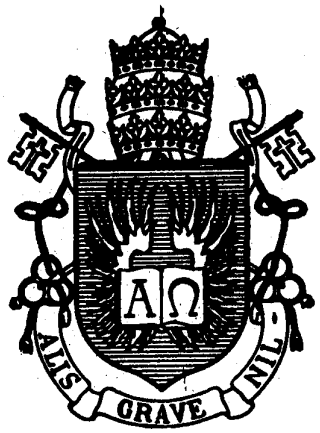

PUC

Series: Monografias em Ciência da Computação
Nº 1/79

MICROPROGRAMMING
PART 0: THEORETICAL EVOLUTION OF THE MICROPROGRAM CONCEPT

by

Ayola N. Akonteh

Departamento de Informática

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO
RUA MARQUÊS DE SÃO VICENTE, 225 – CEP-22453
RIO DE JANEIRO – BRASIL

PUC/RJ - DEPARTAMENTO DE INFORMÁTICA

Series: Monografias em Ciência da Computação
Nº 1/79
Series Editor: Daniel A. Menascé

March,1979

MICROPROGRAMMING
PART 0: THEORETICAL EVOLUTION OF THE MICROPROGRAM CONCEPT*

by

yola N. Akonteh

* This work has been sponsored in part by FINEP.

Abstract:

This is Part 0 of a series on microprogramming. It developes basic concepts and the mathematical representation of each stage in the evolution of microprogramming.

Key words:

Control levels, interpretation, mapping, micro-operation, microinstruction, microroutine.

Resumo:

Este trabalho é a parte 0 da serie de monografias sobre microprogramação. Esta parte desenvolve os conceitos básicos e a representação matemática de cada etapa da evolução de micro - programação.

Palavras Chave:

Níveis de contrôle, interpretação, mapeamento, micro-operação, microinstrução, microrotina.

INDICE

MICROPROGRAMMING

PART 0: THEORETICAL EVOLUTION OF THE MICROPROGRAM CONCEPT.

0.0 Introduction

A digital computer consist basically of 5 major components (see Figure 1), one of which is the control unit. The control unit in a digital system regulates primitive operations in all digital systems components (registers, I/O devices, etc) through control signals called micro-operations. Some of the functions of the control unit in digital systems can be summarised as follows:

(i) - Determination of instruction to be executed;

(ii) - Data paths regulation through control signals;

(iii) - Data interpretation;

(iv) - Timing of memory cycles.

Control in digital systems can be implemented in two forms, viz: Hardwired random logic or microprogramming.

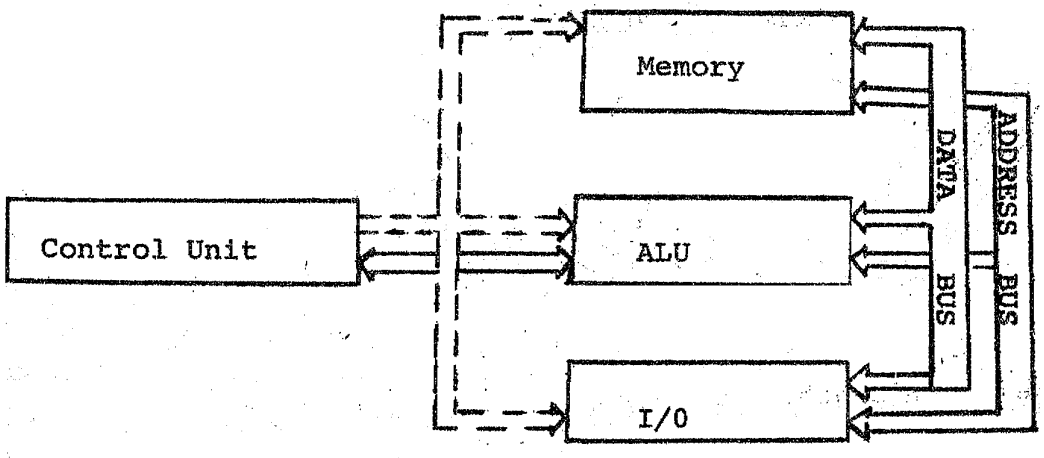

Figure 1 Basic Organization of a digital computer.

This study attempts to place into perspective the concept of micro program control in digital systems as used in todays literature.

0.1 - The general nature of control in digital systems.

Control in digital systems is hierarchical (see figure 2) of which the lowest level is represented by intra-register transfers, usually regulated by clock pulses and other control conditions. This level of control is describable by a register transfer language and can be directly implemented through combinational logic. Seve - ral other levels of control are shown up to the control of a compu - ting environment by an operating system.

Some reciprocity exist between control at some level j, $(K_j(t))$ and interpretation $I_j(t)$. This reciprocity which is represented schematically below (see figure 2) can be functionally stated for control $K_j(t)$ and interpretation $I_j(t)$ as follows:

$$K_j(t) = \phi^{-1}(I_j\text{-}1(t)) \quad \ldots\ldots\ldots\ldots\ldots\ldots (0.1)$$

$$j=1,2\ldots\ldots\ldots n$$

$$t\geq 0$$

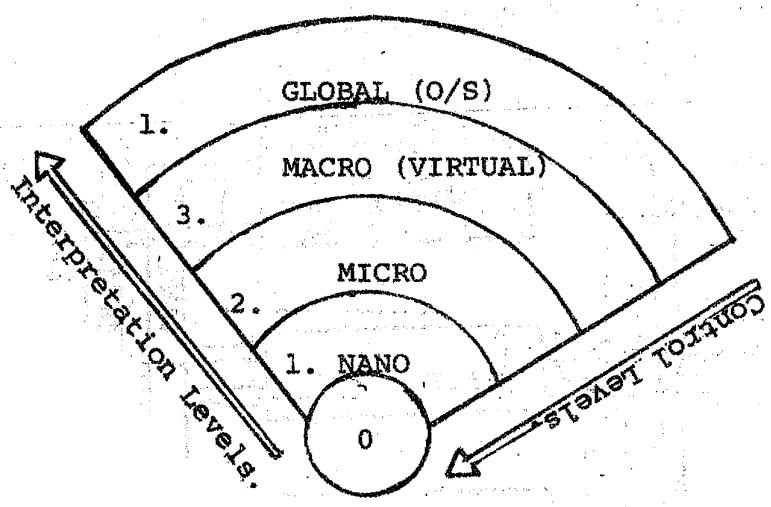

Figure 2 Hierachical control in digital systems.

This study considers the primitive levels (nano and micro) that deal directly with register - to- register transfers, control of computer resources and interpretation of higher levels of control (virtual and global). The focus of the study, however, is not control per se, but the theoretical evolution of microprogrammed control, from hardwired control through the Wilkes model to todays concept of microprogram control as used in the literature.

0.2 - Basic Premises of Control in Digital Systems

Data processing in digital systems is characterised by inter-register data transfers that can be regulated by some control function $K_j(t)$. For example a transfer from register A to B could be functionally represented as

$$K_j(t) : \quad B \leftarrow A.$$

A schematic realisation of the above micro-operation is shown in figure 3 and can be implemented through hardwired logic circuits.

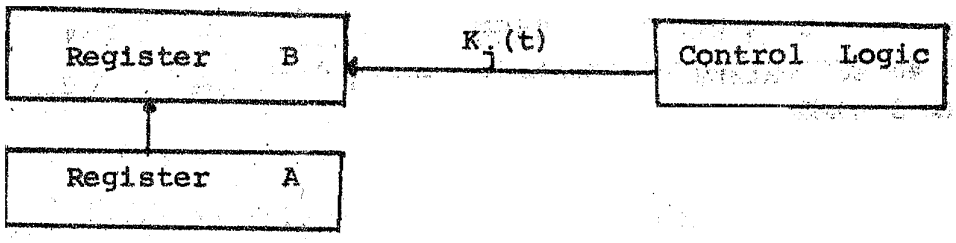

Figure 3 Transfer from register A to B

Though functional, this scheme faces several limitations in larger scale applications. This is so because of the complicated data networks in digital systems with several inter-register transfers usually required to realize a single computational operation. Despite these limitations, hardwired logic has been used over the years, long before the advent of intergrated circuit technology which

made microprogramming a viable alternative.

Microprogramming as a control tool, has evolved through at least 3 distinct theoretical stages. The first stage is marked by the primitive implementation of control functions through use of random logic (early 50s and 60s). The second stage is marked by Wilkes model (control matrix, introduced in 1951 but used commercially for the first time in the 60s) characterised by a diode matrix. The third stage which is marked by sub-stages and is still evolving is marked by the systematic coding of micro programs on the now available semiconductor memories (ROMs, RAMs, EPROMs, etc) born in the 60s. In this study each evolutionary stage is examined briefly to determine the deductive perception that went into developing the sucessor stage.

0.3 Hardwired Control (Random Logic Control)

Historically digital control took the form of randon logic networks from the combinational and ad hoc nature of its design. This scheme (see figure 4) can be derived in terms of a specific system which consist basically of an instruction register R, instruction decoder D, and a clock T_j that emits a periodic pulse j. For a given function $f_k(t)$ (e.g. ADD function) there exist an activating signal $S_k(t)$ defined in terms of other functions and some operators E_k as follows:

$$S_k(t) = \emptyset(f_k(t), E_k) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (0.2)$$

$t \geq o$, $k=1,2, \ldots n$

$E_1, E_2, E_3 \ldots\ldots E_n$ may be assigned the operators OR(+), XOR($\oplus$), AND(.), etc.

Since $t \geq o$, is exogenous to the system, the parameter pair (k,t) define a unique micro-operation $S_k(t)$.

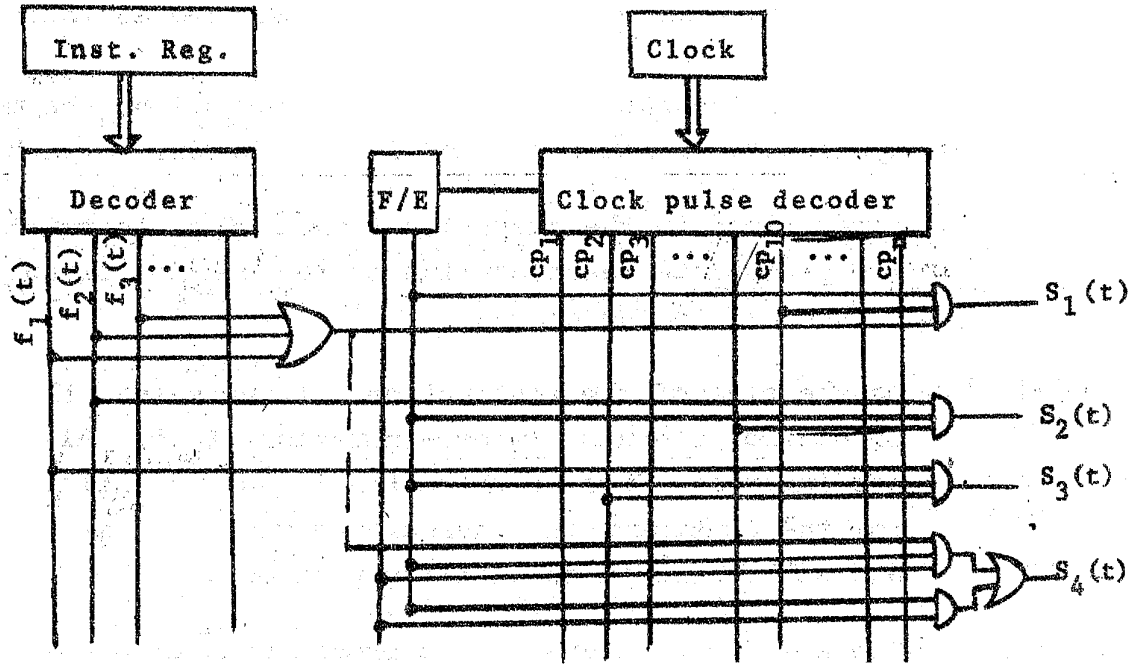

Figure 4. Hardwired Control Layout

Figure 4 above shows an internal layout of a hardwired controlled digital system. The functions CLA (clear), SUB (subtract), ADD (Add), etc can be represented in a general form as $f_1(t)$, $f_2(t)$, and $f_3(t)$ respectively with the controlling signal $S_k(t)$ dependent on the clock pulse CP(k). Each $S_k(t)$ emitted is regarded as a micro-operation directed at controlling some specific resource $r \in R$, through activation of an appropriate logic gate. In the above scheme, the source instruction (macroinstruction) is decoded to activate appropriate functions $f_k(t)$ (ADD, SUB, CLA, etc) that recognizes the memory cycle (fetch or execute) and appropriate clock pulse to generate the control signal $S_k(t)$. Such a signal (micro-operation) $S_k(t)$ for the clear (CLA) function $f_1(t)$ can be expressed as follows:

$$S_1(t) = (f_1(t) + f_2(t) + f_3(t).(E).CP(10).....(0.3)$$

where $t \geq o$, and E indicates that the machine is in the execution cycle. Similar equations can be developed for read (R), add (ADD), subtract (SUB), etc to determine their individual control signals (micro-operation) $S_k(t)$.

This scheme, though functional, is however shrouded with several problems, some of which can be summarised as follows:

(i) - Whatever the nature of the macroinstruction, the memory cycle time Δt remains fixed in most hardwired machines.

(ii) - Alterations in the execution format of an instruction imply complete restructuring of the control circuitry to accomodate the change.

(iii) - As the size of the instruction set increases, the number of control signals (micro-operation, $S_k(t)$ also increases implying large combinational and sequential circuits for the control unit; associated with this may be race and other circuit reliability problems.

(iv) - There is lack of systematic programming and possible modification of $f_k(t)$ in a way that new instruction sets could be introduced after the system is built.

(v) - The scheme does not allow for an internal modification of $S_k(t)$, rather it relies on the generation of $S_k(t)$ based on the built-in functions $f_k(t)$.

The above constraints together with advances in semiconductor technology provide a basis for systematizing control in digital systems. The works of Professor M.V. Wilkes (Cambridge University, 1951) forms the basic and most important contribution in the systematic design of control in digital systems.

0.4 - The Wikes Model

The wilkes model (W) constitute the first attempt to incoparate systematic control in digital systems through substitution of random hardwired control logic with a programmed logic matrix. A summary diagram of the model together with explanatory notes is shown in figure 5 below.

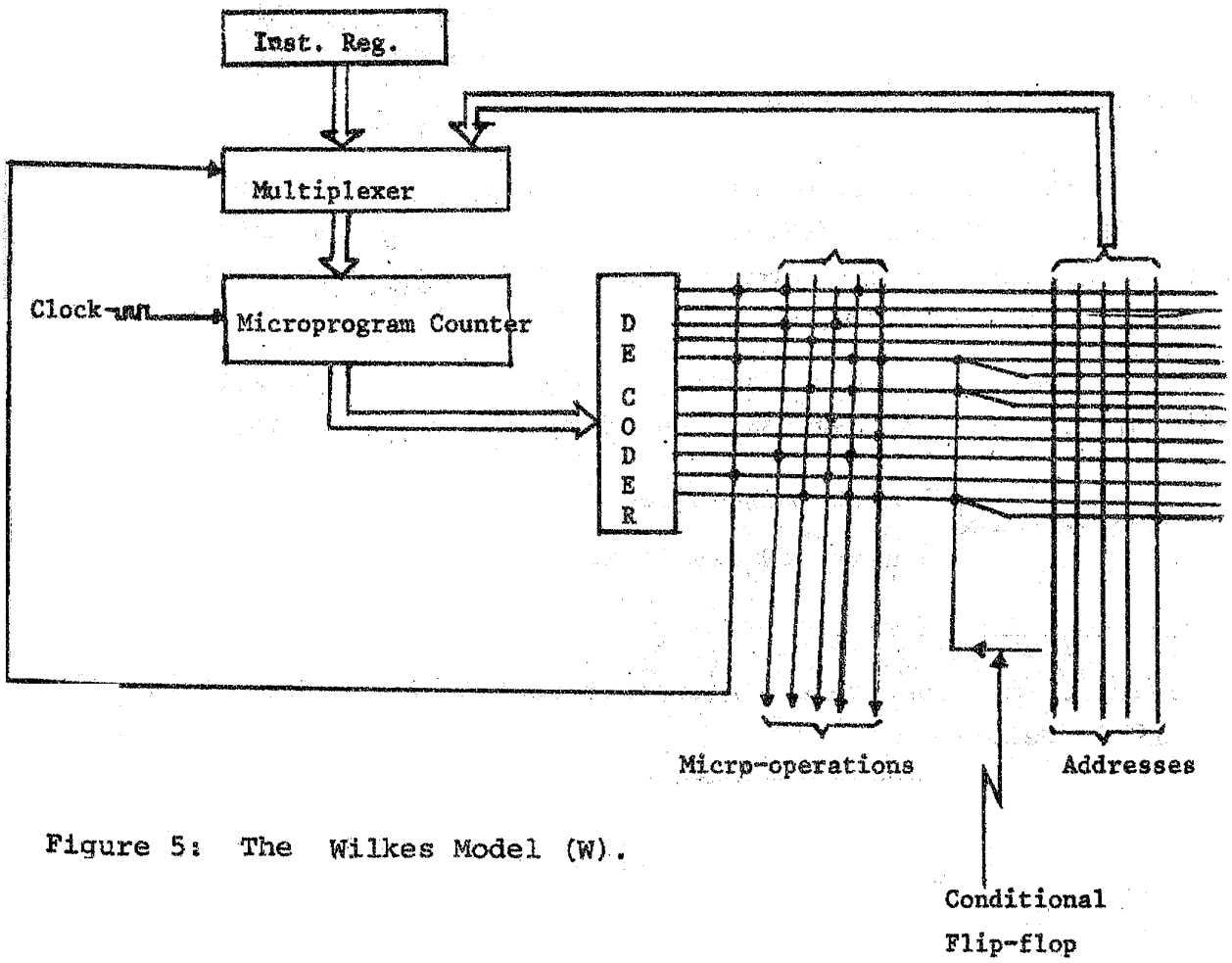

Figure 5: The Wilkes Model (W).

The W scheme can be described in terms of two matrices A, B and a column vector C which are expressed mathematically as follows:

$$W = \psi(A,B,C) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (0.4)$$

'A' is an $(m \times n)$ matrix of which each matrix element a_{ij} for a defined row $i(=1,2,3,\ldots\ldots m)$ and column $j(=1,2,3,\ldots n)$ are interpreted similary as micro-operations.

Definition 1:

A micro-operation is a primitive operation (AND,OR,XOR,etc) realizable in a single clock pulse on information stored in one or more registers and meant to directly control some specific activity or resource in a digital system.

'B' is a $(m \times \ell)$ matrix of which all column elements $j(=n+2,n+3,\ldots\ell)$ for a defined row $i(=1,2,\ldots m)$ constitute the address of the next microinstruction. 'ℓ' represents the total size of the address field.

Every element $c_{ij}(j=n+1)$ of the column vector for any defined row $i(=1,2,3,\ldots m)$ can take one of two possible values, viz

$$c_{ij} = \begin{cases} 1 & \text{next address determined by some condition;} \\ 0 & \text{next address is read off directly from matrix B.} \end{cases}$$

Thus the elements a_{ij}, b_{ij}, c_{ij} for a defined row i $(=1,2,3,\ldots m)$ constitute a microinstruction of the form shown in figure 6 below.

Row 1	$a_{1j}(j=1,2,3\ldots n)$	$c_{1j}(j=n+1)$	$b_{1j}(j=n+2,n+3,\ldots\ell$

Figure 6 The W micoinstruction format.

The operation of W is straight forward; the actual matrix scheme to realize any given micro-operation is derived from a diode matrix (see figure 7).

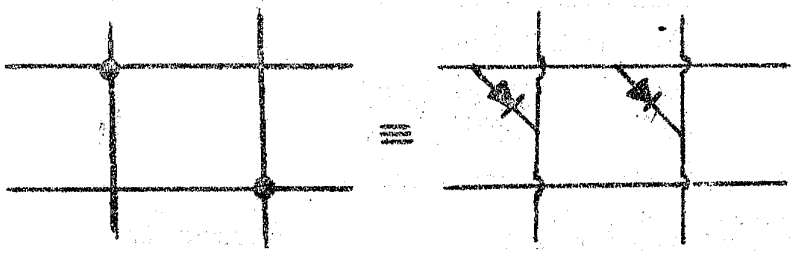

Figure 7 W matrix elements and the equivalent Diodes

In all the W scheme can be summarised (see figure 5) follows:

The multiplexor selects the operation code of the institruction being executed and presents this as the address of the first microinstruction to be executed.

At the arrival of the clock pulse, the microprogram counter synchronously passes this address to the address decoder.

The decoder decodes this address and uses it to activate a pacticular row i of matrix A,B, and vector C (this correspond to a microinstruction).

The activated microinstruction contains the address b_{ij} $(n+1< j\leq \ell)$ of the next microintruction to be executed. This address can be modified pending the value of c_{ij} $(i=1,2,3...m;j=n+1)$.

Hence, W distinguishes at loast 3 instruction fields; micro-operation, condition(s) and address of the next microinstruction. It therefore conforms to the concept of microprogramming as used today; where the control unit is permitted to control basic elements of digital system using primitive signals encoded into micro-instructions. A limitation of the above scheme lies in the need to modify hardware each time the microprogram is modified.

0.5 - Microprogramming

The advent of intergrated circuit technology has made possible the development of advanced memory systems (ROMs,RAMs, PROMs, etc.) capable of implementing control as proposed by Wilkes. Though this set of memory systems offer some flexibility, programs once stored on ROMs (Read-Only-Memories) cannot be modified easily. RAMs (Random-Access-Memories) permit modifications in existing microprograms and EPROMs (Erasable-Programmable ROMs) permit modifications in existing microprograms.

The microinstruction* format stored in these memories vary, but in general can take the encoded form shown in figure 8 below. There are several forms of microinstruction designs and their

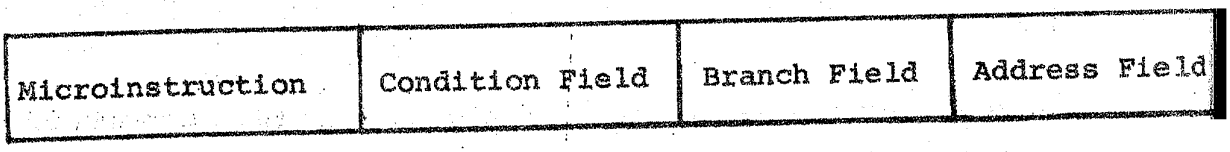

Figure 8 General microinstruction format.

encodings; the most common of which are the horizontal and vertical encodings. Horizontal microinstructions are characteristically long with each bit (micro-operation) directly controlling specific resources.

Vertical instructions on the other hand are encoded into fields with each field controlling some distinct group of resources (for example peripheral devices, ALU, etc).

An instruction in a microprogrammed computer is normally interpreted by a series of microprograms stored in a control memory. One of the methods to determine a microprogram associated with a given instruction is through a mapping process. In mapping the opcode of the given instruction is transformed to the address of the first microinstruction of its interpretive microsubroutine contained in the control memory. There are several intermediate manipulations some of which are shown in figure 9 for the instructio

* A microinstruction is a set of micro-operations (encoded or non-encoded) that control primitive operations in a digital system.

subtract (SUB)

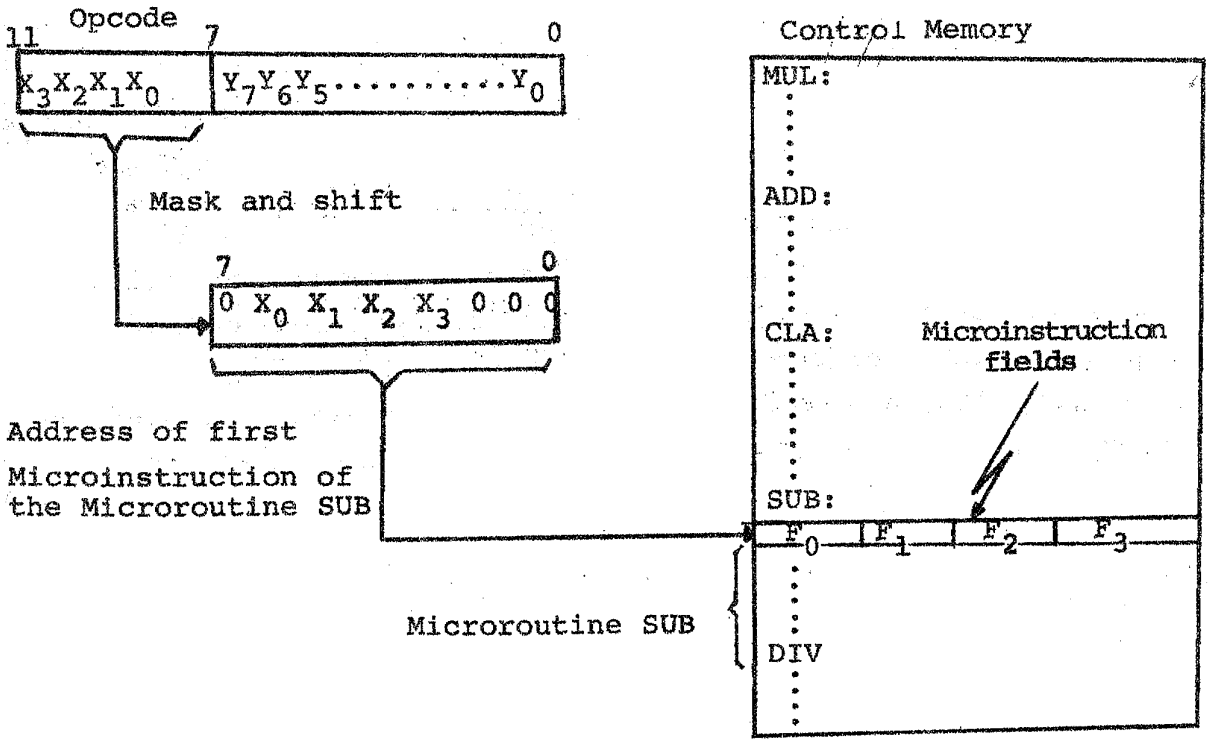

Figure 9 Mapping the instruction SUB into a control memory.

This mapping process constitute an important link between two programming levels: macro - and micro-programming. Micro-instruction processing follow the customary cycles of fetch, decode, execute found in higher level programming.

CONCLUSIONS

The evolution of the microprogramming concept is traceable through the Wilkes model to the original hardwired control in digital systems. The growth in microprogram implementation will increase with evolutions in semiconductor technology and will offer an alternate level of program implementation and greater flexibility in new functions definition in digital systems.

Bibliography:

Akonteh, A.N. Microprogramming: Principles and Developments
Monograph in Computer Science, #4/79, PUC-RJ

Akonteh, A.N. Microprogramming: Trends in Emulation Technology
Monograph in Computer Science, #6/79, PUC-RJ

Wilkes, M.V. Microprogramming Principles and Developments
Infotech State of the Art Report #23, 1975