Memorandum to: Members of Dept. 271 September 7, 1961

Subject: The Multiplier Register on the 7030

The floating point multiplier register $MR is available for programming on K-2 and subsequent 7030 machines. It is not available on X-1 (LASL), the 7950, nor K-1 (LRL).

The following information was obtained from Don Gibson, Dale Kuhns and Gordon Zook from the 7030 improvement project in Kingston.

1) The floating point multiplier register $MR consists of exponent bits occupying $R.48 through $R.59 ($L.112 through $L.123), as well as fraction and sign bits occupying the last 52 bits of the "D" register in the E-box.

2) $MR has no address, and can be referred to only through explicit instructions.

3) The floating point instruction "load multiplier register" (LMR) on the improved machines loads $MR in the same was as the floating point LFT instruction loads $FT on K-1. The bit configurations are the same (10010).

4) The floating point instruction *+ on the improved machines behaves in the same way as the corresponding instruction on K-1; but $MR instead of $FT will be employed to furnish the implied operand. The bit configurations are the same (01110).

5) The floating point instruction "store multiplier (STM)" with bit configuration 11110 will be added to the improved machines. This instruction stores the contents of $MR into the full word specified by the effective address.

6) On the improved machines LMR, *+ and STM are capable of being modified by modifier bits just like other floating point instructions. They each occupy one level of the lookahead.

Incidentally, LFT, *+, M+, M+MG, D/ each consume two look-ahead levels on K-1. On the improved machines these instructions consume only one level each if the operand is in core memory.

7) There is a high degree of compatibility between improved and unimproved machines for binary coded non-VFL programs, except the following:

a) When $R is referred to as the operand address (bits $R.48 through $R.59 are shared between $R and $MR).

b) When $FT'is referred to as the operand address ($FT is still 14.0, however, on the improved machines.)

c) STM does not exist for the unimproved machines.

8) VFL instructions are unaltered on the improved machines, but $MR, aside from the sharing of bits with $R, as seen in (7a), is part of the VFL accumulator, and further is along the path of data flow for some VFL instructions. Thus

a) VFL fetch type instructions with offset less than 16 will alter the exponent bits of $MR. The corresponding store type instructions may store $MR exponent bits.

b) VFL load (L) and load with flag (LWF) automatically clears the $MR exponent to all zeros before the loading operation.

c) The fraction and sign byte parts of $MR are replaced whenever a VFL instruction calling for word-boundary crossover is executed. This is because the D register is needed for all such cases to house the second full word temporarily.

d) The sign byte of $MR is replaced in the following VFL instructions:

LFT, LTRS, LTRCV:
decimal *, decimal /, decimal *+

9) The question of the worth of the installment of $MR as a programming entity can be answered by the following timing comparisons: (all instructions floating point with minimum shifts)

LFT, 2.8 to 4.4 μs on K-1; LMR, 0.9 to 1.2 μs on K-2
*+, 6.5 μs on K-1; 3.9 to 4.5 μs on K-2

The arithmetic time in matrix multiplications is cut in half.

Tien Chi Chen
Special Studies

TCC:tm

P. S. Information to the right of the offset is changed whenever the accumulator has to be recomplemented on subtractions. Due to the bit-sharing with $R, the $MR exponent will therefore be altered by a VFL instruction which changes the sign of the accumulator.