# Theory of Operation

## Core Memory Replacement for IBM 1130


Summary

This is designed as an easy swap for an 8K core memory in an IBM 1130 system. Gate B compartment C1 contains the original core memory stack plus SLT cards for the memory function.

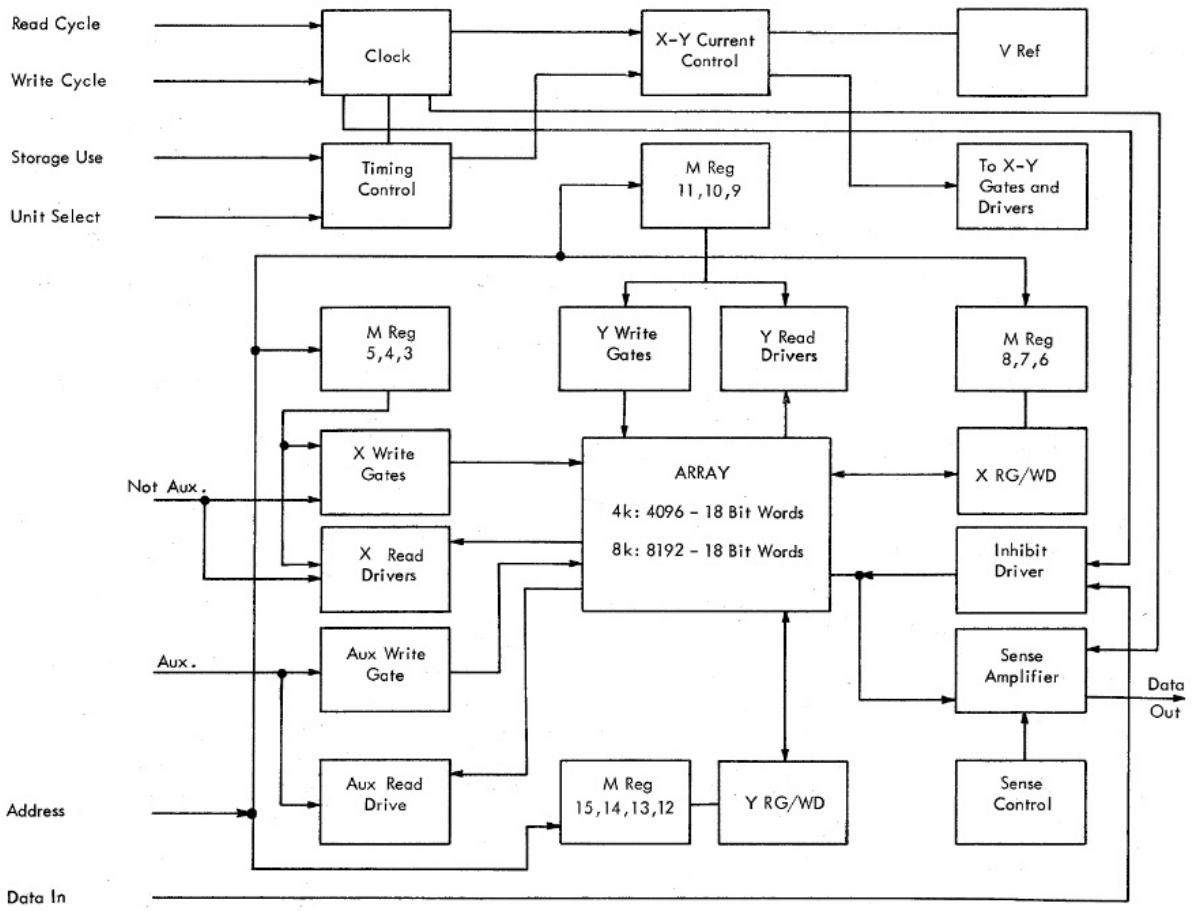
This PCB is mounted inside the card side of the compartment where the three connector cables T1, T3 and T4 plug into the core memory compartment.

Those three cables are instead plugged into this PCB and a connection is made to the 1130 +12V power supply. Compartment C1 is disconnected from the +6, +3, -3 and +12V supplies of the 1130 as all its functions are completely replaced by this board.

The board implements non-volatile storage with 8K words of 18 bit storage, providing 16 data bits plus parity bits P1 and P2. This board retains the content of memory for up to 20 years without power being supplied.

# Theory of Operation
## Core Memory Replacement for IBM 1130

Interface between the 1130 system and the core memory




The IBM 1130 provides 4 control signals to the core memory compartment - *Storage Select*, *Storage Use*, *Storage Read* and *Storage Write*.

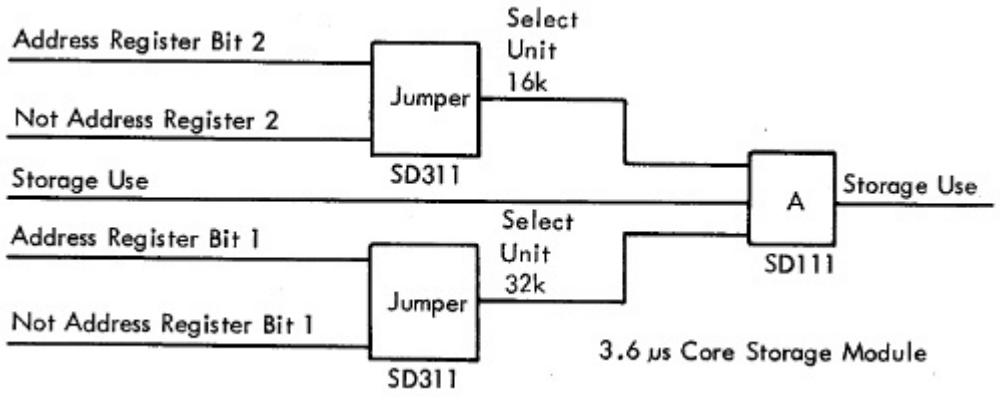
Cable T3 delivers a 13-bit memory address which represents the word within the 8K core memory that is being addressed. The low order bit of an address or a data word is numbered 15, with bit 0 referring to the highest order bit (usually a sign bit) of the data word. The Storage Address Register (*SAR*) bits 3 to 15 are used to select one of 8192 unique word addresses.

# Theory of Operation

## Core Memory Replacement for IBM 1130

For machines configured with larger memories, up to 32K words, the 1130 has multiple core memory compartments. These are in gate D compartment A1, gate D compartment B1, gate E compartment A1 and gate E compartment B1 depending on how many increments of 8K are installed.

The 100 ns pulse *Storage Select* indicates that the *SAR* address is within the 8K range of memory that this compartment manages. For a 32K machine, *Storage Select* is only emitted for the compartment matching the full *SAR* address. On the 8K models such as this board is designed to replace, *Storage Select* always pulses for any memory access.



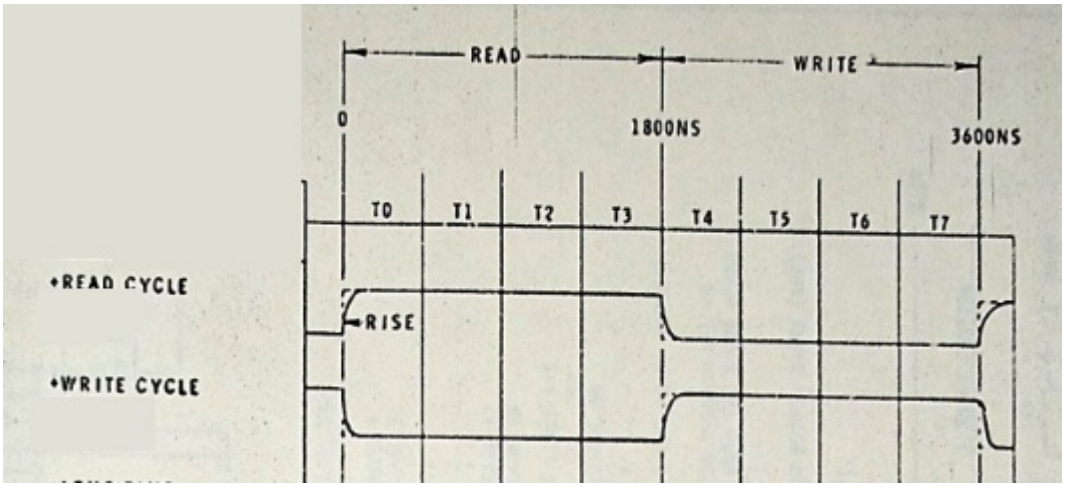

The jumpers referenced above determine what the two high order address bits are for a particular core module (compartment). With both bits at 0, this addresses words 0 to 8,191. Other combinations select other compartments:

01 (8,192 to 16,383),
10 (16,384 to 24,575)
11 (24,576 to 32,767).

# Theory of Operation

## Core Memory Replacement for IBM 1130

Any time the 1130 CPU clock is running, the signals *Storage Read* and *Storage Write* are alternating. Each is active for 1.8 or 1.1 microseconds (depending on whether this is a 3.6 uS or a 2.2 uS model).



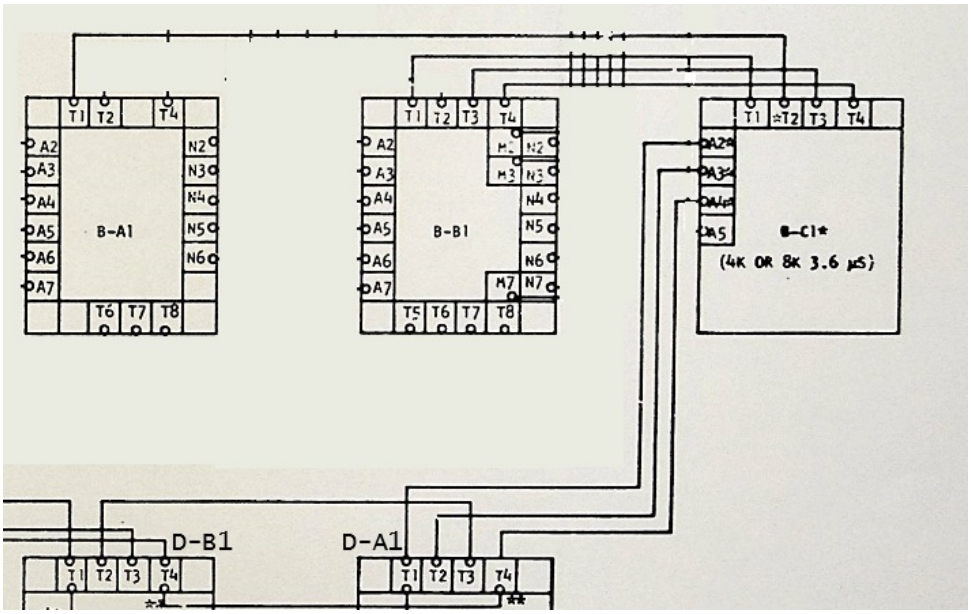

The final control signal *Storage Use* is active when the CPU wants to read and write to memory but is turned off when no memory access is desired. Example - To branch to an interrupt handler, the CPU forces in a Branch and Store IAR (BSI) instruction while turning off *Storage Use* so that nothing comes from memory during that 3.6 or 2.2 uS processor cycle.

There is always a *SAR* address incoming on cable T3. When *Storage Select* arrives and *Storage Use* are asserted, the core memory compartment will perform a read of a word during the time that *Storage Read* is active, then perform a write of a word when *Storage Write* is active.

# Theory of Operation

## Core Memory Replacement for IBM 1130

The data value to be written into memory is delivered on cables T1 and T4. This is called the Storage Buffer Register (*SBR* or B register). Sixteen data bits B0 to B15 are transmitted along with the two parity bits P1 and P2.




When a word is read, while *Storage Read* is active, sense lines S0 through S17 will generate a pulse for any bit of the word that contained the binary value 1. The pulse is generated somewhere during the time *Storage Read* is active. It is not synchronous to the CPU clock or circuitry. Sense lines are output on cables T1 and T4.

In fact the core memory compartment is not synchronous to the 1130 other than by seeing the alternating *Storage Read* and *Storage Write* signals. The core memory captures the *SBR* value sometime during the cycle when *Storage Write* is active. The CPU does not change the *B* register or the *SAR* values except at the start of the *Storage Read* or *Storage Write* signals.

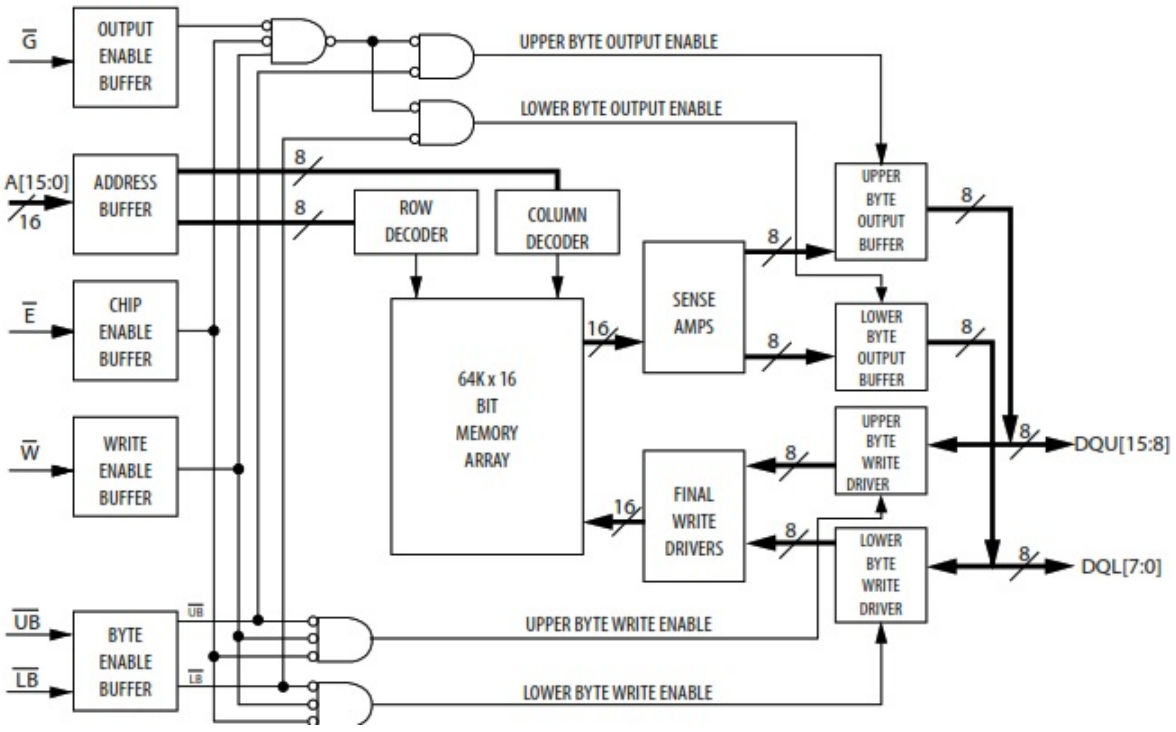
# Theory of Operation

## Core Memory Replacement for IBM 1130

MRAM chip at the heart of the replacement board

One Magnetorestrictive Random Access Memory chip is installed on the board, with a capacity of 64K words of 16 bits. This technology is non-volatile, just as is core memory, meaning the contents from its prior use are still there when the machine is powered up.

The Everspin MR0A16A chip guarantees that the contents remain preserved for at least 20 years after power was last removed. The chip has no wear limits - it can be read and written as much as is desired with no impact on the chip life.



Finally, it is designed to be accessed very similar to Static Random Access Memory (SRAM) chips. The chip completes a read or a write in 35 nanoseconds, far faster than is required by an 1130 system.

The SAR address bits are directly connected to the MRAM chip. Its behavior is determined by three chip control signals - ~E, ~W and ~G.

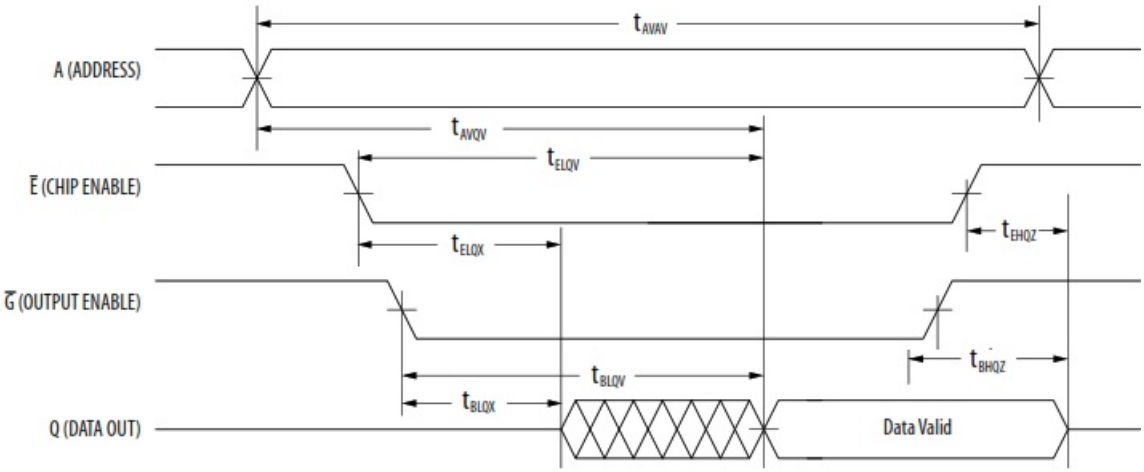
~E enables the chip to perform some function when it is asserted low.

~W directs the chip to write the data on its 16 bidirectional data pins into the location determined by the address bit pins.

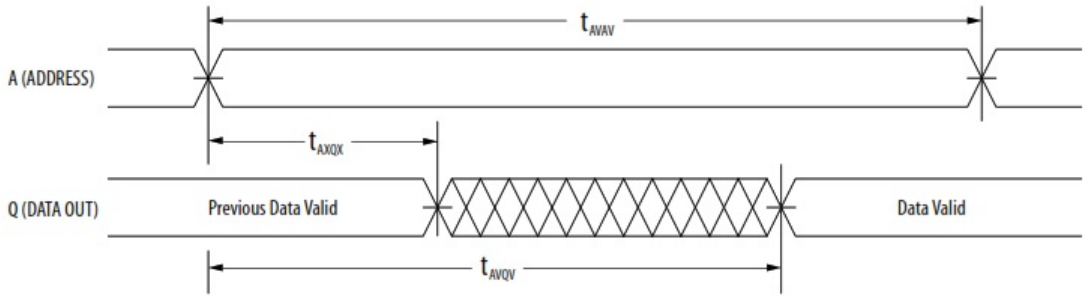
~G controls the bidirectional data pins. When asserted low, the contents of the memory word at the location determined by the address bit pins is output on the 16 pins. When ~G is deasserted (logic high level), the output pins are put into high impedance mode.

Thus to write the ~G must be deasserted and the data word driven onto the 16 data pins. To read, ~G is asserted and the memory contents drive the 16 data pins.

The chip is not sensitive to how long the ~G, ~E and ~W pins are asserted, other than the minimum 35 nanosecond time necessary for the chip to complete the read or the write.



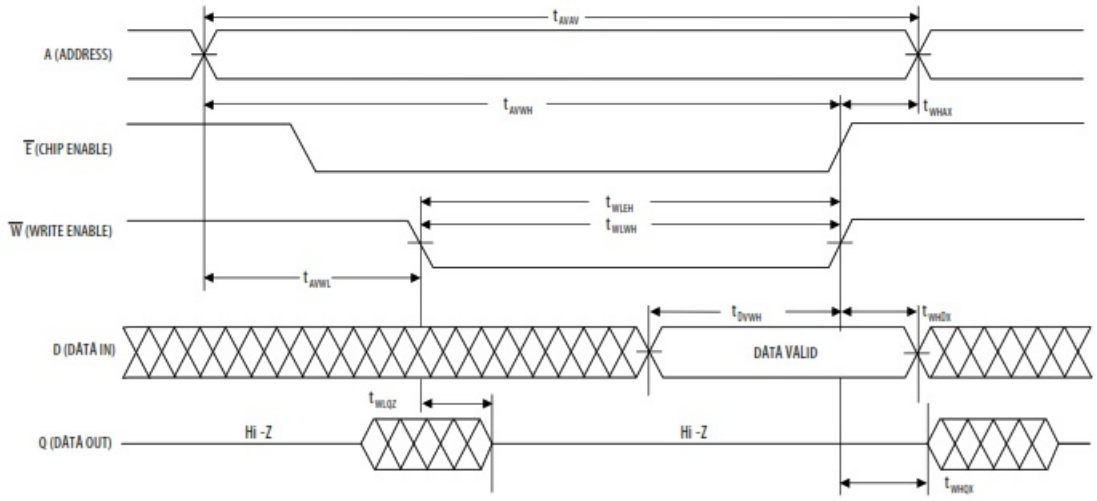

Above is the mode I will use for reads - imagine from the point that data is valid we sit and wait about 800 nanoseconds before sampling the data values. They won't change no matter how long it sits in this state, as shown by the read alternative sequence below:




Note: Device is continuously selected ($\overline{E} \leq V_{IL}$, $\overline{G} \leq V_{IL}$).

# Theory of Operation
## Core Memory Replacement for IBM 1130

For writes I will use a version of the pattern below:




In this case, ~E is asserted for about 800 nanoseconds before I assert ~W during the 80 ns pulse. That is longer than the 35 ns max time needed for the write to complete.

# Theory of Operation
## Core Memory Replacement for IBM 1130

Dealing with parity bits

The core memory in the IBM 1130 is relatively error prone, thus parity bits are included in the core stack and used by the CPU. The modern MRAM chip is so reliable that we don't need parity checking.

Thus when we write a word into memory we only capture the 16 bits of the SBR and ignore the parity bits produced by the CPU. When we return a word from memory during a read, we must present the two parity bits to the CPU because it will test them to check for errors.

Thus, the two parity bits are generated by the board from the 16 data bits retrieved from the MRAM chip. The CPU is presented 16 data and 2 parity bits on sense bits S0 to S17.

The parity scheme in the IBM 1130 is to produce a parity bit for each half of the word. Parity bit P1 is calculated over the values in data bits 0 to 7, while parity bit P2 is generated from the values in data bits 8 to 15.

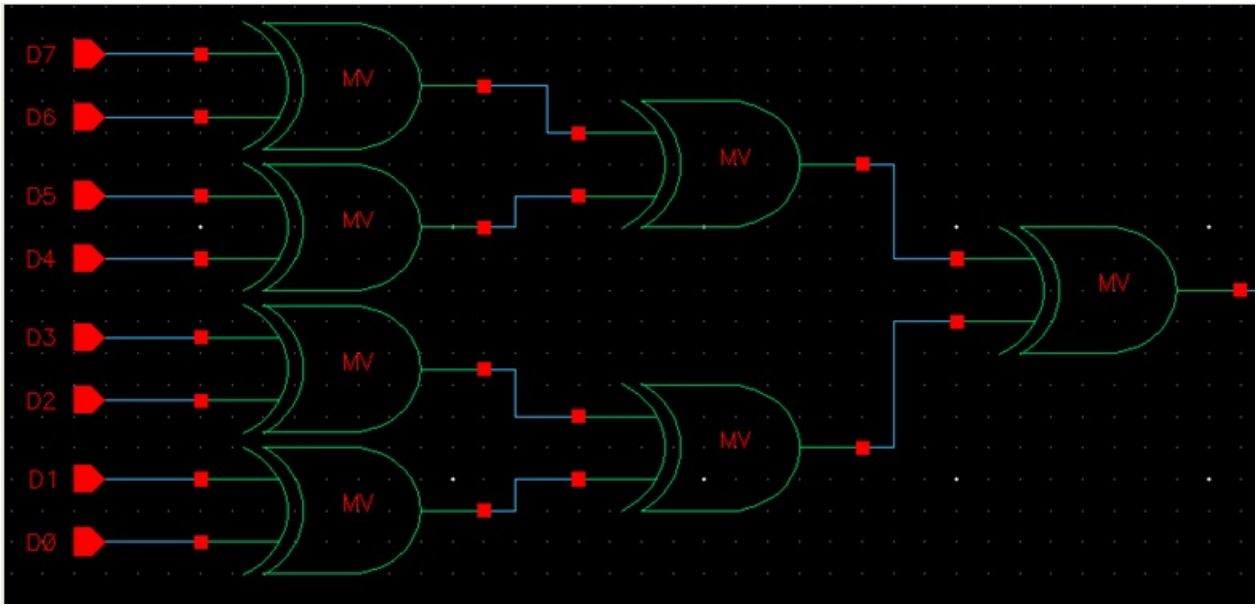
The 1130 uses odd parity. This means that the number of bits that have a value of 1 in the half of a word, plus the associated parity bit, must be an odd number. If the eight bits are 00000000 then the parity bit is turned on to produce an odd count. If the eight bits were 00001000 then the parity bit is set to 0.

# Theory of Operation

## Core Memory Replacement for IBM 1130

This is generated with a cascade of exclusive-OR (XOR) gates. The output of XOR is 0 when the two inputs agree, a 1 when they differ. It doesn't matter if the two bits are 00 or 11, the agree.  Similarly, whether 01 or 10, they disagree.

An XOR gate compares each pair of bits in the halfword - 0 and 1, 2 and 3, etc. The outputs of those four XOR gates are connected to two more XOR gates, comparing pairs of of the four XOR gate outputs. The two gate outputs of the second level of XORS are connected to a single XOR in the third level that produces the inverted value of the parity bit.




When the output of the XOR cascade is 1, it means we already have odd parity thus the parity bit must be zero. Another XOR is used with one input tied to high to create an NOT gate for the output.

# Theory of Operation

## Core Memory Replacement for IBM 1130

The 1130 CPU uses the same three level chain of XOR gates to do its parity generation and parity checking. In this design, we don't bother storing the generated parity bits along with the *SBR*, but we recreate the two parity bits during a read so that the 1130 CPU can validate correct parity.

# Theory of Operation
## Core Memory Replacement for IBM 1130

Connecting SBR signals to the bidirectional data pins

The *SBR* signals coming in over cables T1 and T3 are unidirectional - they are driven by the 1130 CPU at all times. We put buffer chips between the *SBR* and the MRAM data pins - when the buffer is enabled it drives the MRAM data pins with the value coming from the *SBR*.

When disabled, the buffer chip goes to high impedance. This allows the MRAM chip to drive values on the pins during a read. The *SBR* is isolated by the buffer chip whose inputs are unidirectional just like *SBR*.

Because the *SBR* bits from the CPU are inverted, the buffer we use will invert the signals before applying them to the bidirectional data pins of the MRAM chip. Thus the word in memory is not inverted.

The write timer produces an approximately 80 nanosecond pulse that becomes the MRAM chip ~W control signal to perform the write.

The buffer enable is simply the inverted Storage Write signal - it goes low when in a storage write phase of the memory cycle, passing the SBR bits on to the MRAM data pins.

The ~G control signal that drives output from the MRAM chip onto its data pins is connected to the Storage Write signal - when not in a storage write phase we have the memory contents driven on the data pins.

# Theory of Operation
## Core Memory Replacement for IBM 1130

Producing pulses on the sense lines only when the bit was 1

The MRAM chip provides a steady word value on its bidirectional data pins once the read has completed, but the 1130 expects a short pulse for bits that had the value 1.

We make use of a monostable multivibrator as a read timer to produce an approximately 80 nanosecond pulse during Storage Read. The data value from the MRAM data pins is passed though NAND gates, so that only when a data pin is 1 and the pulse is active will be output a logic low to the sense line.

This matches the inverted logic that the 1130 CPU expects - a low going brief pulse will set the SBR to a value of 1. If the memory bit value was 0, then no pulse is emitted from core memory and the SBR bit stays at 0.

The two recreated parity bits are treated similarly, so that pulses are emitted for any parity bit that has a 1 value.

# Theory of Operation
## Core Memory Replacement for IBM 1130

Reading and writing timing consistent with 1130 core memory

The timing of a core memory module is not clock synchronous and the sense pulses fore each bit can arrive at slightly different times. About halfway through the *Storage Read* signal duration, the sense pulses are typically produced.

Thus we use a chain of two Monostable Multivibrators as the write timer. The first is triggered on the rising edge of the *Storage Read* signal. It has a pulse duration of approximately 800 nanoseconds.

The falling edge of this first pulse is the trigger for the second pulse which is produced for about 100 nanoseconds. The effect of this chain is that our 100 ns pulse is started about 800 ns into the 1800 ns duration of the *Storage Read* signal.

It is this delayed 100 ns pulse that is combined with the MRAM data bit value in a NAND gate. The gate either produces a negative 100 ns pulse starting at T+800 ns or it produces nothing.

The MRAM chip has its ~G control pin asserted low when *Storage Write* is low. This starts at T+0 and the chip will have the contents of the word requested by the *SAR* location sitting on the bidirectional data pins in 35 ns.

That value will remain on those pins throughout the entire time *Storage Write* is inactive. This is not a problem for the MRAM chip. Our NAND gate just samples the data pin value when our 80 ns pulse arrives.

*+12V present* and *Storage Use* must both be asserted in order for the ~E control pin to be asserted low. This ensure the MRAM chip only does reads or writes when the CPU wants memory to be accessed and the CPU is not in the midst of powering down or up.

When the *Storage Write* signal is asserted, the *SBR* contains the data word to be written into memory at the location specified by the *SAR*. The values on *SBR* and *SAR* stay constant throughout the *Storage Write* 1.8 uS duration.

We implemented a write timer as a second chain of two Monostable Multivibrators on the board, this chain used to control write activity. The first part of the chain is triggered on the rising edge of *Storage Write* and has a pulse duration of 800 nanoseconds. The falling edge of that long duration pulse is the trigger for the second part of our chain which emits an 80 nanosecond pulse.

We have turned off the ~G control pin because we are in a *Storage Write*. Our buffer enable signal is an inverted version of *Storage Write*, thus causing the buffer chips to drive the *SBR* values onto the MRAM bidirectional data pins. We turn on the ~W signal during the 80 ns pulse to perform the actual write.

# Theory of Operation
## Core Memory Replacement for IBM 1130

Behavioral difference from original core memory

No difference is undetectable by any software running on the IBM 1130 or any peripheral device connected to the machine. No diagnostic program will detect this, nor will any timing be altered.

The original IBM core memory stack is a destructive read memory, which is why the 1130 always performs a storage cycle of a read followed by a write(back). When using the 1130 in any mode other than Single Step, entire read-then-write cycles are performed thus no difference can be detected.

However, Single Step mode advances through the T clock steps T0 to T7, where the read part of a storage cycle occurs during the first four clock steps and the write(back) happens in the last four steps. It is possible to step partway through a storage cycle, then switch the mode or hit the Reset button.

This leaves the currently addressed storage location as all zeros with bad parity on original core memory, but with MRAM the original value and good parity remain. If the operator stopped single stepping during the read part of the cycle then reset the machine so that it eventually read the location that was zeroed, a parity error would be detected.

This is the only situation in which the difference should be observable between this core memory substitute and the original IBM memory stack.

# Theory of Operation

## Core Memory Replacement for IBM 1130

Memory content safety during power shutoff

The board is connected to the IBM 1130's +12VDC supply line. The power on and power of sequencing of the 1130 ensures that the +12V line is connected only when all the primary SLT logic rails are at correct levels (+6, -3 and +3).

When the power switch is turned off, a relay disconnects +12V while the SLT power supplies continue to provide good logic voltages. Because it is switched through a relay, the shutoff is almost instantaneous.

One other case exists with the 1130 power supplies - failure of one of the voltage rails due to some kind of malfunction. As the SLT logic levels degrade a sensing card will shut down the relay for the +12V and then lock out the machine.

The IBM core memory modules are designed to rely on the 12V relay to protect against spurious writes on power issues, thus we should be equally safge.

A resistor divider on the 12V input is connected through an AND gate, so that use of the MRAM is revoked if 12V is no longer present from the CPU. Thus any energy in our power supply to the MRAM chip and other logic chips on the board will not produce any spurious write to memory.

# Theory of Operation
## Core Memory Replacement for IBM 1130

Power supply circuit

I bought a linear voltage regulator module that takes the +12V supply from the 1130 and generates our 3.3V rail for the board. It will be mounted with the PCB inside the memory compartment of the 1130 system.

# Theory of Operation
## Core Memory Replacement for IBM 1130

Variations that could support larger and faster memory sizes

If the 1130 system is a model that uses the newer faster IBM 2.2 microsecond core memory modules, we would have to adjust the resistor and capacitor values on the first stages of both Monostable Multivibrator chains to attain a shorter delay pulse of about 500 ns instead of the 800 ns used with the 3.6 uS core type.

This design is currently correct for a 8KB 3.6 uS model of the 1130, where the core memory is contained in gate B compartment C1. It will need changes for other configuratons.

The memory modules are moved to different locations with larger memory models. These sit in gates D and E, contained in the extension frame attached to the left of the console keyboard in what is informally called the blister. Each gate has room for two compartments; each compartment holds an 8K memory module.

Even the smallest 2.2 uS memory configurations are contained in gate D compartment A1 rather than gate B compartment C1. The signals that come into gate B compartment C1 through T1, T3 and T4, plus cable T2, would flow through some additional SLT logic in that compartment and then exit over other cables that run to gates D and E.

The cables coming out of gate B compartment C1 for these new configurations are connected at A2, A3, A4 and A5 of B-C1. They run to connectors at the top of the compartments on gates D and E. Cable A2 handles extra signals for 2.2 uS models.

# Theory of Operation

## Core Memory Replacement for IBM 1130

Larger memory would require the additional SLT logic implemented in gate B compartment C1. B-C1 no longer holds a core memorystack and related logic cards; it has a different purpose for the alternate configurations. Thus our board would have to connect in gate D compartment A1 even if we replace all the memory in all the compartments of the blister.

Our board does not have pins to connect to cable T2, but that is used to send the appropriate Storage Selection bits to let each compartment respond to its range of 8K addresses out of the total. We need those bits to complete the full addressing of up to 32K words. They should arrive over the cables into gate D but the details are still to be worked out.

Versions of the 1130 with only a single 4K or 8K 3.6 microsecond storage capacity have the memory in gate B compartment C1. All others use the Expanded Storage feature, which puts logic in B-C1 and moves the *Storage Selec*t signal over to cable T2 for 2.2 uS memories.

The MRAM chip has plenty of capacity but we would have to update its memory address pin connections and deal wtih the Storage Selection signals appropriately so that we could answer for all replacement core memory compartments.