MEMOREX 7100

CPU LOGIC DESIGN

G. YEE

R. STALLMAN

7/19/72

CONTENTS

1.0 Introduction

This manual describes the logic design of the Memorex 7100 CPU. It is intended that this manual will provide all the basic information necessary to understand the CPU block diagram and the second-level logic diagrams of the CPU organization.

The implementation discussed is in TTL, and employs MSI elements and ROMS, where applicable, to reduce parts count. The CPU requires a clock frequency of 400 nsec. and a pulse width of approximately 50 nsec. Because of this speed, bipolar ROMS and RAMS were chosen. The design chosen eliminates the need to employ Schottky or high speed elements.

The block diagram has been partitioned for convenient board sizes based on the TTL implementation.

The design, as implemented, is of the architecture detailed in the architecture/microprogram manual.

2.0 Micro-Instruction Decode (Ref. 7100 CPU Block Diagram)

The micro-instructions are loaded into the u-instruction register and held there for one clock cycle. Normally, u-instructions are loaded into the UIR from the UROM, however the UBUS may be used to load the UIR during halt. The UROM output is or'ed with the UBUS when the UIR is specified as a destination by the micro-program. Micro-instruction modification is accomplished with this capability.

The contents of the UIR are preserved during a halt unless the operator loads a new u-instruction from the control panel.

2.1 Micro-op Decode

The micro-op field is 3 bits wide which allows a maximum of 8 micro-instruction formats that may be specified. The micro-op field is in bit positions 0-2 in all 8 micro-instruction formats. Refer to Table 1 of the CPU Architecture/Micro-programming Manual. Each micro-op is decoded and used to enable the field decoding that applies to the particular instruction in the UIR. An example would be one in which the source field is found in micro-ops 1 and 5 only. Thus the source decode is enabled only for these two micro-op formats. When the system is in halt the micro-op decode is disabled.

2.2 Register Source and Destination Decode (Ref Table 2, CPU Arch/Micro-programming Manual)

The micro-instruction format contains a 5 bit field which is the address of the source for information to be placed on the UBUS and a 5 bit field for the destination of information from the UBUS. There are a maximum of 32 possible sources and 32 destinations. The 16 least significant source and destination decodes, address the register file, and the most significant 16 are either discrete registers, not assigned, or special operation codes. A source

or destination of R1I/R2I implies that the source/destination address can be found in the R1I or R2I field of the PIR. The R1I or R2I field can specify only one of the first 8 registers in the register file.

2.3 Memory Port Decode: (Ref. Table 4 of the Architecture/microprogramming manual)

The memory port decode is a three bit field in the ALU, GSD and SCB micro-instructions. There are eight possible commands that can be decoded. All port commands are memory-related. The decoded port field has the capability of providing 8 commands. BMW, TSR, FMW, BPW, FMR, and FPW are decoded in the memory interface control logic. FMR, TSR, and HDM are decoded elsewhere. Since memory operations take more than one cycle the port field is retained in a latch by the CPU memory interface control logic until the memory operation is complete.

2.4 Branch Condition Decoding: (Ref. Table 5 of the Architecture/ Microprogramming manual).

The condition field is a 4-bit field which selects a maximum of 16 possible status conditions for a micro-instruction branch. There is also a reverse sense bit that is used in conjunction with the 4 bit condition field for micro-instruction branching on the not-true condition. The conditions are described in the architecture/ microprogramming manual. The decoding of the condition field is enabled by UOP 5 and UOP 6, the conditional branch micro-instruction formats.

2.5 ALU Control Decode

The ALU up field of 5 bits allows definition of 32 types of arithmetic/logic functions. The ALU control decode uses these 5 bits along with UOP 0 to generate all A & B register and ALU controls.

2.6 CROM Address

The GCD instruction, UOP 2, has an 8-bit field which is an address for a table of 16-bit constants stored in a read only memory (CROM). The output of the CROM is always sourced onto the bus for this type of micro-instruction. The CROM may contain 256 16-bit constants.

3.0 Micro-ROM Addressing (13 bits)

The micro-instructions stored in the UROM are normally executed in consecutive order. The UROM address is incremented by one and the incremented address is stored in the UPNT register.

There are several ways to branch to a different UROM address. For the unconditional branch the 12 least significant bits of the UIR and the most significant bit of the UPNT are used to generate the UROM address. The 8 LSBs of the ACB micro-instruction are combined with the five MSBs of the UPNT to produce another type of branch address. A branch address can also be loaded into the UPNT from the bus as a result of a SCB type micro-instruction.

The UPNT register is 13-bits wide and has the capability of addressing 8k x 16 bit words of UROM. The UPNT register can be loaded by the control panel when halted or during normal operation from the incrementer.

4.0 Micro-program Store (UROM)

The micro-program store is 8K maximum by 16 bits of ROM. The UROM consists of bipolar ROM elements with a worst case access time of 60 nsec. The UROM is accessed every 400 nsec.

5.0 Program Instruction Register (PIR)

The PIR is a 16 bit register that is used to hold macro-program instructions. It is loaded from the UBUS when selected as a destination. The 8 MSBs are used to address a decode ROM (DROM)

which is 256 x 13 bits and contains branch addresses for the UROM. The 8 LSBs of the macro-instruction field are designated as R1 and R2 fields and are used as register file addresses when specified as a source or destination.

6.0 Arithmetic Logic Unit

6.1 General

Included in the arithmetic-logic unit are the A & B feeder registers; the ALU logic; the control for the ALU logic; the control for setting/resetting bits related to Arithmetic-logic unit functions in the micro-condition register; and logic required to perform word, byte, and NYBL manipulation.

All ALU ops are performed on 8 or 16-bit operands previously loaded into the A & B registers. Results of the ALU operations are gated to the destination specified in the destination field of the micro-instruction. Results of these operations also set/reset related bits in the micro-condition register. In some ALU instruction formats the destination field is unused.

Arithmetic operations performed by the ALU are either binary or decimal. These two general categories differ from each other substantially. Binary ops: require one micro-cycle for their execution; involve two 16-bit operands; and require binary numbers to be represented in 2's complement notation. Decimal ops: require two micro-cycles for their execution; involve 8 bit (two decimal digits) operands and outputs; and require digits to be represented in packed BCD (8421) format.

6.2 Control

When an ALU instruction is decoded, if it is of a format having a destination, the arithmetic-logic unit gates to the bus are enabled. All other micro-instructions disable these outputs to the bus.

The five least-significant bits of the ALU micro-instruction determine the operation to be performed (Ref. Table 3 of the Architecture/Microprogramming manual). It is the decode of these 5 bits that control all functions of the arithmetic-logic unit. (Ref. logic diagram - "ALU op Decode")

6.3 Detailed Functional Description
(Ref. Table 3 of the Architecture/Microprogramming manual)

6.3.1 A & B Registers: (Ref. logic diagram - "A & B Registers")
The A & B Registers are 2-16 bit registers serially linked. Each is composed of 2-8 bit MSI Registers serially linked. They perform; right shift, left shift, parallel load, or do nothing. Control for them is via two mode control lines. Broadside loading of the registers is allowed when either is selected as a destination during execution of a micro-instruction. ALU ops SRO and SLO (shift right or left one bit) are linked (32 bit) end-off shifts. The actual shifting of the data in the A & B registers will not be accomplished until the end of the micro-cycle. Since the shifted data must be present at the destination specified in the micro-instruction prior to the end of the cycle, the outputs of the registers must be scaled one position left or right with gates. The resultant output then appears at the destination shifted. Right shifts are scaled through discrete gates. Left shifts are scaled through the MSI ALU.

6.3.2 WORD, BYTE or NYBL Manipulation: (Ref. logic diagram - "A & B Registers")
ALU, ops IN1, INZ & BCB, insert nybles or bytes from the A register into corresponding fields in the word from the B register. AOB & BOB are ops which allow the A or B register contents to be gated to a destination without modification. These ops are performed with gates for IN1, INZ, BCB, and through the MSI ALU for AOB & BOB. INV gates the 1's complement of the A register onto the bus. This function is performed with the MSI ALU.

6.3.3 Bit Test: (Ref. logic diagram - "A & B Registers")
ALU ops BBT and DBT nondestructively test the state of any one of the 16 bits in the A register and stores the state of that bit in bit zero of the micro-condition register. This function is performed with a

1 out of 16 decoder. The address of the bit to be tested is encoded in the low-order 4 bits of the dest field of the micro-instruction for DBT and in the low-order 4 bits of the B register for BBT. This op has no data destination, therefore, when it is executed, destination decoding must be disabled.

6.3.4 Arithmetic Functions:

ADD, BAD, BADX, SUB, BSB, & BSBX are binary arithmetic ops and are performed on the two 16-bit operands with an MSI ALU. (4-74181, 1-74182). The MSI ALU is controlled with 5 mode control lines. Additional control circuitry is required to handle carries and overflows. Carries may be stored in the micro-condition register for use in multiple-precision additions or subtractions. Algorithms and logic for the control of carries and overflow are included on logic diagram - "UCR & UCR CONTROL". Two's complement notation is used in these operations.

DAD, DADX, DSB, & DSBX are decimal ops performed on a right-justified byte in the A and B register. The data is in packed decimal form. The decimal digits are represented by 4-bit BCD (8421) code, therefore each pass through the ALU adds two decimal digits to two decimal digits and a possible carry from the UCR. A carry from the high-order digit is stored in the UCR.

Subtraction of decimal representations are performed using 10's complement additions. This method of subtraction by addition is somewhat analogous to using 2's complement addition when performing binary subtraction. When using 2's complement notation, binary numbers are left in 2's complement form in memory. When decimal data is processed, however, it is stored in true form. When subtracting one number from another number, by adding the 10's complement of one to the other, the addition may or may not have produced a carry. The carry, if it occurs, is dropped and the difference is in true form. If no carry occurs, the difference is in 10's complement form and must be corrected to place it in true form. No carry also indicates that a larger number was subtracted from a smaller number, and thus a negative difference has resulted. To put a BCD number in 10's complement form, the 9's complement is first derived by subtracting each digit from 9 and adding 1 to the LSB.

In a similar fashion, taking the 10's complement of a number in 10's complement form, places it back in true form. When reference to BCD is made, 8421 code is implied.

DAD, DSB

DAD and DSB are ops intended for use in 8 bit BCD addition (DAD) or subtraction (DSB). Specifically, these ops are to be used for the first pass, of a series, through the ALU. That is, when the two least significant decimal digits of a number are added. Carries or borrows from the condition register are ignored during these ops, however, if a carry or borrow is produced during the execution of the op it is stored in the UCR.

DADX and DSBX are ops used in 8 bit BCD addition (DADX) or subtraction (DSBX). These ops are used after the first pass through the ALU, and include a carry or borrow from the UCR that the first pass may have produced. If a carry or borrow is produced during the execution of these ops, it is stored in the UCR.

The Decimal ops are passed through a separate ALU (in parallel with the MSI ALU) composed of 7483 ADDERS and the necessary control circuitry to place it in the Add or Subtract mode. The basic circuit consists of a BCD adder which corrects for illegal BCD counts (over decimal 9). A 9's complementer at the input to the BCD adder is enabled during subtraction. With a forced carry into the low-order decade, a 10's complement addition is performed to effect a subtraction.

6.3.5 Remaining Logic Functions:

(Ref. Table 3 of the Architecture/Microprogramming manual).
XOR, IOR, and AND are performed in the MSI ALU. The only control circuitry for these functions is that involved in decoding the 5-bit ALU op field.

6.3.6 Compares:

CMP, the compare op simultaneously does both an arithmetic (signed) and a logical (unsigned) comparison of the A & B register contents.

The results of the comparison are placed in the micro-condition register (Ref. Fig. 2 of the Architecture/Microprogramming manual). Algorithms for the control of the micro-condition register are included on logic diagram - "UCR & UCR Control".

7.0 Condition Register (UCR)

7.1 General: (Refer to Fig. 2 of the Architecture/Microprogramming manual)

The condition register is a 10 bit register used for: storing the carries and overflows resulting during arithmetic ops, holding the results of compare and bit test instructions, and for holding the relative arithmetic (signed) magnitude of the results of all non-branch instructions (BUS = 0, BUS < 0). These 10 bits form a part of the condition set used by the branch instructions.

7.2 Controls for Clocking the Condition Register: (Ref. logic diagrams - "UCR & UCR Control", "ALU & Control", "UCR & Control")

Bits 0-7 of the UCR are clocked only during ALU instructions with ops requiring bits to be set/reset, or when a micro-instruction specifies the UCR as a destination. The remainder of the instructions and instruction formats do not affect this part of the condition register.

Bits 9 & 10 of the condition register are clocked with all non-branch micro-instructions. These 2 bits are not included as destinations specified in the DEST field of micro-instruction formats.

7.3 Set/Reset Controls for the Condition Register: (Ref. Table 2 of the Architecture/Microprogramming manual)

The algorithms and logic required to set bits 0-9 of the condition register are detailed in logic diagram - "UCR & UCR Control".

8.0 Register file and I/O Control

General (Ref. Logic diagram - "Reg. File & I/O Interface")

The register file contains 80 16-bit RAM registers as shown in Fig. 1 of the architecture/microprogram manual. Addressing of the register file's 16 rows by 16 columns (256 locations) may select one of the 80 registers as a source or destination of data to/from the bus. Certain of the remaining 256 addresses are assigned to registers or command codes external to the CPU (I/O) while other do not exist. This organization of the register file enables the CPU to communicate with I/O devices in the same manner in which it uses its own registers.

Addressing the register file

There are 256 possible address locations in the partially-populated register file, arranged in a 16 row by 16 column configuration. The MSB of the source or destination field in the micro-instruction determines whether the address is of a register in the register file or is of another register in the CPU. If a register file address is being decoded, the register to be used is selected by a 4-bit row and 4-bit column address.

The 4-bit row address comes from the source or destination field in the micro-instruction or from the R1, R2 fields in the PIR. The selection of the proper address source is done with a 4-input mux. This mux is controlled by a source/dest signal and a decode/ R1I, R2I signal. The source/dest signal determines whether the source field or the Dest field from the UIR will be the row address. A decode of 15 or 16 in the UIR source or dest. field indicates that the address is not in the UIR, but will be in the 3-bit R1 or R2 field of the PIR. When the 3-bit field of the PIR is used to address the register, the MSB of the resultant 4-bit address is "0".

The 4-bit column address may originate from one of two sources, the "X" or "P" register. Normal column addressing is done VIA the "P" register. "P" is loaded, by instruction from the CPU, with the output of the priority encoder. The priority encoder encodes data and service requests from 16 I/O Lines. A comparator

compares the existing "P" register contents, with the output of the priority encoder. When a data or service request of higher priority than the one in the "P" register exists, an "interrupt" is routed to the sense demux. As the microprogram checks for "interrupts", "P" may be loaded with a new address if it exhibits higher priority. The other source of column address is from the "x" register. The "x" register may be loaded from the bus when used as a destination in a micro-instruction. The x register is 5 bits wide. Only the four LSB's may be used as a column address. The MSB of the x register is used to select, via a mux, either the x or P register as the column address.

The 4-bit Row and 4-bit column addresses are extended to various I/O devices as previously described.

9.0 Memory Interface

The CPU may access memory via a micro-instruction port operation. The main memory takes three CPU micro-cycles for one memory cycle. The CPU shares the memory with other devices and each device has a fixed priority. If the CPU is not the highest priority device requesting memory, the port operation must remain latched up until the CPU memory request is satisfied. The memory interface control latches up the port request, decodes the port request, and controls the reading or writing from the memory.

The micro-program has the ability to halt the machine until a CPU memory operation is completed by executing a hold on memory (HDM) port command.

The 16-bit memory address register (MAR) is gated onto the bus by "CPU Memory Enable" at the beginning of the memory cycle. The 16-bit Memory Data Register (MDR) is either gated to, or from the memory on a bi-directional bus depending on whether or not, the operation is a write or read. A CPU memory operation may be initiated from the control panel while the CPU is halted. The MAR & MDR may be loaded or displayed by the control panel. The MDR may be loaded one byte at a time from the 8 LSB's of the UBUS. The LSB of the MAR is used to determine which MDR byte is loaded when the DEST. field is SMDR. The 8 LSB's of the MDR may be sourced either onto the left or right byte of the UBUS depending upon the LSB of the MAR when the source field is SMDR. A full 16 bit word may be loaded into the MDR or placed onto the bus from the MDR when DEST./SOURCE MDR is used.

The memory protect function works as follows: The protect register is loaded from the UBUS and contains two bytes. Each byte represents a page limit of the 8 MSBs of memory addressing. The high order byte of the protect register represents the highest protected page and the low order byte represents the lowest protected page. An error condition is set when the protect control is activated by the proper port operation and may be selected as a branch condition.