●特輯● μ-processor의 応用과 現況 및 展望

大学에서의 마이크로 컴퓨터 教育

吳 永 敎

근년, 산업계에서 디지탈技術의 활용이 일반화하고 있었는데, 마이크로 컴퓨터의 등장으로 이 경향이 한층 가속화하게 되었다. 마이크로컴퓨터가 가질 수 있는 높은 성능과, 이에 대조적인 "小型" 및 "廉價" 때문에 사회일반에게 까지 크게 영향을 미칠 것으로 생각되고 있다. 따라서 고급인력을 양성해 내는 대학은 지금 산업계로부터 하나의 긴박한 요구를 받고 있다고 하겠으며, 이러한 시점에서 大學커리큘럼에 마이크로컴퓨터에 관한 과목을 이미 도입한 경우를 살펴보고 우리의 현실에 대비하는 것은 중요한 일이라 생각된다.

커리큘럼 Ⅰ : 주로 講義

마이크로컴퓨터에 대한 교육을 생각할 때 첫째로 유의해야 할 것은, 마이크로컴퓨터는 하아드웨아 및 소프웨아의 기능면에서 볼 때, 기존 전자계산기의 서브시스템이며, 근본적으로 새로운 점(작다는 것을 제외하고는)을 가지고 있지 않다는 점이다. 그러므로 마이크로컴퓨터에 관한 확실한 지식을 얻는 길은 디지탈 시스템에 대한 견전한 교육을 통해서만 가능하며, 이러한 기초없이 다만 어떤 특별한 機種의 마이크로컴퓨터에 관한 해설서만 읽고서 시작한다는 것은 피해야할 문제이다. 한편 마이크로컴퓨터는 프로그램을 가지고 있으므로 종래의 좁은 의미에서의 디지탈 시스템과는 다르다. 현재와 같이 實時間 制御가 주요 응용분야가 되어 있는 경우에는, 주로 하아드웨어 기술자의 관심거리가 되어 있으므로, 이러한 하아드웨어 기술자는 하아드웨어/소프트웨어 양면을 이해해야만 하게 되었다.

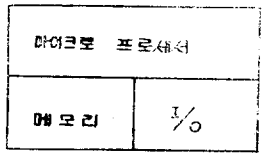

그림 1. 마이크로컴퓨터의 構造

다음에 학부학생들을 위하여 몇몇 대학에서 마련한 마이크로컴퓨터에 관한 카리큘럼을 살펴 보기로 한다.

(1) Berkley大學[1]

(a) 機械語 및 assembley語프로그래밍(프로그래밍實習병행)

(b) 디지탈回路設計槪論(하아드웨어實驗병행)

(c) 入出力機器와 마이크로컴퓨터(마이크로컴퓨터 實驗병행). 이와 동시에 計算機組織槪論

(2) Northwestern大學[2]

(a) 디지탈시스템設計 :

i) Register transfer module(RTM)을 이용한 시스템設計實習

ii) C. Clare著 Designing Logic System Using State Machines: 非同期式制御方式·gate回路와 ROM에 의한 設計法·部分機械의 合成法 등.

iii) T. Blakeslee著 Digital Design With Standard MSI and LSI.

(b) 마이크로컴퓨터를 이용한 시스템設計 : bus에 관하여 protocol의 電氣的 및 論理面의 고찰, 同期와 arbitration. CPU와 周邊回路와의 관계. bit-slice[3],[4]와 마이크로프로그래밍.

(c) 마이크로컴퓨터 소프트웨어設計 : 프로그래밍技法. assembler, loader, debugger, cross-assembler등 소프트웨어 사포오트. documentation技法. 入出力設計. interrupt設計.

(3) 東京工業大學[5]

이 대학에서는 현재까지 강의나 실험실습 단계에서 특별히 마이크로컴퓨터에 대하여 강조하고 있지는 않다.

(a) 情報處理槪論 第1: Fortran 중심

(b) 情報處理槪論 第2: 計算機의 하아드웨아 및 시스템構成

* 正會員 : 漢陽大工大電氣工學科敎授(學會編修委員)

(c) 計算機組織 : F. Hill and G. Peterson著 Digital Systems: Hardware Organization and Design, 2nd ed.; 計算機하아드웨어 및 動作方式

(d) 데이타構造

(e) Assembly語

(f) Fortran 및 Assembly語 프로그래밍實習

(g) 마이크로컴퓨터를 이용한 시스템設計 또는 마이크로 컴퓨터를 위한 시스템프로그램의 開發 : 주로 情報工學科(Dept. of Computer Science) 4학년생들 중 상당수가 卒業硏究로서 이 제목을 선택.

(4) Stanford大學[6]

(a) 計算機組織, 機械 및 Assembly語 槪論 : H.Stone 혹은 H. Stone and D. Siewiorek著 Introduction to Computer Organization and Data Structure; 高級言語에 의한 프로그래밍實習을 전제. 기본적인 計算機組織. 미니컴퓨터用 assembly語. 기본적인 入出力프로그래밍. 간단한 데이타構造.

(b) 디지탈計算機組織 : M. Mano著 Computer System Architecture; 기본적인 스위칭理論 및 論理設計. 演算裝置. 記憶表置. 入出力. 마이크로프로그래밍

(c) 디지탈回路實驗 : J. Wakerly著 Logic Design Products Using Standard Integrated Circuits.

위에서 본 바와 같이 각 대학은 학생들의 기초실력 양성에 힘을 기울이고 있으며, 이 중에서 특히 Stanford大學案이 무난한 것으로 필자에게는 생각된다. 특히 이 大學에서 (c)에 각별히 힘을 기울이고 있는 것은 주목할만하다. 다만 채택된 3권의 책 중에서 내용을 발췌하여 2學期 정도로 마쳐보는 것도 하나의 방안일 것 같다.

마이크로컴퓨터의 프로그램은 비교적 소규모이고 또 되풀이해서 이용되므로 효율이 중요시 되는 것이 그 특징이다. 그런데, 현재 高級言語는 그 object code의 효율이 좋지 않으므로, assembly語가 마이크로컴퓨터에서 주류를 이루고 있다. 그러나 第3世代(intel社의 8086, Zilog社의 Z8000, Motorola社의 M68000등은 1980年代의 初半에 出荷가 순조로울 것으로 보고 있다)에서는 高級言語의 효율이 개선되어 있을 것이므로[7], 高級言語에 의한 프로그래밍 또한 중요한 수단이 될 것이다. 여하튼 assembly語프로그래머는 入出力프로그래밍, interrupt 및 real-time프로그래밍에 익숙하게 되고, 또한 assembly語프로그래밍을 잘하면 高級言語에 의한 프로그래밍은 쉽게 익힐 수 있으나, 그 거꾸로는 잘 안된다는 사실을 감안할 때, assembly語에 대한 교육은 타당성을 유지할 것으로 보인다.

bit-slice마이크로프로쎄서로 CPU를 구성하기는 힘이 드는 것으로 알려져 있으나, 마이크로컴퓨터에 대한 교육 이전에 論理回路設計에 관한 과정에서 부터 bit-slice를 도입한 체계적인 한 커리큘럼이 제안되어 있다.[8]

커리큘럼 II : 주로 實驗

머릿말에서 지적한 바와 같은, 지금 요구되고 있는 사람을 교육하기 위해서는 커리큘럼 1과 아울러서, 이제부터 고찰할 실험실습을 충분히 하므로써만이 가능하다는 것은 자명한 일이다. 기본적인 마이크로컴퓨터 實驗을 어떻게 계획할 것인가에 대하여 생각해 보기로 한다.

(1) 시간배당

週당 4시간을 예정하고 아래와 같이 배당해 본다.

표 1

	총시간	강의시간	실험시간	비　　고
전 반	32	8	24	기 본 과 제
후 반	28	7	21	자 유 선 택

커리큘럼 I 중의 (4)에 해당하는 과정을 마친 학생이 이 실험과정을 거치는 것으로 하는데, 전반에서는 어떤 특정된 마이크로컴퓨터를 이용한 설계와 프로그래밍에 대한 기본 기법을 다루게 된다. 이때에 학생들은 주어진 제목에 대하여 프로그래밍하고, 이 프로그램으로 움직일 하아드웨어를 만들게 된다. 그 다음 후반에서는 가능한 범위에서 학생 스스로 제목을 택하여 거기에 따라 시스템을 구성하도록 한다. 후반기에서의 강의는 여러가지 마이크로컴퓨터의 구조와 성능상의 장단을 비교한다.

전반에 해당하는 실험에 관한 보고가 몇 개 있는데, 그들이 목적했던 바를 간추려 보면 아래와 같다.

(a) 어떤 특정한 入出力機器의 하아드웨어制御回路를 가능한 한 소프트웨어制御로 바꿔보는 것.[1]

(b) 프로그래밍에 의하여 각종 波形을 만들어, 이것을 오실로스코오프로 관측해 보므로써 하아드웨어論理와 소프트웨어論理가 하나가 된 새로운 情報處理空間의 체험.[9]

(c) Direct memory access의 각종 制御方式의 특징, interrupt處理, 소프트웨어와 하아드웨어가 접하는 점의 명확화, 소프트웨어와 하아드웨어 어느 쪽을 택할 것인가에 대한 trade-off의 인식, 操作패널에서의 소프트웨어debug방법 등에 대한 이해.[10] 그리고,

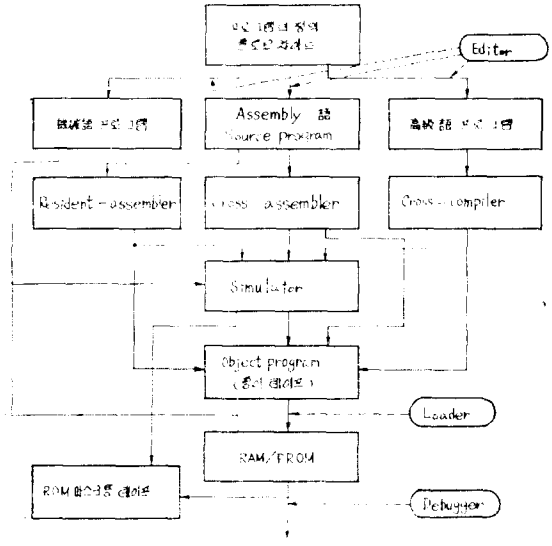

그림 2. 마이크로컴퓨터의 프로그램작성 순서

각 기본과제는 3시간 정도의 준비(집에서)를 할 때, 6시간 정도로 마칠 수 있는 것이 좋을 것 같다.[1)]

(2) 마이크로컴퓨터의 프로그래밍

마이크로컴퓨터는 미니컴퓨터와는 달리, 部品으로서 공급되는 경향이 강하기 때문에, 미니컴퓨터에 비하여 소프트웨어사포오트가 아주 약하다. 따라서 마이크로컴퓨터에 있어서는 프로그래밍이 가장 어려운 문제라 해도 과언이 아니다.

그림 2에 마이크로컴퓨터의 프로그램開發순서를 나타내고 있다.

그림 2에서 assembler에 두가지가 있음을 알 수 있다.

(a) Resident方式 : 프로그램對象으로 하는 마이크로컴퓨터와 같은 프로세서를 가진 마이크로컴퓨터를 이용한다. 다른 計算機에 비해 값이 싸고 同一 프로세서이기 때문에 debugging이 편리하지만, 低速, 사포오트 소프트웨어 및 週邊機器가 약하기 때문에 능률은 높지 못하다.

(b) Cross方式 :

i) Batch; 미니컴퓨터 또는 大型機에서 assemble하여, 그것을 종이 테이프를 메체로 하여 마이크로컴퓨터에 옮긴다. 初期코스트 및 턴아라운드時間이 커진다.

ii) TSS; TSS 端末에서 프로그램을 작성, 시뮬레이션한 후, 종이 테이프를 통하여 마이크로컴퓨터에 이동시킨다. 국내에는 아직 이 방식이 없는것으로 안다.

(3) 마이크로컴퓨터의 선택

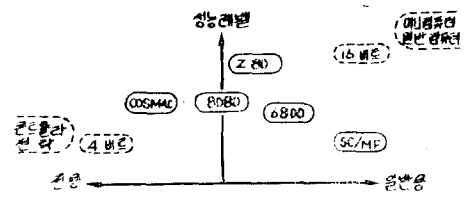

그림 3. 各種마이크로컴퓨터의 성격과 성능

어떤 마이크로컴퓨터를 선택할 것인가는 중요한 문제이나, 학생실험에 쓰이는 만큼 첫째로 出力의 短絡에 강하고 電源이 간단한 것이 좋으며, 一般用(general purpose)인 것이 바람직하다. 그리고 (2)에서 살핀 것과 같이, 프로그램을 어떻게 정확하고 쉽게 만들 수 있는가 하는 점에서 판단하는 것이 중요하다. 그러나 사용목적이 교육용인 만큼 프로그램의 크기는 그다지 크지 않을 것으로 생각되므로 (약 100스텝 내외) 보통 敎育用킷트라는 이름으로 공급되는 것 중에서 택하는 것이 현실적일 것 같다. 이 킷트는 현재 assembler를 갖고 있지 않으므로 hand-assemble (assembler는 hand-assembler, resident-assembler (혹은 self-assembler) 및 cross-assembler로 나눌 수 있다)하여 프로그래밍하게 된다.

현재 이용할 수 있는 킷트는 대개 아래와 같다.

intel SDK 80, 日電 TK 80, 東芝 EX 0, EX 5, Panafacom LK 16, Motorola MEK6800DⅡ, NS SC/MP kit, AMD AM2900 K1, TI LCM 1001.

위에서, 대학 학부과정에 마이크로컴퓨터를 새로이 도입할 경우를 생각하여 몇가지 점을 고찰해 보았다. 이 중에서 實驗實習이 중요한 위치를 차지할 것으로 생각된다. 프로그램 방법으로 hand-assemble을 일단 가정하였으나, 이 방식에 의하면 여러가지 불편이 있을 것은 분명한 일이므로 보다 나은 사포오트를 갖추는 것이 바람직 하다. 그 하나의 방안으로서는 resident方式에 floppy disk와 ASCII keyboard, on-line alphanumeric display 및 小型프린터를 갖추는 방안이다. 이때 floppy disk에 들어 있을 소프트웨어는 필요최소한의 것으로부터 차차 확대해 나가면 된다. 한편 크로스方式을 손쉽게 이용할 수 있다면 高速종이테이프리이더를 갖춰야 한다.

표 2와 그림 4는 실험실을 준비할 때 참고로 하기 위하여, 일반적으로 알려져 있는 소프트웨어開發手段과 debugging手段에 관한 설명이다. 표 2에서는 윗부분에 있는 것들 보다는 아랫쪽에 기록된 수단이 더

표 2. 소프트웨어開發手段[11]

PROM programmer
Oscilloscope
Programmer panel
Prototyping kit
Logic analizer
Microprocessor analizer
In-circuit emulator
Paper-tape-based development system
Minicomputer and large computer system
Disk-based development system

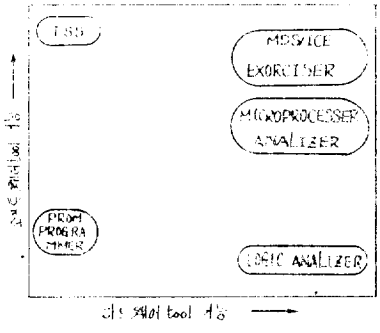

그림 4. debugging手段

효과가 크다는 것을 뜻하고 있다. 여기서 disk-based development system은 floppy disk와 in-circuit emulator[6),13)]를 갖추고 있다. 또 그림 4의 MDS/ICE란 intel社에서 개발한 마이크로컴퓨터開發시스템(MDS)에 in-circuit emulator(ICE)를 함께 쓰고 있다는 것을 나타내고 있고, Exorciser는 Motorola社의 開發시스템이다.

만일 대학원 레벨에서의 마이크로컴퓨터를 생각한다면, 그것은 자연히 project laboratory 위주가 될것이며, 實時間制御시스템 array processor system, pipelining system, local network 등 문제가 다양하다. 產學協同의 좋은 예로서는 Worcester 工大의 경우가 참고가 될 것 같다.[12)]

오늘날의 공학교육은 기초교육의 충실과 아울러서, 급변하는 산업기술에 대응할 수 있는 유연성을 그 어느때보다도 필요로 하고 있다. 대학에서의 마이크로컴퓨터교육은 필자의 얕은 경험만으로는 다루기 힘든 제목이었으나, 그 중요성을 느껴 한번 고찰해 보았다. 보다 나은 플랜을 위해서 독자 여러분의 동참을 바라마지 않는바 이다.

참 고 문 헌

1) Imsong Lee," "Hands-On" Approach to Microcomputer Education," Proceedings of the IEEE, June 1976
2) Bernard W. Jordan, Jr., "Microprocessors in a Digtal System Design Curriculum", Proceedings of the IEEE, June 1976
3) George Reyling, Jr., "Considerations in choosing a microprogrammable Bit-Sliced Architecture", IEEE Computer, July 1974
4) Nikitas A. Alexandridis, "Bit-Sliced Microprocessor Architecture," IEEE Computer, Jan 1978
5) 學部學習案內 및 敎授要目, 東京工業大學 1978.
6) John F. Wakerly and Edward J. McClusky, "Microcomputers in the Computer Engineering Curriculum," IEEE Computer, Jan 1977
7) William Fletcher, "Computer Science and Enigineering Education, The Digital Logic Area," IEEE Computer, Dec 1977
8) Robert Sugarman, "Our Microuniverse expands," IEEE Spectrum, Jan 1979
9) 安田 壽旺, 學部學生實驗을 위한 마이크로컴퓨터, bit, Vol. 10, No. 5, 日本共立出版社, 1978
10) 阿章清滋, "大學敎育과 마이크로컴퓨터", 인터페이스 No. 10, 日本 CQ出版社 June 1977
11) Celeste S. Magers "Managing Software Development in Microprocessor Projects," IEEE Computer, June 1978
12) Tom Collins, "A Radical Experiments in Teaching," IEEE Spectrum, Aug 1978
13) Larry Krummel and Gaymond Schultz, "Advances in Microcomupter Development Systems", IEEE Computer, Feb 1977