INTRODUCTION

The computer industry, during its short life of approximately 20 years, has seen many innovations, evolutionary trends and developments, and has even accommodated several major revolutionary technological breakthroughs. One of these revolutionary-evolutionary developments has been the increasing use of small memories as scratchpads.

A working definition of scratchpad memories is given in this paper. From that definition three specific, nonoverlapping applications are derived. One of these, the use of scratchpad memories for control purposes, has been developed and employed at Honeywell for some time. Other manufacturers have also employed scratchpad memories—for example, Burroughs in their D825, UNIVAC in their 1107, and RCA in the larger members of their SPECTRA 70 systems. Discussions of this particular aspect of scratchpad, or control, memory forms the major portion of this paper. In separate sections, the specific uses of control memory in the Series 800, the Series 200 and the Honeywell 8200 are described.

In the succeeding section some prognostications concerning the state-of-the-art in control memories, based upon the results of extrapolations to 1970, are presented. A simplified tradeoff analysis is also developed and discussed.
THE HONEYWELL SERIES 800 FAMILY OF
DATA PROCESSING SYSTEMS

The Honeywell 800 was announced in early 1958. At the grossest level, the system appears to
be a very conventional system of that time period. However, this is misleading. It was a dynamic and
revolutionary jump beyond the then current state­
of-the-art in computer system organization. A
few of the H-800’s contemporaries and competi­
tors are characterized in Table 1.

The Honeywell 800 differs from machines of its
own era, and indeed from most currently available
systems which have been designed and built during
the interim. Figure 1 presents a diagram of the

Table 1. H-800 Competitor Analysis.

<table>
<thead>
<tr>
<th>Machine</th>
<th>First Instal-</th>
<th>Monthly Rental</th>
<th>Number Add Word</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>lation Date</td>
<td>thousands of</td>
<td></td>
</tr>
<tr>
<td>IBM 7070</td>
<td>6/60</td>
<td>11-31 dollars</td>
<td>6</td>
</tr>
<tr>
<td>H-800</td>
<td>12/60</td>
<td>19-35 dollars</td>
<td>6</td>
</tr>
<tr>
<td>GE 225</td>
<td>1/61</td>
<td>2.5-25</td>
<td>18</td>
</tr>
<tr>
<td>RCA 301</td>
<td>2/61</td>
<td>3.5-25</td>
<td>7</td>
</tr>
<tr>
<td>G-20</td>
<td>4/61</td>
<td>14-22</td>
<td>6</td>
</tr>
</tbody>
</table>

Honeywell 800 system organization. In most respects
it resembles a “conventional” computer. The differ­
ence is in the crosshatched area in the control unit
which is referred to as the “multiprogram control.”

The multiprogram control permits the total system
to operate the I/O devices with a maximum effi­
ciency as well as to time-share the arithmetic cir­
cuits among a maximum of eight separate and dis­
tinct programs.

The physical implementation of the multiprogram
control uses a 256-word scratchpad core memory. This memory—a control memory—was de­
sign ed with a memory cycle time equal in duration
to the memory cycle time of the main high-speed
core memory. However, to facilitate the interleav­
ing of register accesses and to minimize dead time,

the control memory timing was designed to be 180
degrees out of phase with the main memory.

Within this context let us consider the functions
of the control memory. Table 2 presents a list of
functions, each associated with a unique register or
location, for a total of 32 locations. This set is re­
ferred to as a control group. Each control group of
32 registers operates and controls the entire pro­
cessing system for a single program. Each control
group provides the programmer with the powers and
facilities of a conventional computer. Since there are
eight such groups, the Honeywell 800 is eight com­
Table 2. List of Registers Contained in a Single H-800 Control Group.

<table>
<thead>
<tr>
<th>Function</th>
<th>Number of Registers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sequence registers</td>
<td>2</td>
</tr>
<tr>
<td>Sequence history registers</td>
<td>2</td>
</tr>
<tr>
<td>Index registers</td>
<td>8</td>
</tr>
<tr>
<td>Unprogrammed transfer registers</td>
<td>1</td>
</tr>
<tr>
<td>Mask index register</td>
<td>1</td>
</tr>
<tr>
<td>Arithmetic control counters</td>
<td>2</td>
</tr>
<tr>
<td>General-purpose registers</td>
<td>12</td>
</tr>
<tr>
<td>Read/write address counters</td>
<td>4</td>
</tr>
<tr>
<td>TOTAL</td>
<td>32</td>
</tr>
</tbody>
</table>

 Each control group contains two sequence counters which permits the program to select and/or return to alternate sequences or paths of processing on an instruction-by-instruction basis. For each sequence counter in the system, there is a corresponding history register. These registers are used to store the contents of a sequence counter whenever a branch is taken by an instruction. This permits a first level of return from a subroutine and also assists in program debugging.

 Most of the registers listed in Table 2 are self-explanatory; however, two items merit some further comment. The general-purpose registers, like the index registers, are used primarily for address modification. Their use differs from that of index registers in several important respects. For example, the contents of these registers can be incremented or decremented (to an upper limit of 31) following their use. These registers are employed primarily to address an operand or a result location in any bank of memory, but they may also be used as programmed counters, as temporary storage for the contents of other special registers, and for any other purpose the programmer may devise.

 The other set of locations in the control memory which require additional comment are the read/write address counters. To understand their use we must first examine the input/output interface and the traffic control system.

 The traffic control directs the time-sharing of memory by the peripheral units and the central processor. Its main object is the efficient use of the entire system according to a set of priorities which are derived directly from the nature of the peripheral equipment and are independent of the program.

 The creation of a demand signal by any device is shown in the corresponding control stage. When any program has been turned on in the central processor, the switch corresponding to the central processor stage is continuously closed. Traffic control begins each scan at the left end of the band. It proceeds to the right, ignoring all stages which show no demand signal, until a demand stage is reached. This stage is allowed access to the main memory for one memory cycle only. Traffic control then returns to the left end of the band to begin the next scan. Since the control search is anticipatory, no system time is consumed in bypassing stages in which no demand exists.

 Execution of a peripheral instruction automatically loads the data address into the proper read or write counter and initiates activity on the addressed peripheral device. The central processor is then free to continue instruction execution. Whenever the device is ready to transmit a word the stage switch is closed causing traffic control to allocate a memory cycle. The data word is transmitted using the address in the read/write counter and the counter is incremented by one, thus requiring no program reference beyond the original peripheral instruction.

 If no peripheral device requires a memory cycle, it is given to the multiprogram control. The multiprogram control, in turn, allocates the memory cycle to the appropriate control group.

 If only one control group (one program) is active, all cycles allowed the central processor are used in executing instructions from the active program. Since traffic control allows the central processor all available cycles except those needed to honor the intermittent demands of peripheral devices, this case represents that of the conventional single-program computing machine with the ability to implement input/output operations simultaneously with computing.

 When more than one control group is active, central processor cycles must be shared among the sev-
eral programs. For example, if three programs are active, then one instruction is performed in turn from each. With the initiation of an instruction, all succeeding cycles must be devoted to the execution of this instruction until it is completed.

The true power of multiprogram control appears when an active program attempts to execute an instruction which cannot be implemented because of the unavailability of a system component. Sensing that the instruction cannot be executed immediately, multiprogram control places the control group in a "stall" condition. This condition indicates to multiprogram control that (a) this program, although still active, shall not be allowed any central processor cycles as long as the stall indication remains, and (b) when the channel and/or the device involved completes its present task, the stall condition shall be automatically removed and the program restored to its full active status.

Thus, when an instruction cannot proceed because of input/output conflicts with either the same or another program, the central processor cycles which it would have used are made available to the other active programs, enabling processing to proceed faster. The result of the operation of multiprogram control is that there is never an idle memory cycle as long as there is any active program in which an instruction can be executed.

In this respect the 800 automatically subdivides itself into as many computers as there are active programs.

**THE HONEYWELL SERIES 200**

The Series 200 has been designed primarily for business applications and for the mixed environment of business data, communication and scientific processing. In this environment the key performance dimension is throughput. High throughput, in turn, requires that the proper consideration or weights be given to three operations:

1. Inputting of data
2. Data manipulation
3. Outputting of data

The Honeywell Series 200 line of data processors is composed of five members; at the lower end of the spectrum is the H-120; this is successively followed by the H-200, the H-1200, the H-2200, and the H-4200, each more powerful in turn. Key attributes of each processor are shown in Table 3.

**Table 3. Series 200 Members — Key Attributes.**

<table>
<thead>
<tr>
<th>Processor Model</th>
<th>Main Memory Speed (cycles per character)</th>
<th>Memory Capacity (thousands of characters)</th>
<th>Number of Input/Output Simultaneous number of Simultaneous with Computing</th>
<th>Advanced Programming Instructions</th>
<th>Financial Edit Instruction</th>
<th>Multiply and Divide Instructions</th>
<th>Scientific Processing Instructions</th>
<th>Memory Protect Facility</th>
</tr>
</thead>
<tbody>
<tr>
<td>120</td>
<td>3 µsec</td>
<td>2-32</td>
<td>4 controls in processor; 4 I/O trunks available</td>
<td>2-3</td>
<td>Op</td>
<td>Op</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>200</td>
<td>2 µsec</td>
<td>4-65</td>
<td>8-16</td>
<td>3-4</td>
<td>Op</td>
<td>Op</td>
<td>S</td>
<td>NA</td>
</tr>
<tr>
<td>1200</td>
<td>1.5 µsec</td>
<td>8-131</td>
<td>16</td>
<td>4</td>
<td>S</td>
<td>S</td>
<td>S</td>
<td>S</td>
</tr>
<tr>
<td>2200</td>
<td>1 µsec</td>
<td>16-262</td>
<td>16-32</td>
<td>4-8</td>
<td>S</td>
<td>S</td>
<td>S</td>
<td>S</td>
</tr>
<tr>
<td>4200</td>
<td>188 nanosec</td>
<td>32-524</td>
<td>32-64</td>
<td>8-16</td>
<td>S</td>
<td>S</td>
<td>S</td>
<td>S</td>
</tr>
</tbody>
</table>

In the Series 200 the scratchpad memory, or control memory, is organized in a different manner than it is in the H-800. This is due, in part, to the fact that the Series 200 members are character-oriented rather than word-organized processors. Further, the Series 200 processors do not possess capability for parallel processing—i.e., the control memory is organized as a single control group (in H-800 terminology).

Physically, the control memory has a maximum of 64 locations, the number found in any one machine depending upon the model, options and features. The number of bits per position depends upon the size of the main core memory. Unlike the scratchpad memory in the H-800, the control memory in the various members of the Series 200 operates at a faster rate than the main core memory. This permits multiple scratchpad memory ac-
cesses during a main memory cycle. The speed ratio for each member of the series is shown in Table 4.

Table 4. Relative Speed Ratio of Control Memory to Main Memory for the Members of the Series 200.

<table>
<thead>
<tr>
<th>Processor</th>
<th>Main Memory Cycle Time</th>
<th>Control Memory Cycle Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>H-120</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>H-200</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>H-1200</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>H-2200</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>H-4200</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

As noted above, the Series 200 processors are character-oriented. This fact had a direct influence in the specification of the functions and number of registers per function which were designed into the scratchpad memory. These functions are listed in Table 5.

Table 5. Control Memory Function in the Series 200 Processor.

<table>
<thead>
<tr>
<th>Descriptive Functions</th>
<th>Number of Locations</th>
</tr>
</thead>
<tbody>
<tr>
<td>A &amp; B address registers</td>
<td>2</td>
</tr>
<tr>
<td>Sequence register</td>
<td>1</td>
</tr>
<tr>
<td>Change sequence register</td>
<td>1</td>
</tr>
<tr>
<td>R/W channel current location counter</td>
<td>3–16</td>
</tr>
<tr>
<td>R/W channel starting location counter</td>
<td>3–16</td>
</tr>
<tr>
<td>Work registers</td>
<td>3</td>
</tr>
<tr>
<td>External interrupt registers</td>
<td>1</td>
</tr>
<tr>
<td>Internal interrupt register</td>
<td>1</td>
</tr>
<tr>
<td>Floating point accumulation</td>
<td>12</td>
</tr>
</tbody>
</table>

The Series 200 processors are not only character-oriented machines but they operate with instruction formats which may contain zero, one, or two addresses—thus the requirement for two address registers. Further, the input/output interface has been designed to work in conjunction with the control memory. The read/write channels (RWC's) are not scanned as the traffic control was scanned in the H-800, but in a fixed cyclic manner. The first memory cycle is offered to RWC 1. If this RWC cannot use the cycle it is utilized by the central processor. The next memory cycle is offered to RWC 2, etc. The I/O access is accomplished via two registers in the control memory which are directly associated with each RWC.

The remaining counters and registers listed in Table 5 are self-explanatory with the possible exception of the floating point accumulators. These registers are provided in the larger members of the Series 200 where the Scientific Option is available. The availability of four floating point accumulators permits the temporary storage of intermediate results which, if stored in and recalled from main core memory on a character-by-character basis, would be cumbersome and time-consuming. With the results in scratchpad, they are available for manipulation on a register-to-register basis.

The use of the scratchpad memory has allowed the designer to economically implement by hardware many conventional software functions with significant improvements in machine performance. We believe that the proper balance has been struck between cost and capability at a reasonable and marketable price.

THE HONEYWELL 8200

The recently announced H-8200 represents the intelligent integration of the Series 800 and the Series 200 into one system. It has carried the concept of a control memory to its next logical level of utilization—the delegation to a structured hardware element of those systems-control functions which can best be performed by hardware for the greatest speed increase per dollar of cost.

The H-8200 main memory has a cycle time of 750 nanoseconds and a scratchpad which will provide a speed ratio of control memory to main memory of from 4 to 6:1.

The H-8200 system's processor (see Fig. 3) contains 10 programming groups. Nine of these groups control active running programs, while the tenth is a master control group.

![Figure 3. H-8200 system organization.](From the collection of the Computer History Museum (www.computerhistory.org))
The function of the master control group is to provide intercommunication among the active programming groups—and thus, in fact, to monitor the entire system. Of the nine remaining groups, eight handle data and instructions in the H-800 fixed-word mode while the ninth group operates in the H-200 variable-length character mode.

A further function of the master control group is to manage the input/output facility as well. In particular:

1. It maintains identification information regarding memory partitioning.
2. It diagnoses program, peripheral and memory usage violations.
3. It responds to privileged operating system instructions.

The 10 program groups are physically realized in 3 distinct scratchpad memories. This distribution permits overlapping and concurrent operation. Two other scratchpad memories have been employed, one in the memory multiplexor and the other in the input/output multiplexor.

The scratchpad memory in the input/output multiplexor provides the following functions:

1. Read/write channel capability for both the character and word processors.
2. Reservation facilities on a read/write channel basis.
3. Peripheral barricading.

By utilizing scratchpad memories to a greater extent than ever before, the H-8200 design has minimized the memory storage requirement for overhead functions and at the same time produced a faster reacting system. This provides greater throughput, which in turn provides greater economy to the user.

CONTROL MEMORY IMPLEMENTATION TRADEOFF ANALYSIS

The utilization of scratchpad memories at Honeywell has developed over an extended period of time. Their continued use has been reviewed and evaluated for each new product in which they have, or could have, been employed.

The decision has turned, and must always turn, upon one key point: Does the use of a scratchpad memory produce a quantitative increase in the overall performance of the system for the dollars involved? The correct decision involves the answers to many questions. For example, what set of functions will the scratchpad memory perform? Is the set of functions to be performed optimal? What effect will the inclusion of a scratchpad memory (with a given set of functions) have upon the software design? Does the cost of the scratchpad actually reduce the cost of the overall system, or does it increase it?

The approach followed by any group in answering these questions will depend upon backgrounds, personalities, availability of funds, etc.

Let us assume that the decision to employ a scratchpad memory for control purposes has been made and that the decision was based upon a preliminary systems analysis which delineated the functions which the unit would be required to perform. As a result of the specification, the size of the memory (and therefore its capacity in bits) was established.

The next step is to determine the speed of the unit, and the implementation technology.

Recall from Table 4 that the speed ratio between the control memory and the main memory for several members of the Series 200 ranged from 2 to 6. It is probable that this ratio in the future will tend toward the high end of the range, and possibly even extend beyond it. For discussion purposes, let us assume 4 as a minimum and 8 as a maximum. We now need a speed estimate for future main memory systems.

Several studies have been performed recently 9,10 which provide a reasonable engineering estimate of the speed of future main memory systems. One of these studies was performed by Honeywell's Electronic Data Processing Division under the sponsorship of the U.S. Army Electronics Command and produced a set of extrapolated computer characteristics. For one of these extrapolations, concerning the memory cycle time of future data processing systems, the data file has been expanded and updated, and a current extrapolation has been produced. It is shown in Fig. 4.

The data base for Fig. 4 represents 208 distinct machines introduced over a period of nearly 20 years. The machines contained in the data base represent, as far as memory technology is concerned, the full gamut of main memory implementation techniques from mercury delay lines to magnetic thin films.
The upper curve in Fig. 4 represents the least squares fit to the total data base. Once plotted, all points above that curve were eliminated from the data base and a second curve fitted; finally the points above the second curve were dropped and a third curve was generated. The three curves thus represent:

1. The average for the total data base.
2. The average of the leading or, in this case, the lower half of the data base.
3. The average of the leading quarter of the data base.

If a specific datum is selected and the three extrapolations of Fig. 4 are plotted for this data, as in Fig. 5, a trend indicating the leading edge of the technology for the particular time period should emerge.

This hoped-for trend in memory cycle time does not appear to be emerging. This implies, if the extrapolation technique is valid, that a saturation or slowing down of technological developments in this area is occurring.

This type of extrapolation has been made by the writer three times during the past six years. Figure 11 summarizes the three results for the extrapolated memory cycle time obtained for the year 1970. The number of machines considered in each extrapolation is listed in the second column of Table 6.

Returning to the detailed specification of the control memory, we can estimate a main memory cycle time of approximately 300 nanoseconds for 1970. Based upon this figure and the speed ratio
Figure 5. Extrapolated memory cycle time for machines introduced in 1970 (based upon an expanded MILDATA data base).

Table 6. Comparison of Three Extrapolations Concerning Memory Speed in 1970.

<table>
<thead>
<tr>
<th>Date</th>
<th>Data Base (number of machines)</th>
<th>Average Memory Cycle Time 1970 μsec</th>
</tr>
</thead>
<tbody>
<tr>
<td>1959</td>
<td>30</td>
<td>0.25–0.50</td>
</tr>
<tr>
<td>1963</td>
<td>154</td>
<td>0.30</td>
</tr>
<tr>
<td>1965</td>
<td>208</td>
<td>0.330</td>
</tr>
</tbody>
</table>

developed above (control memory cycle time should be 4 to 8 times as fast as the main memory cycle time), the memory cycle for a control memory in 1970 should range from approximately 40 to 75 nanoseconds.

The capacity and the speed of the unit has been established. The final question to be considered in this paper—the selection of the control memory implementation technology—can now be examined.
The control memory could be implemented via one of several different physical implementations—for example, semiconductive flip-flop elements or magnetic thin films.

The general cost (in dollars) to produce (on a production basis) a control memory of a given capacity and speed can be defined as:

\[ C = I + D \cdot B \]  

where

- \( C \) = total product cost in dollars,
- \( I \) = initial "investment" cost per machine for the particular technological approach (the sum of the costs for the drivers, sense amplifiers, address and data registers, etc.) in dollars,
- \( D \) = cost per bit of storage (in dollars), and
- \( B \) = number of bits.

For two distinctly different methods of implementation the crossover point occurs when the total costs of the control memory, in production for both, are equal.

Therefore:

\[ I_A + D_A B = I_B + D_B B \]  

Solving for \( B \), we obtain:

\[ B = \frac{I_A - I_B}{D_B - D_A} = \frac{\Delta I}{\Delta D} \]  

\( \Delta I \) and \( \Delta D \) represent the differential in "investment" and cost per bit respectively.

Figure 6 presents Eq. (3) in a convenient form. A single example will suffice to explain its use. Assume we are comparing a magnetic thin film memory with a set of flip-flop registers for the control memory in a machine. Here \( I_A \) will represent the peripheral cost to support a film memory for this purpose. In practical terms, \( I_A \) is the sum of the costs for the memory drivers, sense amplifiers, address and data registers, etc. Hypothetically \( I_B \) could be considered to be equal to zero. \( D_A \) and \( D_B \) would be the cost per bit for each approach. Summarizing:

\( I_A = $10^3 \)
\( I_B = 0 \)
\( D_A = $0.50 \)
\( D_B = $2.50 \)

Therefore:

\[ \Delta I = 10^3 \quad \text{and} \quad \Delta D = 2.00 \]  

Using Fig. 6, we see for this example that the break-even point is at 500 bits. For larger capacities a core memory is more economical; for less capacity the active flip-flop implementation is more economical.

**SUMMARY AND CONCLUSIONS**

Historically, Honeywell was the first to develop and utilize the concept of scratchpad memories for control purposes within its data processing systems. The H-800 and the Series 200 systems each employed different embodiments of the basic concept. Increasing utilization of the fundamental concepts both by other computer manufacturers and Honeywell (for example the expanded and extended control memory system of the H-8200) substantiates the growing recognition by systems designers of the usefulness of such devices.

Extrapolations of current trends with an eye toward future developments in scratchpad memories leads one to the definite conclusion that scratchpad memories will be a significant factor in the design of future systems. In substantiation, three facts can be noted: (1) the rapid and significant decrease in the cost of small high-speed memories; (2) the definite trend toward the design and development of more sophisticated and complex systems; and (3) the increasing desire of the systems designer to emancipate the programmer from the individual machine characteristics to the greatest extent possible.

**ACKNOWLEDGMENTS**

The author wishes to express his appreciation to the members of Honeywell Electronic Data Processing Division who made the submission and presentation of this paper possible. In particular, Mr. John Gilson, Miss Sonya Shapiro and Mr. Robert Zinn showed a great deal of patience in reviewing and assisting in clarifying my thoughts during the preparation of this paper.
Figure 6. Differential cost analysis, comparing two distinctly different implementation techniques.

REFERENCES