PATENT SPECIFICATION (11) 1 570 791

1 570 791

(21) Application No. 53478/76 (22) Filed 22 Dec. 1976 (19)

(31) Convention Application No. 642844 (32) Filed 22 Dec. 1975 in

(33) United States of America (US)

(44) Complete Specification Published 9 Jul. 1980

(51) INT. CL.[3] G06F 7/52

(52) Index at Acceptance
G4A 17P 2B5 2BY 2F8 MD

(72) Inventor: JERRY L. KINDELL

(54) MULTIPLICATION APPARATUS

(71) We, HONEYWELL INFORMATION SYSTEMS INC. a Corporation organised and existing under the laws of the State of Delaware, United States of America, of 200 Smith Street, Waltham, Massachusetts 02154, United States of America, do hereby declare the invention for which we pray that a patent may be granted to us, and the method by which it is to be performed, to be particularly described in and by the following statement:-

This invention relates to multiplication system and apparatus, and more particularly to a system for multiplying numbers, and more particularly to a system for multiplying numbers in binary form.

In general, it is well known to perform multiplication operations by generating required multiples of a multiplicand during the multiplication operation (in contrast to prestoring multiples). An example of this type of apparatus is disclosed in U.S. Patent 3,730,425. That apparatus operates to multiply the multiplier by two (i.e., shifts the multiplier left one bit) before beginning the multiplication operation. Thereafter, during the multiplication operation, circuits included in the multiplication apparatus select one of a number of multiplication factors in accordance with pairs of multiplier bits. While that apparatus halves the number of cycles normally required for a multiplication operation, the time for performing a multiply operation is still considerable. Also, as discussed in that patent, others have suggested speeding up multiplication by examining multiplier bits in pairs, and adding different multiples to a number of series connected adders. An example of this type of multiplication is described in the text, "The Logic of Computer Arithmetic", by Ivan Flores, published by Prentice-Hall Inc., 1963.

Other types of multiplication apparatus have employed prestored multiples which are generated prior to the multiplication operation, the multiples being selected in accordance with the values of multiplier digits. An example of this type of apparatus is disclosed in U.S. patent 3,641,331. While that apparatus reduces the number of multiples required to be stored and the time required for generating all of the remaining multiples, considerable time is still required for generating and storing the multiples prior to the multiplication operation.

The object of the present invention is to provide an improved multiplication system.

Accordingly, the invention provides a multiplication system in which the multiplier is divided into a plurality of portions and each portion is divided into a plurality of sections, comprising:

a plurality of multiple selection circuits to which the sections of a single portion of the multiplier are applied in parallel, each forming a corresponding multiple of the multiplicand;

a corresponding plurality of multiple registers, fed from the multiple selection circuits; and

a corresponding plurality of carry save adders arranged in cascade, fed from the multiple registers and from the output of the cascade;

and wherein in operation each portion of the multiplier is applied to the multiple selection circuits in a respective cycle of a primary sequence of cycles to generate a set of multiples of the multiplicand for that portion, the set of multiples for that portion are stored in the multiple registers, and the contents of the multiple registers for that portion are added in the cascade concurrently with the generation of the set of multiples for the next portion (if any) of the multiplier.

It should be noted that the sections of the multiplier forming each portion overlap; in the preferred embodiment, each section is 3 bits and overlaps the adjacent sections by 1 bit. Similarly, the portions of the multiplier overlap; in the preferred embodiment, each portion is

7 bits, giving 3 sections, and overlaps the adjacent portions by 1 bit.

A carry save adder is an adder consisting of a set of individual 1-bit full adders *without* carry through. Thus a carry save adder has 3 inputs to each bit position, and produces 2 outputs at each bit position; the 2 outputs represent the sum and carry of the 3 input bits. By providing a cascade of such adders, either linear or branched, a plurality of words can be added together. The advantage of the carry save adder is that there is no need to allow for long delays for carry propagation across, potentially, the entire width of the words. A carry through adder is a more conventional adder in which the carry from one bit position is fed to the next bit position, so that there are just 2 "free" inputs plus the carry input to each position, and 1 output.

In the preferred embodiment, there are three carry save adder circuits, each of which receives the multiple signals from one of a corresponding one of three multiple selection circuits. A common timing source includes circuits for generating timing control signals which are applied to three multiple registers fed from the multiple selection circuits. The multiple registers store the multiple signals until the timing source enables their replacement by the multiple generated during a previous cycle of operation. In this manner, the apparatus enables multiple generation operations to be overlapped and to proceed in parallel with the generation of partial products by the adder circuits. This reduces to a minimum the number of cycles normally required to perform a multiplication operation.

A multiplication system embodying the invention will now be described, by way of example, with reference to the accompanying drawings, in which:

Figure 1 is a block diagram of the multiplication system.

Figure 2a shows in greater detail the decoder circuits of Figure 1.

Figure 2b shows in greater detail the clock control circuits of Figure 1.

Figure 2c shows in greater detail a simplified version of the carry save adder network and binary multiple generator circuits of Figure 1.

Figure 2d shows the carry save adder network of Figure 1.

Figure 3 shows several of the timing signals produced by the clock control circuits of Figure 1.

Figure 4 is a cycle sequence diagram of the operation of Figure 1.

Figure 1 shows a system for multiplying a pair of binary numbers. A 36-bit multiplicand which can be extended with ZEROS to fill a 72-bit multiplicand is multiplied by a 28, 36, or 64-bit multiplier depending upon the particular instruction being executed as explained herein. A 72-bit multiplicand is used in the case of a floating point multiply instruction.

In the case of a 28-bit multiplier, when the bits are not evenly divisible by the number of bits being examined per cycle, the mutliplier is extended with ZEROS at the least significant bit end to make the division come out even. As explained herein, with three levels of carry save addition, seven bits of the multiplier are examined at a time, three overlapped groups of multiplier bits (e.g. for the most significant bits 0 to 6, the different groups correspond to bits 0-2, 2-4 and 4-6). The groups each select the appropriate multiples for the multiplier digits having actual values 0-7.

The multiplication of the two binary words is controlled by the various control signal levels provided by a microprogrammed control store 40 which forms part of the control portion of a computer system employing the multiplication system. From Figure 1, during a cycle of operation, it is seen that the microinstruction contents of an addressed storage location of the store 40 are read out and stored in a control point register 404 for decoding by the decoder circuits of block 406.

The timing signals for the system of Figure 1 are provided by external clock circuits of block 20; these circuits provide clock pulses at 100 nanosecond intervals on two sets of lines. The clock pulses on each line are offset from one another by 180°. The clock pulses are applied to the clock control circuits of block 401 of a timing and control section 400 which controls the distribution of the clock pulses to the system of Figure 1 in addition to controlling whether the pulses are distributed at 50 or 100 nanosecond intervals as explained herein.

From Figure 1, it is seen that the system 10 comprises: the timing and control section 400 mentioned above; an input section 300; a multiple generator section 100; a carry save adder section 200; and a full adder and storage section 250.

The Timing And Control Section 400

This section includes the clock control circuits and mode flip-flops of block 401, the control point register 404, the decoder circuits of block 406, a four-stage binary counter 408 and decoder circuits of block 410. The circuits of block 401, shown in greater detail in Figure 2b, under microprogram control distribute clock pulses at either 100 nanosecond or 50 nanosecond rate. More specifically, the multiplier system is controlled by bits 88-97 of microinstruction word, part of which is loaded into the control point register 404 at the beginning of each cycle of operation. The microinstruction word bits are coded as follows:

(1) bits 88-89 are type bits which identify the operation or function to be performed. The

code 01 identifies the multiply function;

(2) bit 90 controls the strobing of the multiplier holding register RCH, not shown, which connects to the input buffer gates 301 via the input lines RCHO-35. The state of this bit is not resident in the control point register 404, but is stored in another control point register, not shown. When bit 90 is a ZERO, no strobe pulse is generated and when bit 90 is a binary ONE, a strobe pulse is applied to the RCH register.

(3) bits 91 to 93 designate the type of multiply operation and are used to load the counter 408 which controls the operation of a ZMR switch 310 of section 300. These bits are coded as follows:

000 = NOP
001 = MPF = Multiply fractional, fixed point;
010 = MPY = Multiply integer, fixed point;
011 = FMP = Single precision, normalized, floating point multiply;
100 = UFM = Single precision, unnormalized, floating point multiply;
101 = DFMP = Double precision, normalized, floating point multiply; and
110 = DUFM = Double precision, unnormalized, floating point multiply.

(4) bit 94 controls the timing mode for the multiplier system by establishing when the circuits of block 401 will provide clock pulses at 50 and 100 nanosecond rates. When a binary ZERO, the bit designated as SET50N10 inhibits the circuits of block 401 from generating clock pulses at 50 nanosecond clock rate beginning with the next 100 nanosecond cycle. When a binary ONE, the SET50NS10 signal causes the clock circuits of block 401 to generate clock pulses at a 50 nanosecond rate beginning with the next 100 nanosecond cycle.

(5) bit 95 is used to control an output switch, not shown, to which the adder output is delivered.

(6) bits 96-97 control the strobing of a register included in the data output path, not shown, and an input RCA register 302 of section 300 defined by the codes 00 and 01, respectively.

Figure 2b shows block 401 in detail. In operation, whenever the SET50NS10 signal is a binary ONE, this causes the mode flip-flops 401-5 and 401-4 to switch to a binary ONE which forces signals FF50NS-MODE 10 and FF50NS-MODE 11 to a binary ONE. This specifies the 50 nanosecond mode of operation in which clock pulses are delivered at a 50 nanosecond rate. As long as the clock bit 94 is a binary ONE, the system will remain in the 50 second nanosecond mode.

The mode flip-flop 401-4 when switched to a binary ONE forces another mode flip-flop 401-3 to a binary ONE forcing signal DL-FF-50NS-MODE 00 to a binary ZERO. This flip-flop maintains the multiply system in the 50 nanosecond mode throughout the last 100 nanosecond cycle prior to switching out of the 50 nanosecond mode of operation. At the end of the 100 nanosecond interval, flip-flop 401-3 is reset to a binary ZERO.

The binary ZERO side of mode flip-flop 401-5 is applied to the pair of gates 401-30 and 401-32 which control the operation of a multiplier selection switch 316 (ZIER). The gate 401-30 forces the output of the ZIER switch to ZEROS in the 100 nanosecond cycle before entering the 50 nanosecond mode in the case of floating point instructions (i.e., signal FFFLOATING10 = 1).

It will be noted that the outputs of the two flip-flops are "ored" together by gate 401-28 whose outputs then indicate whether the system is operating in 50 or 100 nanosecond mode. The state of the 100NSMODE 10 signal is used to control the operation of the counter 408 via gate 401-22 and the strobing of signals applied to the gates 304 via gate 401-20 when enabled by gate 401-18. The gate 401-24 generates the signal SET50NS12 in response to signals FF100NSMODE 10 and SET50NS10 which identify the last 100 ns cycle before the system enters the 50 ns mode. The SET50NS12 signal controls the loading of the RMN register 308.

The gate 401-6 when enabled by mode flip-flop 401-5 applies the set of clock pulses to the driver circuit 401-8. The driver circuit 401-8 also receives the second set of pulses from gate 401-14. The pulses applied from both inputs are "ored" by the driver circuit 401-8 to produce the clock signal CLOCK MLT310. This signal is delayed by selector circuit 401-10 to produce a delayed clock signal CLOCK DL-2-10. As seen from Figure 1, the CLOCK MLT310 signal is applied to the flip-flops, RS and RC registers 250-4 and 250-6 in addition to the multiplier selection registers. To avoid any transfer race conditions, the signal CLOCK DL-2-10 is applied to the multiple generator latch circuits. The normal clock signal CLOCK 100NS120 is provided by the driver circuit 401-12 in response to the input signal CLOCK 100NS10 which is inverted by gate 401-16. The CLOCK100NS120 signal is applied to register 404 and strobes the register at 100 nanosecond intervals which corresponds to the

read cycle of the ROM40. The various clock signals discussed above are illustrated in Figure 3.

Figure 2a shows in greater detail the decoder and encoder circuits of block 406. The octal decoder circuit 406-1 decodes bits 91-93 of the microinstruction word stored in the control point register 404 and generates signals indicating the type of multiply instruction. From the floating point signals produced by circuit 406-1, circuit 406-3 encodes these signals to define further the type of floating point operation such as normalized, unnormalized, etc.

As mentioned above, the signals from the block 406 are used to control the selection of the 50 nanosecond mode of operation and the selection of multiplier bits. The counter 408 in response to signals from block 406 controls the selection of multiplier bits through the ZMR switch 310 of input section 300. The counter 408 is intially loaded with a starting count under the control of bits 91 to 93 and is decremented by one during each cycle when the system is operating in the 50 nanosecond mode (or when the SET50NS10 signal is a binary ONE).

Initially, the three most signficant bits of the counter are set to binary ONES and the least significant bit is set to a binary ONE for floating point instructions and to a ZERO for other instructions. The loading of counter 408 takes place during the cycle when the RCA register 302 is being loaded with the multiplicand. The output signals from the counter 408 are decoded by the circuits 410 and the resulting signals are used to select one of the eight positions of switch 310. The switch 310 feeds switch 312. The basic control depends upon whether the system is executing a floating point instruction. In the case of no floating point instruction, the output from switch 312 corresponds to the first position (ZMRO-6) and in the case of a floating point instruction the output is from the second position (ZMR2-8). The selection of switches and counter states are as illustrated in the table herein.

INSTRN.	MLTPLR BITS	-MULTIPLIER BIT SELECTION ACTIONS-	COUN-TER STATE	ZMR SWTCH CNTRL STATE	ZZMR SWTCH CNTRL STATE	ZIER SWTCH CNTRL STATE
MPF	30-35,X	ZIER←RCH30-35, RMNO; RMN←RCH0-8; RMR←RCH24-30	1110	6	0	1
MPY	24-30	ZIER←RMR; RMR←RCH18-24	1101	5	0	0
	18-24	ZIER←RMR; RMR←RCH12-18	1100	4	0	0
	12-18	ZIER←RMR; RMR←RCH6-12	1011	3	0	0
	6-12	ZIER←RMR; RMR←RMN0-6	1010	2	0	0
	0-6	ZIER←RMR	1001	1	0	0
FMP	XXXXXXX	Turn Off ZIER Enable; RMN←RCH0-8; RMR←RCH32-35, 000	1111	7	1	*
UFM	32-35,XXX	ZIER←RMR; RMR←RCH26-32	1110	6	1	0
	26-32	ZIER←RMR; RMR←RCH20-26	1101	5	1	0
	20-26	ZIER←RMR; RMR←RCH14-20	1100	4	1	0
	14-20	ZIER←RMR; RMR←RCH8-14	1011	3	1	0
	8-14	ZIER←RMR	1010	2	1	0
DFMP	XXXXXXX	Turn Off ZIER Enable; RMN←RCH0-8; RMR←RCH30-32, 000	1111	7	1	*
DUFM	68-71,XXX	ZIER←RMR; RMR←RCH26-32	1110	6	1	0
	62-68	ZIER←RMR; RMR←RCH20-26	1101	5	1	0
	56-62	ZIER←RMR; RMR←RCH14-20	1100	4	1	0
	50-56	ZIER←RMR; RMR←RCH8-14	1011	3	1	0
	44-50	ZIER←RMR; RMR←RMN2-8	1010	2	1	0
	38-44	ZIER←RMR; RMR←RCH32-35, RMN0-2	1001	1	1	0
	32-38	ZIER←RMR; RMR←RCH26-32	1000	0	1	0
	26-32	ZIER←RMR; RMR←RCH20-26	0111	5	1	0
	20-26	ZIER←RMR; RMR←RCH14-20	0110	4	1	0
	14-20	ZIER←RMR; RMR←RCH8-14	0101	3	1	0
	8-14	ZIER←RMR	0100	0	1	0

MULTIPLE SELECTION TABLE

NOTE: X is always a ZERO.

The RMN register is always initialized to ZEROS prior to the beginning of the multiplier bit selection process.

* The enable control to the switch is turned off during this cycle which forces the ZIER output to ZEROS.

Input Section 300

This section includes gate and register circuits 301, 302, 304 and 306 which receive signals representative of the multiplicand and multiplier, a pair of registers 308 and 314 and delay circuits 320, in addition to the circuits 310, 312, and 316 which select the different bits of the multiplier in the proper sequence. The sequences and multiplier bit groupings for the different instructions are also indicated in the above table. The circuits of the section adapt the multiplier system to a 36 bit wide bus. (It will be obvious that this section would be considerably less complex if the system were connected to a 72 bit wide bus).

The RMN register 308 is a 9 bit register which is cleared to ZERO whenever the RCA register 302 is loaded (i.e., when bits 91-93 specify such loading). Whenever the SET50NS12 signal is a binary ONE, register 308 is loaded with bits 0-8 from a register RCH, not shown. This confines loading to the last full 100 nanosecond cycle before the system begins operating in the 50 nanosecond mode. In the case of double precision multipliers, the RMN register 308 provides a smooth transition in transfers of the lower and upper words of the multiplier.

The RCA register 302 is a 36 bit register which provides storage for one word of the multiplicand. As shown in Figure 1, its outputs are applied in parallel to each of the multiple generator circuits 304 in a similar fashion are connected in parallel to the multiple generator circuits of section 100. In the case of a double precision (floating point instruction) operation, these circuits apply the least significant 36 bits of the multiplicand from an input ZAQ but which are held there throughout the execution of the instruction. In the case of a single precision operation, the input buffers are disabled by signal SUPZAQ10 from block 401. This results in ZEROS being applied to the circuits of section 100.

Considering the multiplier selection switches in greater detail, it is seen that the ZMR switch 310 is a 9 bit wide, one of eight data selector switch. Under the control of counter 408, the ZMR switch 310 performs the primary multiplier bit selection. The ZZMR switch 312 is a 7 bit wide one of two data selector switch which selects which set of the output multiplier bit signals (ZMRO-8) is loaded into the 7 bit wide RMR register 314. The ZIER switch 316 is a 7 bit wide one of two data selector switch which selects which multiplier bit signals are applied to the multiple generator circuits of section 100.

During initial examination of the multiplier, the ZIER switch 316 either selects the bits directly from the input RCH register when the multiply instruction specifies a fixed point operation or forces its output to ZEROS when the multiply instruction specifies a floating point operation. The initial examination always takes place during a full 100 nanosecond cycle. All other examinations of the multiplier take place during 50 nanosecond cycles during which time the ZIER switch 316 selects the RMR register.

During the 100 nanosecond cycle, there is enough time to select the multiplier bits from the RCH register and generate the multiple. The RMR register ensures that multiplier bit selection occurs in one cycle and multiple generation takes place in the next cycle.

During the initial examination, the least significant bit selected is from the input RMN0. This bit is chosen because it is ZERO for the fixed point multiply instructions and for the start of the first multiplier word of a quad precision multiply instruction. Also, it corresponds to the most significant bit from the previous word during the switch of multiplier words for the quad precision multiply instruction.

As explained herein, the multiply operation is arranged so that the last partial product summation in the carry save adder network occurs in the last half of the final 100 nanosecond cycle while the system is operating in the 50 nanosecond cycle while the system is operating in the 50 nanosecond mode. This provides for a smooth transition back to 100 nanosecond mode. Thus, the last multiplier examination must take place during the first half of that last 100 nanosecond cycle.

The number of 50 nanosecond cycles is chosen to be always even and the number of multiplier bit selection cycles is equal to the number of 50 nanosecond cycles minus one plus one for the selection of the first multiplier bit during the last 100 nanosecond cycle preceding the 50 nanosecond cycles. Thus, the number of multiplier bit selection cycles must also be even. While this is acceptable for the 36 bit multiplier used in the case of fixed point operations, the 28 and 64 bit multipliers used in the case of floating point operations require an odd number of cycles to complete the examination of the multiplier. The number of cycles is made even by forcing the first multiplier bits selected to be all ZEROS. The actual bits from the multiplier are then used starting with the next selection.

As seen from Figure 1, the multiplier bits selected by the ZIER switch 316 are delayed by circuits included within block 320 and then applied in parallel to each of the multiple generator circuits of section 100. The delay is such that strobing or loading of the latch circuits can take place following the strobing at the other registers. This ensures that signals being scanned and applied to the RC and RS registers 200-10 and 200-12 are being strobed or loaded. For this reason, the clock or strobing signal CLOCK DL210 applied to the latch circuits is also delayed.

Multiple Generator Section 100

This section includes the multiple generator circuits 100-2, 100-4 and 100-6 in addition to the carry generation and decode circuits of block 100-10. These circuits in response to the multiplier bits applied thereto control the generation of multiples of the multiplicand during the multiplication. As mentioned, each of the circuits 100-2, 100-4 and 100-6 receives as data inputs the multiplicand signals from the RCA register 302 and the ZAQ input bus signals from buffer gate circuits of block 304.

The function performed by these circuits is illustrated in Figure 2c, at the top left. The multiple generator circuit functions somewhat as a one of five data selector switch, the output of which feeds a number of latch circuits. The latch circuits enable the formation of the multiple and the summing of the multiple to the partial product to occur concurrently over an interval of two 50 nanosecond cycles. It will be noted that each generator circuit is 73 bits wide. This width allows the 72 bit multiplicands to be shifted right by one bit position without being truncated. The bit must be stored until it has been added to the partial product and until all carries resulting from the addition have been fully propagated. The shifting is denoted in Figure 1 by the two ZEROS (i.e. 0,0). In the other positions, fixed values of ZERO and ONE are wired into positions so as to provide the appropriate values for the least significant bits of these multiples.

ZIER SELECTED MULTIPLIER BITS	ZIER INPUT TO V114 CHIP	MULTIPLE FORMATION REQUIRED BY ALGORITHM	ACTION OF MULTIPLE GENERATOR	RESULT OF CARRY IN GENERATION
000	111	0☆M'icand	0	0
001	110	1/2☆M'icand	M'icand right one	0
010	101	1/2☆M'icand	M'icand right one	0
011	100	1☆M'icand	M'icand	0
100	011	-1☆M'icand	$\overline{\text{M'icand}}$	1
101	010	-1/2☆M'icand	$\overline{\text{M'icand right one}}$	1
110	001	-1/2☆M'icand	$\overline{\text{M'icand right one}}$	1
111	000	0☆M'icand	0	0

From the table, it is seen that when a "negative" multiple is specified for generation, the multiple generator circuit produces the one's complement (inverse) of the multiple together with a carry-in bit provided by the circuits of block 100-10. When the complement of the multiple is added to the partial product by the carry-save network of section 200, the carry-in bit is also added to the least significant bit position of the multiple. This completes the complementing of the multiple (two's complement) for a required subtraction operation.

The circuits of block 100-10 include decoder circuits, conventional in design, which decode the different groups of multiplier bits and generate the appropriate carry-out signals having the values indicated in the above table.

Additionally, the block 100-10 includes a plurality of flip-flops and delay circuits, conventional in design. The delay circuits delay the carry-out signals generated by the decoder circuits. The signals are thereafter stored in clocked flip-flops whose binary ONE output are

applied as inputs to the carry save adder section 200 and to the full adder section 250. The output signals correspond to signals DLFFCARRYA10, FFCARRYB10, and FFCARRYC10. The decoded signal DLFFCARRYA10 generated in response to the group of multiplier bits for the A level is applied of two series connected flip-flops before being applied to the section 250 in order to provide for proper timing as explained herein.

Carry Save Adder Section 200

This section includes a plurality of carry save adder array networks 200-2, 200-4, and 200-6 in addition to a carry register (RC) 200-10 and a sum register (RS) 200-12. The three levels, designated A, B and C, of carry save adder array networks add the multiples to the partial product accumulated in the registers 200-10 and 200-12.

The different levels are shown in greater detail in Figures 2c and 2d. Referring to Figure 2c, there are shown the connections to each of the carry save adder network levels for multiplying an 8 bit multiplicand by a 6 bit multiplier. It will be noted that the carry save adder networks broaden by two carry save adder circuit stages at both the most significant and least significant ends of each level. Thus, at each level, the multiple input signals are shifted left by 2 bit positions and at the bottom level, four carry save adder circuits do not receive input signals from the multiple generator circuit 100-6. The widening at the most significant end positions the multiples at the proper bit positions which are to be added to the partial product. This accomplishes the 2 bit left shift of the multiples.

Each of the blocks within each level (e.g. CSA, CSB, and CSC) represents a carry save adder circuit for performing the addition of a pair of addended bits and a carry bit. The ACin, BCin, and CCin signals correspond to the output carry-in signals generated by the carry-in generation circuits. The ACin signal instead of being applied to the carry save adder networks is stored in a flip-flop, not shown, and then applied to the carry adder of section 250 during a following cycle of operation.

The RC and RS registers 200-10 and 200-12 store the carry and sum output signals generated by the C level carry save adder network. The output signals from these registers, as shown are applied as inputs to the A level carry save adder network and to the ripple carry adder of section 250.

Figure 2d illustrates the structural arrangement of carry save adder array networks employed in Figure 1. The left and right sides of the figure show the most significant bit structure and least significant bit structure, respectively. The RC and RS registers 200-10 and 200-12 represent the preceding level for adder level A. The carry-in signals FFCARRYC and FFCARRYB produced by the carry generation and decode circuits of block 100-10 and are inserted in levels in the places shown. The carry-in signal FFCARRYA is inserted into the ripple carry adder of section 250 at the stage which generates bit 76 as explained herein.

Full Adder Section 250

This section includes a 78 bit adder 250-2 with carry lookahead propagation in addition to carry out storage as illustrated in Figure 2c (i.e., flip-flop 250-8). The adder 250-2 adds the carry signals to the sum signals of the partial product to form the final product.

In operation, during the main multiply cycles, the adder 250-2 generates a carry out signal for bits of the product which are to be discarded during that cycle (i.e. bits 71-76). The carry out signal from bit position 71 is stored in a flip-flop, such as flip-flop 250-8 of Figure 2c, and is applied as the carry input signal (i.e. RIPCIN in Figure 2c) to the adder 250-2 during the next cycle.

As illustrated in Figure 2c, the RS register 200-12 stages are connected to corresponding stages of the adder 250-2. The RC register 200-10 stages are in effect shifted one bit position to the left and then connected to stages of the adder 250-2. This leaves the input to the stage of bit position 76 or to position 12 of Figure 2c free for the insertion of the carry-in signal from level A. The extra bit position in the adder 250-2 corresponds to the most significant bit position and serves as an extended sign bit. This stage always generates the actual sign and can be used in the detection and correction of overflow conditions. The input lebeled "PROP BIT" in Figure 2c is connected to a binary ONE enabling the carry out signal to be propagated through that stage. The stages in Figure 1 are also similarly connected to binary ONES where such stages are absent a second input from the carry save adder circuits or other circuits associated therewith.

DESCRIPTION OF OPERATION

General

Before illustrating the operation of the system in detail, the overall operation will be described in general terms with reference to Figure 4 and a table included herein.

As mentioned, the operations performed by the various sections of the system are overlapped. The table included herein shows the major actions that occur in each cycle of operation during the execution of a typical instruction. The table only shows those cycles associated with the different sections in which actions take place which are related in producing the final

product. However, it will be appreciated that all of the sections are always performing some action.

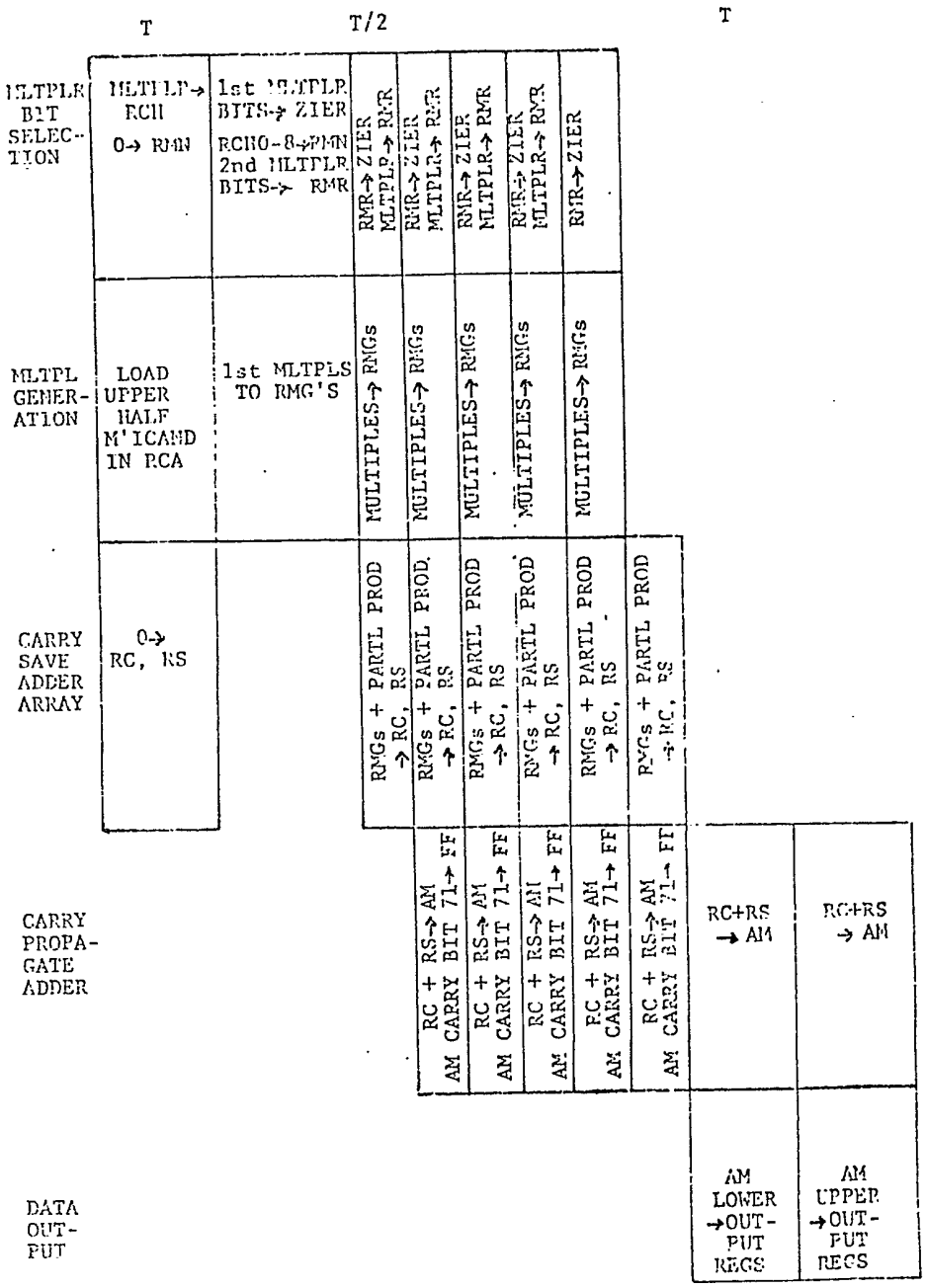


figure 4 illustrates the actions for a fixed binary integer multiply (MPY) which involves a 36 bit multiplicand. Referring to the figure, it will be noted that the first cycle is a set up cycle. During this cycle, the multiplier bit signals are loaded into a register where they can be examined and the multiplicand is loaded into the RCA register. In this instance, ZEROS are being loaded into the RMN, RC, and RS registers during the same cycle.

During the second cycle, the system receives the second half of the multiplicand from the ZAQ bus. Since the second half of the multiplicant for this type instruction is always ZERO, the input bus is forced to ZEROS. At this time, the system is set to the 50 nanosecond mode of operation at the start of the next 100 nanosecond cycle by the circuits of block 401. Also, the counter 408 will also have been forced to state "11100".

It will be noted that during this cycle, the input section 300 examines the first 7 bits of the multiplier which causes the first multiple generated by each of the multiple generator circuits 100-2, 100-4, and 100-6 to be loaded into the latch circuits associated therewith. That is, the ZIER switch 316 distributes the different groups of multiplier bits to each of the generator circuits which in turn select the proper multiple to be stored by the latch circuits. At the same time, a second set of multiplier bits is selected and loaded into the RMR register 314. Also, the upper bits are loaded into the RMN register 308.

As seen from Figure 4, the system begins a series of four 50 nanosecond cycles which repetitively perform the same actions. During each cycle, a next group of multiplier bits is selected and loaded into the RMR register 316. In parallel, the multiplier bits previously stored in the RMR register 316 select the next group of multiples which is stored in the latch circuits. The multiples previously stored in the latch circuits are added by the carry save adder networks 200-2, 200-4 and 200-6 to the RC and RS registers 200-20 and 200-12 with the result being then stored in the RC and RS registers 200-10 and 200-12. It will be appreciated that the counter 408 is decremented by the circuits of block 401 during each cycle.

After the operations outlined above are performed, the loading of the multiplier bits is complete. During the next cycle, the multiplier bits previously stored in the RMR register 316 select the last group of multiples. The multiple generator circuits generat ethe multiples which are stored in the latch circuits associated therewith while the carry save adder networks sum the multiples previously stored in the latch circuits. During a final cycle of execution, the carry save adder networks sum the multiples stored in the latch circuits as shown in Figure 4. At this point, the carry and sum registers 200-10 and 200-12 store the complete product. Also, during this cycle, the system is conditioned to be switched back into the 100 nanosecond mode of operation by the circuits of block 401.

The next two cycles serve as "clean up" cycles during which the full adder 250-2 adds the contents of the RC and RS registers 200-10 and 200-12. Also, during these cycles, indicators (i.e. ZERO and sign) are examined and signals representative of the final product are transferred to various registers located in other units for storage.

It will be appreciated that there are slight variations in the cycles as well as in the set up operations for the different instructions. For example, as mentioned previously, in the case of a floating point multiply instruction, during the initial 100 nanosecond cycle, the input signals are forced to ZEROS in order to have the number of 50 nanosecond cycles come out even. This ensures that the last half cycle occurs in synchronism with the end of a 100 nanosecond cycle interval. That is, the multiplication begins on a full or half cycle depending upon whether the multiplier is divisible by six.

DETAILED DESCRIPTION OF OPERATION

The operation of the multiplication system will now be described with reference to a specific example. The example is a multiply fractional instruction (MPF) in which the multiplier, multiplicand, and final product have the following values.

Multiplier	= 0.101 1000 1110 1100 0000 . . .0000	Binary
	= 0.543540000000	Octal
	= 0.6947021484375	Decimal
Multiplicand	= 0.111 0000 0000 . . .0000	Binary
	= 0.7	Octal
	= 0.875	Decimal
Product	= 0.100 1101 1100 1110 1000 . . .0	Binary
	= 0.467164	Octal
	= 0.6078643798828125	Decimal

The values for the multiplier have been selected so that each of the eight possible combinations of multiplier bits is included. The multiplicand for ease of illustration was chosen to be all ZEROS except for one digit. The equivalent octal and decimal values are also given to facilitate noting of the results generated during each cycle of operation. The contents of the registers have octal values indicated during the cycle. The values are also given in binary form in those instances where it facilitates an understanding of the system's operation.

The first cycle is a start execution cycle. During this cycle, the multiplier applied initially from a data-in register, now shown, is applied via a temporary storage register, RCH, now shown, to the lines RCH0-35. Also, the multiplicand is transferred from an operand register, not shown, to the RCA register 302. In the example, they have the following values.

Multiplier = 0.543540000000
Multiplicand = 0.700000000000

The contents of other registers, e.g., RCH, RMN, RCA, RC, RS, RMR, and latch circuits RMGA, RMGB, and RMGC can store any values since they are not relevant (i.e. represent "don't care" situations).

During the second cycle, the first set of multiples is prepared and the multiplier bits examined by the ZIER switch 316 correspond to RCH bits 30-35 and RMN bit 0. In this case, all seven multiplier bits are ZEROS. Therefore, the decoder circuit of each multiple generator citcuit selects ZEROS for all three levels and the ZEROS are stored in each of the latch circuits RMGA, RMGB, and RMGC. Also, the RCH bits 24-30 are selected by ZMR switch 310 and loaded into the RMR register 314 via the ZZMR switch 312. The RCH bits 0-8 are loaded into the RMN register 308 in response to the SET50NS12 signal. The SET50NS12 signal also conditions the circuits of block 40 to switch into 50 nanosecond mode of operation during the next cycle.

During the second cycle, the pertinent registers and circuits store and generate signals having the following values:

RCHO = 0.5435400000000
RCA = 0.7000000000000
ZAQ = 0.0000000000000
RMN = 000
RC = 0.0.....0
RS = 0.0.....0
ZIER = 000
CARRYA = 0
CARRYB = 0
CARRYC = 0

During this cycle (i.e., first 50 nanosecond cycle), the multiples prepared in the preceding cycle are added together by the carry save adder networks to form a ZERO partial product represented by the contents of the carry and sum registers. That is, since all the input multiple signals are ZEROS, all adder sum and carry output signals generated are ZEROS. The multiplier bits stored in the RMR register 314 are ZEROS, therefore, the multiples being prepared during this cycle are ZEROS. In addition, the carry output signals are ZEROS. The multiplier signals RCH 18-24 are selected and placed in the RMR register 314.

During this cycle (i.e., first 50 nanosecond cycle), the pertinent registers and circuits store and generate signals having the following values.

RCHO	= 0.543540000000
RCA	= 0.70.....0
ZAQ	= 0.0.....0
RMN	= 261
RMGA	= 0.0.....0
RMGB	= 0.0.....0
RMGC	= 0.0.....0
RC	= 0.0.....0
RS	= 0.0.....0
ACSA CARRY	= 0.0.....0
ACSA SUM	= 0.0.....0
ACSB CARRY	= 0.0.....0
ACSB SUM	= 0.0.....0
ACSC CARRY	= 0.0.....0
ACSC SUM	= 0.0.....0
FF-CARRY A 10	= 0
FF-CARRY B 10	= 0
FF-CARRY C 10	= 0
RMR	= 000
ZIER	= 000
CARRY A	= 0
CARRY B	= 0
CARRY C	= 0

During the next cycle (i.e., second 50 nanosecond cycle), the pertinent registers and circuits store and generate signals having the following values.

RCHO	= 0.543540000000
RCA	= 0.70.....0
ZAQ	= 0.0.....0
RMN	= 261
RMGA	= 0.0.....0
RMGB	= 0.0.....0
RMGC	= 0.0.....0
RC	= 0.0.....0
RS	= 0.0.....0
ACSA CARRY	= 0.0.....0
ACSA SUM	= 0.0.....0
ACSB CARRY	= 0.0.....0
ACSB SUM	= 0.0.....0
ACSC CARRY	= 0.0.....0
ACSC SUM	= 0.0.....0
FF-CARRY A 10	= 0
FF-CARRY B 10	= 0
FF-CARRY C 10	= 0
RMR	= 000
ZIER	= 000
CARRY A	= 0
CARRY B	= 0
CARRY C	= 0
DL-FF-CARRY A 10	= 0
FULL ADDER (AM)	= 0.0.....0
ADDER CARRY OUT BIT71	= 0

It will be noted that, again, multiples having a ZERO value are added by the carry save adder networks producing a ZERO partial product corresponding to the sum and carry component signals stored in the RS and RC registers 200-12 and 200-10. Also, new ZERO multiples are generated by the multiple generator circuits since the multiplier bits stored in the RMR register 314 are ZEROS. Multiplier bits RCH 12-18 are selected and stored in the RMR register 314. The full carry adder 250-2 sums the contents of RC and RS registers which produces a ZERO carry out signal for the product bits which will be discarded (not required for the final result).

During the next cycle of operation (i.e., third 50 nanosecond cycle), the carry save adder networks again sum the ZERO multiples stored in the latch circuits to a ZERO partial product resulting in a ZERO partial product. The full carry adder also produces a ZERO

final product result. However, during this cycle, the multiplier bits stored with RMR register 314 are not all ZEROS but have the binary value "1100000". This means that the groups of multiplier bits for the A, B, and C levels are "000", "000", and "110", respectively. Therefore, the multiple generator circuits 100-2 and 100-4 generate ZERO multiples while the multiple generator circuits 100-6 generates a multiple having a value -1/2 times the multiplicant. The RMGC multiple generator circuit generates the multiple by shifting the input multiplicand signals one bit position to the right and then complementing or inverting each bit signal. The carry output signal is also a binary ONE to complete the subtraction operation by 2's complement addition. Lastly, the bits RCG 6-12 are selected and stored in the RMR register 314.

Thus, during this cycle, the pertinent registers and circuits store and generate signals having the following values.

RCHO	= 0.543540000000
RCA	= 0.70.....0
ZAQ	= 0.0.....0
RMN	= 261
RMGA	= 0.0.....0
RMGB	= 0.0.....0
RMGC	= 0.0.....0
RC	= 0.0.....0
RS	= 0.0.....0
ACSA CARRY	= 0.0.....0
ACSA SUM	= 0.0.....0
ACSB CARRY	= 0.0.....0
ACSB SUM	= 0.0.....0
ACSC CARRY	= 0.0.....0
ACSC SUM	= 0.0.....0
FF-CARRY A 10	= 0
FF-CARRY B 10	= 0
FF-CARRY C 10	= 0
RMR	= 140
ZIER	= 140
CARRY A	= 0
CARRY B	= 0
CARRY C	= 1
DL-FF-CARRY A 10	= 0
FULL ADDER (AM)	= 0.0.....0
ADDER CARRY OUT BIT71	= 0
ADDER F-CARRY-IN BIT76	= 0

The octal 1 in the RMR register 314 corresponds to the value of the least significant bit position of the next number in binary (i.e., 5).

During this cycle (i.e., the fourth 50 nanosecond cycle), the pertinent registers and circuits store and generate signals having the following values.

RCHO	= 0.543540000000
RCA	= 0.70.....0
ZAQ	= 0.0.....0
RMN	= 261
RMGA	= 0.0.....0
RMGB	= 0.0.....0
RMGC	= 1.437.....7
RC	= 0.0.....0
RS	= 0.0.....0
ACSA CARRY	= 0.0.....0
ACSA SUM	= 0.0.....0
ACSB CARRY	= 0.0.....0
ACSB SUM	= 0.0.....04
ACSC CARRY	= 0.0.....0100
ACSC SUM	= 1.437.....7600
FF-CARRY A 10	= 0
FF-CARRY B 10	= 0
FF-CARRY C 10	= 1
RMR	= 035
ZIER	= 035
CARRY A	= 1
CARRY B	= 0
CARRY C	= 0
DL-FF-CARRY A 10	= 0
FULL ADDER (AM)	= 0.0.....0
ADDER CARRY OUT BIT71	= 0
ADDER F-CARRY-IN BIT76	= 0

It will be noted that during this cycle, the carry save adder networks add the first non-ZERO multiple to the partial product. Since signals from the RC and RS registers 200-10 and 200-12 and the RMGA latch circuits are ZEROS, the sum and carry output signals from the ACSA carry save adder network are ZEROS. The stored C level carry output signal FFCARRYC is inserted into the B level carry save adder network at position ACSB-70. This, in turn, causes the ACSB carry save adder network to produce a non-ZERO sum as indicated above (i.e., since all signals are shifted two bit positions to the left, the sum = 4).

The multiple signals from the latch circuits of multiple generator circuit RMGC are summed to the partial product by the carry save adder network ACSC which gives the result indicated. During this cycle, the full adder 250-2 still is summing ZEROS.

During this cycle, the multiplier bits in the RMR register 314 have the binary value "0011101". The multiplier bit groups for levels A, B and C are "101", "111", and "001", respectively. Thus, the multiple generator circuit 100-2 generates a -1/2 times the multiplicand for addition to the partial product. Accordingly, the multiplicand signals to the multiple generator circuit 100-2 are shifted right by one, inverted, and stored in the RMGA latch circuits. Also, the carry output signal is forced to a binary ONE and stored. The multiplier signals "111" cause the multiple generator circuit 100-4 to generate a ZERO multiple. Thus, the multiple generator circuit causes ZEROS to be loaded into the RMGB latch circuits and a ZERO the carry output signal to be generated and stored.

The "001" multiplier signals cause the multiple generator circuit 100-6 to generate a +1/2 times the multiplicand. Accordingly, the circuit 100-6 shifts the input multiplicand signals right by one and loads them into the RMGC latch circuits for later addition to the partial product. A binary ZERO carry output signal is generated and stored. Lastly, bits RMNO-6 are selected and stored in the RMR register 314.

During the next cycle, (i.e., the fifth 50 nanosecond cycle), the pertinent registers and circuits store and generate signals having the following values.

RCHO	= 0.543540000000
RCA	= 0.70 0
ZAQ	= 0.0 0
RMN	= 261
RMGA	= 1.437 7
RMGB	= 0.0 0
RMGC	= 0.340 0
RC	= 0.0 0100
RS	= 1.437 7600
ACSA CARRY	= 1.407 76
ACSA SUM	= 0.330 0
ACSB CARRY	= 0.0020 03
ACSB SUM	= 1.6657 74
ACSC CARRY	= 0.3410 0114
ACSC SUM	= 1.41437 7654
FF-CARRY A 10	= 1
FF-CARRY B 10	= 0
FF-CARRY C 10	= 0
RMR	= 054
ZIER	= 054
CARRY A	= 1
CARRY B	= 0
CARRY C	= 0
DL-FF-CARRY A 10	= 0
FULL ADDER (AM)	= 3.440 0
ADDER CARRY OUT BIT71	= 1
ADDER F-CARRY-IN BIT76	= 0

The multiplier bits stored in the RMR register 314 have the binary value "0101100". This means that the multiplier bit groups for levels A, B, and C are "100", "011", and "010", respectively. Thus, the multiple generator circuit 100-2 requires the generation of -1 times the multiplicand for addition to the partial product. Accordingly, the circuit 100-2 is operative to invert the multiplicand input signals and store them in the RMGA latch circuits. Also, the carry output signal for level A is forced to a binary ONE to complete the 2's complement addition. Since the multiplier bits for level B are "011", the multiple generator circuit 100-4 is required to generate 1 times the multiplicand. Accordingly, the circuit 100-4 is operative to store the input multiplicand signals in the RMGB latch circuits. Also, the carry output signal for level B remains a binary ZERO. The multiplier bits for level C are "010" which requires the multiple generator circuit 100-6 to generate +1/2 times the multiplicand. Accordingly, the circuit 100-6 shifts the input multiplicand signals right by one and loads them into the RMGC latch circuits. The carry output signal for level C is a binary ZERO. At this point, it will be noted that all of the multiplier bits have been examined.

During this cycle, the carry save adder networks sum the previously stored multiples in each of the RMGA, RMGB, and RMGC latch circuits to the partial product in the manner previously described. During each cycle, the adder 250-2 adds the contents of the RC and RS registers 200-10 and 200-12 and the carry signal produced from bit position (i.e. 76) during the next cycle. This takes into account any effect the discarded bits will have on the final product.

During the next cycle (i.e., sixth 50 nanosecond cycle), the carry save adder networks sum the multiples stored in the latch circuits RMGA, RMGB, and RMGC to the partial product. The adder 250-2 generates the carry out signal. The pertinent registers and circuits store and generate signals having the following values.

RCHO	= 0.543540000000
RCA	= 0.70 0
ZAQ	= 0
RMN	= 261
RMGA	= 1.07 7
RMGB	= 0.70 0
RMGC	= 0.340 0
RC	= 0.3410 0114
RS	= 1.41437 7654
ACSA CARRY	= 1.16347 76
ACSA SUM	= 0.61430 0
ACSB CARRY	= 0.541020 03
ACSB SUM	= 1.232657 710
ACSC CARRY	= 0.240410 014
ACSC SUM	= 1.7661437 740
FF-CARRY A 10	= 1
FF-CARRY B 10	= 0
FF-CARRY C 10	= 0
DL-FF-CARRY A 10	= 1
FULL ADDER (AM)	= 0.31640 0114
ADDER CARRY OUT BIT71	= 1
ADDER F-CARRY-IN BIT76	= 1

This completes the series of 50 nanosecond cycles and the circuits of block 401 condition the system to switch back to 100 nanosecond mode during the start of the next cycle. In this cycle, the pertinent registers and circuits have the following values.

RC	= 0.2404100 014
RS	= 1.7661437 740
DL-FF-CARRY A 10	= 1
F-CARRY-7176	= 1
FULL ADDER (AM)	= 0.4671640 0

during this cycle, the contents of the RC register 200-10 shifted left one is added to the contents of the RS register 200-12 to form a final product in stages 0-71 of adder 250-2. The most significant bit position of the adder is used to signal overflow conditions. Since the formation of the final product is complete, the example can be viewed as complete. The additional "clean up" cycles discussed previously simply provide for storage of the final product in the appropriate registers and the setting of the indicators.

For completeness, the following table is included illustrating the states of the carry save adder networks and adder during the last four cycles.

– CYCLE 6 –

ACSA	RS Right 2	000 000
	RC Right 1	000 000
	RMGA	000 000
ACSB	ACSA SUM Left 1	00000 00000
	ACSA CARRY	00000 00100
	RMGB	00000 00000
ACSC	ACSB SUM Left 1	0000000 0010000
	ACSB CARRY	0000000 0000000
	RMGC	1100011 1110000
	ACSC SUM	1100011 1100000
	ACSC CARRY	0000000 0010000

– CYCLE 7 –

ACSA	RS Right 2	111 100 011 111 111
	RC Right 1	000 000 000 000 000
	RMGA	110 001 111 111 111
ACSB	ACSA SUM	00001101100000 00011
	ACSA CARRY Left 1	11100000111 11000
	RMGA	00000000000 00011
ASCS	ACSB SUM	1111101101011111 1100011
	ACSB CARRY Left 1	0000000001000 0011011
	RMGC	0011100000000 0000011
	ACSC SUM	1100001100011111 1111011
	ACSC CARRY	0011100001000000 0000011
AM	RS	1110001111 11100000
	RC Left 1	0000000000 00100000
		0
	AM	1110010 00000

– CYCLE 8 –

ACSA	RS Right 2	111 100 001 100 011 111 111
	RC Right 1	000 111 000 010 000 000
	RMGA	100 011 111 111
ACSB	ACSA SUM	00011000110001100000 00011
	ACSA CARRY Left 1	11001110011100111 11000
	RMGB	01110000 00011
ACSC	ACSB SUM	11101001101011011011111 100011
	ACSB CARRY Left 1	0010110000100001000 011000
	RMGC	0011100000 000011
	ACSC SUM	11111101100011001111 111000
	ACSC CARRY	0010100000100001000 000011
AM	RS	1110000110001111 1011
	RC Left 1	0011100001000 0111
		1
	AM	0001100111010000 0011

– CYCLE 9 –

AM	RS	11111110110001100011111 1000
	RC Left 1	0010100000100001000 0111
		1
	AM	00100110111001110100000 0000

From the foregoing, it is seen that the system is able to perform multiplication operations at high speeds by employing a plurality of multiple generator circuits in combination with a corresponding number of carry save adder circuits. Under the control of timing circuits, the system generates groups of multiple signals in parallel while the carry save adder networks sum previously generated and stored multiple signals to the carry and sum components of a partial product, thereby reducing to a minimum the overall time required to execute a multiply instruction.

WHAT WE CLAIM IS:-

1. A multiplication system in which the multiplier is divided into a plurality of portions and each portion is divided into a plurality of sections, comprising:

a plurality of multiple selection circuits to which the sections of a single portion of the multiplier are applied in parallel, each forming a corresponding multiple of the multiplicand;

a corresponding plurality of multiple registers, fed from the multiple selection circuits; and

a corresponding plurality of carry save adders arranged in cascade, fed from the multiple registers and from the output of the cascade;

and wherein in operation each portion of the multiplier is applied to the multiple selection circuits in a respective cycle of a primary sequence of cycles to generate a set of multiples of the multiplicand for that portion, the set of multiples for that portion are stored in the multiple registers, and the contents of the multiple registers for that portion are added in the cascade concurrently with the generation of the set of multiples for the next portion (if any) of the multiplier.

2. A multiplication system according to claim 1 including sum and carry registers fed from the cascade and feeding back to the cascade.

3. A multiplication system according to either previous claim wherein each multiple selection circuit is fed with the multiplicand and generates the appropriate multiple afresh on each cycle.

4. A multiplication system according to claim 3 wherein each multiple selection circuit includes a plurality of sets of gating circuits each coupling the multiplicand in a different form to the corresponding multiple register, and a decoder controlled by the respective section of the multiplier to select one of the sets of gating circuits.

5. A multiplication system according to any previous claim wherein in operation preliminary and concluding cycles are performed in which operations ancillary to the primary sequence of cycles are carried out and which are of a different length to the cycles of the primary sequence.

6. A multiplication system substantially as herein described with reference to the accompanying drawings.

M.G. HARMAN
Chartered Patent Agent,
Agent for the Applicants

Printed for Her Majesty's Stationery Office, by Croydon Printing Company Limited, Croydon, Surrey, 1980.
Published by The Patent Office, 25 Southampton Buildings, London, WC2A 1AY,from
which copies may be obtained.

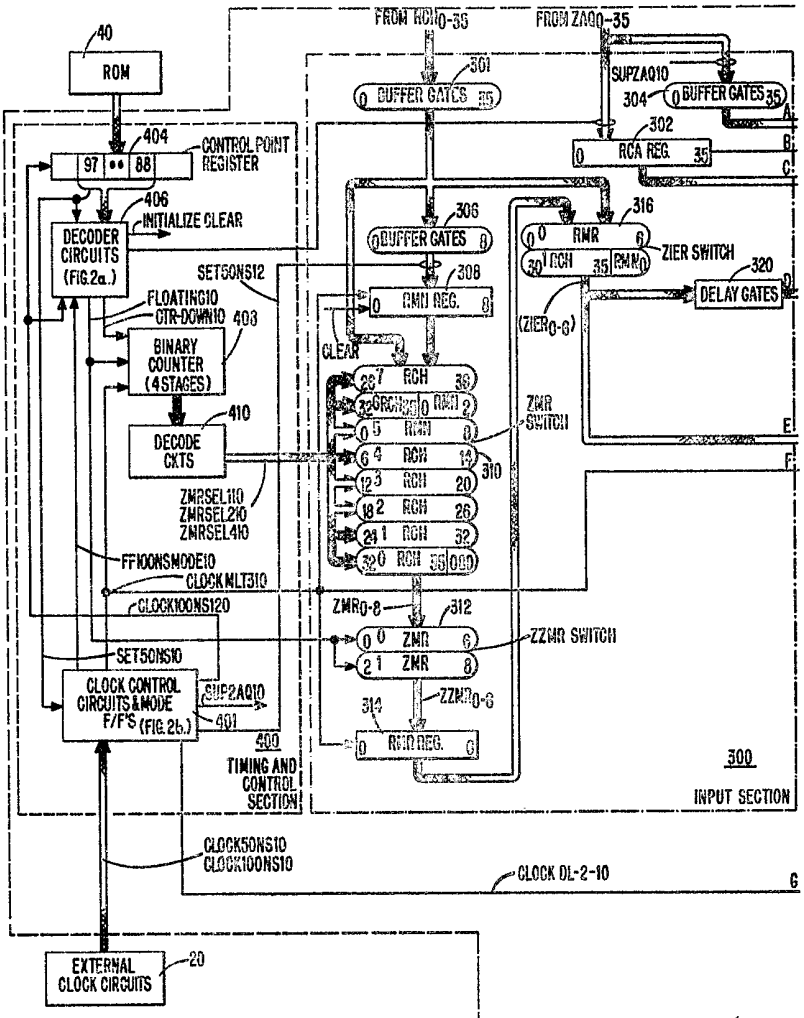
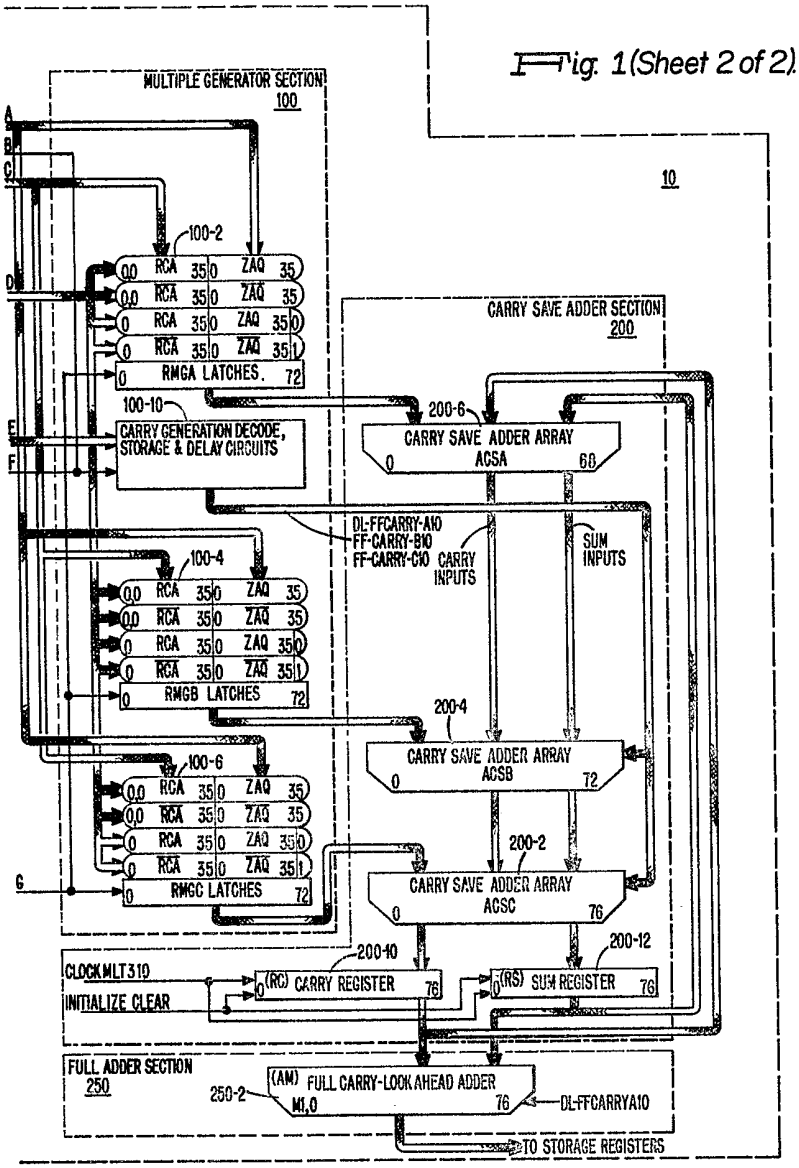


Fig. 1 (Sheet 1 of 2).

Fig. 1 (Sheet 2 of 2)



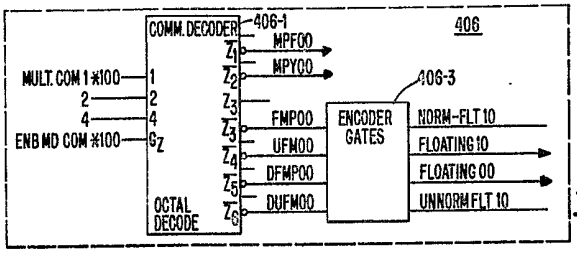


Fig. 2a.

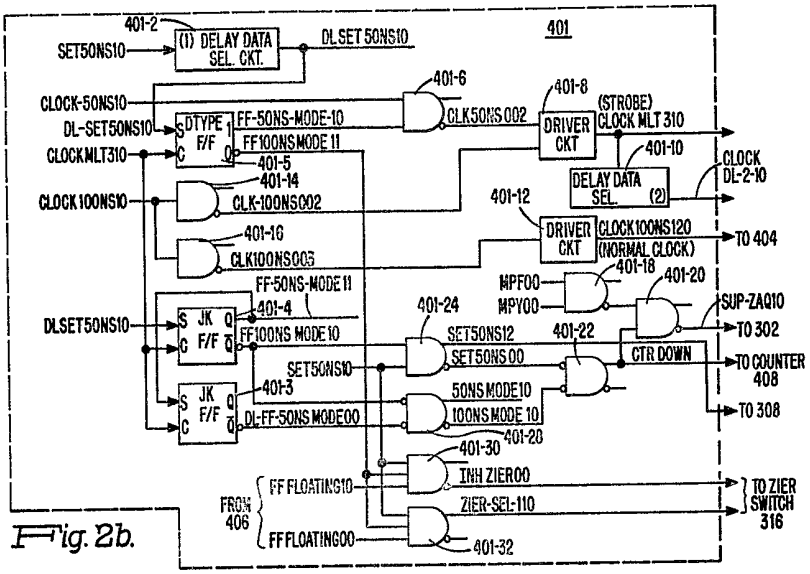


Fig. 2b.

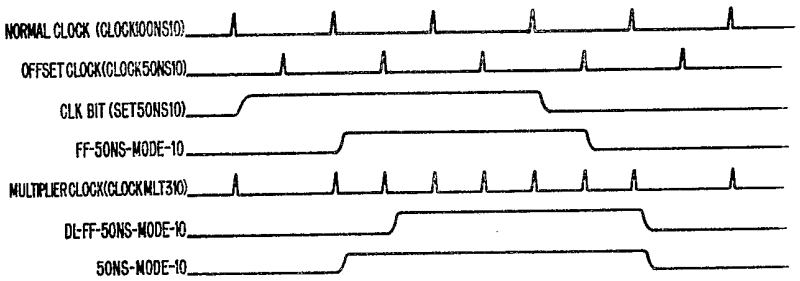


Fig. 3.

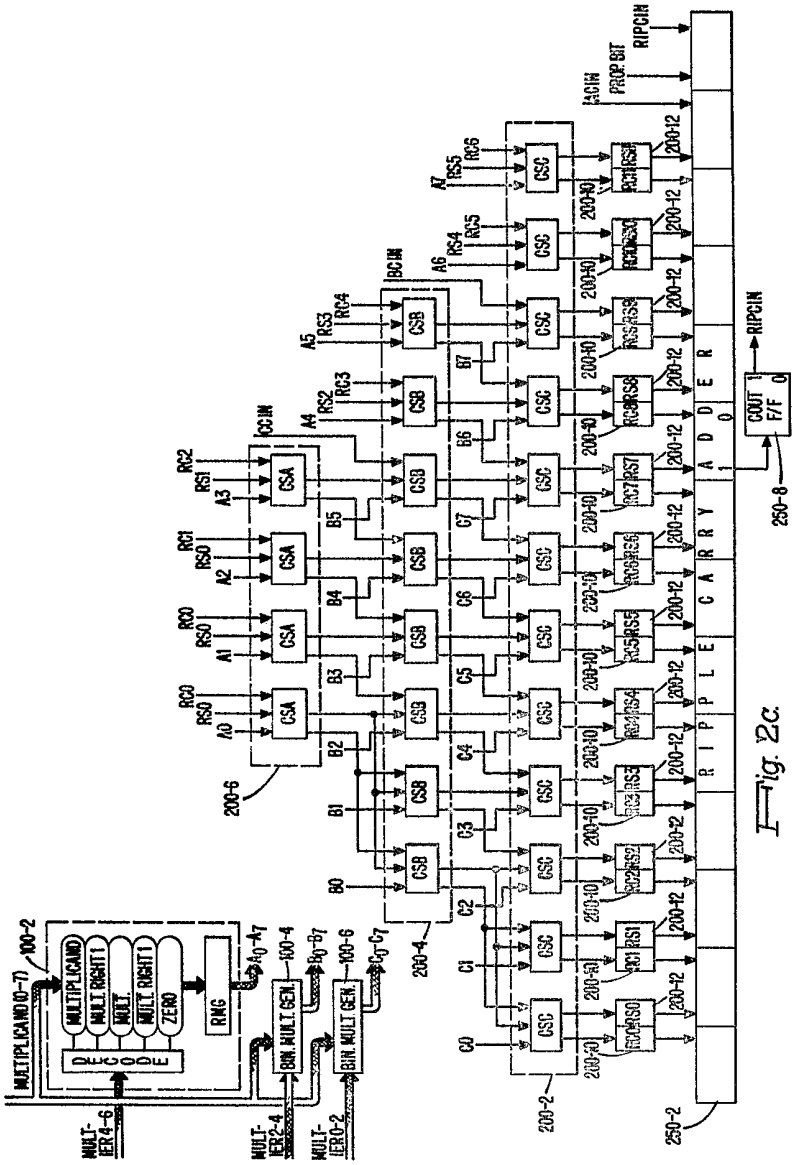

MULTIPLICAND (0-7)
100-2
MULTIPLICAND
MULT RIGHT 1
MULT
MULT RIGHT 1
ZERO
DECODE
RNG
BIN. MULT GEN.
100-4
BIN. MULT GEN.
100-6
MULT-IER 4-6
MULT-IER 2-4
MULT-IER 0-2
200-6
200-4
200-2
250-2
250-8
CSA
CSB
CSC
CARRY RIPPLE ADDER
COUT 1 F/F 0
RIPCIN
PROP BIT

Fig. 2c.

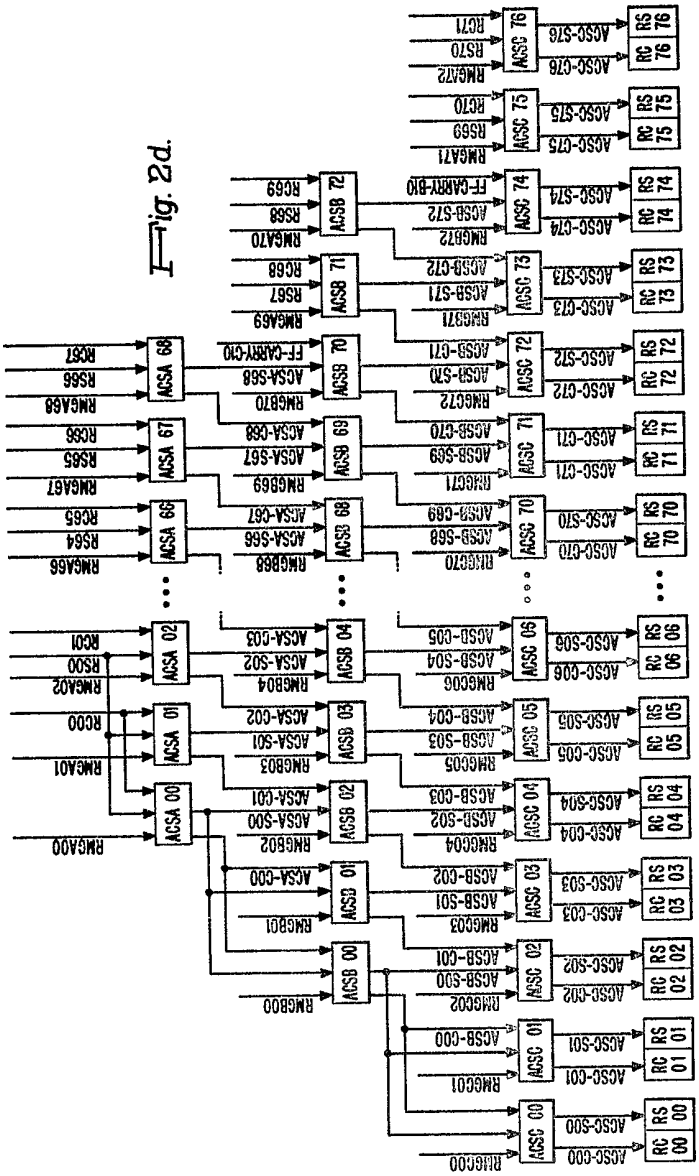


Fig. 2d.

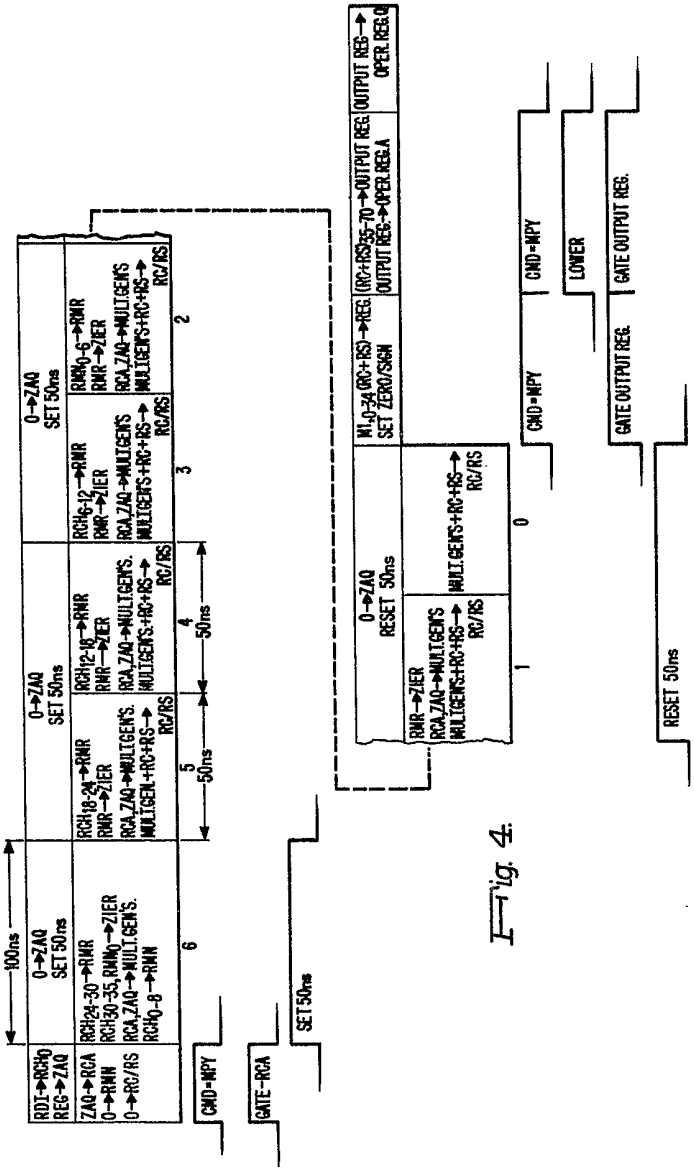

100ns
RDI→RCH0
REG→ZAQ
0→ZAQ
SET 50ns
ZAQ→RCA
0→RMR
0→RC/RS
RCH24-30→RMR
RCH30-35, RMM0→ZIER
RCA,ZAQ→MULT.GEN'S.
RCH0-8→RMR
CMD=MPY
GATE-RCA
SET 50ns
0→ZAQ
SET 50ns
RCH18-24→RMR
RMR→ZIER
RCA,ZAQ→MULTGEN'S.
MULTGEN.+RC+RS→
RC/RS
RCH12-18→RMR
RMR→ZIER
RCA,ZAQ→MULTGEN'S.
MULTGEN'S.+RC+RS→
RC/RS
50ns
50ns
0→ZAQ
SET 50ns
RCH6-12→RMR
RMR→ZIER
RCA,ZAQ→MULTGEMS
MULTGEMS+RC+RS→
RC/RS
RMM0-6→RMR
RMR→ZIER
RCA,ZAQ→MULTGEMS
MULTGEMS+RC+RS→
RC/RS
M1,0-24 (RC+RS)→REG.
SET ZERO/SIGN
(RC+RS)35-70→OUTPUT REG.
OUTPUT REG.→OPER.REG.A
OUTPUT REG→
OPER.REG.Q
RMR→ZIER
RCA,ZAQ→MULTGEM'S
MULTGEMS+RC+RS→
RC/RS
0→ZAQ
RESET 50ns
MULTGEMS+RC+RS→
RC/RS
CMD=MPY
CMD=MPY
LOWER
GATE OUTPUT REG.
GATE OUTPUT REG.
RESET 50ns
0
1
2
3
4
5
6

Fig. 4.