the device for a long time. Once these were solved, MOS digital integrated circuits started to take off in full in the early 1970s. Remarkably, the first MOS logic gates introduced were of the CMOS variety [Wanlass63], and this trend continued till the late 1960s. The complexity of the manufacturing process

ここから次のページ。

- An intriguing overview of the evolution of digital integrated circuits can be found in [Murphy93]. (Most of the data in this overview has been extracted from this reference). It is accompanied by some of the historically ground-breaking publications in the domain of digital IC's.

- delayed the full exploitation of these devices for two more decades. Instead, the first practical MOS integrated circuits were implemented in PMOS-only logic and were used in applications such as calculators. The second age of the digital integrated circuit revolution was inaugurated with the introduction of the first microprocessors by Intel in 1972 (the 4004) and 1974 (the 8080) [Shima74]. These processors were implemented in NMOS-only logic, that has the advantage of higher speed over the PMOS logic. Simultaneously, MOS technology enabled the realization of the first high-density semiconductor memories. For instance, the first 4Kbit MOS memory was introduced in 1970 [Hoff70].

EX_107_21

STE-101-102

覇権

Ultimately, bipolar digital logic lost the battle for hegemony in the digital design world for exactly the reasons that haunted the vacuum tube approach: the large power consumption per gate puts an upper limit on the number of gates that can be reliably integrated on a single die, package, housing, or box. Although attempts were made to develop high integration density, low-power bipolar families (such as *I^2L—Integrated Injection Logic* [Hart72]), the torch was gradually passed to the MOS digital integrated circuit approach.

消費電力

たいまつ

The basic principle behind the MOSFET transistor (originally called IGFET) was proposed in a patent by J. Lilienfeld (Canada) as early as 1925, and, independently, by O. Heil in England in 1935. Insufficient knowledge of the materials and gate stability problems, however, delayed the practical usability of the device for a long time. Once these were solved, MOS digital integrated circuits started to take off in full in the early 1970s. Remarkably, the first MOS logic gates introduced were of the CMOS variety [Wanlass63], and this trend continued till the late 1960s. The complexity of the manufacturing process

原理

特許

独立に

材料

変種

複雑性

製造プロセス

安定性

[1] An intriguing overview of the evolution of digital integrated circuits can be found in [Murphy93]. (Most of the data in this overview has been extracted from this reference). It is accompanied by some of the historically ground-breaking publications in the domain of digital IC's.

STC-101-103

実現

嶋正利（ビジコン社）

遅延させる

利用

驚くべき進化

同時に

delayed the full exploitation of these devices for two more decades. Instead, the first practical MOS integrated circuits were implemented in PMOS-only logic and were used in applications such as calculators. The second age of the digital integrated circuit revolution was inaugurated with the introduction of the first microprocessors by Intel in 1972 (the 4004) and 1974 (the 8080) [Shima74]. These processors were implemented in NMOS-only logic, that has the advantage of higher speed over the PMOS logic. Simultaneously, MOS technology enabled the realization of the first high-density semiconductor memories. For instance, the first 4Kbit MOS memory was introduced in 1970 [Hoff70].

宣誓

- These events were at the start of a truly astounding evolution towards ever higher integration densities and speed performances, a revolution that is still in full swing right now. The road to the current levels of integration has not been without hindrances, however. In the late 1970s, NMOS-only logic started to suffer from the same plague that made high-density bipolar logic unattractive or infeasible: power consumption. This realization, combined with progress in manufacturing technology, finally tilted the balance towards the CMOS technology, and this is where we still are today. Interestingly enough, power consumption concerns are rapidly becoming dominant in CMOS design as well, and this time there does not seem to be a new technology around the corner to alleviate the problem.
- Although the large majority of the current integrated circuits are implemented in the MOS technology, other technologies come into play when very high performance is at stake. An example of this is the BiCMOS technology that combines bipolar and MOS devices on the same die. BiCMOS is effectively used in high-speed memories and gate arrays. When even higher performance is necessary, other technologies emerge besides the already mentioned bipolar silicon ECL family—Gallium-Arsenide, Silicon-Germanium and even superconducting technologies. While these circuits only fill in a small niche in the overall digital integrated circuit design scene, it is worth examining some of the issues emerging in the design of these circuits. With the continuing increase in performance of digital MOS circuits, design problems currently encountered in these high-speed technologies might come to haunt CMOS as well in the foreseeable future.

These events were at the start of a truly astounding evolution towards ever higher integration densities and speed performances, a revolution that is still in full swing right now. The road to the current levels of integration has not been without hindrances, however. In the late 1970s, NMOS-only logic started to suffer from the same plague that made high-density bipolar logic unattractive or infeasible: power consumption. This realization, combined with progress in manufacturing technology, finally tilted the balance towards the CMOS technology, and this is where we still are today. Interestingly enough, power consumption concerns are rapidly becoming dominant in CMOS design as well, and this time there does not seem to be a new technology around the corner to alleviate the problem.

Although the large majority of the current integrated circuits are implemented in the MOS technology, other technologies come into play when very high performance is at stake. An example of this is the BiCMOS technology that combines bipolar and MOS devices on the same die. BiCMOS is effectively used in high-speed memories and gate arrays. When even higher performance is necessary, other technologies emerge besides the already mentioned bipolar silicon ECL family—Gallium-Arsenide, Silicon-Germanium and even superconducting technologies. While these circuits only fill in a small niche in the overall digital integrated circuit design scene, it is worth examining some of the issues emerging in the design of these circuits. With the continuing increase in performance of digital MOS circuits, design problems currently encountered in these high-speed technologies might come to haunt CMOS as well in the foreseeable future.

電力消費

軽減する

超伝導

遭遇する

出現する

すきま
ニッチ

- Integration density and performance of integrated circuits have gone through an astounding revolution in the last couple of decades. In the 1960s, Gordon Moore, then with Fairchild Corporation and later cofounder of Intel, predicted that the number of transistors that can be integrated on a single die would grow exponentially with time. This prediction, later called Moore's law, has proven to be amazingly visionary. Its validity is best illustrated with the aid of a set of graphs. Figure 1.2 plots the integration density of both logic ICs and memory as a function of time. As can be observed, integration complexity doubles approximately every 1 to 2 years. As a result, memory density has increased by more than a thousandfold since 1970.
- An intriguing case study is offered by the microprocessor. From its inception in the early seventies, the microprocessor has grown in performance and complexity at a steady and predictable pace. The number of transistors and the clock frequency for a number of landmark designs are collected in Figure 1.3. The million-transistor/chip barrier was crossed in the late eighties. Clock frequencies double every three years and have reached into the 100 MHz range.

1.2 Issues in Digital Integrated Circuit Design

課題 革命 共同創業者 予想

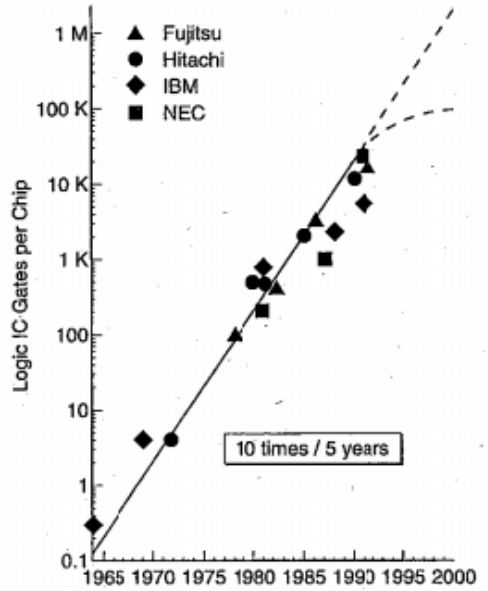
Integration density and performance of integrated circuits have gone through an astounding revolution in the last couple of decades. In the 1960s, Gordon Moore, then with Fairchild Corporation and later cofounder of Intel, predicted that the number of transistors that can be integrated on a single die would grow exponentially with time. This prediction, later called *Moore's law*, has proven to be amazingly visionary. Its validity is best illustrated with the aid of a set of graphs. Figure 1.2 plots the integration density of both logic ICs and memory as a function of time. As can be observed, integration complexity doubles approximately every 1 to 2 years. As a result, memory density has increased by more than a thousandfold since 1970.

作図する 正当性

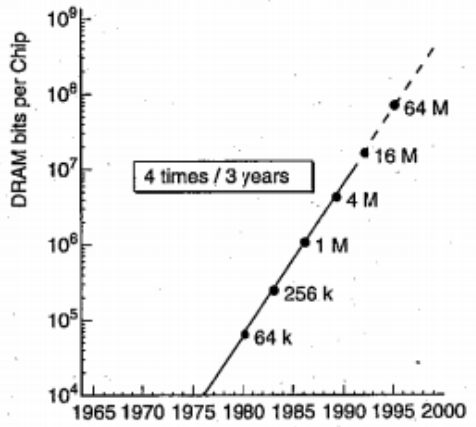
An intriguing case study is offered by the microprocessor. From its inception in the early seventies, the microprocessor has grown in performance and complexity at a steady and predictable pace. The number of transistors and the clock frequency for a number of landmark designs are collected in Figure 1.3. The million-transistor/chip barrier was crossed in the late eighties. Clock frequencies double every three years and have reached into the 100 MHz range.

目印 集積密度は4倍/3年 周波数 発端

EX_107_21

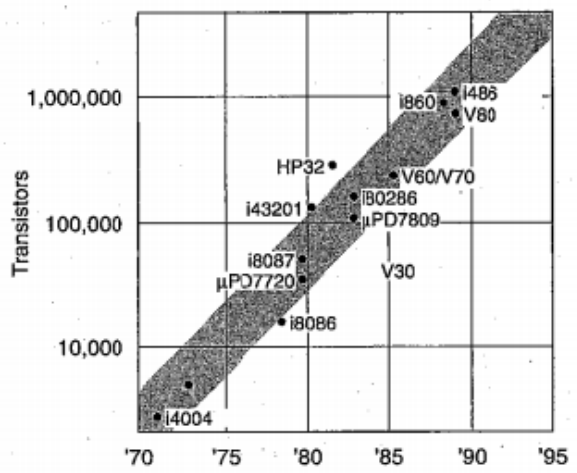


(a) Trends in logic IC complexity

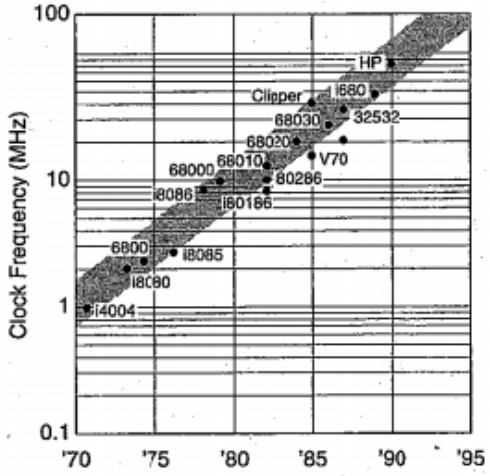


(b) Trends in memory complexity

Figure 1.2 Evolution of integration complexity of logic ICs and memories as a function of time (from [Masaki92]).



(a) Trends in transistor count



(b) Trends in clock frequency

- Figure 1.3 Evolution of microprocessor transistor count and clock frequency (from [Sasaki911).
- into the 100 MHz range. An even more important observation is that, as of now, these trends have not shown any signs of a slow-down.
- It should be no surprise to the reader that this revolution has had a profound impact on how digital circuits are designed. Early designs were frilly hand-crafted. Every transistor was laid out and optimized individually and carefully fitted into its environment. This is adequately illustrated in Figure 1.4a, which shows the design of the Intel 4004 microprocessor. This approach is, obviously, not appropriate when more than a million devices have to be created and assembled. With the rapid evolution of the design technology, time-to-market is one of the crucial factors in the ultimate success of a component.

Figure 1.4 Comparing the design methodologies of the Intel 4004 (1971) and PentiumTM (1994) microprocessors (reprinted with permission from Intel).

Figure 1.3 Evolution of microprocessor transistor count and clock frequency (from [Sasaki91]).

into the 100 MHz range. An even more important observation is that, as of now, these trends have not shown any signs of a slow-down.

It should be no surprise to the reader that this revolution has had a profound impact on how digital circuits are designed. Early designs were truly hand-crafted. Every transistor was laid out and optimized individually and carefully fitted into its environment. This is adequately illustrated in Figure 1.4a, which shows the design of the Intel 4004 microprocessor. This approach is, obviously, not appropriate when more than a million devices have to be created and assembled. With the rapid evolution of the design technology, time-to-market is one of the crucial factors in the ultimate success of a component.

手作業

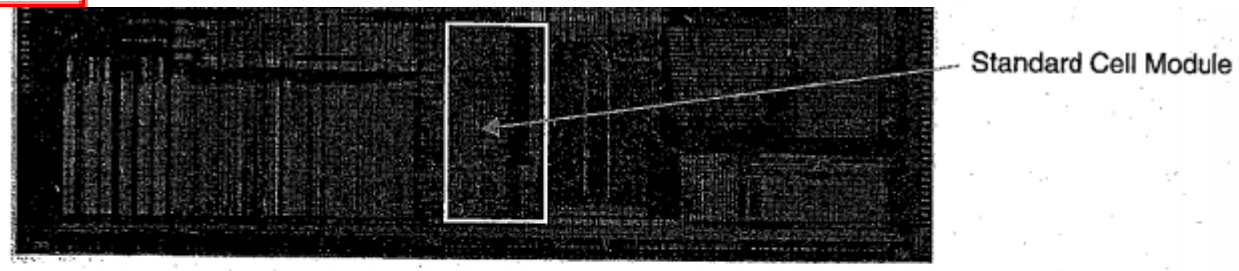
観察

最適化する

組み立てる

進化

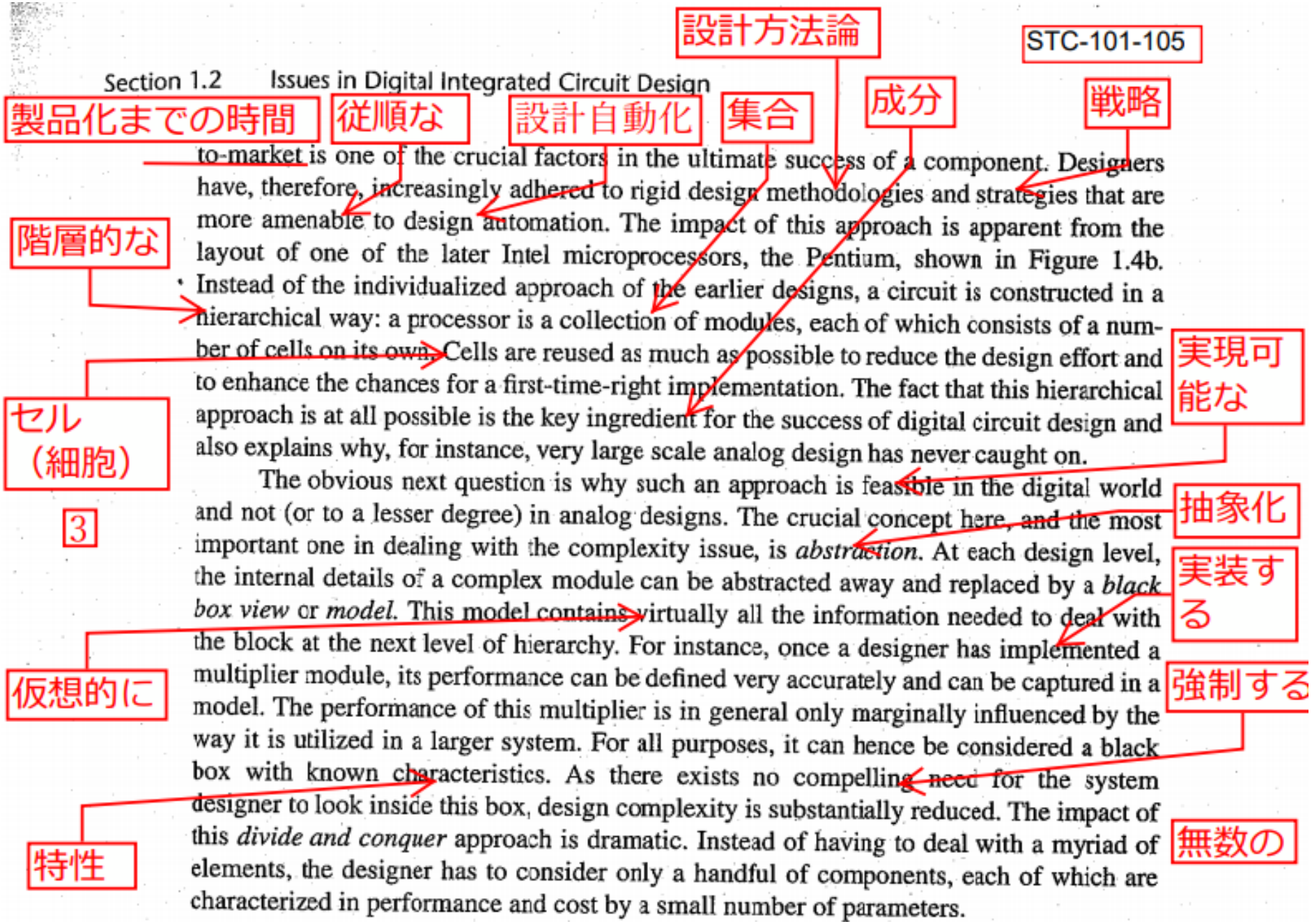
環境



(b) The Pentium™ microprocessor (see also back cover)

Figure 1.4 Comparing the design methodologies of the Intel 4004 (1971) and Pentium™ (1994) microprocessors (reprinted with permission from Intel).

- Designers have, therefore, increasingly adhered to rigid design methodologies and strategies that are more amenable to design automation. The impact of this approach is apparent from the layout of one of the later Intel microprocessors, the Pentium, shown in Figure 1.4b. Instead of the individualized approach of the earlier designs, a circuit is constructed in a hierarchical way: a processor is a collection of modules, each of which consists of a number of cells on its own. Cells are reused as much as possible to reduce the design effort and to enhance the chances for a first-time-right implementation. The fact that this hierarchical approach is at all possible is the key ingredient for the success of digital circuit design and also explains why, for instance very large scale analog design has never caught on.
- The obvious next question is why such an approach is feasible in the digital world and not (or to a lesser degree) in analog designs. The crucial concept here, and the most important one in dealing with the complexity issue, is abstraction. At each design level, the internal details of a complex module can be abstracted away and replaced by a black box view or model. This model contains virtually all the information needed to deal with the block at the next level of hierarchy. For instance, once a designer has implemented a multiplier module, its performance can be defined very accurately and can be captured in a model. The performance of this multiplier is in general only marginally influenced by the way it is utilized in a larger system. For all purposes, it can hence be considered a black box with known characteristics. As there exists no compelling need for the system designer to look inside this box, design complexity is substantially reduced. The impact of this divide and conquer approach is dramatic. Instead of having to deal with a myriad of elements, the designer has to consider only a handful of components, each of which are characterized in performance and cost by a small number of parameters.



設計方法論
STC-101-105
Section 1.2 Issues in Digital Integrated Circuit Design
製品化までの時間
従順な
設計自動化
集合
成分
戦略
to-market is one of the crucial factors in the ultimate success of a component. Designers have, therefore, increasingly adhered to rigid design methodologies and strategies that are more amenable to design automation. The impact of this approach is apparent from the layout of one of the later Intel microprocessors, the Pentium, shown in Figure 1.4b. Instead of the individualized approach of the earlier designs, a circuit is constructed in a hierarchical way: a processor is a collection of modules, each of which consists of a number of cells on its own. Cells are reused as much as possible to reduce the design effort and to enhance the chances for a first-time-right implementation. The fact that this hierarchical approach is at all possible is the key ingredient for the success of digital circuit design and also explains why, for instance, very large scale analog design has never caught on.
階層的な
セル（細胞）
実現可能な
The obvious next question is why such an approach is feasible in the digital world and not (or to a lesser degree) in analog designs. The crucial concept here, and the most important one in dealing with the complexity issue, is *abstraction*. At each design level, the internal details of a complex module can be abstracted away and replaced by a *black box view* or *model*. This model contains virtually all the information needed to deal with the block at the next level of hierarchy. For instance, once a designer has implemented a multiplier module, its performance can be defined very accurately and can be captured in a model. The performance of this multiplier is in general only marginally influenced by the way it is utilized in a larger system. For all purposes, it can hence be considered a black box with known characteristics. As there exists no compelling need for the system designer to look inside this box, design complexity is substantially reduced. The impact of this *divide and conquer* approach is dramatic. Instead of having to deal with a myriad of elements, the designer has to consider only a handful of components, each of which are characterized in performance and cost by a small number of parameters.
抽象化
実装する
仮想的に
強制する
特性
無数の
3

- This is analogous to a software designer using a library of software routines such as input/output drivers. Someone writing a large program does not bother to look inside those library routines. The only thing he cares about is the intended result of calling one of those modules. Imagine what writing software programs would be like if one had to fetch every bit individually from the disk and ensure its correctness instead of relying on handy "file open" and "get string" operators.
- Typically used abstraction levels in digital circuit design are, in order of increasing abstraction, the device, circuit, gate, functional module (e.g., adder) and system levels (e.g., processor), as illustrated in Figure 1.5. A semiconductor device is an entity with a very complex behavior. No circuit designer will ever seriously consider the solid-state physics equations governing the behavior of the device when designing a digital gate. Instead he will use a simplified model that adequately describes the input-output behavior of the transistor. For instance, an AND gate is adequately described by its Boolean expression (Z = A.B), its bounding box, the position of the input and output terminals, and the delay between the inputs and the output.
- This design philosophy has been the enabler for the emergence of elaborate computer-aided design (CAD) frameworks for digital integrated circuits; without it the current design complexity would not have been achievable. Design tools include simulation at the various complexity levels, design verification, layout generation, and design synthesis, An overview of these tools and design methodologies is given in Chapter 11 of this textbook.
- Furthermore, to avoid the redesign and reverification of frequently used cells such as basic gates and arithmetic and memory modules, designers most often resort to dell libraries.

This is analogous to a software designer using a library of software routines such as input/output drivers. Someone writing a large program does not bother to look inside those library routines. The only thing he cares about is the intended result of calling one of those modules. Imagine what writing software programs would be like if one had to fetch every bit individually from the disk and ensure its correctness instead of relying on handy "file open" and "get string" operators.

Typically used abstraction levels in digital circuit design are, in order of increasing abstraction, the device, circuit, gate, functional module (e.g., adder) and system levels (e.g., processor), as illustrated in Figure 1.5. A semiconductor device is an entity with a very complex behavior. No circuit designer will ever seriously consider the solid-state physics equations governing the behavior of the device when designing a digital gate. Instead he will use a simplified model that adequately describes the input-output behavior of the transistor. For instance, an AND gate is adequately described by its Boolean expression ($Z = A.B$), its bounding box, the position of the input and output terminals, and the delay between the inputs and the output.

This design philosophy has been the enabler for the emergence of elaborate *computer-aided design* (CAD) frameworks for digital integrated circuits; without it the current design complexity would not have been achievable. Design tools include simulation at the various complexity levels, design verification, layout generation, and design synthesis. An overview of these tools and design methodologies is given in Chapter 11 of this textbook.

Furthermore, to avoid the redesign and reverification of frequently used cells such as basic gates and arithmetic and memory modules, designers most often resort to *cell librar-ies.*

類似の

正確さ

固体物理

動作

設計理念

可能にするもの

精巧な

合成

検証

ツール（道具）

方法論

生成

頼る

- These libraries contain not only the layouts, but also provide complete documentation and characterization of the behavior of the cells. The use of cell libraries is, for instance, apparent in the layout of the Pentium. processor (Figure 1.4b). The integer and floating point unit, just to name a few, contain large sections designed using the so-called standard cell approach. In this approach, logic gates are placed in rows of cells of equal height and interconnected using routing Channels. The layout of such a block can be generated automatically given that a library of cells is available.
- The preceding analysis demonstrates that design automation and modular design practices have effectively addressed some of the complexity issues incurred in contemporary digital design. This leads to the following pertinent question. If design automation solves all our design problems, why should we be concerned with digital circuit design at all? Will the next-generation digital designer ever have to worry about transistors or parasitics, or is the smallest design entity he will ever consider the gate and the module?
- The truth is that the reality is more complex, and various reasons exist as to why an insight into digital circuits and their intricacies will still be an important asset for a long time to come.
- • First of all, someone still has to design and implement the module libraries. Semiconductor technologies continue to advance from year to year, as demonstrated in Figure 1.2, where the minimum MOS device dimensions are plotted as a function of time.

特性化

Figure 1.5 Design abstraction leve

整数演算、浮動小数点演算

文書

les. These libraries contain not only the layouts, but also provide complete documentation and characterization of the behavior of the cells. The use of cell libraries is, for instance, apparent in the layout of the Pentium processor (Figure 1.4b). The integer and floating-point unit, just to name a few, contain large sections designed using the so-called *standard cell approach*. In this approach, logic gates are placed in rows of cells of equal height and interconnected using routing channels. The layout of such a block can be generated automatically given that a library of cells is available.

相互配線

関連する

招く

生成

The preceding analysis demonstrates that design automation and modular design practices have effectively addressed some of the complexity issues incurred in contemporary digital design. This leads to the following pertinent question. If design automation solves all our design problems, why should we be concerned with digital circuit design at all? Will the next-generation digital designer ever have to worry about transistors or parasitics, or is the smallest design entity he will ever consider the gate and the module?

現代の

The truth is that the reality is more complex, and various reasons exist as to why an insight into digital circuits and their intricacies will still be an important asset for a long time to come.

洞察

複雑性

- First of all, someone still has to *design and implement the module libraries*. Semiconductor technologies continue to advance from year to year, as demonstrated in Figure 1.2, where the minimum MOS device dimensions are plotted as a function of time.

- Figure 1.2, where the minimum MOS device dimensions are plotted as a function of time. Until one has developed a fool-proof approach towards "porting" a cell from one technology to another, each change in technology—which happens approximately every two years—requires a redesign of the library.
- Creating an adequate *model* of a cell or module requires an in-depth understanding of its internal operation. For instance, to identify the dominant performance parameters of a given design, one has to recognize the critical timing path first.
- The library-based approach works fine when the design constraints (speed, cost or power) are not stringent. This is the case for a large number of *application-specific designs*, where the main goal is to provide a more integrated system solution, and performance requirements are easily within the capabilities of the technology.
- Unfortunately for a large number of other products such as microprocessors, success hinges on high performance, and designers therefore tend to push technology to its limits. At that point, the hierarchical approach tends to become somewhat less attractive. To resort to our previous analogy to software methodologies, a programmer tends to "customize" software routines when execution speed is crucial; compilers—or design tools—are not yet to the level of what human sweat or ingenuity can deliver.

STC-101-107

馬鹿でも間違えない

移植する

Figure 1.2, where the minimum MOS device dimensions are plotted as a function of time. Until one has developed a fool-proof approach towards "porting" a cell from one technology to another, each change in technology—which happens approximately every two years—requires a redesign of the library.

厳格な

- Creating an adequate *model* of a cell or module requires an in-depth understanding of its internal operation. For instance, to identify the dominant performance parameters of a given design, one has to recognize the critical timing path first.

設計制約

特定用途向け

- The library-based approach works fine when the design constraints (speed, cost or power) are not stringent. This is the case for a large number of *application-specific designs*, where the main goal is to provide a more integrated system solution, and performance requirements are easily within the capabilities of the technology. Unfortunately for a large number of other products such as microprocessors, success hinges on high performance, and designers therefore tend to push technology to its limits. At that point, the hierarchical approach tends to become somewhat less attractive. To resort to our previous analogy to software methodologies, a programmer tends to "customize" software routines when execution speed is crucial; compilers—or design tools—are not yet to the level of what human sweat or ingenuity can deliver.

専用化

- Even more important is the observation that the abstraction-based approach is only correct to a certain degree. The performance of, for instance, an adder can -be substantially influenced by the way it is connected to its environment. The interconnection wires themselves contribute to delay as they introduce parasitic capacitances, resistances and even inductances. The impact of the interconnect parasitics is bound to increase in the years to come with the scaling of the technology.
- Scaling tends to emphasize some other deficiencies of the abstraction-based model. Some design entities tend to be global or external (to resort anew to the software analogy). Examples of global factors are the clock signals, used for synchronization in a digital design, and the supply lines. Increasing the size of a digital design has a profound effect on these global signals. For instance, connecting more cells to a sup ply line can cause a voltage drop over the wire, which, in its turn, can slow down all the connected cells. Issues such as clock distribution, circuit synchronization, and supply-voltage distribution are becoming more and more critical. Coping with them requires a profound understanding of the intricacies of digital circuit design.
- Another impact of technology evolution is that new design issues and constraints tend to emerge over time. A typical example of this is the periodical reemergence of power dissipation as a constraining factor, as was already illustrated in the historical overview. Another example is the changing ratio between device and interconnect parasitics. To cope with these unforeseen factors, one must at least be able to model and analyze their impact, requiring once again a profound insight into circuit topology and behavior.
- Finally, when things can go wrong, they do. A fabricated circuit does not always exhibit the exact waveforms one might expect from advance simulations. Deviations can be caused by variations in the fabrication process parameters, or by the inductance of the package, or by a badly modeled clock signal.

EX_107_21

- Even more important is the observation that the abstraction-based approach is only correct to a certain degree. The performance of, for instance, an adder can be substantially influenced by the way it is connected to its environment. The interconnection wires themselves contribute to delay as they introduce parasitic capacitances, resistances and even inductances. The impact of the *interconnect parasitics* is bound to increase in the years to come with the scaling of the technology.
- Scaling tends to emphasize some other deficiencies of the abstraction-based model. Some design entities tend to be *global or external* (to resort anew to the software analogy). Examples of global factors are the clock signals, used for synchronization in a digital design, and the supply lines. Increasing the size of a digital design has a profound effect on these global signals. For instance, connecting more cells to a supply line can cause a voltage drop over the wire, which, in its turn, can slow down all the connected cells. Issues such as clock distribution, circuit synchronization, and supply-voltage distribution are becoming more and more critical. Coping with them requires a profound understanding of the intricacies of digital circuit design.
- Another impact of technology evolution is that *new design issues* and constraints tend to emerge over time. A typical example of this is the periodical reemergence of power dissipation as a constraining factor, as was already illustrated in the historical overview. Another example is the changing ratio between device and interconnect parasitics. To cope with these unforeseen factors, one must at least be able to model and analyze their impact, requiring once again a profound insight into circuit topology and behavior.
- Finally, when things can go wrong, they do. A fabricated circuit does not always exhibit the exact waveforms one might expect from advance simulations. Deviations can be caused by variations in the fabrication process parameters, or by the induc-

寄生容量
比例縮小
不足
電源線
同期
技術進化
分配
電力消費
ばらつき
洞察力
位相幾何学
偏差

inductance of the package, or by a badly modeled clock signal.

- Troubleshooting a design requires circuit expertise.
- For all the above reasons, it is my belief that an in-depth knowledge of digital circuit design techniques and approaches is an essential asset for a digital-system designer. Even though she might not have to deal with the details of the circuit on a daily basis, the under standing will help her to cope with unexpected circumstances and to determine the dominant effects when analyzing a design.
- Example 1.1 Clocks Defy Hierarchy
- To illustrate some of the issues raised above, let us examine the impact of deficiencies in one of the most important global signals in a design, the clock. The function of the clock signal in a digital design is to order the multitude of events happening in the circuit. This task can be compared to the function of a traffic light that determines which cars are allowed to move. It also makes sure that all operations are completed before the next one starts—a traffic light should be green long enough to allow a car or a pedestrian to cross the road. Under ideal. circumstances, the clock signal periodic step waveform with abrupt transitions between the low and the high values (Figure 1.6a).

問題解決

tance of the package, or by a badly modeled clock signal. *Troubleshooting* a design requires circuit expertise.

For all the above reasons, it is my belief that an in-depth knowledge of digital circuit design techniques and approaches is an essential asset for a digital-system designer. Even though she might not have to deal with the details of the circuit on a daily basis, the understanding will help her to cope with unexpected circumstances and to determine the dominant effects when analyzing a design.

Example 1.1 Clocks Defy Hierarchy

To illustrate some of the issues raised above, let us examine the impact of deficiencies in one of the most important global signals in a design, the *clock*. The function of the clock signal in a digital design is to order the multitude of events happening in the circuit. This task can be compared to the function of a traffic light that determines which cars are allowed to move. It also makes sure that all operations are completed before the next one starts—a traffic light should be green long enough to allow a car or a pedestrian to cross the road. Under ideal circumstances, the clock signal is a periodic step waveform with abrupt transitions between the low and the high values (Figure 1.6a).

Exercise: EX_107

EX_107-1：Slide 07/09/11/15 (原本p2~4)までを読んで、和文または英文でここまでの要点を短くまとめてください(どのようなことをテーマにしているか) -箇条書きでもかまいません。分量が多いと思う人は、キーワードと思う単語を書き出してもいいです。

EX_107-2：Slide 17/20/22/24/26/28/30 (原本p4~9)までを読んで、理解の難しかった英文を2つ挙げてそのまま書いて(コピペしてもいいです)ください。

EX_107-3：Slide 17/20/22/24/26/28/30 (原本p4~9)までを読んで、意味を推定できなかった英単語のなかで、動詞を5語、Technical Termを10語あげてください。

- 注1:自動和訳(翻訳)サイトを使ってもいいですが、その前に必ず英文を通読してください。
- 注2:英文で提出する場合は事前に自動翻訳サイトで、エラーチェックをかけてください。
- 注3:速く読み概要を掴む演習なので、正確さにとらわれてReadingをストップしないように
- 提出はClass Web “レポート” にて木曜まで
- 毎回のレポートは、最低A4 1ページ以上は書いてください。余白には、今回の授業の内容、資料についての感想や要望を記入してください。

難単語とTech_Term ヒント: EX_107

- English Words ほとんどの人が指摘した難単語
  - Adhead (adhere –英検1級)　使われる場面で意味が変わるので、イメージでとらえるといい。
  - Adhesive 接着剤　(剥がれないほど強くくっつく)
  - ついでに覚える sticky べたべたくっつく けどとれる
  - 付箋紙(post-it) sticky note
- いくつかの単語を程度ごとにまとめてイメージで覚えるとよい。
  - Good > Fair > Poor : Very をつけると拡張できる
  - Always >> Usually > Often > Sometimes >>Never

難単語とTech_Term ヒント: EX_107

- 分野別に整理してみる 1
- Computer Aided Design（CAD）関連
  - Computer を使用して ハードウェア設計を自動化することとそのためのツール(Application Software) LSI 設計や基板（PCB –Printed Circuit Board）の設計に使用される。設計仕様の記述(Behavior , Function , Topology) 記述(入力)からの自動設計(Synthesis, Compilation)、検証（Simulation, Verification）に分けられる。LSI設計では、階層(Hierarchy)設計を行う。最上位の実体をEntity という。
- Methodology(方法論)-計算機アルゴリズム
  - Divide and Concur (分割統治法) bitの重みづけのあるバイナリ-2進数で考えてみる。:項の数(ビットの数)をnとするとビットの増加は線形(n+1)でも表現できる数は、2^(n+1)と計算量が指数関数で増加する。よって分割(Divide)を行い計算量を削減して分割した単位ごとに計算を完了(Concur)する。FFT(高速フーリエ変換では、2^nの計算量をnlognにする)

難単語とTech_Term ヒント: EX_107

EX_113_21

Technical Term “topology”

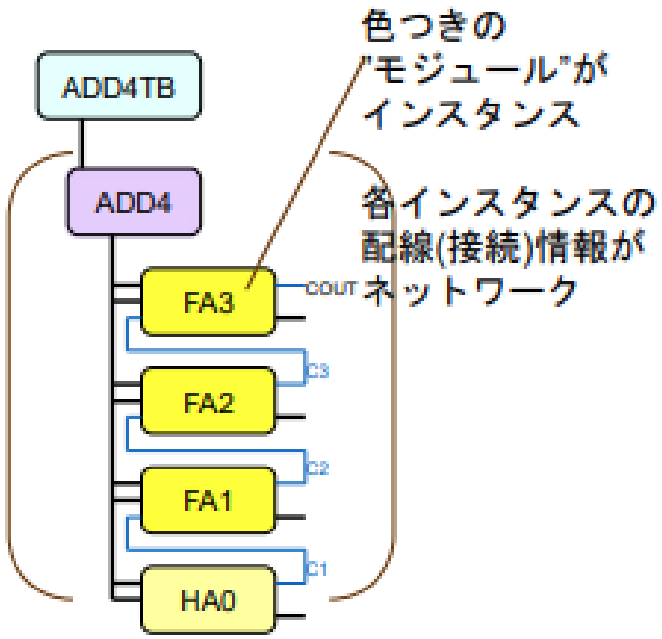
- EX_113 で topology という単語が出てきます。これは、(physical structure / instance and network というニュアンスで) behavior に対する概念で使われます。
- 無理に日本語に訳して解釈しようとすると無理が生じます。Behavior との対比でイメージ(概念)的に理解するとうまくいきます。
- HDLと回路図の対比で具体的に解説します。
- Behavior は“動作”あるは“振舞い”です
- Topology は”具体的な回路要素とその接続”です

難単語とTech_Term ヒント: EX_107

EX_113_21

4 bit 加算器の Topology / Behavior 表現

Add4のTopology (Structure –構造)
記述例：インスタンス(要素/素子)とネットワーク(接続)で対象を表現する。



Add4 のbehavior
(ふるまい、動作)
記述例(HDL記述)

A[a0..a3]

B[b0..b3]

Y[y0..y3]

Y = A + B;

秋学期：EX_209 から

難単語とTech_Term ヒント: EX_107

- 分野別に整理してみる 2
- 回路（Circuit / Technology）関連
  - Digital / Analog
  - Solid State 固体状態 (目視上動かない)
  - SSD (Solid State Drive 半導体メモリ – HDD は回転している)
  - Arithmetic (算術) 整数の四則演算(を行う回路、除算を除くことがある)計算機ではスカラ演算、行列の計算は乗算と加算だが、高速化のためにベクトル演算として専用並列プロセッサを使う。
- 回路に付随する用語
  - Parasitic (寄生的) に発生する容量 (capacitance) など
  - 回路は、能動素子(固有の動作特性を持つ-トランジスタなど)と受動素子(一定温度で固定値を持つが、これら素子の想定固定値以外に回路内に発生する微小な特性が存在する。場合によっては回路動作に影響を与える

Memo

予習：次回の資料に必ず目を通しておいてください。

フォローアップURL (Revised)

http://mikami.a.la9.jp/meiji/MEIJI.htm

担当講師

三上廉司(みかみれんじ)

Renji_Mikami(at_mark)nifty.com

mikami(at_mark)meiji.ac.jp (Alternative)

http://mikami.a.la9.jp/_edu.htm