In the Matter of
Certain Wireless Consumer Electronics Devices and Components Thereof

Investigation No. 337-TA-853

Complainants' Technology Tutorial

Before the Honorable E. James Gildea

March 5, 2013

Pioneering Technology

- Co-inventors Charles Moore and Russell Fish
- In 1988 and 1989, Moore and Fish designed a 32-bit microprocessor called “ShBoom”
- The IEEE identified ShBoom as one of “The 25 Microchips that Shook the World”
- Moore was inducted into the Computer Design Hall of Fame and received a Presidential Commendation, among other awards

Pioneering Technology

- On August 3, 1989, the inventors filed a comprehensive patent application, from which seven patents issued
- The ’336 patent has been subjected to six *ex parte* reexamination requests since 2006
- Over 600 separate prior art references have been cited against the ’336 patent in the reexaminations

Pioneering Technology

- Over 100 global electronics firms have licensed the Moore patents, including:
  - Sony
  - Intel
  - Hewlett Packard
  - Motorola
  - General Electric
  - Panasonic

The ’336 Patent

- The ’336 patent teaches the use of two independent clocks in a microprocessor system:
  - an on-chip first clock to time the CPU, and
  - a second, independent, clock to time the input/output (I/O) interface.

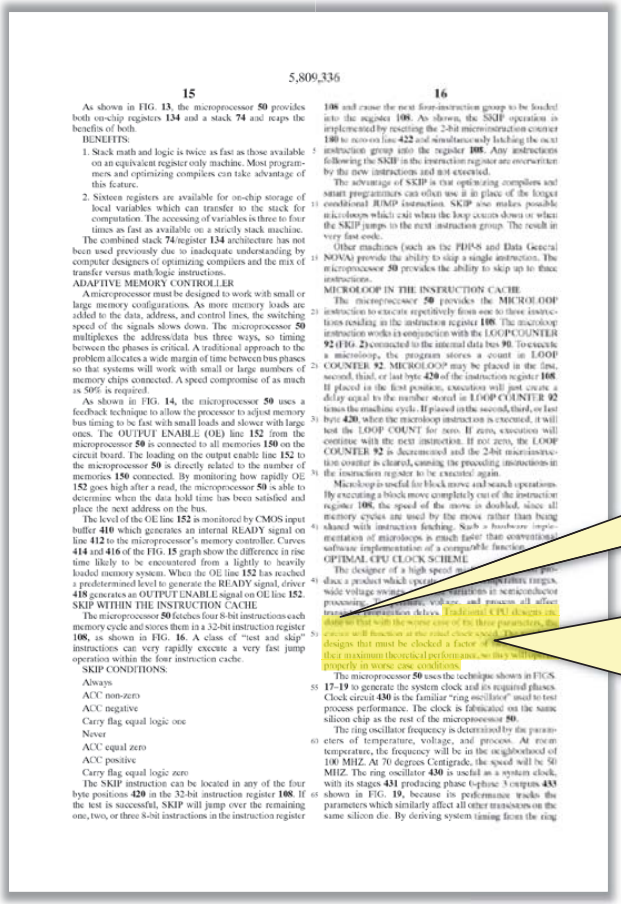
- Decoupling the system clock from the I/O clock allows the clocks to run independently (or “asynchronously”), and thus allows the CPU to run faster when needed (or more slowly to conserve power).

The ’336 Patent

- The on-chip oscillator that is used to clock the CPU is constructed on the same integrated circuit as the CPU
- The invention takes advantage of the laws of physics
- Because the clock is on the same silicon die as the CPU, the capability of both will vary similarly with respect to variations in temperature, voltage, and manufacturing

The ’336 Patent

Old Way of Clocking CPU



5,809,336

15

As shown in FIG. **13**, the microprocessor **50** provides both on-chip registers **134** and a stack **74** and reaps the benefits of both.

BENEFITS:

1. Stack math and logic is twice as fast as those available on an equivalent register only machine. Most programmers and optimizing compilers can take advantage of this feature.
2. Sixteen registers are available for on-chip storage of local variables which can transfer to the stack for computation. The accessing of variables is three to four times as fast as available on a strictly stack machine.

The combined stack **74**/register **134** architecture has not been used previously due to inadequate understanding by computer designers of optimizing compilers and the mix of transfer versus math/logic instructions.

ADAPTIVE MEMORY CONTROLLER

A microprocessor must be designed to work with small or large memory configurations. As more memory loads are added to the data, address, and control lines, the switching speed of the signals slows down. The microprocessor **50** multiplexes the address/data bus three ways, so timing between the phases is critical. A traditional approach to the problem allocates a wide margin of time between bus phases so that systems will work with small or large numbers of memory chips connected. A speed compromise of as much as 50% is required.

As shown in FIG. **14**, the microprocessor **50** uses a feedback technique to allow the processor to adjust memory bus timing to be fast with small loads and slower with large ones. The OUTPUT ENABLE (OE) line **152** from the microprocessor **50** is connected to all memories **150** on the circuit board. The loading on the output enable line **152** to the microprocessor **50** is directly related to the number of memories **150** connected. By monitoring how rapidly OE **152** goes high after a read, the microprocessor **50** is able to determine when the data hold time has been satisfied and place the next address on the bus.

The level of the OE line **152** is monitored by CMOS input buffer **410** which generates an internal READY signal on line **412** to the microprocessor's memory controller. Curves **414** and **416** of the FIG. **15** graph show the difference in rise time likely to be encountered from a lightly to heavily loaded memory system. When the OE line **152** has reached a predetermined level to generate the READY signal, driver **418** generates an OUTPUT ENABLE signal on OE line **152**.

SKIP WITHIN THE INSTRUCTION CACHE

The microprocessor **50** fetches four 8-bit instructions each memory cycle and stores them in a 32-bit instruction register **108**, as shown in FIG. **16**. A class of "test and skip" instructions can very rapidly execute a very fast jump operation within the four instruction cache.

SKIP CONDITIONS:

Always
ACC non-zero
ACC negative
Carry flag equal logic one
Never
ACC equal zero
ACC positive
Carry flag equal logic zero

The SKIP instruction can be located in any of the four byte positions **420** in the 32-bit instruction register **108**. If the test is successful, SKIP will jump over the remaining one, two, or three 8-bit instructions in the instruction register

16

108 and cause the next four-instruction group to be loaded into the register **108**. As shown, the SKIP operation is implemented by resetting the 2-bit microinstruction counter **100** to zero on line **422** and simultaneously latching the next instruction group into the register **108**. Any instructions following the SKIP in the instruction register are overwritten by the new instructions and not executed.

The advantage of SKIP is that optimizing compilers and smart programmers can often use it in place of the longer conditional JUMP instruction. SKIP also makes possible microloops which exit when the loop counts down or when the SKIP jumps to the next instruction group. The result is very fast code.

Other machines (such as the PDP-8 and Data General NOVA) provide the ability to skip a single instruction. The microprocessor **50** provides the ability to skip up to three instructions.

MICROLOOP IN THE INSTRUCTION CACHE

The microprocessor **50** provides the MICROLOOP instruction to execute repetitively from one to three instructions residing in the instruction register **108**. The microloop instruction works in conjunction with the LOOP/COUNTER **92** (FIG. **2**) connected to the internal data bus **90**. To execute a microloop, the program stores a count in LOOP COUNTER **92**. MICROLOOP may be placed in the first, second, third, or last byte **420** of the instruction register **108**. If placed in the first position, execution will just create a delay equal to the number stored in LOOP COUNTER **92** times the machine cycle. If placed in the second, third, or last byte **420**, when the microloop instruction is executed, it will test the LOOP COUNT for zero. If zero, execution will continue with the next instruction. If not zero, the LOOP COUNTER **92** is decremented and the 2-bit microinstruction counter is cleared, causing the preceding instructions in the instruction register to be executed again.

Microloop is useful for block move and search operations. By executing a block move completely out of the instruction register **108**, the speed of the move is doubled, since all memory cycles are used by the move rather than being shared with instruction fetching. Such a hardware implementation of microloops is much faster than conventional software implementation of a comparable function.

OPTIMAL CPU CLOCK SCHEME

The designer of a high speed [illegible] produce a product which operates [illegible] wide voltage swings, [illegible] variations in semiconductor processing. Temperature, voltage, and process all affect transistor switching delays. Traditional CPU designs do so that with the worse case of the three parameters, the circuit will function at the rated clock speed. The result are designs that must be clocked a factor of [illegible] their maximum theoretical performance, so they will operate properly in worse case conditions.

The microprocessor **50** uses the technique shown in FIGS. **17–19** to generate the system clock and its required phases. Clock circuit **430** is the familiar "ring oscillator" used to test process performance. The clock is fabricated on the same silicon chip as the rest of the microprocessor **50**.

The ring oscillator frequency is determined by the parameters of temperature, voltage, and process. At room temperature, the frequency will be in the neighborhood of 100 MHZ. At 70 degrees Centigrade, the speed will be 50 MHZ. The ring oscillator **430** is useful as a system clock, with its stages **431** producing phase 0-phase 3 outputs **433** shown in FIG. **19**, because its performance tracks the parameters which similarly affect all other transistors on the same silicon die. By deriving system timing from the ring

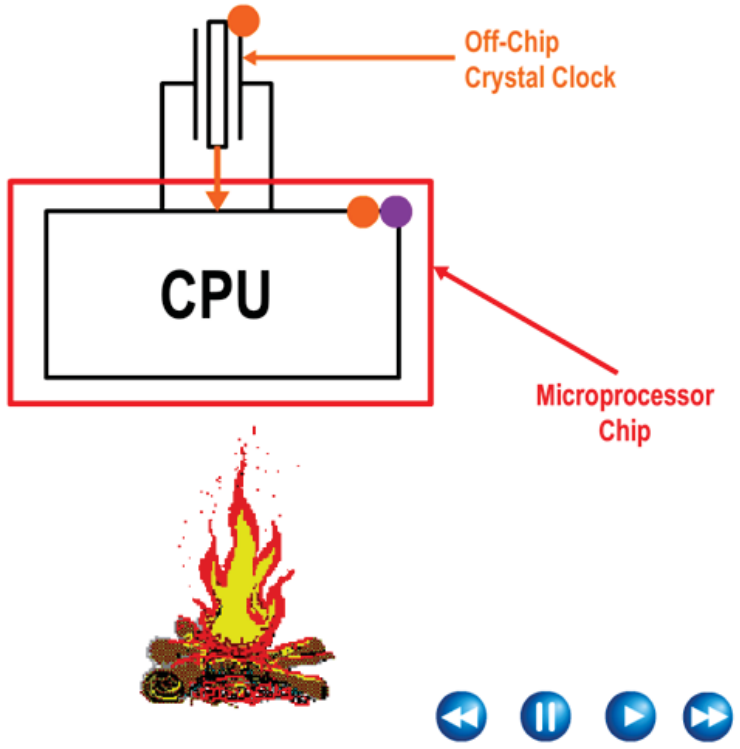
JXM-0001 (’336 patent) col. 16:48-53

“Traditional CPU designs are done so that with the worse case of the three parameters [temperature, voltage and variations in semiconductor processing], the circuit will function at the rated clock speed.”

“The result are designs that must be clocked a factor of two slower than their maximum theoretical performance, so they will operate properly in worse case conditions.”

The ’336 Patent

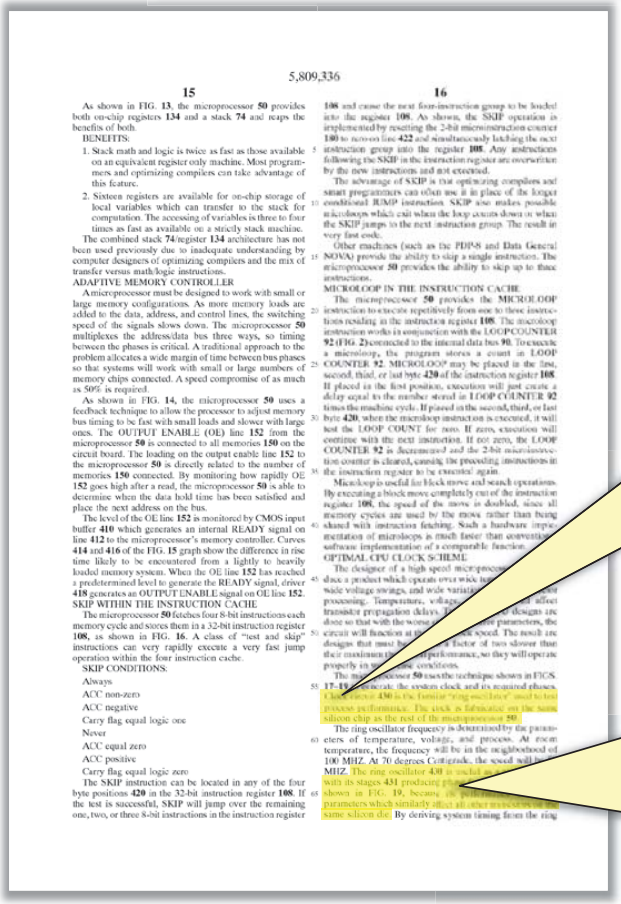
Old Way of Clocking CPU



The ’336 Patent

The Moore Solution: On-Chip System Clock

On-Chip Devices of the CPU and System Clock Vary Together



5,809,336

15

As shown in FIG. **13**, the microprocessor **50** provides both on-chip registers **134** and a stack **74** and reaps the benefits of both.

BENEFITS:

1. Stack math and logic is twice as fast as those available on an equivalent register only machine. Most programmers and optimizing compilers can take advantage of this feature.
2. Sixteen registers are available for on-chip storage of local variables which can transfer to the stack for computation. The accessing of variables is three to four times as fast as available on a strictly stack machine.

The combined stack **74**/register **134** architecture has not been used previously due to inadequate understanding by computer designers of optimizing compilers and the mix of transfer versus math/logic instructions.

ADAPTIVE MEMORY CONTROLLER

A microprocessor must be designed to work with small or large memory configurations. As more memory loads are added to the data, address, and control lines, the switching speed of the signals slows down. The microprocessor **50** multiplexes the address/data bus three ways, so timing between the phases is critical. A traditional approach to the problem allocates a wide margin of time between bus phases so that systems will work with small or large numbers of memory chips connected. A speed compromise of as much as 50% is required.

As shown in FIG. **14**, the microprocessor **50** uses a feedback technique to allow the processor to adjust memory bus timing to be fast with small loads and slower with large ones. The OUTPUT ENABLE (OE) line **152** from the microprocessor **50** is connected to all memories **150** on the circuit board. The loading on the output enable line **152** to the microprocessor **50** is directly related to the number of memories **150** connected. By monitoring how rapidly OE **152** goes high after a read, the microprocessor **50** is able to determine when the data hold time has been satisfied and place the next address on the bus.

The level of the OE line **152** is monitored by CMOS input buffer **410** which generates an internal READY signal on line **412** to the microprocessor's memory controller. Curves **414** and **416** of the FIG. **15** graph show the difference in rise time likely to be encountered from a lightly to heavily loaded memory system. When the OE line **152** has reached a predetermined level to generate the READY signal, driver **418** generates an OUTPUT ENABLE signal on OE line **152**.

SKIP WITHIN THE INSTRUCTION CACHE

The microprocessor **50** fetches four 8-bit instructions each memory cycle and stores them in a 32-bit instruction register **108**, as shown in FIG. **16**. A class of "test and skip" instructions can very rapidly execute a very fast jump operation within the four instruction cache.

SKIP CONDITIONS:

Always
ACC non-zero
ACC negative
Carry flag equal logic one
Never
ACC equal zero
ACC positive
Carry flag equal logic zero

The SKIP instruction can be located in any of the four byte positions **420** in the 32-bit instruction register **108**. If the test is successful, SKIP will jump over the remaining one, two, or three 8-bit instructions in the instruction register

16

108 and cause the next four-instruction group to be loaded into the register **108**. As shown, the SKIP operation is implemented by resetting the 2-bit microinstruction counter **180** to zero on line **422** and simultaneously latching the next instruction group into the register **108**. Any instructions following the SKIP in the instruction register are overwritten by the new instructions and not executed.

The advantage of SKIP is that optimizing compilers and smart programmers can often use it in place of the longer conditional JUMP instruction. SKIP also makes possible microloops which exit when the loop counts down or when the SKIP jumps to the next instruction group. The result in very fast code.

Other machines (such as the PDP-8 and Data General NOVA) provide the ability to skip a single instruction. The microprocessor **50** provides the ability to skip up to three instructions.

MICROLOOP IN THE INSTRUCTION CACHE

The microprocessor **50** provides the MICROLOOP instruction to execute repetitively from one to three instructions residing in the instruction register **108**. The microloop instruction works in conjunction with the LOOP/COUNTER **92** (FIG. **2**) connected to the internal data bus **90**. To execute a microloop, the program stores a count in LOOP COUNTER **92**. MICROLOOP may be placed in the first, second, third, or last byte **420** of the instruction register **108**. If placed in the first position, execution will just create a delay equal to the number stored in LOOP COUNTER **92** times the machine cycle. If placed in the second, third, or last byte **420**, when the microloop instruction is executed, it will test the LOOP COUNT for zero. If zero, execution will continue with the next instruction. If not zero, the LOOP COUNTER **92** is decremented and the 2-bit microinstruction counter is cleared, causing the preceding instructions in the instruction register to be executed again.

Microloop is useful for block move and search operations. By executing a block move completely out of the instruction register **108**, the speed of the move is doubled, since all memory cycles are used by the move rather than being shared with instruction fetching. Such a hardware implementation of microloops is much faster than conventional software implementation of a comparable function.

OPTIMAL CPU CLOCK SCHEME

The designer of a high speed microprocessor must balance the choice of a product which operates over wide temperature, wide voltage swings, and wide variations in [illegible] processing. Temperature, voltage, and process affect transistor propagation delays. Designers are often forced to design circuits that must be done so that with the worst combination of parameters, the circuit will function at the desired clock speed. The result are designs that must be a factor of two slower than their maximum [illegible] performance, so they will operate properly under [illegible] conditions.

The microprocessor **50** uses the technique shown in FIGS. **17–19** to generate the system clock and its required phases. Clock circuit **430** is the familiar "ring oscillator" used to test process performance. The clock is fabricated on the same silicon chip as the rest of the microprocessor **50**.

The ring oscillator frequency is determined by the parameters of temperature, voltage, and process. At room temperature, the frequency will be in the neighborhood of 100 MHZ. At 70 degrees Centigrade, the speed will [illegible] MHZ. The ring oscillator **430** is [illegible] with its stages **431** producing [illegible] shown in FIG. **19**, because all other parameters which similarly affect all other transistors on the same silicon die. By deriving system timing from the ring

The patent discloses a microprocessor embodiment where “[t]he clock is fabricated on the same silicon chip as the rest of the microprocessor 50.”

The “[c]lock circuit 430 is the familiar ‘ring oscillator’ ….” “The ring oscillator 430 is useful as a system clock … because its performance tracks the parameters which similarly affect all other transistors on the same silicon die.”

JXM-0001 (’336 patent) col. 16:57-58; 16:63-67

The ’336 Patent

- A “ring oscillator” – An oscillator having a multiple, odd number of inversions arranged in a loop.
  - Specifically, a Ring Oscillator must have:
    - At least 3 inversion stages that invert the inputted value.
- The multiple odd number of inversions ensure a change of state (Hi to Low, or Low to Hi) at each inversion stage.
  - Even numbers become stuck at logic high or low state.
- While other types of oscillators exist they should not be confused with the “continual inversion” phenomenon that causes the oscillation in a ring oscillator.

The ’336 Patent: Ring Oscillator

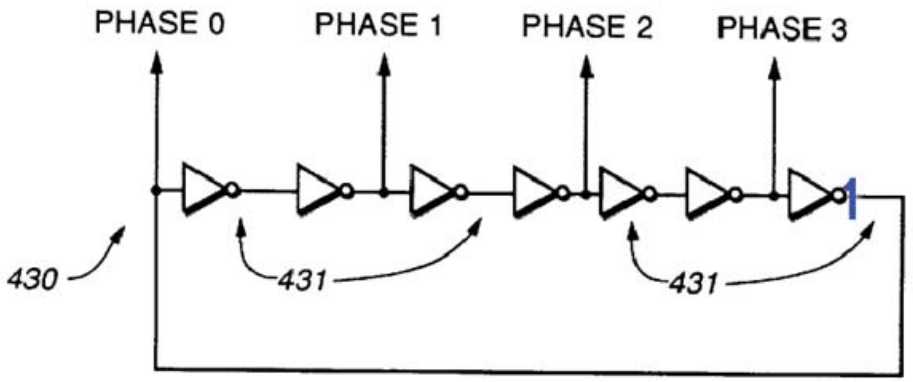


FIG._18

CDXM-0002.011

The ’336 Patent

- On-Chip oscillators vary due to one or more of “PVT”:
  - The semiconductor manufacturing or fabrication Process
  - The Voltage supplied
  - The Temperature of the circuit
- Since the CPU and the on-chip oscillator are on the same integrated circuit, they will be similarly affected by Process, Voltage, Temperature variations.

The ’336 Patent

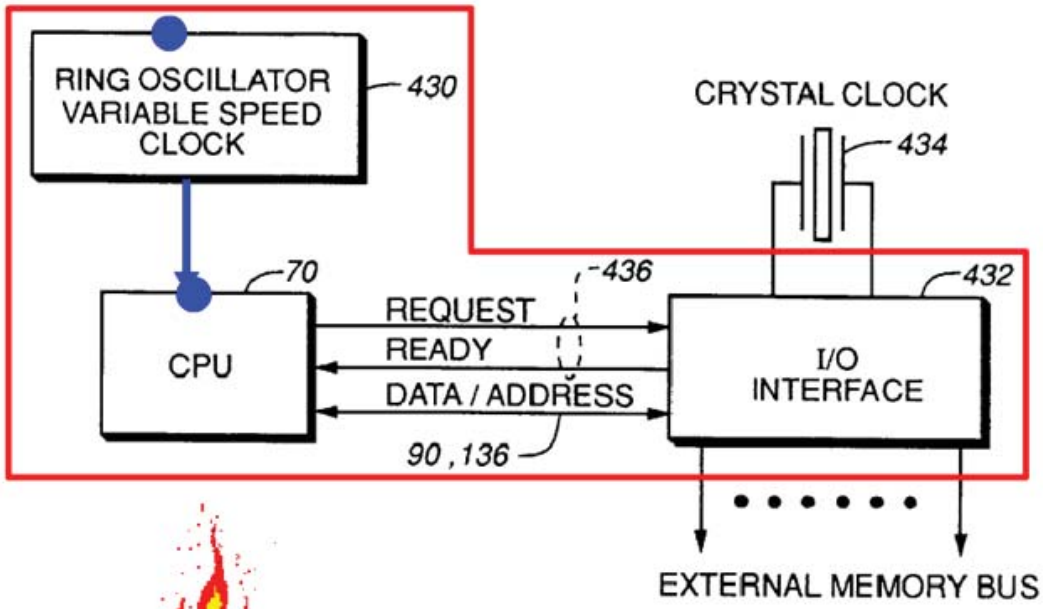


FIG._17

The ’336 Patent

I/O Interface Timed by Independent Off-Chip Clock

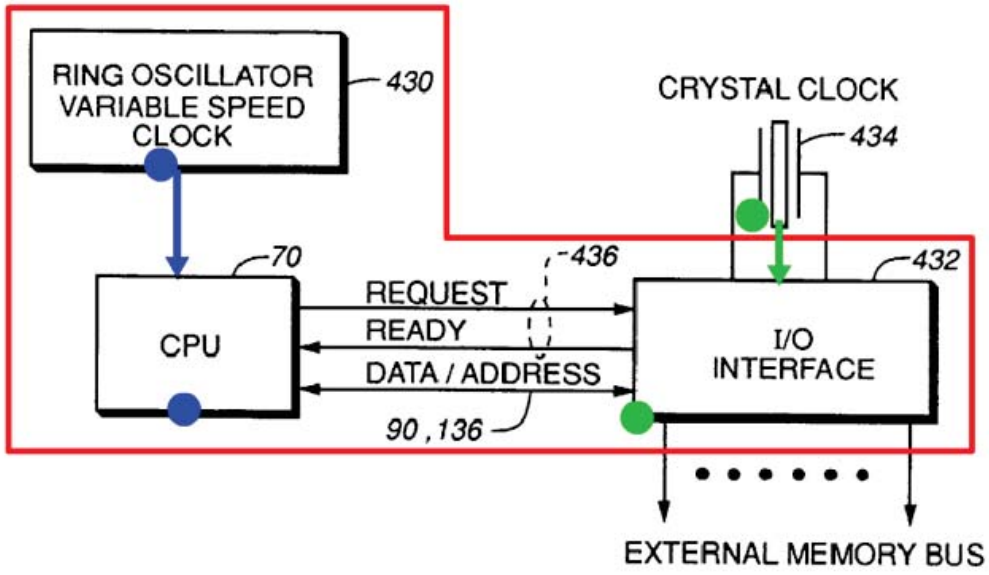


FIG._17

Thank you