R. Emery

CAM #13 KMPD January 2, 1957

Machine Organization Simulation Group (Job #1011) Department 575

Subject: Machine Logic Organization Simulation General Flow Chart and Summary

The simulation program and its stored data input are assumed to be in memory at the START point in the flow diagram.

The input program of instructions for the simulated machine is loaded into a section of memory. A control specifying the address of the first instruction to be executed is read into the Program Counter. A signal is then sent to read out the instruction at that address.

It must be determined next whether the read out signal is being blocked by the gate control on the signal line. Blocking levels may be generated by other portions of the simulation. Instructions of the TRANSFER class are known to cause such a condition while the contents of the Program Counter are being modified.

A control word is forwarded to the Decoding Program when the read out signal is gated. This control specifies the location of the register flow information for the instruction read out of memory. When decoded this will eventually transmit the instruction to the first decoding register.

The Decoding Program converts the register flow data into "signal bits". (i.e., binary decision elements). Such conversion is properly timed to sequence the machine operations where overlapping instructions are involved. Logic gates are also specified here in coded form if applicable to the instruction. These simulated gates are equivalent to gate circuits where the level input controls the passage of the signal input. The level input in this case is a means of expressing a restriction upon the signal or data transmission.

The program block following the Decoding Program controls the establishment of data transmission lines between pairs of registers. Actually, these lines are specified through the decoding operation. It remains for the Control Program to close the switch on the connections. This switching is accomplished in the precise pulse time that a transmission between two registers is to be initiated. The closing of a switch is indicated by two bits. One signifies the action to the transmitting register and the other to the receiving register.

In order to initiate the decoding of any given instruction in the machine, it must be determined when the operation code has entered the decoding register. This is done by interrogating the receiving end switch on that register. If it is closed, it is assumed that an instruction has reached the point where it may be considered for decoding in the next pulse time. The operation code is compared with a table of instruction classes or types until a match is found. Control words in the table at that point are forwarded to the Decoding Program.

The special subroutines indicated on the flow diagram may be used to cover conditions not readily handled in the general structure of the simulation program. It is hoped that there will be very few of these cases that require special handling.

Once the switch is closed on a transmission, it essentially activates a holding relay which keeps the line connected when the switch is released. In the simulation the switch remains closed for one pulse time. The Scan Program checks for lines which have been established in that pulse time. Upon finding one, the status indicators of the registers at either end of the transmission line are changed to indicate that a transmission in is progress. The transmitter is timed to the end of the transmission, and the receiver is clocked to the end of some minimum time during which it must retain the data received. At the final count down of a given time, the register is changed in status to indicate its ability to continue with the next operation. This changed status is relayed to the Control Program when it is significant to the establishement of new data transmission lines.

The next step in the program asks if any timing data is to be collected. A data collection program is available for this work. Its function is to gather information on specified registers in each pulse time. This information is stored for a print routine which converts it to a timing chart format.

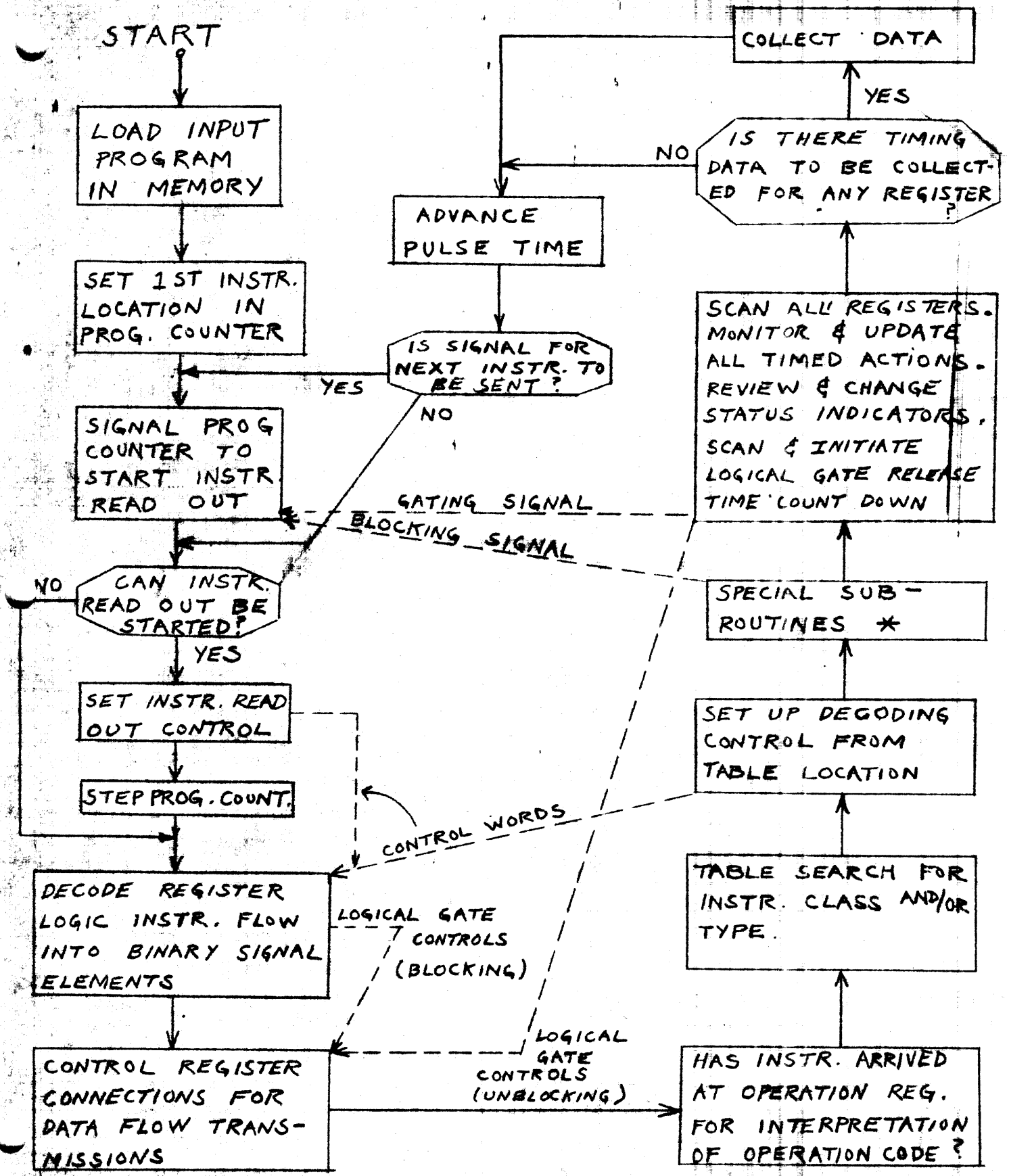
Preparatory to the next cycle through the simulation, the pulse time counter is advanced. According to the type machine being simulated a signal may be emitted from some source at a calculated time to call out the next instruction from memory. This completes the overall simulation loop. As can be seen, the simulation is now under the control of the instructions being simulated. The registers will send and receive data according to the dictates of these instructions in keeping with the logical configuration of the simulated machine.

R. F. Wolf

et

MACHINE LOGIC ORGANIZATION SIMULATION FLOW DIAGRAM

1-2-57



* TRANSFER CLASS OF INSTRUCTIONS MAY EXERT SPECIAL CONTROL ON READ OUT OF NEXT INSTRUCTION (SHORT PROGRAMS TO COVER UNUSUAL OR UNFORESEEN CASES FALL IN THIS CATEGORY)

Index Memory
Memory bus contrl out (data + instrs from memory)
Mem. bus (data from mem)
Mem IN Bus
Memory Control unit
checker out bus
checker out bus
From LA
addr out bus
IC Adder
IC
Y1 Reg
Y2 Reg
X Reg
Z Reg
W Reg
IAU Adder
updated Ind Reg
IAU CHECKER
addr in bus A
addr in bus B
Branch Test
fetch addr
store addr
data store
Mem out Bus
Load Look-ahead lines
checker in bus
To IC
Effective Address Reg
IC Buffer
IC
OP CODE
IND
Data
address modifier
Execution Register Decoder + Controls
data store
CD bus
Accumulator bus
upper Bound
lower Bound
Indicator Register
Interrupt mech.
Mask Reg
Arith checker
C Reg
D reg
Accumulator A B
Acc Sign
Left Zeros count
all ones count
VFL CONTROLS
Serial Arith. Unit
Parallel Floating Point Unit
High Speed Multiplier Unit
data count
arithmetic bus